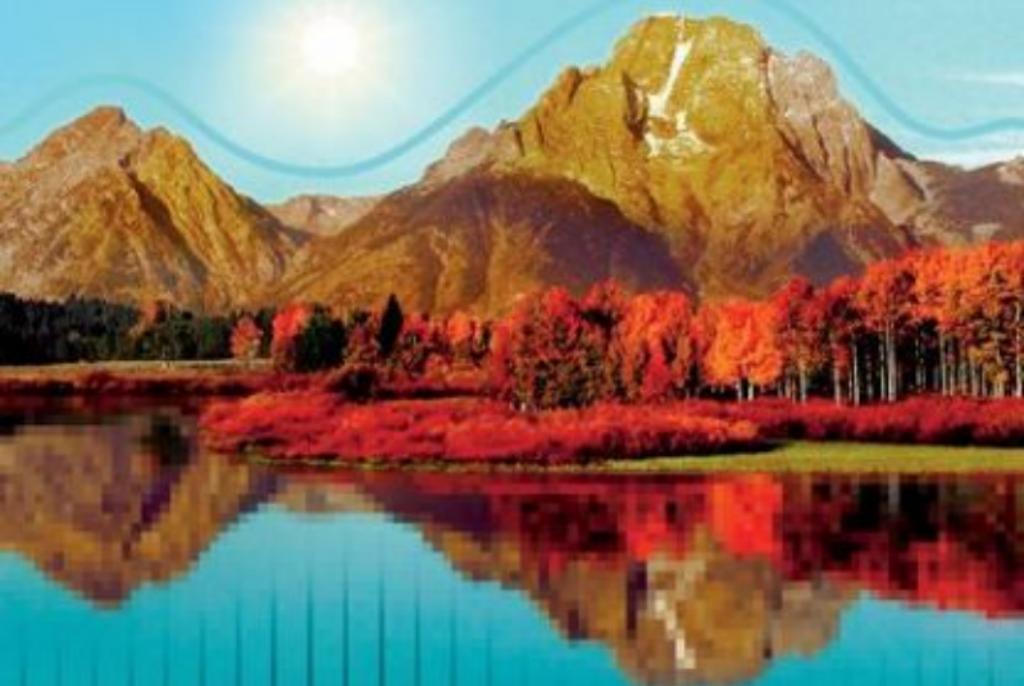


Essentials of Digital Signal Processing



B. P. Lathi
Roger Green

Essentials of Digital Signal Processing

This textbook offers a fresh approach to digital signal processing (DSP) that combines heuristic reasoning and physical appreciation with sound mathematical methods to illuminate DSP concepts and practices. It uses metaphors, analogies, and creative explanations along with carefully selected examples and exercises to provide deep and intuitive insights into DSP concepts.

Practical DSP requires hybrid systems including both discrete- and continuous-time components. This book follows a holistic approach and presents discrete-time processing as a seamless continuation of continuous-time signals and systems, beginning with a review of continuous-time signals and systems, frequency response, and filtering. The synergistic combination of continuous-time and discrete-time perspectives leads to a deeper appreciation and understanding of DSP concepts and practices.

Notable Features

1. Written for upper-level undergraduates
2. Provides an intuitive understanding and physical appreciation of essential DSP concepts without sacrificing mathematical rigor
3. Illustrates concepts with 500 high-quality figures, more than 170 fully worked examples, and hundreds of end-of-chapter problems
4. Encourages student learning with more than 150 drill exercises, including complete and detailed solutions
5. Maintains strong ties to continuous-time signals and systems concepts, with immediate access to background material with a notationally consistent format, helping readers build on their previous knowledge
6. Seamlessly integrates MATLAB throughout the text to enhance learning
7. Develops MATLAB code from a basic level to reinforce connections to underlying theory and sound DSP practice

B. P. Lathi holds a PhD in Electrical Engineering from Stanford University and was previously a Professor of Electrical Engineering at California State University, Sacramento. He is the author of eight books, including *Signal Processing and Linear Systems* (second ed., 2004) and, with Zhi Ding, *Modern Digital and Analog Communications Systems* (fourth ed., 2009).

Roger Green is an Associate Professor of Electrical and Computer Engineering at North Dakota State University. He holds a PhD from the University of Wyoming. He is co-author, with B. P. Lathi, on the second edition of *Signal Processing and Linear Systems*.

Essentials of Digital Signal Processing

Bhagawandas P. Lathi and Roger Green

*Sacramento State University,
North Dakota State University*



CAMBRIDGE

UNIVERSITY PRESS

32 Avenue of the Americas, New York, NY 10013-2473, USA

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org

Information on this title: www.cambridge.org/9781107059320

© Bhagawandas P. Lathi, Roger Green 2014

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press.

First published 2014

Printed in the United States of America

A catalog record for this publication is available from the British Library.

Library of Congress Cataloging in Publication Data

ISBN 978-1-107-05932-0 Hardback

Cambridge University Press has no responsibility for the persistence or accuracy of URLs for external or third-party Internet Web sites referred to in this publication and does not guarantee that any content on such Web sites is, or will remain, accurate or appropriate.

Contents

Preface	vii
1 Review of Continuous-Time Signals and Systems	1
1.1 Signals and Signal Categorizations	2
1.1.1 Continuous-Time and Discrete-Time Signals	3
1.1.2 Analog and Digital Signals	3
1.2 Operations on the Independent CT Variable	4
1.2.1 CT Time Shifting	4
1.2.2 CT Time Scaling	5
1.2.3 CT Time Reversal	5
1.2.4 Combined CT Time Shifting and Scaling	6
1.3 CT Signal Models	7
1.3.1 CT Unit Step Function $u(t)$	7
1.3.2 CT Unit Gate Function $\Pi(t)$	8
1.3.3 CT Unit Triangle Function $\Lambda(t)$	8
1.3.4 CT Unit Impulse Function $\delta(t)$	9
1.3.5 CT Exponential Function e^{st}	12
1.3.6 CT Interpolation Function $\text{sinc}(t)$	13
1.4 CT Signal Classifications	15
1.4.1 Causal, Noncausal, and Anti-Causal CT Signals	15
1.4.2 Real and Imaginary CT Signals	16
1.4.3 Even and Odd CT Signals	18
1.4.4 Periodic and Aperiodic CT Signals	21
1.4.5 CT Energy and Power Signals	21
1.4.6 Deterministic and Probabilistic Signals	25
1.5 CT Systems and Properties	25
1.5.1 Linearity	26
1.5.2 Time Invariance	27
1.5.3 The Zero-State Response of an LTIC System	28
1.5.4 Causality	29
1.5.5 Stability	29
1.6 Foundations of Frequency-Domain Analysis	30
1.6.1 LTIC System Response to an Everlasting Exponential e^{st}	30
1.7 The Fourier Series	33
1.7.1 Exponential Form of the Fourier Series	34
1.7.2 Trigonometric and Compact Trigonometric Forms	37
1.7.3 Convergence of a Series	41
1.8 The Fourier Transform	45
1.9 Fourier Transform Properties	50
1.9.1 Duality Property	51

1.9.2	Linearity Property	52
1.9.3	Complex-Conjugation Property	52
1.9.4	Scaling Property	53
1.9.5	Time-Shifting Property	54
1.9.6	Time-Differentiation and Time-Integration Properties	59
1.9.7	Time-Domain Convolution Property	59
1.9.8	Correlation and the Correlation Property	61
1.9.9	Extending Fourier Transform Properties to the Fourier Series	66
1.10	The Laplace Transform	68
1.10.1	Connection between the Fourier and Laplace Transforms	70
1.10.2	Laplace Transform Properties	72
1.11	Summary	73
2	Continuous-Time Analog Filters	85
2.1	Frequency Response of an LTIC System	85
2.1.1	Pole-Zero Plots	89
2.2	Signal Transmission through LTIC Systems	92
2.2.1	Distortionless Transmission	94
2.2.2	Real Bandpass Systems and Group Delay	97
2.3	Ideal and Realizable Filters	100
2.4	Data Truncation by Windows	104
2.4.1	Impairments Caused by Windowing	104
2.4.2	Lowpass Filter Design Using Windows	106
2.4.3	Remedies for Truncation Impairments	109
2.4.4	Common Window Functions	109
2.5	Specification of Practical Filters	112
2.6	Analog Filter Transformations	113
2.6.1	Lowpass-to-Lowpass Transformation	115
2.6.2	Lowpass-to-Highpass Transformation	116
2.6.3	Lowpass-to-Bandpass Transformation	117
2.6.4	Lowpass-to-Bandstop Transformation	118
2.7	Practical Filter Families	120
2.7.1	Butterworth Filters	120
2.7.2	Chebyshev Filters	129
2.7.3	Inverse Chebyshev Filters	139
2.7.4	Elliptic Filters	144
2.7.5	Bessel-Thomson Filters	147
2.8	Summary	149
3	Sampling: The Bridge from Continuous to Discrete	155
3.1	Sampling and the Sampling Theorem	155
3.1.1	Practical Sampling	161
3.2	Signal Reconstruction	164
3.3	Practical Difficulties in Sampling and Reconstruction	168
3.3.1	Aliasing in Sinusoids	173
3.4	Sampling of Bandpass Signals	176
3.5	Time-Sampling Dual: The Spectral Sampling Theorem	181
3.6	Analog-to-Digital Conversion	185
3.6.1	Analog-to-Digital Converter Transfer Characteristics	189
3.6.2	Analog-to-Digital Converter Errors	194
3.6.3	Analog-to-Digital Converter Implementations	196
3.7	Digital-to-Analog Conversion	199

3.7.1	Sources of Distortion in Signal Reconstruction	200
3.8	Summary	202
4	Discrete-Time Signals and Systems	212
4.1	Operations on the Independent DT Variable	214
4.1.1	DT Time Shifting	214
4.1.2	DT Time Reversal	215
4.1.3	DT Time Scaling: Sampling Rate Conversion	216
4.2	DT Signal Models	219
4.2.1	DT Unit Step Function $u[n]$	219
4.2.2	DT Unit Impulse Function $\delta[n]$	220
4.2.3	DT Exponential Function z^n	222
4.3	DT Signal Classifications	231
4.3.1	Causal, Noncausal, and Anti-Causal DT Signals	231
4.3.2	Real and Imaginary DT Signals	232
4.3.3	Even and Odd DT Signals	232
4.3.4	Periodic and Aperiodic DT Signals	233
4.3.5	DT Energy and Power Signals	236
4.4	DT Systems and Examples	238
4.4.1	The Order and General Form of Difference Equations	245
4.4.2	Kinship of Difference Equations to Differential Equations	246
4.4.3	Advantages of Digital Signal Processing	248
4.5	DT System Properties	248
4.5.1	Time Invariance	248
4.5.2	Linearity	250
4.5.3	The Zero-State Response of an LTID System	252
4.5.4	Causality	254
4.5.5	Stability	256
4.5.6	Memory	256
4.5.7	Invertibility	257
4.6	Digital Resampling	257
4.7	Summary	261
5	Time-Domain Analysis of Discrete-Time Systems	270
5.1	Iterative Solutions to Difference Equations	270
5.2	Operator Notation	275
5.3	The Zero-Input Response	277
5.3.1	Insights into the Zero-Input Behavior of a System	282
5.4	The Unit Impulse Response	284
5.4.1	Closed-Form Solution of the Impulse Response	285
5.5	The Zero-State Response	288
5.5.1	Convolution Sum Properties	291
5.5.2	Graphical Procedure for the Convolution Sum	294
5.5.3	Interconnected Systems	300
5.5.4	LTID System Response to an Everlasting Exponential z^n	303
5.6	Total Response	304
5.7	System Stability	305
5.7.1	External (BIBO) Stability	306
5.7.2	Internal (Asymptotic) Stability	306
5.8	Intuitive Insights into System Behavior	311
5.8.1	Dependence of System Behavior on Characteristic Modes	311
5.8.2	Response Time of a System: The System Time Constant	312

5.8.3	Time Constant and Rise Time of a System	314
5.8.4	Time Constant and Filtering	314
5.8.5	Time Constant and Pulse Dispersion	315
5.8.6	The Resonance Phenomenon	315
5.9	Classical Solution of Linear Difference Equations	317
5.10	Summary	322
6	Discrete-Time Fourier Analysis	331
6.1	The Discrete-Time Fourier Transform	331
6.1.1	The Nature of Fourier Spectra	337
6.1.2	Obtaining the DTFT from the CTFT	338
6.1.3	DTFT Tables and the Nuisance of Periodicity	340
6.2	Properties of the DTFT	343
6.2.1	Duality	343
6.2.2	Linearity Property	343
6.2.3	Complex-Conjugation Property	343
6.2.4	Time Scaling and the Time-Reversal Property	344
6.2.5	Time-Shifting Property	345
6.2.6	Frequency-Differentiation Property	350
6.2.7	Time-Domain and Frequency-Domain Convolution Properties	351
6.2.8	Correlation and the Correlation Property	354
6.3	LTID System Analysis by the DTFT	355
6.3.1	Distortionless Transmission	359
6.3.2	Ideal and Realizable Filters	362
6.4	Connection between the DTFT and the CTFT	364
6.5	Digital Processing of Analog Signals	370
6.5.1	A Mathematical Representation	371
6.5.2	Time-Domain Criterion: The Impulse Invariance Method	373
6.6	Digital Resampling: A Frequency-Domain Perspective	379
6.6.1	Using Bandlimited Interpolation to Understand Resampling	380
6.6.2	Downsampling and Decimation	383
6.6.3	Interpolation and Upsampling	387
6.6.4	Time-Domain Characterizations	391
6.6.5	Fractional Sampling Rate Conversion	394
6.7	Generalization of the DTFT to the z -Transform	395
6.8	Summary	397
7	Discrete-Time System Analysis Using the z-Transform	410
7.1	The z -Transform	410
7.1.1	The Bilateral z -Transform	410
7.1.2	The Unilateral z -Transform	416
7.2	The Inverse z -Transform	419
7.2.1	Inverse z -Transform by Power Series Expansion	425
7.3	Properties of the z -Transform	427
7.3.1	Linearity Property	427
7.3.2	Complex-Conjugation Property	427
7.3.3	Time Scaling and the Time-Reversal Property	428
7.3.4	Time-Shifting Property	428
7.3.5	z -Domain Scaling Property	432
7.3.6	z -Domain Differentiation Property	433
7.3.7	Time-Domain Convolution Property	433
7.3.8	Initial and Final Value Theorems	435

7.4	<i>z</i> -Transform Solution of Linear Difference Equations	436
7.4.1	Zero-State Response of LTID Systems: The Transfer Function	439
7.5	Block Diagrams and System Realization	445
7.5.1	Direct Form Realizations	447
7.5.2	Transposed Realizations	451
7.5.3	Cascade and Parallel Realizations	453
7.6	Frequency Response of Discrete-Time Systems	457
7.6.1	Frequency-Response from Pole-Zero Locations	462
7.7	Finite Word-Length Effects	469
7.7.1	Finite Word-Length Effects on Poles and Zeros	469
7.7.2	Finite Word-Length Effects on Frequency Response	472
7.8	Connection between the Laplace and <i>z</i> -Transforms	474
7.9	Summary	476
8	Digital Filters	485
8.1	Infinite Impulse Response Filters	485
8.1.1	The Impulse Invariance Method Revisited	486
8.1.2	The Bilinear Transform	491
8.1.3	The Bilinear Transform with Prewarping	497
8.1.4	Highpass, Bandpass, and Bandstop Filters	501
8.1.5	Realization of IIR Filters	508
8.2	Finite Impulse Response Filters	511
8.2.1	Linear Phase FIR Filters	511
8.2.2	Realization of FIR Filters	515
8.2.3	Windowing in FIR Filters	517
8.2.4	Time-Domain Methods of FIR Filter Design	521
8.2.5	Window Method FIR Filter Design for Given Specifications	529
8.2.6	Frequency-Domain Methods of FIR Filter Design	537
8.2.7	Frequency-Weighted Least-Squares FIR Filter Design	544
8.3	Summary	552
9	Discrete Fourier Transform	559
9.1	The Discrete Fourier Transform	560
9.1.1	The Picket Fence Effect and Zero Padding	563
9.1.2	Matrix Representation of the DFT and Its Inverse	565
9.1.3	DFT Interpolation to Obtain the DTFT	567
9.2	Uniqueness: Why Confine $x[n]$ to $0 \leq n \leq N - 1$?	569
9.2.1	Modulo- N Operation	572
9.2.2	Circular Representation of an N -Length Sequence	573
9.3	Properties of the DFT	579
9.3.1	Duality Property	579
9.3.2	Linearity Property	579
9.3.3	Complex-Conjugation Property	580
9.3.4	Time-Reversal Property	580
9.3.5	Circular Shifting Properties	580
9.3.6	Circular Convolution Properties	581
9.3.7	Circular Correlation Property	582
9.4	Graphical Interpretation of Circular Convolution	583
9.4.1	Circular and Linear Convolution	585
9.4.2	Aliasing in Circular Convolution	588
9.5	Discrete-Time Filtering Using the DFT	590
9.5.1	Block Convolution	593

9.6	Goertzel's Algorithm	600
9.7	The Fast Fourier Transform	603
9.7.1	Decimation-in-Time Algorithm	604
9.7.2	Decimation-in-Frequency Algorithm	609
9.8	The Discrete-Time Fourier Series	612
9.9	Summary	617
A	MATLAB	625
B	Useful Tables	640
C	Drill Solutions	646
Index		731

Preface

Since its emergence as a field of importance in the 1970s, digital signal processing (DSP) has grown in exponential lockstep with advances in digital hardware. Today's digital age requires that undergraduate students master material that was, until recently, taught primarily at the graduate level. Many DSP textbooks remain rooted in this graduate-level foundation and cover an exhaustive (and exhausting!) number of topics. This book provides an alternative. Rather than cover the broadest range of topics possible, we instead emphasize a narrower set of core digital signal processing concepts. Rather than rely solely on mathematics, derivations, and proofs, we instead balance necessary mathematics with a physical appreciation of subjects through heuristic reasoning, careful examples, metaphors, analogies, and creative explanations. Throughout, our underlying goal is to make digital signal processing as accessible as possible and to foster an intuitive understanding of the material.

Practical DSP requires hybrid systems that include both discrete-time and continuous-time components. Thus, it is somewhat curious that most DSP textbooks focus almost exclusively on discrete-time signals and systems. This book takes a more holistic approach and begins with a review of continuous-time signals and systems, frequency response, and filtering. This material, while likely familiar to most readers, sets the stage for sampling and reconstruction, digital filtering, and other aspects of complete digital signal processing systems. The synergistic combination of continuous-time and discrete-time perspectives leads to a deeper and more complete understanding of digital signal processing than is possible with a purely discrete-time viewpoint. A strong foundation of continuous-time concepts naturally leads to a stronger understanding of discrete-time concepts.

Notable Features

Some notable features of this book include the following:

1. This text is written for an upper-level undergraduate audience, and topic treatment is appropriately geared to the junior and senior levels. This allows a sufficiently detailed mathematical treatment to obtain a solid foundation and competence in DSP without losing sight of the basics.
2. An underlying philosophy of this textbook is to provide a simple and intuitive understanding of essential DSP concepts without sacrificing mathematical rigor. Much attention has been paid to provide clear, friendly, and enjoyable writing. A physical appreciation of the topics is attained through a balance of intuitive explanations and necessary mathematics. Concepts are illustrated using nearly 500 high-quality figures and over 170 fully worked examples. Further reinforcement is provided through over 150 drill exercises, complete detailed solutions of which are provided as an appendix to the book. Hundreds of end-of-chapter problems provide students with additional opportunities to learn and practice.
3. Unlike most DSP textbooks, this book maintains strong ties to continuous-time signals and systems concepts, which helps readers to better understand complete DSP systems. Further, by leveraging off a solid background of continuous-time concepts, discrete-time concepts are more easily and completely understood. Since the continuous-time background material is

included, readers have immediate access to as much or little background material as necessary, all in a notationally-consistent format.

4. MATLAB is effectively utilized throughout the text to enhance learning. This MATLAB material is tightly and seamlessly integrated into the text so as to seem a natural part of the material and problem solutions rather than an added afterthought. Unlike many DSP texts, this book does not have specific “MATLAB Examples” or “MATLAB Problems” any more than it has “Calculator Examples” or “Calculator Problems.” Modern DSP has evolved to the point that sophisticated computer packages (such as MATLAB) should be used every bit as naturally as calculus and calculators, and it is this philosophy that guides the manner that MATLAB is incorporated into the book.

Many DSP books rely on canned MATLAB functions to solve various digital signal processing problems. While this produces results quickly and with little effort, students often miss how problem solutions are coded or how theory is translated into practice. This book specifically avoids high-level canned functions and develops code from a more basic level; this approach reinforces connections to the underlying theory and develops sound skills in the practice of DSP. Every piece of MATLAB code precisely conforms with book concepts, equations, and notations.

Book Organization and Use

Roughly speaking, this book is organized into five parts.

1. Review of continuous-time signals and systems (Ch. 1) and continuous-time (analog) filtering (Ch. 2).
2. Sampling and reconstruction (Ch. 3).
3. Introduction to discrete-time signals and systems (Ch. 4) and the time-domain analysis of discrete-time systems (Ch. 5).
4. Frequency-domain analysis of discrete-time systems using the discrete-time Fourier transform (Ch. 6) and the z -transform (Ch. 7).
5. Discrete-time (digital) filtering (Ch. 8) and the discrete-Fourier transform (Ch. 9).

The first quarter of this book (Chs. 1 and 2, about 150 pages) focuses on continuous-time concepts, and this material can be scanned or skipped by those readers who possess a solid background in these areas. The last three quarters of the book (Chs. 3 through 9, about 450 pages) cover traditional discrete-time concepts that form the backbone of digital signal processing. The majority of the book can be covered over a semester in a typical 3 or 4 credit-hour undergraduate-level course, which corresponds to around 45 to 60 lecture-hours of contact.

As with most text books, this book can be adapted to accommodate a range of courses and student backgrounds. Students with solid backgrounds in continuous-time signals and systems can scan or perhaps altogether skip the first two chapters. Students with knowledge in the time-domain analysis of discrete-time signals and systems can scan or skip Chs. 4 and 5. Courses that do not wish to emphasize filtering operations can eliminate coverage of Chs. 2 and 8. Many other options exist as well. For example, students enter the 3-credit Applied Digital Signal Processing and Filtering course at North Dakota State University having completed a 4-credit Signals and Systems course that covers both continuous-time and discrete-time concepts, including Laplace and z -transforms but not including discrete-time Fourier analysis. Given this student background, the NDSU DSP course covers Chs. 2, 3, 6, 8, and 9, which leaves enough extra time to introduce (and use) digital signal processing hardware from Texas Instruments; Chs. 1, 4, 5, and 7 are recommended for reading, but not required.

Acknowledgments

We would like to offer our sincere gratitude to the many people who have generously given their time and talents to the creation, improvement, and refinement of this book. Books, particularly sizable ones such as this, involve a seemingly infinite number of details, and it takes the combined efforts of a good number of good people to successfully focus these details into a quality result. During the six years spent preparing this book, we have been fortunate to receive valuable feedback and recommendations from numerous reviewers, colleagues, and students. We are grateful for the reviews provided by Profs. Zekeriya Aliyazicioglu of California State Polytechnic University-Pomona, Mehmet Celenk of Ohio University, Liang Dong of Western Michigan University, Jake Gunther of Utah State University, Joseph P. Hoffbeck of the University of Portland, Jianhua Liu of Embry-Riddle Aeronautical University, Peter Mathys of the University of Colorado, Phillip A. Mlsna of Northern Arizona University, S. Hossein Mousavinezhad of Idaho State University, Kalyan Mondal of Fairleigh Dickinson University, Anant Sahai of UC Berkeley, Jose Sanchez of Bradley University, and Xiaomu Song of Widener University. We also offer our heartfelt thanks for the thoughtful comments and suggestions provided by the many anonymous reviewers, who outnumbered the other reviewers more than two-to-one. We wish that we could offer a more direct form of recognition to these reviewers. Some of the most thoughtful and useful comments came from students taking the Applied Digital Signal Processing and Filtering course at North Dakota State University. Two students in particular – Kyle Kraning and Michael Boyko – went above the call of duty, providing over one hundred corrections and comments. For their creative contributions of cartoon ideas, we also give thanks to NDSU students Stephanie Rosen (Chs. 1, 4, and 5) and Tanner Voss (Ch. 2). Book writing is a time-consuming activity, and one that inevitably causes hardship to those who are close to an author. Thus, we offer our final thanks to our families for their sacrifice, support, and love.

B. P. Lathi

R. A. Green



DSP is always on the future's horizon!

Chapter 1

Review of Continuous-Time Signals and Systems

This chapter reviews the basics of continuous-time (CT) signals and systems. Although the reader is expected to have studied this background as a prerequisite for this course, a thorough yet abbreviated review is both justified and wise since a solid understanding of continuous-time concepts is crucial to the study of digital signal processing.

Why Review Continuous-Time Concepts?

It is natural to question how continuous-time signals and systems concepts are relevant to digital signal processing. To answer this question, it is helpful to first consider elementary signals and systems structures.

In the most simplistic sense, the study of signals and systems is described by the block diagram shown in Fig. 1.1a. An input signal is fed into a system to produce an output signal. Understanding this block diagram in a completely general sense is quite difficult, if not impossible. A few well-chosen and reasonable restrictions, however, allow us to fully understand and mathematically quantify the character and behavior of the input, the system, and the output.

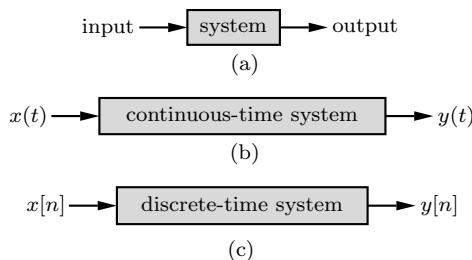


Figure 1.1: Elementary block diagrams of (a) general, (b) continuous-time, and (c) discrete-time signals and systems.

Introductory textbooks on signals and systems often begin by restricting the input, the system, and the output to be continuous-time quantities, as shown in Fig. 1.1b. This diagram captures the basic structure of continuous-time signals and systems, the details of which are reviewed later in this chapter and the next. Restricting the input, the system, and the output to be discrete-time (DT) quantities, as shown in Fig. 1.1c, leads to the topic of discrete-time signals and systems.

Typical digital signal processing (DSP) systems are hybrids of continuous-time and discrete-time systems. Ordinarily, DSP systems begin and end with continuous-time signals, but they process

signals using a digital signal processor of some sort. Specialized hardware is required to bridge the continuous-time and discrete-time worlds. As the block diagram of Fig. 1.2 shows, general DSP systems are more complex than either Figs. 1.1b or 1.1c allow; both CT and DT concepts are needed to understand complete DSP systems.

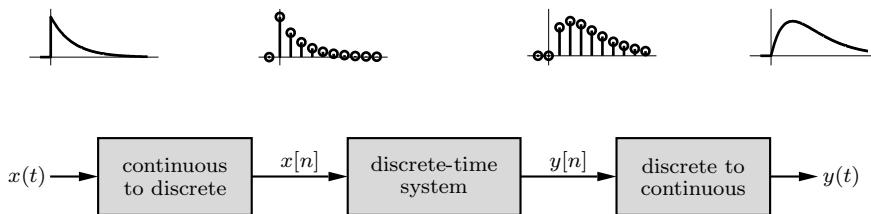


Figure 1.2: Block diagram of a typical digital signal processing system.

A more detailed explanation of Fig. 1.2 helps further justify why it is important for us to review continuous-time concepts. The continuous-to-discrete block converts a continuous-time input signal into a discrete-time signal, which is then processed by a digital processor. The discrete-time output of the processor is then converted back to a continuous-time signal.[†] Only with knowledge of continuous-time signals and systems is it possible to understand these components of a DSP system. Sampling theory, which guides our understanding of the CT-to-DT and DT-to-CT converters, can be readily mastered with a thorough grasp of continuous-time signals and systems. Additionally, the discrete-time algorithms implemented on the digital signal processor are often synthesized from continuous-time system models. All in all, continuous-time signals and systems concepts are useful and necessary to understand the elements of a DSP system.

Nearly all basic concepts in the study of continuous-time signals and systems apply to the discrete-time world, with some modifications. Hence, it is economical and very effective to build on the previous foundations of continuous-time concepts. Although discrete-time math is inherently simpler than continuous-time math (summation rather than integration, subtraction instead of differentiation), students find it difficult, at first, to grasp basic discrete-time concepts. The reasons are not hard to find. We are all brought up on a steady diet of continuous-time physics and math since high school, and we find it easier to identify with the continuous-time world. It is much easier to grasp many concepts in continuous-time than in discrete-time. Rather than fight this reality, we might use it to our advantage.

1.1 Signals and Signal Categorizations

A *signal* is a set of data or information. Examples include telephone and television signals, monthly sales of a corporation, and the daily closing prices of a stock market (e.g., the Dow Jones averages). In all of these examples, the signals are functions of the independent variable *time*. This is not always the case, however. When an electrical charge is distributed over a body, for instance, the signal is the charge density, a function of *space* rather than time. In this book we deal almost exclusively with signals that are functions of time. The discussion, however, applies equally well to other independent variables.

Signals are categorized as either continuous-time or discrete-time and as either analog or digital. These fundamental signal categories, to be described next, facilitate the systematic and efficient analysis and design of signals and systems.

[†]As we shall later see, the continuous-to-discrete block is typically comprised of a signal conditioning circuit followed by a CT-to-DT converter and an analog-to-digital converter (ADC). Similarly, the discrete-to-continuous block is typically comprised of a digital-to-analog converter (DAC) followed by a DT-to-CT converter and finally another conditioning circuit.

1.1.1 Continuous-Time and Discrete-Time Signals

A signal that is specified for every value of time t is a *continuous-time signal*. Since the signal is known for every value of time, precise event localization is possible. The tidal height data displayed in Fig. 1.3a is an example of a continuous-time signal, and signal features such as daily tides as well as the effects of a massive tsunami are easy to locate.

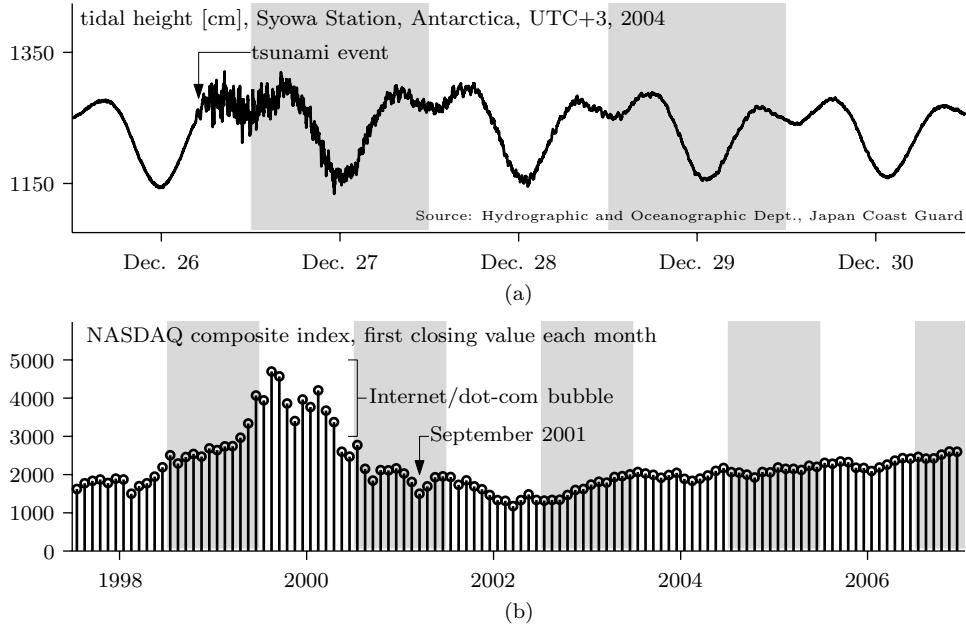


Figure 1.3: Examples of (a) continuous-time and (b) discrete-time signals.

A signal that is specified only at discrete values of time is a *discrete-time signal*. Ordinarily, the independent variable for discrete-time signals is denoted by the integer n . For discrete-time signals, events are localized within the sampling period. The technology-heavy NASDAQ composite index displayed in Fig. 1.3b is an example of a discrete-time signal, and features such as the Internet/dot-com bubble as well as the impact of the September 11 terrorist attacks are visible with a precision that is limited by the one month sampling interval.

1.1.2 Analog and Digital Signals

The concept of continuous-time is often confused with that of analog. The two are not the same. The same is true of the concepts of discrete-time and digital. A signal whose amplitude can take on any value in a continuous range is an *analog signal*. This means that an analog signal amplitude can take on an infinite number of values. A *digital signal*, on the other hand, is one whose amplitude can take on only a finite number of values. Signals associated with typical digital devices take on only two values (binary signals). A digital signal whose amplitudes can take on L values is an *L -ary signal* of which binary ($L = 2$) is a special case.

The terms “continuous-time” and “discrete-time” qualify the nature of a signal along the time (horizontal) axis. The terms “analog” and “digital,” on the other hand, qualify the nature of the signal amplitude (vertical axis). Using a sinusoidal signal, Fig. 1.4 demonstrates the various differences. It is clear that analog is not necessarily continuous-time and that digital need not be discrete-time. Figure 1.4c shows, for example, an analog, discrete-time signal. We shall discuss later a systematic procedure for A/D conversion, which involves quantization (rounding off), as explained in Sec. 3.6.

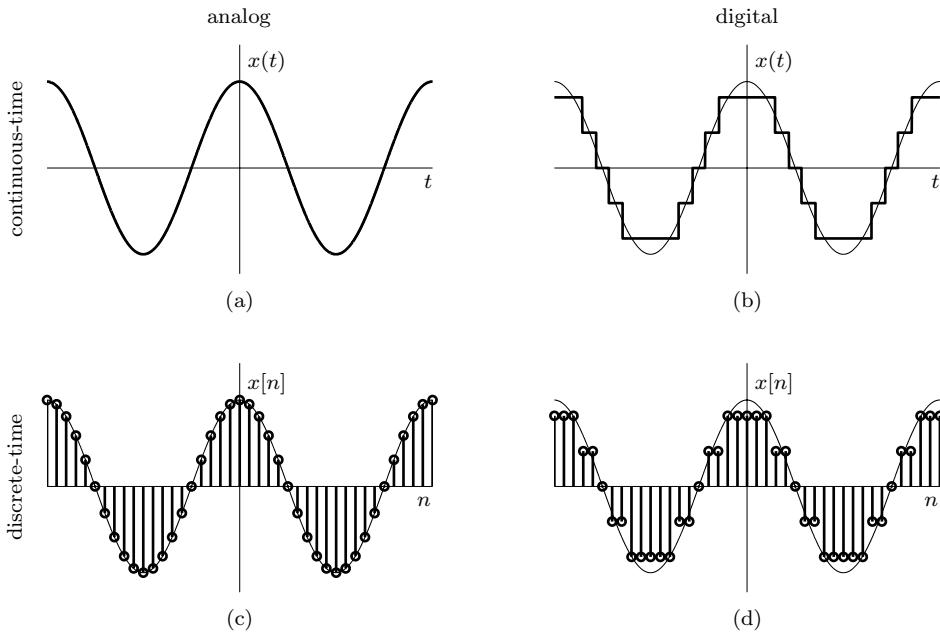


Figure 1.4: Examples of (a) analog, continuous-time, (b) digital, continuous-time, (c) analog, discrete-time, and (d) digital, discrete-time sinusoids.

Signals in the physical world tend to be analog and continuous-time in nature (Fig. 1.4a). Digital, continuous-time signals (Fig. 1.4b) are not common in typical engineering systems. As a result, when we refer to a continuous-time signal, an analog continuous-time signal is implied.

Computers operate almost exclusively with digital, discrete-time data (Fig. 1.4d). Digital representations can be difficult to mathematically analyze, so we often treat computer signals as if they were *analog* rather than digital (Fig. 1.4c). Such approximations are mathematically tractable and provide needed insights into the behavior of DSP systems and signals.

1.2 Operations on the Independent CT Variable

We shall review three useful operations that act on the independent variable of a CT signal: shifting, scaling, and reversal. Since they act on the independent variable, these operations do not change the shape of the underlying signal. Detailed derivations of these operations can be found in [1]. Although the independent variable in our signal description is time, the discussion is valid for functions having continuous independent variables other than time (e.g., frequency or distance).

1.2.1 CT Time Shifting

A signal $x(t)$ (Fig. 1.5a) delayed by $b > 0$ seconds (Fig. 1.5b) is represented by $x(t - b)$. Similarly, the signal $x(t)$ advanced by $b > 0$ seconds (Fig. 1.5c) is represented by $x(t + b)$. Thus, to time shift a signal $x(t)$ by b seconds, we replace t with $t - b$ everywhere in the expression for $x(t)$. If b is positive, the shift represents a time delay; if b is negative, the shift represents a time advance by $|b|$. This is consistent with the fact that a time delay of b seconds can be viewed as a time advance of $-b$ seconds.

Notice that the time-shifting operation is on the independent variable t ; the function itself remains unchanged. In Fig. 1.5, the function $x(\cdot)$ starts when its argument equals T_1 . Thus, $x(t - b)$ starts

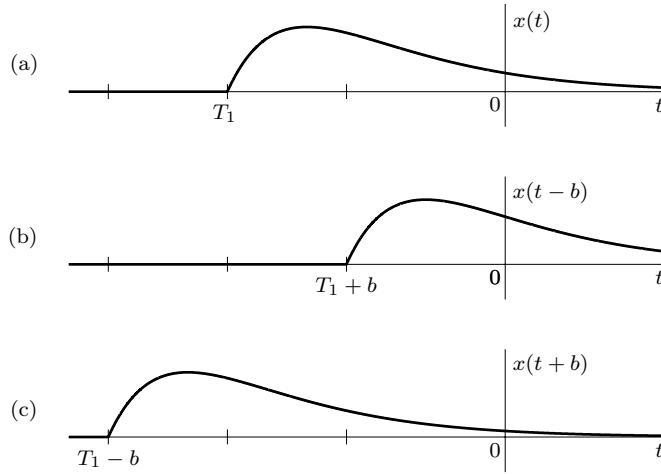


Figure 1.5: Time shifting a CT signal: (a) original signal, (b) delay by b , and (c) advance by b .

when its argument $t - b$ equals T_1 , or $t = T_1 + b$. Similarly, $x(t + b)$ starts when $t + b = T_1$, or $t = T_1 - b$.

1.2.2 CT Time Scaling

A signal $x(t)$, when time compressed by factor $a > 1$, is represented by $x(at)$. Similarly, a signal time expanded by factor $a > 1$ is represented by $x(t/a)$. Figure 1.6a shows a signal $x(t)$. Its factor-2 time-compressed version is $x(2t)$ (Fig. 1.6b), and its factor-2 time-expanded version is $x(t/2)$ (Fig. 1.6c). In general, to time scale a signal $x(t)$ by factor a , we replace t with at everywhere in the expression for $x(t)$. If $a > 1$, the scaling represents time compression (by factor a), and if $0 < a < 1$, the scaling represents time expansion (by factor $1/a$). This is consistent with the fact that time compression by factor a can be viewed as time expansion by factor $1/a$.

As in the case of time shifting, time scaling operates on the independent variable and does not change the underlying function. In Fig. 1.6, the function $x(\cdot)$ has a maximum value when its argument equals T_1 . Thus, $x(2t)$ has a maximum value when its argument $2t$ equals T_1 , or $t = T_1/2$. Similarly, $x(t/2)$ has a maximum when $t/2 = T_1$, or $t = 2T_1$.

▷ Drill 1.1 (CT Time Scaling)

Show that the time compression of a sinusoid by a factor a ($a > 1$) results in a sinusoid of the same amplitude and phase, but with the frequency increased a -fold. Similarly, show that the time expansion of a sinusoid by a factor a ($a > 1$) results in a sinusoid of the same amplitude and phase, but with the frequency reduced by a factor a . Verify your conclusions by sketching the sinusoid $\sin(2t)$ and the same sinusoid compressed by a factor 3 and expanded by a factor 2.

△

1.2.3 CT Time Reversal

Consider the signal $x(t)$ in Fig. 1.7a. We can view $x(t)$ as a rigid wire frame hinged at the vertical axis. To time reverse $x(t)$, we rotate this frame 180° about the vertical axis. This time reversal, or reflection of $x(t)$ about the vertical axis, gives us the signal $x(-t)$ (Fig. 1.7b); whatever happens in Fig. 1.7a at some instant t also happens in Fig. 1.7b at the instant $-t$. Thus, the mirror image of

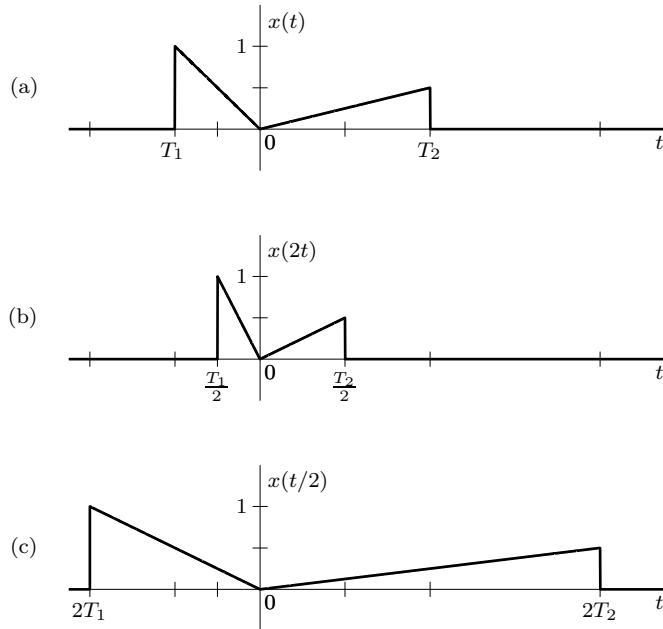


Figure 1.6: Time scaling a CT signal: (a) original signal, (b) compress by 2, and (c) expand by 2.

$x(t)$ about the vertical axis is $x(-t)$. Notice that time reversal is a special case of the time-scaling operation $x(at)$ where $a = -1$.

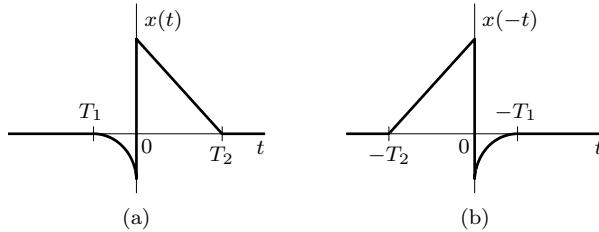


Figure 1.7: Time reversing a CT signal: (a) original signal and (b) its time reverse.

1.2.4 Combined CT Time Shifting and Scaling

Many circumstances require simultaneous use of more than one of the previous operations. The most general case is $x(at - b)$, which is realized in two possible sequences of operations:

1. Time shift $x(t)$ by b to obtain $x(t - b)$. Now time scale the shifted signal $x(t - b)$ by a (i.e., replace t with at) to obtain $x(at - b)$.
2. Time scale $x(t)$ by a to obtain $x(at)$. Now time shift $x(at)$ by $\frac{b}{a}$ (i.e., replace t with $[t - \frac{b}{a}]$) to obtain $x(a[t - \frac{b}{a}]) = x(at - b)$.

For instance, the signal $x(2t - 6)$ can be obtained in two ways. First, delay $x(t)$ by 6 to obtain $x(t - 6)$ and then time compress this signal by factor 2 (replace t with $2t$) to obtain $x(2t - 6)$. Alternately, we first time compress $x(t)$ by factor 2 to obtain $x(2t)$; next, replace t with $t - 3$ to delay this signal and produce $x(2t - 6)$.

When a is negative, $x(at)$ involves time scaling as well as time reversal. The procedure, however, remains the same. Consider the case of a signal $x(-2t + 3)$ where $a = -2$. This signal can be generated by advancing the signal $x(t)$ by 3 seconds to obtain $x(t + 3)$. Next, compress and reverse this signal by replacing t with $-2t$ to obtain $x(-2t + 3)$. Alternately, we may compress and reverse $x(t)$ to obtain $x(-2t)$; next, replace t with $t - 3/2$ to delay this signal by $3/2$ and produce $x(-2[t - 3/2]) = x(-2t + 3)$.

▷ Drill 1.2 (Combined CT Operations)

Using the signal $x(t)$ shown in Fig. 1.6a, sketch the signal $y(t) = x(-3t - 4)$. Verify that $y(t)$ has a maximum value at $t = \frac{T_1+4}{-3}$.

△

1.3 CT Signal Models

In the area of signals and systems, the unit step, the unit gate, the unit triangle, the unit impulse, the exponential, and the interpolation functions are very useful. They not only serve as a basis for representing other signals, but their use benefits many aspects of our study of signals and systems. We shall briefly review descriptions of these models.

1.3.1 CT Unit Step Function $u(t)$

In much of our discussion, signals and processes begin at $t = 0$. Such signals can be conveniently described in terms of unit step function $u(t)$ shown in Fig. 1.8a. This function is defined by

$$u(t) = \begin{cases} 1 & t > 0 \\ \frac{1}{2} & t = 0 \\ 0 & t < 0 \end{cases}. \quad (1.1)$$

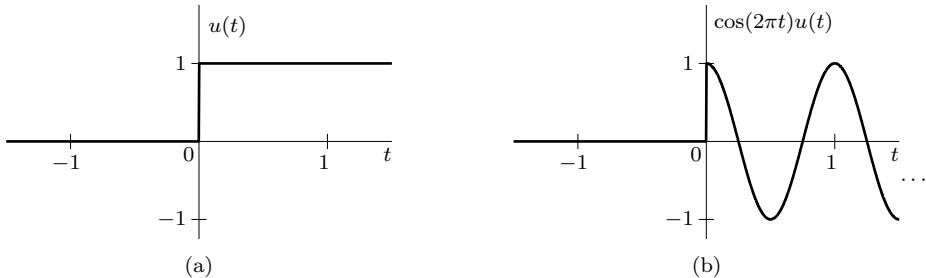


Figure 1.8: (a) CT unit step $u(t)$ and (b) $\cos(2\pi t)u(t)$.

If we want a signal to start at $t = 0$ and have a value of zero for $t < 0$, we only need to multiply the signal by $u(t)$. For instance, the signal $\cos(2\pi t)$ represents an everlasting sinusoid that starts at $t = -\infty$. The causal form of this sinusoid, illustrated in Fig. 1.8b, can be described as $\cos(2\pi t)u(t)$. The unit step function and its shifts also prove very useful in specifying functions with different mathematical descriptions over different intervals (piecewise functions).

A Meaningless Existence?

It is worth commenting that not everyone defines the point $u(0)$ as $1/2$. Some texts define $u(0)$ as 1, others define it as 0, and still others refuse to define it at all. While each definition has

its own advantages, $u(0) = 1/2$ is particularly appropriate from a theoretical signals and systems perspective. For real-world signals applications, however, it makes no practical difference how the point $u(0)$ is defined as long as the value is finite. A single point, $u(0)$ or otherwise, is just one among an uncountably infinite set of peers. Lost in the masses, any single, finite-valued point simply does not matter; its individual existence is meaningless.

Further, notice that since it is everlasting, a true unit step cannot be generated in practice. One might conclude, given that $u(t)$ is physically unrealizable and that individual points are inconsequential, that the whole of $u(t)$ is meaningless. This conclusion is false. *Collectively* the points of $u(t)$ are well behaved and dutifully carry out the desired function, which is greatly needed in the mathematics of signals and systems.

1.3.2 CT Unit Gate Function $\Pi(t)$

We define a unit gate function $\Pi(x)$ as a gate pulse of unit height and unit width, centered at the origin, as illustrated in Fig. 1.9a. Mathematically,[†]

$$\Pi(t) = \begin{cases} 1 & |t| < \frac{1}{2} \\ \frac{1}{2} & |t| = \frac{1}{2} \\ 0 & |t| > \frac{1}{2} \end{cases}. \quad (1.2)$$

The gate pulse in Fig. 1.9b is the unit gate pulse $\Pi(t)$ expanded by a factor τ and therefore can be expressed as $\Pi(t/\tau)$. Observe that τ , the denominator of the argument of $\Pi(t/\tau)$, indicates the width of the pulse.

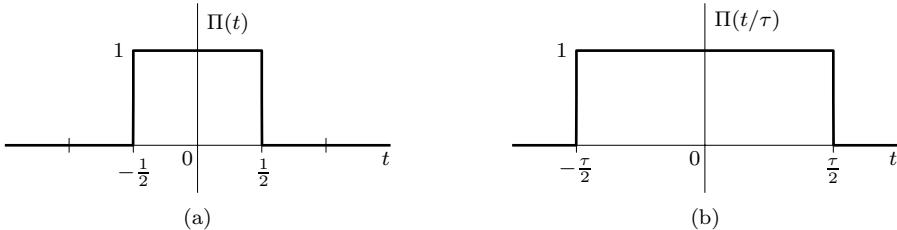


Figure 1.9: (a) CT unit gate $\Pi(t)$ and (b) $\Pi(t/\tau)$.

▷ Drill 1.3 (CT Unit Gate Representations)

The unit gate function $\Pi(t)$ can be represented in terms of time-shifted unit step functions. Determine the value b that ensures $u(t+b) - u(t-b)$ is equal to $\Pi(t)$. Next, represent $\Pi(t)$ using only time-shifted and *reflected* unit step functions.

▫

1.3.3 CT Unit Triangle Function $\Lambda(t)$

We define a unit triangle function $\Lambda(t)$ as a triangular pulse of unit height and unit width, centered at the origin, as shown in Fig. 1.10a. Mathematically,

$$\Lambda(t) = \begin{cases} 1 - 2|t| & |t| \leq \frac{1}{2} \\ 0 & |t| > \frac{1}{2} \end{cases}. \quad (1.3)$$

[†]At $|t| = \frac{1}{2}$, we desire $\Pi(t) = 0.5$ because the inverse Fourier transform of a discontinuous signal converges to the mean of the two values at either side of the discontinuity. As in the case of the unit step, the particular value assigned to a point of discontinuity, while perhaps theoretically convenient, has little practical significance.

The pulse in Fig. 1.10b is $\Lambda(t/\tau)$. Observe that here, as for the gate pulse, the denominator τ of the argument of $\Lambda(t/\tau)$ indicates the pulse width.

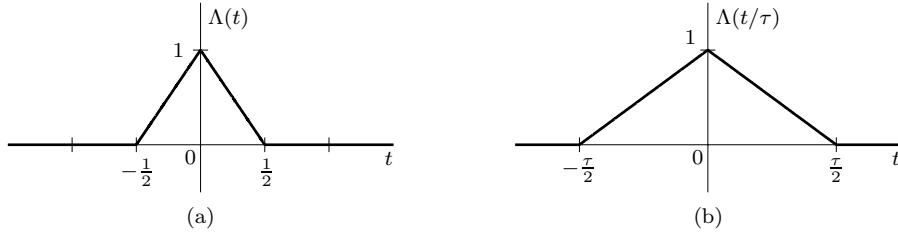


Figure 1.10: (a) CT unit triangle $\Lambda(t)$ and (b) $\Lambda(t/\tau)$.

1.3.4 CT Unit Impulse Function $\delta(t)$

The CT unit impulse function $\delta(t)$ is one of the most important functions in the study of signals and systems. Often called the Dirac delta function, $\delta(t)$ was first defined by P. A. M. Dirac as

$$\begin{aligned} \delta(t) &= 0 \text{ for } t \neq 0 \\ &\text{and} \\ &\int_{-\infty}^{\infty} \delta(t) dt = 1. \end{aligned} \quad (1.4)$$

We can visualize this impulse as a tall, narrow rectangular pulse of unit area, as illustrated in Fig. 1.11b. The width of this rectangular pulse is a very small value ϵ , and its height is a very large value $1/\epsilon$. In the limit $\epsilon \rightarrow 0$, this rectangular pulse has infinitesimally small width, infinitely large height, and unit area, thereby conforming exactly to the definition of $\delta(t)$ given in Eq. (1.4). Notice that $\delta(t) = 0$ everywhere except at $t = 0$, where it is undefined. For this reason a unit impulse is represented by the spear-like symbol in Fig. 1.11a.

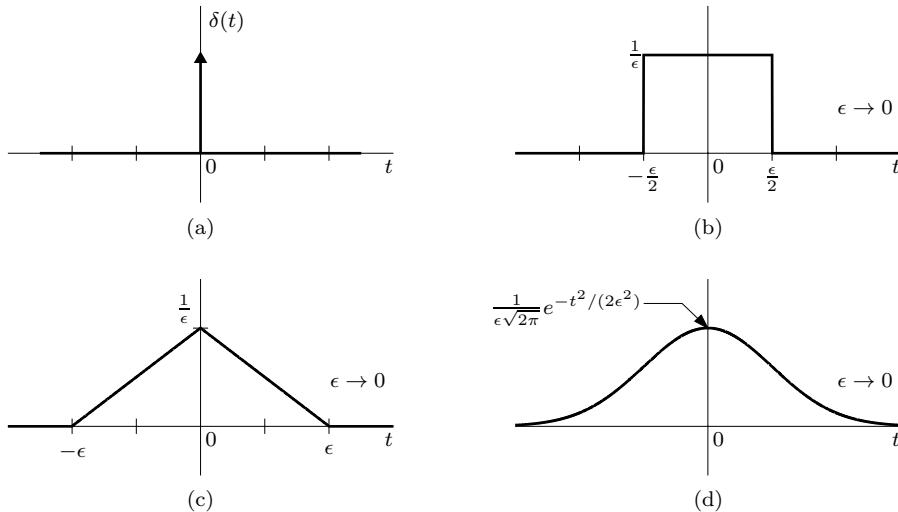


Figure 1.11: (a) CT unit impulse $\delta(t)$ and (b)–(d) visualizing $\delta(t)$ using various functions in the limit $\epsilon \rightarrow 0$.

Other pulses, such as the triangle pulse shown in Fig. 1.11c or the Gaussian pulse shown in Fig. 1.11d, may also be used to develop the unit impulse function. The important feature of $\delta(t)$

is not its shape but the fact that its effective duration (pulse width) approaches zero while its area remains at unity. Both the triangle pulse (Fig. 1.11c) and the Gaussian pulse (Fig. 1.11d) become taller and narrower as ϵ becomes smaller. In the limit as $\epsilon \rightarrow 0$, the pulse height $\rightarrow \infty$, and its width or duration $\rightarrow 0$. Yet, the area under each pulse is unity regardless of the value of ϵ .

From Eq. (1.4), it follows that the function $k\delta(t) = 0$ for all $t \neq 0$, and its area is k . Thus, $k\delta(t)$ is an impulse function whose area is k (in contrast to the unit impulse function, whose area is 1). Graphically, we represent $k\delta(t)$ by either scaling our representation of $\delta(t)$ by k or by placing a k next to the impulse.

Properties of the CT Impulse Function

Without going into the proofs, we shall enumerate properties of the unit impulse function. The proofs may be found in the literature (see, for example, [1]).

1. **Multiplication by a CT Impulse:** If a function $\phi(t)$ is continuous at $t = 0$, then

$$\phi(t)\delta(t) = \phi(0)\delta(t).$$

Generalizing, if $\phi(t)$ is continuous at $t = b$, then

$$\phi(t)\delta(t - b) = \phi(b)\delta(t - b). \quad (1.5)$$

2. **The Sampling Property:** If $\phi(t)$ is continuous at $t = 0$, then Eqs. (1.4) and (1.5) yield

$$\int_{-\infty}^{\infty} \phi(t)\delta(t) dt = \phi(0) \int_{-\infty}^{\infty} \delta(t) dt = \phi(0).$$

Similarly, if $\phi(t)$ is continuous at $t = b$, then

$$\int_{-\infty}^{\infty} \phi(t)\delta(t - b) dt = \phi(b). \quad (1.6)$$

Equation (1.6) states that *the area under the product of a function with a unit impulse is equal to the value of that function at the instant where the impulse is located*. This property is very important and useful and is known as the *sampling* or *sifting* property of the unit impulse.

3. **Relationships between $\delta(t)$ and $u(t)$:** Since the area of the impulse is concentrated at one point $t = 0$, it follows that the area under $\delta(t)$ from $-\infty$ to 0^- is zero, and the area is unity once we pass $t = 0$. The symmetry of $\delta(t)$, evident in Fig. 1.11, suggests the area is 1/2 at $t = 0$. Hence,

$$\int_{-\infty}^t \delta(\tau) d\tau = u(t) = \begin{cases} 0 & t < 0 \\ \frac{1}{2} & t = 0 \\ 1 & t > 0 \end{cases}. \quad (1.7)$$

From Eq. (1.7) it follows that

$$\delta(t) = \frac{d}{dt}u(t). \quad (1.8)$$

The Unit Impulse as a Generalized Function

The definition of the unit impulse function given in Eq. (1.4) is not rigorous mathematically, which leads to serious difficulties. First, the impulse function does not define a unique function. For example, it can be shown that $\delta(t) + \dot{\delta}(t)$ also satisfies Eq. (1.4). Moreover, $\delta(t)$ is not even a true function in the ordinary sense. An ordinary function is specified by its values for all time t . The

impulse function is zero everywhere except at $t = 0$, and at this only interesting part of its range it is undefined. These difficulties are resolved by defining the impulse as a generalized function rather than an ordinary function. A *generalized function* is defined by its effect on other functions instead of by its value at every instant of time.

In this approach, the impulse function is defined by the sampling property of Eq. (1.6). We say nothing about what the impulse function is or what it looks like. Instead, the impulse function is defined in terms of its effect on a test function $\phi(t)$. We define a unit impulse as a function for which the area under its product with a function $\phi(t)$ is equal to the value of the function $\phi(t)$ at the instant where the impulse is located. Thus, we can view the sampling property of Eq. (1.6) as a consequence of the classical (Dirac) definition of the unit impulse in Eq. (1.4) or *as the definition of the impulse function in the generalized function approach*.

A House Made of Bricks

In addition to serving as a definition of the unit impulse, Eq. (1.6) provides an insightful and useful way to view an arbitrary function.[†] Just as a house can be made of straw, sticks, or bricks, a function can be made of different building materials such as polynomials, sinusoids, and, in the case of Eq. (1.6), Dirac delta functions.

To begin, let us consider Fig. 1.12b, where an input $x(t)$ is shown as a sum of narrow rectangular strips. As shown in Fig. 1.12a, let us define a basic strip of unit height and width $\Delta\tau$ as $p(t) = \Pi(t/\Delta\tau)$. The rectangular pulse centered at $n\Delta\tau$ in Fig. 1.12b has a height $x(n\Delta\tau)$ and can be expressed as $x(n\Delta\tau)p(t - n\Delta\tau)$. As $\Delta\tau \rightarrow 0$ (and $n\Delta\tau \rightarrow \tau$), $x(t)$ is the sum of all such pulses. Hence,

$$x(t) = \lim_{\Delta\tau \rightarrow 0} \sum_{n=-\infty}^{\infty} x(n\Delta\tau)p(t - n\Delta\tau) = \lim_{\Delta\tau \rightarrow 0} \sum_{n=-\infty}^{\infty} x(n\Delta\tau) \left(\frac{p(t - n\Delta\tau)}{\Delta\tau} \right) \Delta\tau.$$

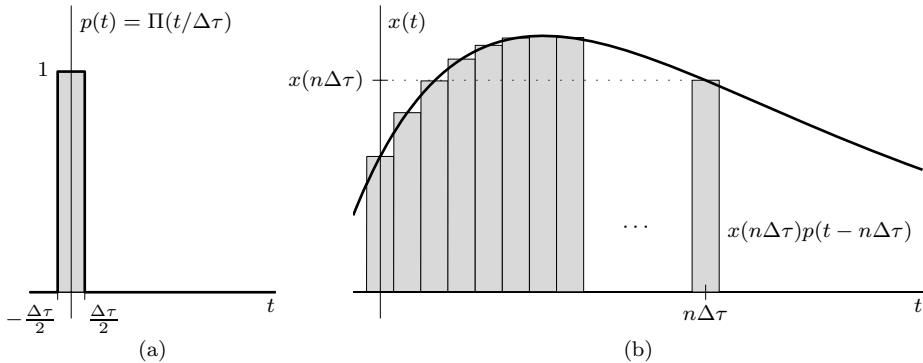


Figure 1.12: Signal representation in terms of impulse components.

Consistent with Fig. 1.11b, as $\Delta\tau \rightarrow 0$, $p(t - n\Delta\tau)/\Delta\tau \rightarrow \delta(t - n\Delta\tau)$. Therefore,

$$x(t) = \lim_{\Delta\tau \rightarrow 0} \sum_{n=-\infty}^{\infty} x(n\Delta\tau)\delta(t - n\Delta\tau) \Delta\tau = \underbrace{\sum_{-\infty}^{\infty} \underbrace{x(\tau)}_{\text{scaled}} \underbrace{\delta(t - \tau)}_{\substack{\text{shifted} \\ \text{impulses}}} d\tau}_{\text{sum}} \quad (1.9)$$

Equation (1.9), known as the sifting property, tells us that an arbitrary function $x(t)$ can be represented as a sum (integral) of scaled (by $x(\tau)$) and shifted (by τ) delta functions. Recognizing

[†]Actually, the function should be continuous for Eq. (1.6) to hold.

that $\delta(t - \tau) = \delta(\tau - t)$, we also see that Eq. (1.9) is obtained from Eq. (1.6) by substituting τ for t , t for b , and $x(\cdot)$ for $\phi(\cdot)$. As we shall see in Sec. 1.5, Eq. (1.9) is very much a house of bricks, more than able to withstand the big bad wolf of linear, time-invariant systems.

▷ Drill 1.4 (CT Unit Impulse Properties)

Show that

$$\begin{array}{ll} \text{(a)} & (t^3 + 2t^2 + 3t + 4)\delta(t) = 4\delta(t) \\ \text{(c)} & e^{-2t}\delta(t+1) = e^2\delta(t+1) \\ \text{(e)} & \int_{-\infty}^{\infty} \delta(\tau-2) \cos\left(\frac{\pi\tau}{4}\right) d\tau = 0 \end{array} \quad \begin{array}{ll} \text{(b)} & \delta(t) \sin\left(t^2 - \frac{\pi}{2}\right) = -\delta(t) \\ \text{(d)} & \int_{-\infty}^{\infty} \delta(\tau)e^{-j\omega\tau} d\tau = 1 \\ \text{(f)} & \int_{-\infty}^{\infty} e^{-2(t-\tau)}\delta(2-\tau) d\tau = e^{-2(t-2)} \end{array}$$

△

1.3.5 CT Exponential Function e^{st}

One of the most important functions in the area of signals and systems is the exponential signal e^{st} , where s is complex in general and given by

$$s = \sigma + j\omega.$$

Therefore,

$$e^{st} = e^{(\sigma+j\omega)t} = e^{\sigma t} e^{j\omega t} = e^{\sigma t} [\cos(\omega t) + j \sin(\omega t)]. \quad (1.10)$$

The final step in Eq. (1.10) is a substitution based on Euler's familiar formula,

$$e^{j\omega t} = \cos(\omega t) + j \sin(\omega t). \quad (1.11)$$

A comparison of Eq. (1.10) with Eq. (1.11) suggests that e^{st} is a generalization of the function $e^{j\omega t}$, where the frequency variable $j\omega$ is generalized to a complex variable $s = \sigma + j\omega$. For this reason we designate the variable s as the *complex frequency*.

For all $\omega \neq 0$, e^{st} is complex valued. Taking just the real portion of Eq. (1.10) yields

$$\operatorname{Re}\{e^{st}\} = e^{\sigma t} \cos(\omega t). \quad (1.12)$$

From Eqs. (1.10) and (1.12) it follows that the function e^{st} encompasses a large class of functions. The following functions are special cases of e^{st} :

1. a constant $k = ke^{0t}$ (where $s = 0 + j0$),
2. a monotonic exponential $e^{\sigma t}$ (where $s = \sigma + j0$),
3. a sinusoid $\cos(\omega t) = \operatorname{Re}\{e^{j\omega t}\}$ (where $s = 0 + j\omega$), and
4. an exponentially varying sinusoid $e^{\sigma t} \cos(\omega t) = \operatorname{Re}\{e^{(\sigma+j\omega)t}\}$ (where $s = \sigma + j\omega$).

Figure 1.13 shows these functions as well as the corresponding restrictions on the complex frequency variable s . The absolute value of the imaginary part of s is $|\omega|$ (the *radian frequency*), which indicates the frequency of oscillation of e^{st} ; the real part σ (the *neper frequency*) gives information about the rate of increase or decrease of the amplitude of e^{st} . For signals whose complex frequencies lie on the real axis (σ -axis, where $\omega = 0$), the frequency of oscillation is zero. Consequently these signals are constants ($\sigma = 0$), monotonically increasing exponentials ($\sigma > 0$), or monotonically decreasing exponentials ($\sigma < 0$). For signals whose frequencies lie on the imaginary axis (ω -axis, where $\sigma = 0$), $e^{\sigma t} = 1$. Therefore, these signals are conventional sinusoids with constant amplitude.

Figure 1.14 shows the demarcation of the s -plane into the *left half-plane* (LHP), which corresponds to exponentially decaying signals ($\sigma < 0$), and the *right half-plane* (RHP), which corresponds to exponentially growing signals ($\sigma > 0$). The imaginary axis separates the two regions and corresponds to signals of constant amplitude.

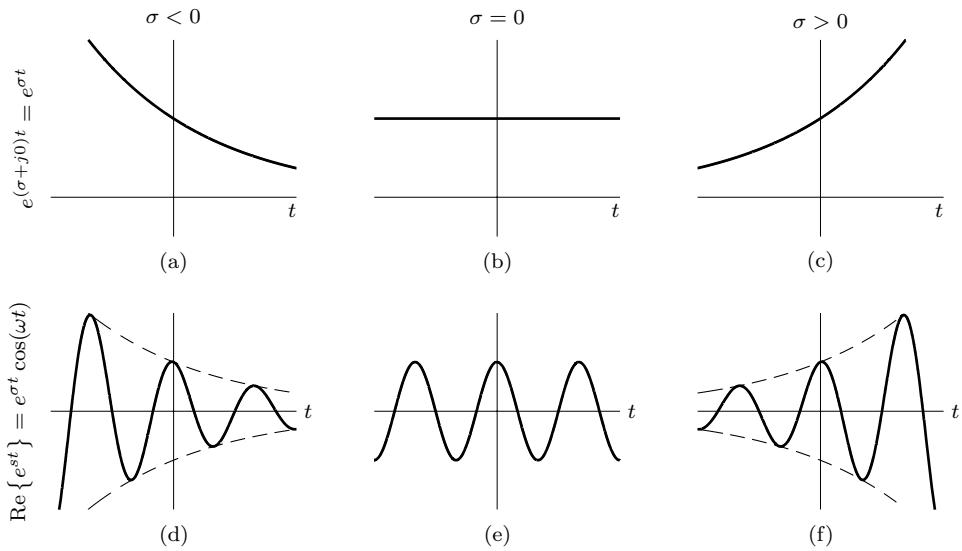


Figure 1.13: Various manifestations of $e^{(\sigma+j\omega)t} = e^{\sigma t}$ and $\operatorname{Re}\{e^{st}\} = e^{\sigma t} \cos(\omega t)$.

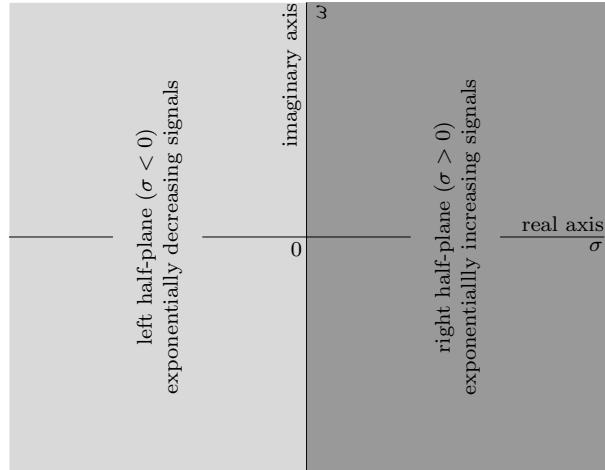


Figure 1.14: Complex frequency plane.

1.3.6 CT Interpolation Function $\operatorname{sinc}(t)$

The “sine over argument” function, or sinc function, plays an important role in signal processing.[†] It is also known as the *filtering* or *interpolating* function. We define

$$\operatorname{sinc}(t) = \frac{\sin(\pi t)}{\pi t}. \quad (1.13)$$

Inspection of Eq. (1.13) shows the following:

[†]sinc(t) is also denoted by Sa(t) in the literature. Some authors define sinc(t) as

$$\operatorname{sinc}(t) = \frac{\sin(t)}{t}.$$

1. The sinc function is symmetric about the vertical axis (an even function).
2. Except at $t = 0$ where it appears indeterminate, $\text{sinc}(t) = 0$ when $\sin(\pi t) = 0$. This means that $\text{sinc}(t) = 0$ for $t = \pm 1, \pm 2, \pm 3, \dots$
3. Using L'Hôpital's rule, we find $\text{sinc}(0) = 1$.
4. Since it is the product of the oscillating signal $\sin(\pi t)$ and the decreasing function $1/(\pi t)$, $\text{sinc}(t)$ exhibits sinusoidal oscillations with amplitude rapidly decreasing as $1/(\pi t)$.

Figure 1.15a shows $\text{sinc}(t)$. Observe that $\text{sinc}(t) = 0$ for integer values of t . Figure 1.15b shows $\text{sinc}(2t/3)$. The argument $2t/3 = 1$ when $t = 3/2$. Therefore, the first zero of this function for $t > 0$ occurs at $t = 3/2$.

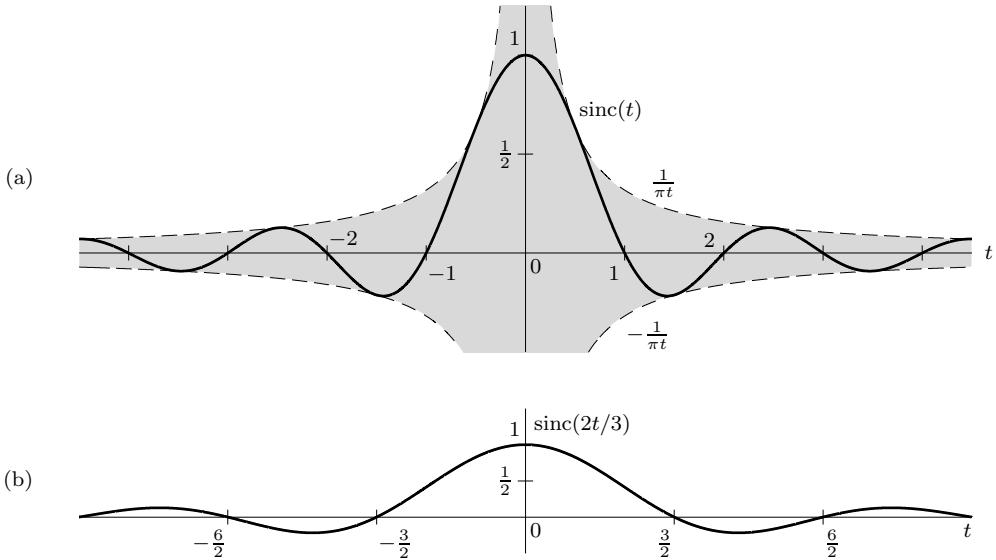


Figure 1.15: The sinc function.

▷ Example 1.1 (Plotting Combined Signals)

Defining $x(t) = e^{-t}u(t)$, accurately plot the signal $y(t) = x\left(\frac{-t+3}{3}\right) - \frac{3}{4}x(t-1)$ over the interval $(-1.5 \leq t \leq 4.5)$.

This problem involves several concepts, including exponential and unit step functions and operations on the independent variable t . The causal decaying exponential $x(t) = e^{-t}u(t)$ is itself easy to sketch by hand, and so too are the individual components $x\left(\frac{-t+3}{3}\right)$ and $-\frac{3}{4}x\left(\frac{t-1}{2}\right)$. The component $x\left(\frac{-t+3}{3}\right)$ is a left-sided signal with jump discontinuity at $\frac{-t+3}{3} = 0$ or $t = 3$. The component $-\frac{3}{4}x(t-1)$ is a right-sided signal with jump discontinuity at $t-1 = 0$ or $t = 1$. The combination $y(t) = x\left(\frac{-t+3}{3}\right) - \frac{3}{4}x(t-1)$, due to the overlap region between $(1 \leq t \leq 3)$, is difficult to accurately plot by hand. MATLAB, however, makes accurate plots easy to generate.

```

01 u = @(t) 1.0*(t>0)+0.5*(t==0);
02 x = @(t) exp(-t).*u(t); y = @(t) x((-t+3)/3)-3/4*x(t-1);
03 t = (-1.5:.0001:4.5); plot(t,y(t)); xlabel('t'); ylabel('y(t)');

```

In line 01, the unit step function is created as an anonymous function using relational operators. Anonymous functions provide a convenient way to quickly specify and manipulate simple functions. Furthermore, anonymous functions can be used in the definition of other anonymous functions, which is a convenience that we use in line 02 when defining $x(t)$ and $y(t)$. In line 03, an appropriate

time vector is created, and the plots are generated. It is important that the time vector t is created with sufficiently fine resolution to adequately represent the jump discontinuities present in $y(t)$. Figure 1.16 shows the result including the individual components $x\left(\frac{-t+3}{3}\right)$ and $-\frac{3}{4}x(t-1)$ added for clarity.

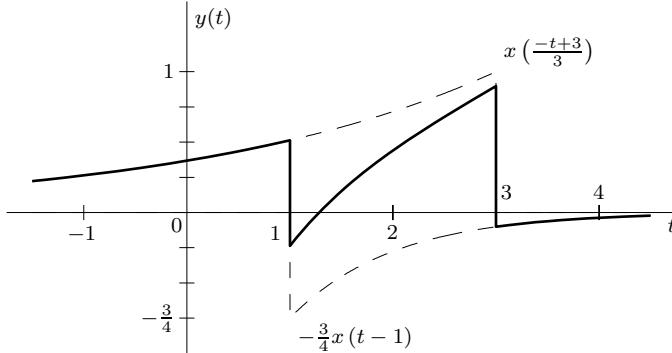


Figure 1.16: A plot of $y(t) = x\left(\frac{-t+3}{3}\right) - \frac{3}{4}x(t-1)$, where $x(t) = e^{-t}u(t)$.

Example 1.1 □

▷ **Drill 1.5 (Plotting CT Signal Models)**

Plot each of the following signals:

- | | | |
|---|--|---|
| (a) $x_a(t) = 2u(t+2) - u(3-3t)$ | (b) $x_b(t) = \Pi(\pi t)$ | (c) $x_c(t) = \Lambda(t/10)$ |
| (d) $x_d(t) = \operatorname{Re} \left\{ e^{(1-j2\pi)t} \right\} u(1-t)$ | (e) $x_e(t) = \operatorname{sinc} \left(\frac{2t}{\pi} \right)$ | (f) $x_f(t) = \operatorname{sinc}(t)\Pi(t/4)$ |

□

1.4 CT Signal Classifications

There are many possible signal classifications that are useful to better understand and properly analyze a signal. In addition to the continuous-time/discrete-time and the analog/digital signal classifications already discussed, we will also investigate the following classifications, which are suitable for the scope of this book:

1. causal, noncausal, and anti-causal signals,
2. real and imaginary signals,
3. even and odd signals,
4. periodic and aperiodic signals,
5. energy and power signals, and
6. deterministic and probabilistic signals.

1.4.1 Causal, Noncausal, and Anti-Causal CT Signals

A *causal* signal $x(t)$ extends to the right, beginning no earlier than $t = 0$. Mathematically, $x(t)$ is causal if

$$x(t) = 0 \quad \text{for } t < 0. \quad (1.14)$$

The signals shown in Fig. 1.8 are causal. Any signal that is not causal is said to be *noncausal*. Examples of noncausal signals are shown in Figs. 1.6 and 1.7. An *anti-causal* signal $x(t)$ extends to the left of $t = 0$. Mathematically, $x(t)$ is anti-causal if

$$x(t) = 0 \quad \text{for } t \geq 0. \quad (1.15)$$

Notice that any signal $x(t)$ can be decomposed into a causal component plus an anti-causal component.

A *right-sided* signal extends to the right, beginning at some point T_1 . In other words, $x(t)$ is right-sided if $x(t) = 0$ for $t < T_1$. The signals shown in Fig. 1.5 are examples of right-sided signals. Similarly, a *left-sided* signal extends to the left of some point T_1 . Mathematically, $x(t)$ is left-sided if $x(t) = 0$ for $t \geq T_1$. If we time invert a right-sided signal, then we obtain a left-sided signal. Conversely, the time reversal of a left-sided signal produces a right-sided signal. A causal signal is a right-sided signal with $T_1 \geq 0$, and an anti-causal signal is a left-sided signal with $T_1 \leq 0$. Notice, however, that right-sided signals are not necessarily causal, and left-sided signals are not necessarily anti-causal. Signals that stretch indefinitely in both directions are termed *two-sided* or *everlasting* signals. The signals in Figs. 1.13 and 1.15 provide examples of two-sided signals.

Comment

We postulate and study everlasting signals despite the fact that, for obvious reasons, a true everlasting signal cannot be generated in practice. Still, as we show later, many two-sided signal models, such as everlasting sinusoids, *do* serve a very useful purpose in the study of signals and systems.

1.4.2 Real and Imaginary CT Signals

A signal $x(t)$ is *real* if, for all time, it equals its own complex conjugate,

$$x(t) = x^*(t). \quad (1.16)$$

A signal $x(t)$ is *imaginary* if, for all time, it equals the negative of its own complex conjugate,

$$x(t) = -x^*(t). \quad (1.17)$$

The real portion of a complex signal $x(t)$ is found by averaging the signal with its complex conjugate,

$$\text{Re}\{x(t)\} = \frac{x(t) + x^*(t)}{2}. \quad (1.18)$$

The imaginary portion of a complex signal $x(t)$ is found in a similar manner,

$$\text{Im}\{x(t)\} = \frac{x(t) - x^*(t)}{2j}. \quad (1.19)$$

Notice that $\text{Im}\{x(t)\}$ is a *real* signal. Further, notice that Eq. (1.18) obeys Eq. (1.16) and j times Eq. (1.19) obeys Eq. (1.17), as expected.

Adding Eq. (1.18) and j times Eq. (1.19), we see that any complex signal $x(t)$ can be decomposed into a real portion plus (j times) an imaginary portion,

$$\text{Re}\{x(t)\} + j\text{Im}\{x(t)\} = \frac{x(t) + x^*(t)}{2} + j\frac{x(t) - x^*(t)}{2j} = x(t)$$

or just

$$x(t) = \text{Re}\{x(t)\} + j\text{Im}\{x(t)\}. \quad (1.20)$$

This representation is the familiar rectangular form.

▷ **Drill 1.6 (Variations of Euler's Formula)**

Using Euler's formula $e^{jt} = \cos(t) + j \sin(t)$ and Eqs. (1.18) and (1.19), show that

$$(a) \quad \cos(t) = \frac{e^{jt} + e^{-jt}}{2} \quad (b) \quad \sin(t) = \frac{e^{jt} - e^{-jt}}{2j}$$

△

Real Comments about the Imaginary

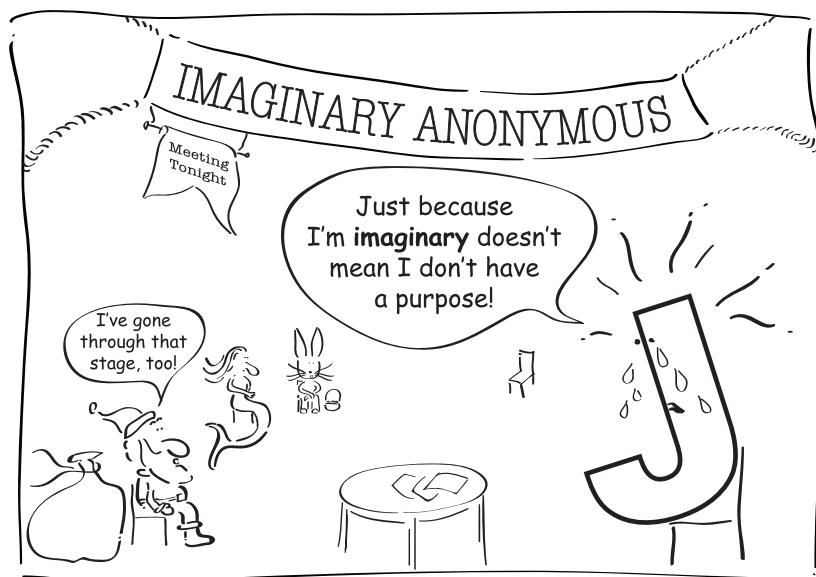
There is an important and subtle distinction between an *imaginary signal* and the *imaginary portion* of a signal: an imaginary signal is a complex signal whose real portion is zero and is thus represented as j times a real quantity, while the imaginary portion of a signal is real and has no j present. One way to emphasize this difference is to write Eq. (1.17) in an alternate but completely equivalent way. A signal $x(t)$ is imaginary if, for all time,

$$x(t) = j \operatorname{Im}\{x(t)\}.$$

From this expression, it is clear that an imaginary signal is never equal to its imaginary portion but rather is equal to j times its imaginary portion. We can view the j in Eq. (1.20) as simply a mechanism to keep the two real-valued components of a complex signal separate.

Viewing complex signals as pairs of separate real quantities offers tangible benefits. Such a perspective makes clear that *complex signal processing*, which is just signal processing on complex-valued signals, *is easily accomplished in the real world by processing pairs of real signals*. There is a frequent misconception that complex signal processing is not possible with analog systems. Again this is simply untrue. Complex signal processing is readily implemented with traditional analog electronics by simply utilizing *dual* signal paths.

It is worthwhile to comment that the historical choices of the terms “complex” and “imaginary” are quite unfortunate, particularly from a signal processing perspective. The terms are prejudicial; “complex” suggests difficult, and “imaginary” suggests something that cannot be realized. Neither case is true. More often than not, complex signal processing is more simple than the alternative, and complex signals, as we have just seen, are easily realized in the analog world.



What's in a name?

▷ **Drill 1.7 (The Imaginary Part Is Real)**

Determine the imaginary portions of (a) $1 + j$ and (b) j . Hint: Neither answer is j !

△

1.4.3 Even and Odd CT Signals

A signal $x(t)$ is *even* if, for all time, it equals its own reflection,

$$x(t) = x(-t). \quad (1.21)$$

A signal $x(t)$ is *odd* if, for all time, it equals the negative of its own reflection,

$$x(t) = -x(-t). \quad (1.22)$$

As shown in Figs. 1.17a and 1.17b, respectively, an even signal is symmetrical about the vertical axis while an odd signal is antisymmetrical about the vertical axis.

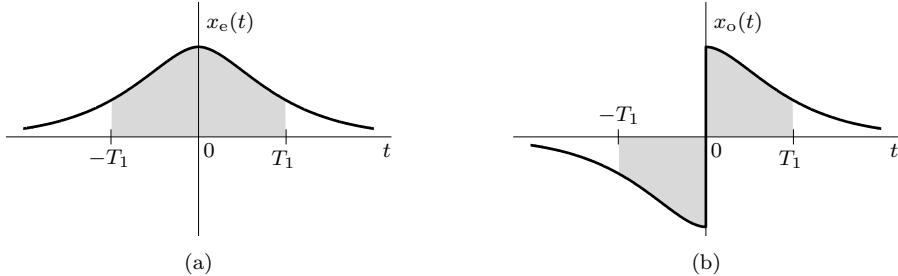


Figure 1.17: Even and odd symmetries: (a) an even signal $x_e(t)$ and (b) an odd signal $x_o(t)$.

The even portion of a signal $x(t)$ is found by averaging the signal with its reflection,

$$x_e(t) = \frac{x(t) + x(-t)}{2}. \quad (1.23)$$

As required by Eq. (1.21) for evenness, notice that $x_e(t) = x_e(-t)$. The odd portion of a signal $x(t)$ is found in a similar manner,

$$x_o(t) = \frac{x(t) - x(-t)}{2}. \quad (1.24)$$

As required by Eq. (1.22) for oddness, $x_o(t) = -x_o(-t)$.

Adding Eqs. (1.23) and (1.24), we see that any signal $x(t)$ can be decomposed into an even portion plus an odd portion,

$$x_e(t) + x_o(t) = \frac{x(t) + x(-t)}{2} + \frac{x(t) - x(-t)}{2} = x(t)$$

or just

$$x(t) = x_e(t) + x_o(t). \quad (1.25)$$

Notice that Eqs. (1.23), (1.24), and (1.25) are remarkably similar to Eqs. (1.18), (1.19), and (1.20), respectively.

Because $x_e(t)$ is symmetrical about the vertical axis, it follows from Fig. 1.17a that

$$\int_{-T_1}^{T_1} x_e(t) dt = 2 \int_0^{T_1} x_e(t) dt.$$

It is also clear from Fig. 1.17b that

$$\int_{-T_1}^{T_1} x_o(t) dt = 0.$$

These results can also be proved formally by using the definitions in Eqs. (1.21) and (1.22).

▷ **Example 1.2 (Finding the Even and Odd Portions of a Function)**

Determine and plot the even and odd components of $x(t) = e^{-at}u(t)$, where a is real and > 0 .

Using Eq. (1.23), we compute the even portion of $x(t)$ to be

$$x_e(t) = \frac{x(t) + x(-t)}{2} = \frac{e^{-at}u(t) + e^{at}u(-t)}{2}.$$

Similarly, using Eq. (1.24), the odd portion of $x(t)$ is

$$x_o(t) = \frac{x(t) - x(-t)}{2} = \frac{e^{-at}u(t) - e^{at}u(-t)}{2}.$$

Figures 1.18a, 1.18b, and 1.18c show the resulting plots of $x(t)$, $x_e(t)$, and $x_o(t)$, respectively.

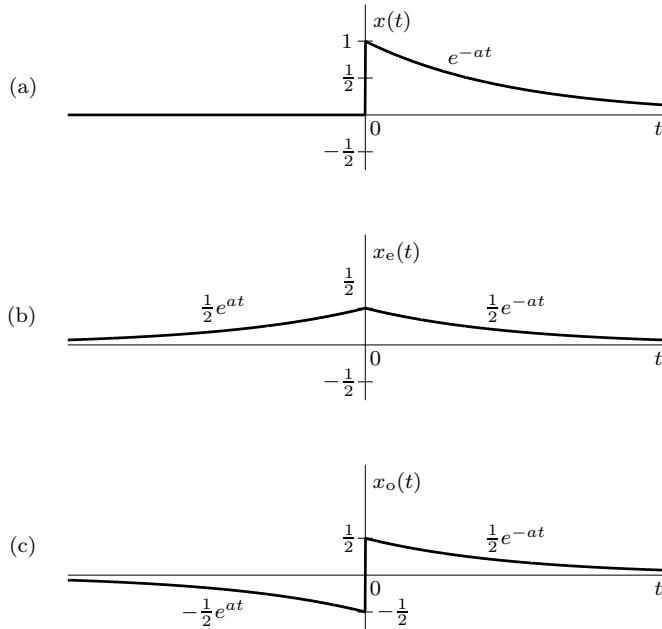


Figure 1.18: Finding the even and odd components of $x(t) = e^{-at}u(t)$.

Setting $a = 1$ for convenience, these plots are easily generated using MATLAB.

```

01 t = linspace(-2,2,4001); x = @t) exp(-t).* (t>0) + 0.5*(t==0);
02 xe = (x(t)+x(-t))/2; xo = (x(t)-x(-t))/2;
03 subplot(311); plot(t,x(t)); xlabel('t'); ylabel('x(t)');
04 subplot(312); plot(t,xe); xlabel('t'); ylabel('x_e(t)');
05 subplot(313); plot(t,xo); xlabel('t'); ylabel('x_o(t)');

```

Example 1.2 ◀

▷ **Drill 1.8 (Even and Odd Decompositions)**

Show that the even and the odd components of $x(t) = e^{j\omega t}$ are $x_e(t) = \cos(\omega t)$ and $x_o(t) = j \sin(\omega t)$, respectively.

△

A Different Prescription for Complex CT Signals

While a complex signal can be viewed using an even and odd decomposition, doing so is a bit like a far-sighted man wearing glasses intended for the near-sighted. The poor prescription blurs rather than sharpens the view. Glasses of a different type are required. Rather than an even and odd decomposition, the preferred prescription for complex signals is generally a conjugate-symmetric and conjugate-antisymmetric decomposition.

A signal $x(t)$ is *conjugate symmetric*, or *Hermitian*, if

$$x(t) = x^*(-t). \quad (1.26)$$

A conjugate-symmetric signal is even in its real portion and odd in its imaginary portion. Thus, a signal that is both conjugate symmetric and real is also an even signal.

A signal $x(t)$ is *conjugate antisymmetric*, or *skew Hermitian*, if

$$x(t) = -x^*(-t). \quad (1.27)$$

A conjugate-antisymmetric signal is odd in its real portion and even in its imaginary portion. Thus, a signal that is both conjugate antisymmetric and real is also an odd signal.

The conjugate-symmetric portion of a signal $x(t)$ is given by

$$x_{cs}(t) = \frac{x(t) + x^*(-t)}{2}. \quad (1.28)$$

As required by Eq. (1.26), we find that $x_{cs}(t) = x_{cs}^*(-t)$. The conjugate-antisymmetric portion of a signal $x(t)$ is given by

$$x_{ca}(t) = \frac{x(t) - x^*(-t)}{2}. \quad (1.29)$$

As required by Eq. (1.27), notice that $x_{ca}(t) = -x_{ca}^*(-t)$.

Adding Eqs. (1.28) and (1.29), we see that any signal $x(t)$ can be decomposed into a conjugate-symmetric portion plus a conjugate-antisymmetric portion,

$$x_{cs}(t) + x_{ca}(t) = \frac{x(t) + x^*(-t)}{2} + \frac{x(t) - x^*(-t)}{2} = x(t)$$

or just

$$x(t) = x_{cs}(t) + x_{ca}(t). \quad (1.30)$$

▷ **Drill 1.9 (Conjugate-Symmetric and Conjugate-Antisymmetric Decompositions)**

Determine the conjugate-symmetric and conjugate-antisymmetric portions of the following signals:

$$(a) \quad x_a(t) = e^{jt} \quad (b) \quad x_b(t) = je^{jt} \quad (c) \quad x_c(t) = \sqrt{2}e^{j(t+\pi/4)}$$

△

1.4.4 Periodic and Aperiodic CT Signals

A CT signal $x(t)$ is said to be *T-periodic* if, for some positive constant T ,

$$x(t) = x(t - T) \quad \text{for all } t. \quad (1.31)$$

The *smallest* value of T that satisfies the periodicity condition of Eq. (1.31) is the *fundamental period* T_0 of $x(t)$. The signal in Fig. 1.19 is a T_0 -periodic signal. A signal is *aperiodic* if it is not

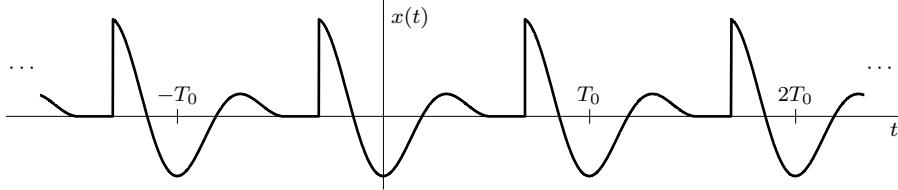


Figure 1.19: A T_0 -periodic signal.

periodic. The signals in Figs. 1.5, 1.6, and 1.8 are examples of aperiodic waveforms.

By definition, a periodic signal $x(t)$ remains unchanged when time shifted by one period. For this reason *a periodic signal, by definition, must start at $t = -\infty$ and continue forever*; if it starts or ends at some finite instant, say $t = 0$, then the time-shifted signal $x(t - T)$ will start or end at $t = T$, and Eq. (1.31) cannot hold. Clearly, a periodic signal is an everlasting signal; not all everlasting signals, however, are periodic, as Fig. 1.5 demonstrates.

Periodic Signal Generation by Periodic Replication of One Cycle

Another important property of a periodic signal $x(t)$ is that $x(t)$ can be generated by *periodic replication* of any segment of $x(t)$ of duration T_0 (the period). As a result, we can generate $x(t)$ from any segment of $x(t)$ with a duration of one period by placing this segment and the reproduction thereof end to end ad infinitum on either side. Figure 1.20 shows a periodic signal $x(t)$ with period $T_0 = 6$ generated in two ways. In Fig. 1.20a, the segment $(-1 \leq t < 5)$ is repeated forever in either direction, resulting in signal $x(t)$. Figure 1.20b achieves the same end result by repeating the segment $(0 \leq t < 6)$. This construction is possible with any segment of $x(t)$ starting at any instant as long as the segment duration is one period.

1.4.5 CT Energy and Power Signals

The size of any entity is a number that indicates the largeness or strength of that entity. Generally speaking, signal amplitude varies with time. How can a signal that exists over time with varying amplitude be measured by a single number that will indicate the signal size or signal strength? Such a measure must consider not only the signal amplitude but also its duration.

Signal Energy

By defining signal size as the area under $x^2(t)$, which is always positive for real $x(t)$, both signal amplitude and duration are properly acknowledged. We call this measure the *signal energy* E_x , defined (for a real signal) as

$$E_x = \int_{-\infty}^{\infty} x^2(t) dt.$$

This definition can be generalized to accommodate complex-valued signals as

$$E_x = \int_{-\infty}^{\infty} |x(t)|^2 dt = \int_{-\infty}^{\infty} x(t)x^*(t) dt. \quad (1.32)$$

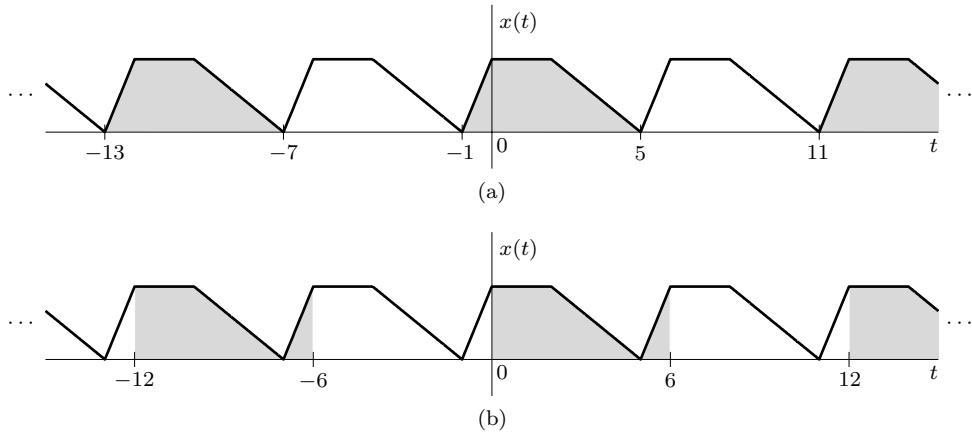


Figure 1.20: Generation of the ($T_0 = 6$)-periodic signal $x(t)$ by periodic replication using (a) the segment $(-1 \leq t < 5)$ and (b) the segment $(0 \leq t < 6)$.

There are also other possible measures of signal size, such as the area under $|x(t)|$. The energy measure, however, is not only more tractable mathematically but is also more meaningful (as shown later) in the sense that it is indicative of the energy that can be extracted from the signal.

Signal energy must be finite for it to be a meaningful measure of the signal size. A necessary condition for the energy to be finite is that the signal amplitude $\rightarrow 0$ as $|t| \rightarrow \infty$ (Fig. 1.21a). Otherwise the integral in Eq. (1.32) will not converge. A signal with finite energy is classified as an *energy signal*.

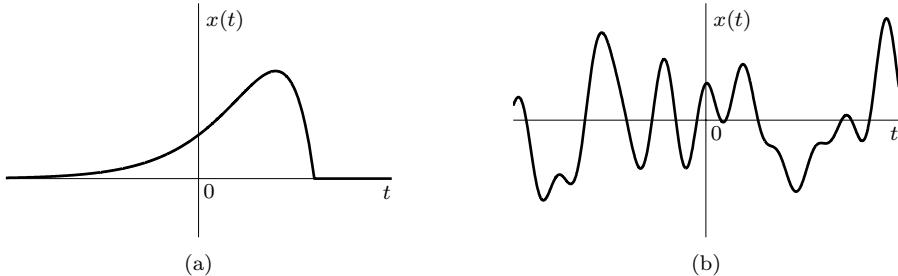


Figure 1.21: Examples of (a) finite energy and (b) finite power signals.

▷ **Example 1.3 (Computing Signal Energy)**

Compute the energy of

$$(a) \quad x_a(t) = 2\Pi(t/2) \quad (b) \quad x_b(t) = \text{sinc}(t)$$

(a)

In this particular case, direct integration is simple. Using Eq. (1.32), we find that

$$E_{x_a} = \int_{-1}^1 (2)^2 dt = 4t \Big|_{-1}^1 = 8.$$

(b)

In this case, the direct integration of $\text{sinc}^2(t)$ is quite difficult. Although Parseval's theorem, to be

discussed in Sec. 1.9.8, makes it easy to determine that the energy is $E_{x_b} = 1$, it is instructive to try and obtain this answer by estimating the integral in Eq. (1.32). We begin by plotting $x_b^2(t) = \text{sinc}^2(t)$ since energy is simply the area under this curve. As shown in Fig. 1.22, $\text{sinc}^2(t)$ decays very quickly and is close to zero for $|t| > 5$.

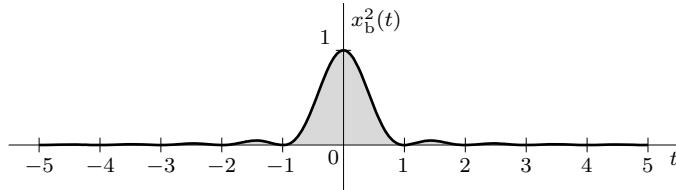


Figure 1.22: Integrating $x_b^2(t) = \text{sinc}^2(t)$ to determine energy.

In the spirit of Fig. 1.12, we can estimate Eq. (1.32) using a rectangular approximation

$$E_{x_b} \approx \sum_{n=-N}^N x_b^2(n\Delta t)\Delta t.$$

Here, Δt must be chosen sufficiently small to capture the detail of x_b^2 , and N must be chosen sufficiently large so that the interval $(-N\Delta t \leq t \leq N\Delta t)$ includes most of the energy. Using our interval of $(-5 \leq t \leq 5)$, MATLAB computes E_{x_b} with about 2% error.

```
01 Deltat = .001; N = 5/Deltat; n = -N:N; sum((sinc(n*Deltat)).^2*Deltat)
ans = 0.9798
```

The same result is also obtained by using MATLAB's built-in numerical integration function `quad`, which is more robust than the simple rectangular approximation.

```
02 x_b = @(t) (sinc(t)).^2; quad(x_b,-5,5)
ans = 0.9798
```

Wider intervals, such as $(-100 \leq t \leq 100)$, improve the approximation.

```
03 x_b = @(t) (sinc(t)).^2; quad(x_b,-100,100)
ans = 0.9986
```

Example 1.3 \triangleleft

▷ Drill 1.10 (Signal Energy)

Sketch the signal $x(t) = \sin(2\pi t)\Pi(t - 1/2)$, and show that its energy is $E_x = 1/2$.

\triangleleft

Signal Power

In some cases, for instance when the amplitude of $x(t)$ does not $\rightarrow 0$ as $|t| \rightarrow \infty$ (Fig. 1.21b), signal energy is infinite. A more meaningful measure of the signal size in such a case is time-normalized energy, if it exists. This measure is called the *power* of the signal. For a real signal $x(t)$, we define its power P_x as

$$P_x = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} x^2(t) dt.$$

Again, we can generalize this definition to accommodate complex signals as

$$P_x = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |x(t)|^2 dt = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t)x^*(t) dt. \quad (1.33)$$

Generally, the mean of an entity averaged over a large time interval approaching infinity exists if the entity is either periodic or exhibits statistical regularity with time. If such conditions are not satisfied, the average may not exist. For instance, a causal ramp signal $x(t) = tu(t)$ increases indefinitely as $t \rightarrow \infty$, and neither the energy nor the power exists for this signal. However, the unit step function, which is not periodic nor has statistical regularity, does have a finite power. A signal with finite and nonzero power is termed a *power signal*.

Observe that the signal power P_x is the time average (mean) of the signal amplitude squared, that is, the *mean square* value of $x(t)$. Indeed, the square root of P_x is the familiar *rms* (root mean square) value of $x(t)$. Thought of another way, power is just energy normalized by an infinitely large time window. Thus, a signal with finite energy has zero power, and a signal with finite power has infinite energy. A signal cannot both be an energy and a power signal.

When $x(t)$ is periodic, $|x(t)|^2$ is also periodic. Hence, the power of $x(t)$ can be computed from Eq. (1.33) by averaging $|x(t)|^2$ over one period T_0 ,

$$P_x = \frac{1}{T_0} \int_{T_0} |x(t)|^2 dt = \frac{1}{T_0} \int_{T_0} x(t)x^*(t) dt. \quad (1.34)$$

The notation \int_{T_0} represents integration over an interval of T_0 seconds starting at any instant.

▷ Drill 1.11 (Signal Power)

Show that

- (a) $x_a(t) = C$ has $P_{x_a} = |C|^2$
- (b) $x_b(t) = u(t)$ has $P_{x_b} = 0.5$
- (c) $x_c(t) = Ce^{j\omega_0 t}$ has $P_{x_c} = |C|^2$
- (d) $x_d(t) = C \cos(\omega_0 t + \theta)$ has $P_{x_d} = \frac{|C|^2}{2}$
- (e) $x_e(t) = C_1 \cos(\omega_1 t + \theta_1) + C_2 \cos(\omega_2 t + \theta_2)$ has $P_{x_e} = \frac{|C_1|^2 + |C_2|^2}{2}$ if $0 \neq \omega_1 \neq \omega_2 \neq 0$
and has $P_{x_e} = \frac{1}{2} (|C_1|^2 + (C_1 C_2^* + C_1^* C_2) \cos(\theta_1 - \theta_2) + |C_2|^2)$ if $\omega_1 = \omega_2 \neq 0$

△

▷ Drill 1.12 (Neither Energy nor Power)

Show that an everlasting exponential $x(t) = e^{-at}$ is neither an energy nor a power signal for any real and nonzero value of a .

△

Comments on Energy and Power

The signal energy as defined in Eq. (1.32) does not indicate signal energy in the conventional sense because signal energy depends not only on the signal but also on the load. It can, however, be interpreted as the energy dissipated in a normalized load if a voltage $x(t)$ were to be applied across a 1-ohm resistor (or if a current $x(t)$ were to be passed through a 1-ohm resistor). The measure of “energy” is, therefore, indicative of the energy capability of the signal and not the actual energy. For this reason the concepts of conservation of energy should not be applied to this “signal energy.” Parallel observation applies to “signal power” defined in Eq. (1.33).

The units of energy and power, as defined in Eqs. (1.32) and (1.33), depend on the nature of the signal $x(t)$. If $x(t)$ is a voltage signal, its energy E_x has units of $V^2 s$ (volts squared-seconds), and its

power P_x has units of V^2 (volts squared). If $x(t)$ is a current signal, these units are A^2 s (amperes squared-seconds) and A^2 (amperes squared), respectively.

More fundamentally, signal energy and power are but convenient indicators of signal size, which prove useful in many applications. For instance, if we approximate a signal $x(t)$ by another signal $\hat{x}(t)$, the error in the approximation is $e(t) = x(t) - \hat{x}(t)$. The energy (or power) of $e(t)$ is a convenient indicator of the goodness of the approximation. It provides us with a quantitative measure to determine the closeness of the approximation. In communication systems, message signals are corrupted by unwanted signals (noise) during transmission over a channel. The quality of the received signal is judged by the relative sizes of the desired signal and the unwanted noise. In this case, the ratio of the message signal and noise signal powers (signal to noise power ratio) is a good indication of the received signal quality.

All practical signals have finite energies and are therefore energy signals. A power signal must necessarily have infinite duration; otherwise its power will not approach a nonzero limit. Clearly, it is impossible to generate a true power signal in practice because such a signal has infinite duration and infinite energy. Still, as in the case of everlasting signals, power signal models serve a useful purpose in the study of signals and systems.

1.4.6 Deterministic and Probabilistic Signals

A signal whose physical description is known completely, either in a mathematical form or a graphical form, is a *deterministic signal*. A signal whose values cannot be predicted precisely but are known only in terms of probabilistic description, such as mean value, mean square value, and so on, is a *probabilistic* or *random signal*. In this book we shall deal almost exclusively with deterministic signals. Random signals are beyond the general scope of this study.

1.5 CT Systems and Properties

Signals are processed by *systems*, which may modify them or extract additional information from them. Thus, a system is an entity that *processes* a set of signals (*inputs*) to yield another set of signals (*outputs*). A system may be made up of physical components, as in electrical, mechanical, or hydraulic systems (hardware realization), or it may be an algorithm that computes an output from an input signal (software realization). A system that operates on continuous-time signals is naturally called a continuous-time system. By using the *operator* H to represent the system function, the system output $y(t)$ is conveniently represented in terms of the input $x(t)$ as

$$y(t) = H \{x(t)\}.$$

Using a system operator H to represent an integrator, for example, implies that $y(t) = H \{x(t)\} = \int_{-\infty}^t x(\tau) d\tau$. Figure 1.23 illustrates the general concept.

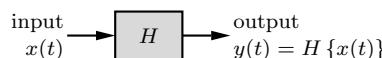


Figure 1.23: Block diagram for continuous-time system H .

It is quite difficult to characterize the behavior H of a CT system in a completely general sense. By restricting our attention to linear, time-invariant systems, however, an elegant and tractable mathematical characterization is possible. In addition to linearity, time invariance, and their application to characterize linear time-invariant continuous (LTIC) systems, the important system properties of causality and stability are also reviewed.

1.5.1 Linearity

We shall consider the linearity concept first. A system whose output is proportional to its input is an *example* of a linear system. But linearity implies more than this; it also implies the *additivity property*, meaning that if several causes (inputs) are acting on a system, then the total effect on the system due to all these inputs can be determined by considering each input separately while assuming all the other inputs to be zero. The total output is then the sum of all the component outputs. This property may be expressed as follows: for a linear system, if input $x_1(t)$ acting alone produces output $y_1(t) = H\{x_1(t)\}$, and if another input $x_2(t)$, also acting alone, produces output $y_2(t) = H\{x_2(t)\}$, then, with both inputs acting on the system, the combined output will be $H\{x_1(t) + x_2(t)\} = y_1(t) + y_2(t)$.

In addition to additivity, a linear system must satisfy the *homogeneity or scaling property*, which states that for an arbitrary number c , if a cause is increased c -fold, then the effect also increases c -fold. That is, if $H\{x(t)\} = y(t)$, then $H\{cx(t)\} = cy(t)$ for all real or complex c .

Thus, linearity requires two properties: homogeneity (scaling) and additivity.[†] Both these properties can be combined into the *superposition property*, which states that if

$$y_1(t) = H\{x_1(t)\} \quad \text{and} \quad y_2(t) = H\{x_2(t)\},$$

then, for all possible constants c_1 and c_2 ,

$$H\{c_1x_1(t) + c_2x_2(t)\} = c_1\underbrace{H\{x_1(t)\}}_{y_1(t)} + c_2\underbrace{H\{x_2(t)\}}_{y_2(t)}. \quad (1.35)$$

As Fig. 1.24 helps to illustrate, the linearity property is satisfied if a sum of scaled inputs applied to a system (Fig. 1.24a) produces the same result as summing and scaling the individual outputs of each input (Fig. 1.24b). Thought of another way, *linearity implies that a system operator H commutes with the operations of summing and scaling*. With a linear system, it does not matter whether summing and scaling precede the system or vice versa. This apparently trivial observation has profound implications for analyzing linear systems, as shown in Sec. 1.5.3.

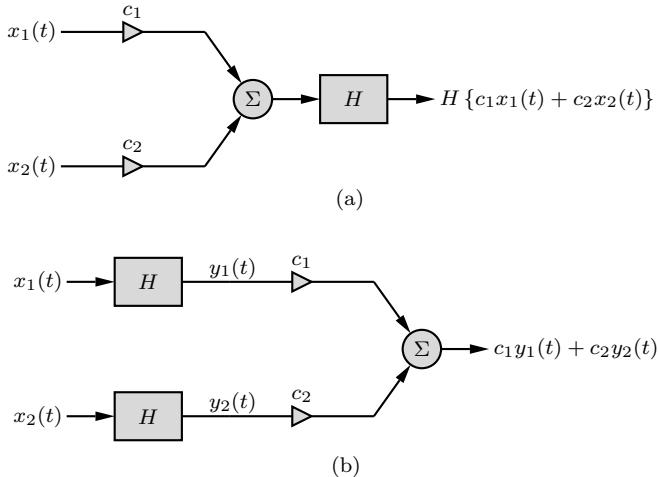


Figure 1.24: Visualizing linearity: (a) summing and scaling precede the system and (b) the system precedes summing and scaling.

[†] A linear system must also satisfy the additional condition of *smoothness*, where small changes in the system's inputs must result in small changes in its outputs [3].

▷ **Drill 1.13 (Additivity Does Not Imply Homogeneity)**

It may appear that additivity implies homogeneity, but this is not the case. Show that a system with the input $x(t)$ and the output $y(t) = \operatorname{Re}\{x(t)\}$ satisfies the additivity property but violates the homogeneity property. Hence, such a system is not linear. Hint: Show that Eq. (1.35) is not satisfied when c is complex.

△

The Total Response of a Linear System

While Eq. (1.35) might seem to suggest that the output of a system is purely the result of *external* inputs, this is not entirely true. In general, the total response of a linear system for $t \geq 0$ is the result of two independent causes: the *external* input $x(t)$ for $t \geq 0$ and the *internal* initial conditions, or state, of the system at $t = 0^-$. Any arbitrary starting point can be used; the starting point $t = 0$ is chosen as a matter of convenience.[†] If a system is to be linear, the output must be the sum of the two components resulting from these two causes: first, the *zero-state response* (ZSR) component that results only from the input $x(t)$ for $t \geq 0$ when the initial conditions at $t = 0^-$ are assumed to be zero and, second, the *zero-input response* (ZIR) component that results only from the initial conditions at $t = 0^-$ with the input $x(t) = 0$ for $t \geq 0$. That is,

$$\text{total response} = \text{zero-state response} + \text{zero-input response} = \text{ZSR} + \text{ZIR}. \quad (1.36)$$

This property of linear systems, which permits the separation of an output into components resulting from the initial conditions and from the input, is called the *decomposition property*.

Equation (1.36) follows directly from the superposition principle of Eq. (1.35) if the causes $x_n(t)$ are generalized to include both *external* inputs and *internal* conditions. Thus, linearity implies that both the zero-input and zero-state components must obey the principle of superposition with respect to each of their respective causes. For example, if we increase the initial condition k -fold, the zero-input component must also increase k -fold. Similarly, if we increase the input k -fold, the zero-state component must also increase k -fold.

In this review of CT systems, we shall deal primarily with the zero-state response. When all the appropriate initial conditions are zero, the system is said to be in *zero state*. The system output is zero when the input is zero only if the system is in zero state.

Comments on Linear and Nonlinear Systems

Almost all CT systems observed in practice become nonlinear when large enough signals are applied to them. The analysis of nonlinear systems is generally difficult. Nonlinearities can arise in so many ways that describing them with a common mathematical form is impossible. Not only is each system a category in itself, but even for a given system, changes in the initial conditions or input amplitudes may change the nature of the problem. On the other hand, the superposition property is a powerful unifying principle that greatly simplifies the analysis of linear systems and allows for a general solution. It is therefore fortunate that many practical engineering systems exhibit linear behavior, at least for small signals.

1.5.2 Time Invariance

Systems whose parameters do not change with time are *time-invariant* (also *constant-parameter*) systems. For such a system, if the input is delayed by b seconds, the output is the same as before but delayed by b (assuming identical initial conditions). In other words, if the response of a time-invariant system to an input (cause) $x(t)$ is $y(t)$, then the response of the same system to an input $x(t - b)$ is $y(t - b)$.

[†]For a discussion of why it is desirable to place the initial conditions at $t = 0^-$ (rather than, say, $t = 0$ or $t = 0^+$), refer to [1].

A system described by the differential equation

$$a_K \frac{d^K y}{dt^K} + a_{K-1} \frac{d^{K-1} y}{dt^{K-1}} + \cdots + a_1 \frac{dy}{dt} + a_0 y = b_L \frac{d^L x}{dt^L} + b_{L-1} \frac{d^{L-1} x}{dt^{L-1}} + \cdots + b_1 \frac{dx}{dt} + b_0 x$$

is a linear system. More compactly, we express the system as

$$\sum_{k=0}^K a_k \frac{d^k}{dt^k} y(t) = \sum_{l=0}^L b_l \frac{d^l}{dt^l} x(t). \quad (1.37)$$

Often, Eq. (1.37) is normalized so that $a_K = 1$. The coefficients a_k and b_l in this equation can be constants or functions of time. If these coefficients are constants, then the system is linear and time invariant (LTI). If they are functions of time, the system is linear and time variant.

As Fig. 1.25 helps to illustrate, the time-invariance property is satisfied if a system acting on a delayed input (Fig. 1.25a) produces the same result as delaying the output of the system in response to the original input (Fig. 1.25b). Thought of another way, *time invariance implies that a system operator H commutes with the time-shifting operation H_{delay}* . With a time-invariant system, the ordering of delay operator and system does not matter. Coupled with linearity, the time-invariance property proves extremely useful to mathematically characterize a system's output, as shown next.

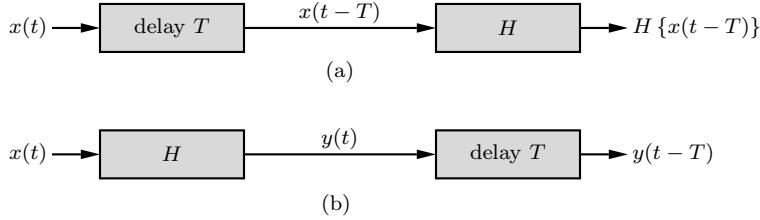


Figure 1.25: Visualizing time invariance: (a) a CT system acting on a delayed input and (b) the delayed output of a CT system.

1.5.3 The Zero-State Response of an LTIC System

We shall use the superposition and time-invariance properties to facilitate finding a system's zero-state response to an arbitrary input $x(t)$. To begin, recall that Eq. (1.9) provides us with a convenient representation of $x(t)$,

$$\underbrace{x(t)}_{\text{input}} = \underbrace{\int_{-\infty}^{\infty} \underbrace{x(\tau)}_{\text{scaled}} \underbrace{\delta(t - \tau)}_{\text{shifted impulses}} d\tau. \quad (1.38)$$

The output $y(t)$ of an LTIC system H is thus

$$y(t) = H \{x(t)\} = H \left\{ \int_{-\infty}^{\infty} x(\tau) \delta(t - \tau) d\tau \right\}. \quad (1.39)$$

Using the property of linearity, we know that summing (integration) and scaling commute with the system operation. Thus, Eq. (1.39) becomes

$$y(t) = \int_{-\infty}^{\infty} x(\tau) H \{\delta(t - \tau)\} d\tau. \quad (1.40)$$

Designating $h(t)$ as the response of an LTIC system to a unit impulse $\delta(t)$, the time-invariance property allows us to write Eq. (1.40) as

$$\underbrace{y(t)}_{\text{output}} = \underbrace{\int_{-\infty}^{\infty}}_{\text{sum}} \underbrace{x(\tau)}_{\text{scaled}} \underbrace{h(t - \tau)}_{\text{shifted impulse responses}} d\tau. \quad (1.41)$$

where $h(t) = H\{\delta(t)\}$. Notice the similar structures of Eqs. (1.38) and (1.41). The integral in Eq. (1.41) is known as the *convolution integral*, expressed symbolically as $x(t) * h(t)$. One can readily show that convolution is commutative [1]. Hence, an LTIC system response can be expressed as

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau) d\tau = \int_{-\infty}^{\infty} h(\tau)x(t - \tau) d\tau = h(t) * x(t). \quad (1.42)$$

The DNA of a System

Much like a single cell of a person contains the DNA that describes the person's entire bodily appearance, a single *impulse response* $h(t) = H\{\delta(t)\}$ contains the information needed to describe an LTIC system's zero-state responses *to the entirety of all possible inputs*. It is as if $h(t)$ contains a system's genetic blueprint. Knowing a system's $h(t)$ provides complete information about the system's ZSR. Further, systems with the same $h(t)$ will be identical twins, indistinguishable in their outward appearance, with regard to their zero-state responses. Still, just as two identical twins can have different personalities inside, systems with identical $h(t)$ can differ internally and produce unique zero-input responses.

1.5.4 Causality

A system is *causal* if the response does not start before the input. Alternately stated, a system is causal if the output at instant $t = t_0$ depends on the values of the input for $t \leq t_0$. A system that violates this condition is a noncausal system. A noncausal system response begins even before the input is applied. Clearly, all practical systems are causal since noncausal systems are physically unrealizable.[†]

Because $\delta(t)$ starts at $t = 0$, a system whose impulse response $h(t) = 0$ for $t < 0$ is a *causal system*. If $h(t)$ starts before $t = 0$, the system is noncausal. When a causal input $x(t)$ is applied to a causal system, Eq. (1.42) simplifies to

$$y(t) = x(t) * h(t) = \int_0^t x(\tau)h(t - \tau) d\tau = \int_0^t h(\tau)x(t - \tau) d\tau = h(t) * x(t). \quad (1.43)$$

1.5.5 Stability

System stability is an important and complex topic. Because of the great variety of possible system behaviors, there are several definitions of stability. Roughly speaking, a system is *stable* if *small* or *bounded* input or initial conditions yield bounded output. Here, we shall introduce what is known as *external stability*, which is also known as *bounded-input bounded-output (BIBO) stability*. A system is BIBO stable if and only if every bounded input $x(t)$ results in bounded output $y(t)$. More formally, BIBO system stability requires that, for all t ,

$$|y(t)| \leq K_y < \infty \text{ for all } x(t) \text{ with } |x(t)| \leq K_x < \infty, \quad (1.44)$$

[†]This statement is true only for temporal systems, where the independent variable is time. For nontemporal systems, such as those where the independent variable is distance, the response can precede the input. In such cases, noncausal systems are physically realizable. The study of noncausal systems is valuable even for temporal systems [1].

where K_x and K_y are some constants. It is important to stress again that for a system to be BIBO stable, the stability condition of Eq. (1.44) must hold for every possible input. If the system violates this condition for even one input, it is unstable.

Internal stability, also known as *asymptotic stability*, is defined in a similar way except that the test involves bounded initial conditions rather than bounded inputs. For asymptotic stability, we demand bounded output for bounded initial conditions. If bounded initial conditions (with zero input) can result in an unbounded response, the system is unstable. If the response is neither unbounded, nor decays to zero, the system is *marginally stable*. In practice, with few exceptions, external stability and internal stability are equivalent [1].

▷ **Drill 1.14 (Testing LTIC Systems for BIBO Stability)**

Show that an LTIC system is BIBO stable if its impulse response $h(t)$ is absolutely integrable, that is,

$$\int_{-\infty}^{\infty} |h(\tau)| d\tau \leq K_h < \infty,$$

where K_h is some constant. Hint: Substitute Eq. (1.41) into Eq. (1.44) and simplify the result.

△

1.6 Foundations of Frequency-Domain Analysis

The convolution method, where an LTIC system input is expressed as a sum of impulses and the response is found as the sum of the system's responses to all the impulse components, is a *time-domain* method of analysis. We now introduce another equally fruitful approach, the *frequency-domain* method, where the input is expressed as a sum of sinusoids (or exponentials), and the system response is obtained by adding the system's responses to all the sinusoidal (or exponential) components of the input. Such a method is attractive only if the response of an LTIC system to a sinusoid (or exponential) is simple and easy to obtain. This is precisely the case, as the following analysis shows.

1.6.1 LTIC System Response to an Everlasting Exponential e^{st}

We now show that an LTIC system response to an everlasting exponential input e^{st} is, within a multiplicative constant, the same exponential. That is, in the case of an exponential input, the shape of the output matches the shape of the input. Observe that here we are considering an *everlasting* exponential starting at $t = -\infty$, in contrast to a *causal* exponential $e^{st}u(t)$ starting at $t = 0$. The system response to an everlasting exponential e^{st} is

$$y(t) = h(t) * e^{st} = \int_{-\infty}^{\infty} h(\tau) e^{s(t-\tau)} d\tau = e^{st} \int_{-\infty}^{\infty} h(\tau) e^{-s\tau} d\tau.$$

The integral on the right-hand side, being a function of s but a constant with respect to t , will be denoted by $H(s)$. Thus,[†]

$$y(t) = H(s)e^{st} = H(s)x(t)|_{x(t)=e^{st}}, \quad (1.45)$$

where

$$H(s) = \int_{-\infty}^{\infty} h(\tau) e^{-s\tau} d\tau. \quad (1.46)$$

For causal $h(t)$, the integral on the right-hand side would range from the limits 0 to ∞ . For an everlasting exponential input, the output is the same as the input within the multiplicative constant

[†]This result is valid only for the values of s lying in the region of convergence (or existence) of $H(s)$, that is, where the integral $\int_{-\infty}^{\infty} h(\tau) e^{-s\tau} d\tau$ exists (converges). Region of convergence is further explained later in Sec. 1.10.

$H(s)$, called the *transfer function* of the system. No other input has this property.[†] As we shall see in Sec. 1.10, Eq. (1.46) states that a system's transfer function $H(s)$ is the *Laplace transform* of its impulse response $h(t)$.

For the input $x(t) = e^{st}$ (and only for e^{st}), the output is $y(t) = H(s)e^{st} = H(s)x(t)$. Thus, a rearrangement of Eq. (1.45) may be considered an alternate definition of the transfer function $H(s)$ of an LTIC system:

$$H(s) = \frac{\text{output signal}}{\text{input signal}} \Big|_{\substack{\text{input = everlasting exponential}}} = \frac{y(t)}{x(t)} \Big|_{x(t)=e^{st}}. \quad (1.47)$$

▷ Example 1.4 (Transfer Functions of Common Systems)

Use Eq. (1.47) to derive the transfer functions of

- | | |
|--|--|
| (a) an ideal time delay of T seconds | (b) an ideal differentiator |
| (c) an ideal integrator | (d) an LTIC system specified by Eq. (1.37) |

(a) An Ideal Time Delay of T Seconds

In this case, when the input is e^{st} , the output is $e^{s(t-T)}$. Hence, according to Eq. (1.47), the transfer function of an ideal T -second delay is

$$H(s) = \frac{e^{s(t-T)}}{e^{st}} = e^{-sT}. \quad (1.48)$$

(b) An Ideal Differentiator

In this case, when the input is e^{st} , the output is se^{st} . Using Eq. (1.47), the transfer function of an ideal differentiator is thus

$$H(s) = \frac{se^{st}}{e^{st}} = s. \quad (1.49)$$

(c) An Ideal Integrator

In this case, when the input is e^{st} , the output is $\frac{1}{s}e^{st}$. Using Eq. (1.47), the transfer function of an ideal integrator is thus

$$H(s) = \frac{e^{st}/s}{e^{st}} = \frac{1}{s}. \quad (1.50)$$

(d) An LTIC System Specified by Eq. (1.37)

When $x(t) = e^{st}$, the response $y(t) = H(s)e^{st}$. Under these conditions,

$$\frac{d^l}{dt^l}x(t) = \frac{d^l}{dt^l}e^{st} = s^l e^{st} \quad \text{and} \quad \frac{d^k}{dt^k}y(t) = \frac{d^k}{dt^k}H(s)e^{st} = H(s)\frac{d^k}{dt^k}e^{st} = s^k H(s)e^{st}.$$

Substitution of these results into Eq. (1.37) yields

$$\underbrace{(a_K s^K + a_{K-1} s^{K-1} + \cdots + a_1 s + a_0)}_{A(s)} H(s) e^{st} = \underbrace{(b_L s^L + b_{L-1} s^{L-1} + \cdots + b_1 s + b_0)}_{B(s)} e^{st}.$$

Solving for $H(s)$ yields

$$H(s) = \frac{b_L s^L + b_{L-1} s^{L-1} + \cdots + b_1 s + b_0}{a_K s^K + a_{K-1} s^{K-1} + \cdots + a_1 s + a_0} = \frac{B(s)}{A(s)}. \quad (1.51)$$

Expressing the numerator and the denominator in factored form yields

$$H(s) = \left(\frac{b_L}{a_K} \right) \frac{(s - z_1)(s - z_2)(s - z_3) + \cdots + (s - z_L)}{(s - p_1)(s - p_2)(s - p_3) + \cdots + (s - p_K)} = \left(\frac{b_L}{a_K} \right) \frac{\prod_{l=1}^L (s - z_l)}{\prod_{k=1}^K (s - p_k)}. \quad (1.52)$$

[†]The function with such property is called the *eigenfunction*, and the constant $H(s)$ is called the *eigenvalue* of the system.

Because $H(s)|_{s=z_l} = 0$ for ($l = 1, 2, \dots, L$), z_1, z_2, \dots, z_L are known as the *zeros* of $H(s)$. Similarly, p_1, p_2, \dots, p_K are known as the *poles* of $H(s)$ because $H(s)|_{s=p_k} = \infty$ for ($k = 1, 2, \dots, K$). The poles are also called the *characteristic roots* of the system. Systems with only LHP poles and zeros are termed *minimum phase systems*.

The poles and zeros of a system help provide us with an intuitive understanding of system behavior. When an everlasting exponential input $x(t) = e^{st}$ has complex frequency s that is close to a system zero, $H(s)$ is small, and so is the corresponding output. Conversely, if the frequency s is close to a system pole, both $H(s)$ and the corresponding output become large. In fact, when an input's frequency matches a system pole, *resonance* occurs, and new modes (different from the original input) are possible at the output. At first, it may seem that a resonant mode violates our previous conclusion that the output of an LTIC system to an everlasting exponential is the same everlasting exponential modified only in gain and phase. However, as cautioned earlier, such behavior is guaranteed only if the input is in the region of convergence of $H(s)$. In the case of resonance, the input is not in the region of convergence of $H(s)$. In summary, system zeros tend to suppress the system response, and system poles tend to amplify it. Further, as we shall see next, a system's poles are directly linked to system stability.

Stability Revisited

In Sec. 1.5.5, we introduced the notions of external (BIBO) stability and internal (asymptotic) stability. We can show that if all the poles (characteristic roots) of $H(s)$ lie in the LHP (have negative real parts), the system is BIBO stable [1]. Otherwise, it is BIBO unstable.

Similarly, we can, almost always, reliably establish asymptotic stability based on $H(s)$ as follows.[†] If the poles of $H(s)$ lie in the LHP, the system is asymptotically stable. The system is asymptotically unstable if some poles (even one) lie in the RHP or if repeated poles lie on the imaginary axis. If some simple poles lie on the imaginary axis and all the remaining poles lie in the LHP, the system is marginally stable.

Example 1.4 ◀

▷ Drill 1.15 (Computing the Transfer Functions of Common Systems)

- (a) Determine the impulse response $h(t)$ of an ideal delay of T seconds and then use this result in Eq. (1.46) to determine the corresponding system transfer function.
- (b) Find the differential equation of the form in Eq. (1.37) for an ideal differentiator and for an ideal integrator. From these differential equations, find their transfer functions using the result in Eq. (1.51).

◀

The Jackpot

The importance of Eq. (1.45) cannot be stressed enough. It states that when the input to an LTIC system is an everlasting exponential e^{st} , then, within the multiplicative constant $H(s)$, the output is identical to the input. Only everlasting exponentials possess this amazing property where the shape of the input is preserved at the system's output. The basis for frequency-domain analysis rests solidly on Eq. (1.45).

Given that Eq. (1.45) makes it so very simple to find the LTIC system response to an everlasting exponential, would it not be nice if all the inputs in the world were everlasting exponentials (or sinusoids)? But reality being what it is, the next best thing is to hope that every signal in practice

[†]Since $H(s)$ is an external description of a system, examining its poles may not indicate internal instabilities unless the system is both *controllable* and *observable*. Fortunately, practical systems are, almost always, both controllable and observable [1].

can be expressed as a sum of such exponentials (or sinusoids). Here we hit the jackpot. It is known that every signal that can be generated in practice can be expressed as a sum of everlasting exponentials (or sinusoids) [1]. The sum may be discrete or over a continuum of frequencies. This is true even for most signals that cannot be generated in practice but are of theoretical significance, such as a unit step, an impulse, or an exponentially growing signal.

We can show (see [1]) that a periodic signal $x(t)$ can be expressed as a sum of exponentials as

$$x(t) = \sum_k X_k e^{s_k t}. \quad (1.53)$$

The response $y(t)$ of an LTIC system with transfer function $H(s)$ to this input $x(t)$ is

$$y(t) = \sum_k H(s_k) X_k e^{s_k t} = \sum_k Y_k e^{s_k t}, \quad (1.54)$$

where $Y_k = H(s_k)X_k$.

An aperiodic signal $x(t)$ can be expressed as a sum of exponentials over a continuum of frequencies as

$$x(t) = \frac{1}{j2\pi} \int_{S_x} X(s) e^{st} ds, \quad (1.55)$$

and the corresponding output is given by

$$y(t) = \frac{1}{j2\pi} \int_{S_y} H(s) X(s) e^{st} ds = \frac{1}{j2\pi} \int_{S_y} Y(s) e^{st} ds, \quad (1.56)$$

where $Y(s) = H(s)X(s)$.

1.7 The Fourier Series

Much as Eq. (1.9) represents a signal $x(t)$ in terms of delta functions, the Fourier series tells us that we can represent a periodic signal $x(t)$ in terms of everlasting complex exponentials or, equivalently, in terms of everlasting sinusoids. Given the simplicity of system analysis for exponential inputs, as seen in Sec. 1.6, such a representation holds much appeal.

There are two primary forms of the Fourier series: the exponential form and the trigonometric form. In the case of real signals, the trigonometric form also leads to a compact trigonometric form. Each form carries identical information. We prefer the exponential form because, when compared with the trigonometric forms, it is very compact. Furthermore, an LTIC system response to exponential signals is also simpler (more compact) than the system response to sinusoids. Moreover, mathematical manipulation and handling of the exponential form (which simplifies multiplication, division, differentiation, and integration) prove much easier than the trigonometric forms.

A minor disadvantage of the exponential form is that it, being a complex signal, is less easy to visualize than a construction based on sinusoids. Fortunately, this difficulty can be readily overcome because of the close connection between exponential and trigonometric coefficients and spectra. For the purpose of mathematical analysis, we shall use exponential signals and spectra, but to promote an intuitive or qualitative understanding, we shall speak in terms of a sinusoidal decomposition. Thus, although most of our future mathematical manipulations will be in terms of exponential spectra, we shall speak of exponentials and sinusoids interchangeably. This is an important point; readers should make extra effort in familiarizing themselves with the three forms of spectra, their relationships, and their convertibility.

1.7.1 Exponential Form of the Fourier Series

A periodic signal $x(t)$ with fundamental period T_0 (fundamental frequency $f_0 = 1/T_0$) can be expressed as a sum of a complex exponential of period T_0 and its harmonics:

$$x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}. \quad (1.57)$$

Here, $\omega_0 = 2\pi f_0 = 2\pi/T_0$ is the fundamental (radian) frequency. Often, Eq. (1.57) is referred to the Fourier series *synthesis equation*.

The coefficients X_k are complex, in general, and are given by

$$X_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk\omega_0 t} dt. \quad (1.58)$$

Often, Eq. (1.58) is referred to the Fourier series *analysis equation*. A discussion of basic Fourier series properties is delayed to Sec. 1.8, where they are presented alongside the corresponding properties of the Fourier transform. To denote the Fourier series relation between $x(t)$ and X_k , we adopt the compact notation of

$$x(t) \iff X_k.$$

For obvious reasons, the Fourier series coefficients for an arbitrary signal are labeled in a manner that is consistent with the time-domain function; that is, Y_k are the Fourier series coefficients of signal $y(t)$, Z_k for $z(t)$, and so forth.

The alert reader will note that Eq. (1.57) is just a special case of Eq. (1.53) where $s_k = jk\omega_0$. Thus, the output of an LTIC system $H(s)$ to a periodic input can, given minor conditions on $H(s)$, be readily computed knowing the input's Fourier series and using Eq. (1.54) with $s_k = jk\omega_0$. In understanding the output of an LTIC system in response to a periodic input, we just hit the jackpot.

The Fourier Series Spectrum

The exponential Fourier series in Eq. (1.57) indicates that a periodic signal $x(t)$ can be expressed as a weighted sum of harmonically related complex exponentials $e^{jk\omega_0 t}$. The weights X_k constitute what is known as the *frequency spectrum* of $x(t)$. To plot the complex coefficients X_k as a function of k or, more informatively, as a function of radian frequency $k\omega_0$, we need two plots: the real and the imaginary parts of X_k or the magnitude and the angle of X_k . We prefer the latter because it is more intuitive and informative.

The *magnitude spectrum* $|X_k|$ tells us the strength of each exponential component $e^{jk\omega_0 t}$, and the *phase spectrum* $\angle X_k$ tells us the phase of each exponential component. To facilitate plotting the magnitude and phase spectra, the coefficients X_k should be expressed in polar form as $|X_k|e^{j\angle X_k}$.

More intuitively, these spectra show at a glance the magnitudes and the phases of the sinusoids that make up the composition of the signal $x(t)$. Knowing these spectra, we can reconstruct or synthesize the signal $x(t)$ according to Eq. (1.57). Therefore, frequency spectra, which provide an alternative way of describing a periodic signal $x(t)$, are in every way equivalent to the plot of $x(t)$ as a function of t . In contrast to the *time-domain description* $x(t)$, the frequency spectra $|X_k|$ and $\angle X_k$ constitute the *frequency-domain description* of the signal. *Signals, therefore, have dual identities: the time-domain identity and the frequency-domain identity. The two identities complement each other; taken together, they provide a better understanding of a signal.*

In the special case of $k = 0$, notice that X_0 is just the average value of $x(t)$ taken over one period. This *dc component* of the signal can often be determined by inspection of $x(t)$. Moreover, Eq. (1.58) shows that for real $x(t)$, X_k is conjugate symmetric ($X_k = X_{-k}^*$), which ensures that

$$|X_k| = |X_{-k}| \quad \text{and} \quad \angle X_k = -\angle X_{-k}. \quad (1.59)$$

In other words, a real signal $x(t)$ has an even magnitude spectrum $|X_k|$ and an odd phase spectrum $\angle X_k$.

▷ **Example 1.5 (The Exponential Fourier Series and Spectra)**

Find the exponential Fourier series for the periodic signal $x(t)$ shown in Fig. 1.26. Sketch the magnitude and phase spectra for $x(t)$.

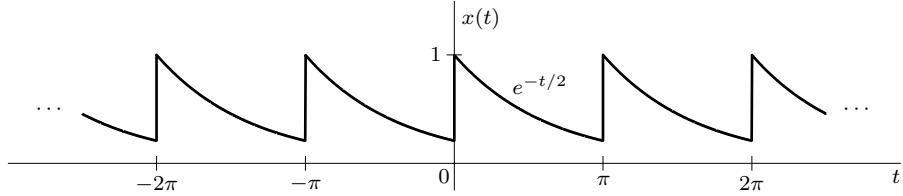


Figure 1.26: A π -periodic signal $x(t)$ with $x(t) = e^{-t/2}$ over $(0 \leq t < \pi)$.

In this case $T_0 = \pi$ s, $\omega_0 = 2\pi/T_0 = 2$ rad/s, and $x(t) = \sum_{k=-\infty}^{\infty} X_k e^{j2kt}$, where

$$X_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-j2kt} dt = \frac{1}{\pi} \int_0^\pi e^{-t/2} e^{-j2kt} dt = \frac{2}{\pi} \left(\frac{1 - e^{-\pi/2}}{1 + j4k} \right) \approx \frac{0.504}{1 + j4k}. \quad (1.60)$$

Using Eq. (1.60), the magnitude and phase spectra are conveniently plotted using MATLAB.

```
01 omega0 = 2; k = -5:5; Xk = (2/pi)*(1-exp(-pi/2))./(1+1j*4*k);
02 subplot(121); stem(k*omega0,abs(Xk)); xlabel('k\omega_0'); ylabel('|X_k|');
03 subplot(122); stem(k*omega0,angle(Xk)); xlabel('k\omega_0'); ylabel('\angle X_k');
```

The results are shown in Fig. 1.27. Observe that the signal has significant dc content and that the coefficients X_k are complex. Moreover, X_k and X_{-k} are conjugates, as expected coming from a real signal. Thus, the magnitude spectrum is an even function, and the angle spectrum is an odd function. This confirms the observation in Eq. (1.59).

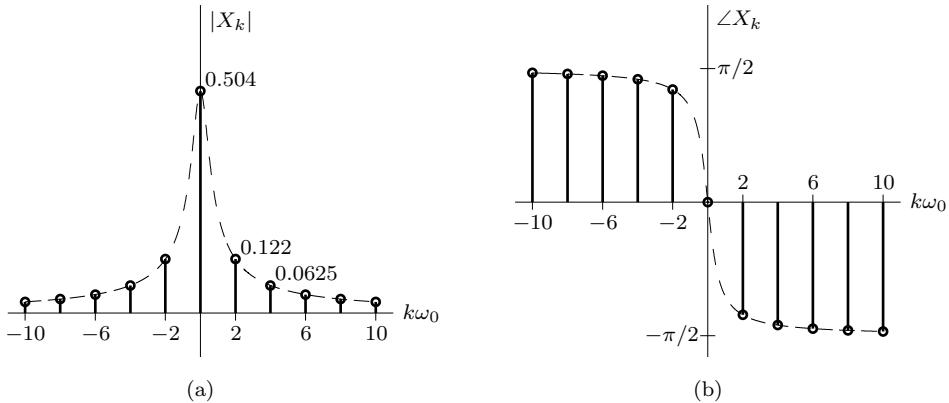


Figure 1.27: Spectra for π -periodic signal $x(t) = e^{-t/2}$ over $(0 \leq t < \pi)$: (a) $|X_k|$ and (b) $\angle X_k$.

Example 1.5 ◀

An interesting aspect of Fourier series is that whenever there is a jump discontinuity in $x(t)$, the series at the point of discontinuity converges to an average of the left-hand and right-hand limits of $x(t)$ at the instant of discontinuity, a behavior that is dictated by the minimizing mean square error property of the Fourier series [1]. In Ex. 1.5, for instance, $x(t)$ is discontinuous at $t = 0$ with $x(0^+) = 1$ and $x(0^-) = e^{-\pi/2} = 0.2079$. The corresponding Fourier series converges to a value $(1+0.2079)/2 = 0.6039$ at $t = 0$. This is easily verified by noting from Eq. (1.57) that $x(0) = \sum_k X_k$, which MATLAB readily computes.

```
01 k = -1000:1000; Xk = (2/pi)*(1-exp(-pi/2))./(1+1j*4*k); sum(Xk)
ans = 0.6039-0.0000i
```

The issue of convergence is discussed in greater depth in Sec. 1.7.3.

What Is a Negative Frequency?

Figure 1.27 illustrates an interesting feature of the exponential Fourier series, namely, that spectra exist for positive as well as negative values of $k\omega_0$ (the radian frequency). The existence of negative frequencies is somewhat disturbing because, by common definition, frequency (the number of repetitions per second) is a positive quantity. How do we interpret a negative frequency?

To understand this apparent mystery, we need to carefully understand what is meant by *frequency*. Consider a 1-Hz sinusoid and a 1-Hz square wave. Both have an identical repetition rate, or frequency, but we know that the signals themselves are different and thus have different Fourier series. To accommodate these differences, Fourier spectra necessarily regard frequency in a more restrictive, and more precise, way to indicate complex exponential (or sinusoidal) content.

Consider two components of our Fourier spectra: $e^{j\omega_0 t}$ and $e^{-j\omega_0 t}$. The first is a positive-frequency component and the second is a negative-frequency component. Mathematically, the distinction is clear, but intuitively this explanation is lacking. Notice, however, that both $e^{j\omega_0 t}$ and $e^{-j\omega_0 t}$ are periodic functions that share the same period $T_0 = 2\pi/\omega_0$. Thus, both $e^{j\omega_0 t}$ and $e^{-j\omega_0 t}$ repeat the same number of times per second. The manner in which $e^{j\omega_0 t}$ and $e^{-j\omega_0 t}$ repeat, however, is different, and herein lies the essence of what is meant by negative frequency.

Figure 1.28 helps demonstrate this distinction. Both $e^{j\omega_0 t}$ and $e^{-j\omega_0 t}$ are unit-length complex vectors that, as functions of t , rotate in the complex plane. The positive-frequency component $e^{j\omega_0 t}$ rotates in a counterclockwise direction, while the negative-frequency component $e^{-j\omega_0 t}$ rotates in the opposite, or clockwise, direction. The negative sign simply reverses the direction of rotation, and the meaning of negative frequency becomes clear.

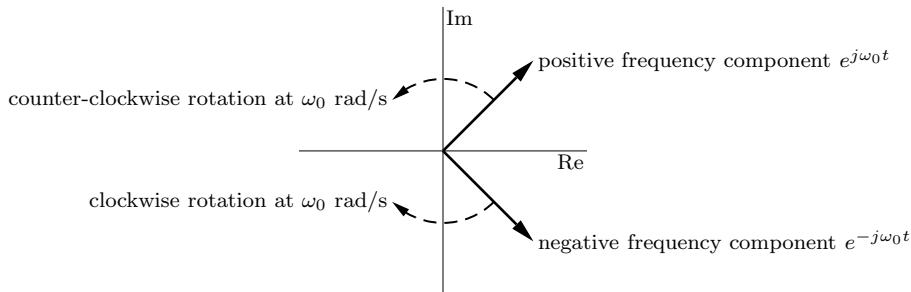


Figure 1.28: Understanding negative frequency.

The situation is analogous to the case when two people run in opposite directions with the same speed. Although the term negative velocity seems rather strange, it allows us to distinguish the different directions of the two runners. Similar is the case with these exponentials and the term frequency. Although negative frequency may seem strange, it allows us to distinguish the rotation direction of our complex exponential components.

Figure 1.28 also provides insight into *why* negative frequencies are required in most Fourier series. It would be impossible, for example, to generate a purely real signal using only positive-frequency components. Real signals require a carefully choreographed conjugate-symmetric dance between positive and negative frequencies where the imaginary component of $e^{jk\omega_0 t}$ can only be countered by its negative-frequency mirror $e^{-jk\omega_0 t}$.

Finality Property of the Fourier Series

One of the important properties of the Fourier series is the *finality* property. Consider a signal $x(t)$ and its Fourier series representation $x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}$. If we approximate $x(t)$ by a truncated (finite number of terms) Fourier series as

$$x(t) \approx x_K(t) = \sum_{k=-K}^K X_k e^{jk\omega_0 t}, \quad (1.61)$$

then according to the finality property, this approximation is the optimum for a given K . Thought of another way, the coefficients X_k are themselves optimal, and each coefficient is independent of the others. There is no choice for X_k other than the Fourier coefficients that will improve the approximation. The optimum here is in the sense of minimizing the energy over one cycle of the error $x(t) - x_K(t)$.

1.7.2 Trigonometric and Compact Trigonometric Forms

The trigonometric form of the Fourier series re-expresses Eq. (1.57) as

$$x(t) = a_0 + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)], \quad (1.62)$$

where

$$\begin{aligned} a_0 &= \frac{1}{T_0} \int_{T_0} x(t) dt, \\ a_k &= \frac{2}{T_0} \int_{T_0} x(t) \cos(k\omega_0 t) dt, \\ \text{and } b_k &= \frac{2}{T_0} \int_{T_0} x(t) \sin(k\omega_0 t) dt. \end{aligned} \quad (1.63)$$

Equation (1.62) tells us what we already know: we can decompose a periodic signal into a sum of sines and cosines. Noting the equivalence of the exponential and trigonometric forms, it takes little effort to demonstrate that

$$\begin{aligned} a_k &= X_k + X_{-k}, & b_k &= j(X_k - X_{-k}), \\ X_0 &= a_0, & \text{and } X_k &= (a_k - jb_k)/2. \end{aligned} \quad (1.64)$$

The results derived so far are general and apply whether $x(t)$ is a real or a complex function of t . However, when $x(t)$ is real, coefficients a_k and b_k are real for all k , and using trigonometric identities, we can express the Fourier series in Eq. (1.62) in the compact trigonometric form as

$$x(t) = C_0 + \sum_{k=1}^{\infty} C_k \cos(k\omega_0 t + \theta_k), \quad (1.65)$$

where

$$\begin{aligned} C_0 &= a_0 = X_0, \\ C_k &= +\sqrt{a_k^2 + b_k^2} = 2|X_k|, \\ \text{and } \theta_k &= \tan^{-1} \left(\frac{-b_k}{a_k} \right) = \angle X_k. \end{aligned} \quad (1.66)$$

Notice that the coefficients C_k are always positive, although C_0 may be positive or negative. Further, θ_k should always be computed using a two-argument arc-tangent function. For example, if $a_k = -1$ and $b_k = 1$, θ_k lies in the third quadrant, and

$$\theta_k = \tan^{-1} \left(\frac{-1}{-1} \right) = -3\pi/4 \neq \tan^{-1}(1) = \pi/4.$$

▷ Example 1.6 (Trigonometric Fourier Series and Spectra)

Find the trigonometric and compact trigonometric Fourier series for the periodic signal $x(t)$ shown in Fig. 1.29. Plot the corresponding spectra.

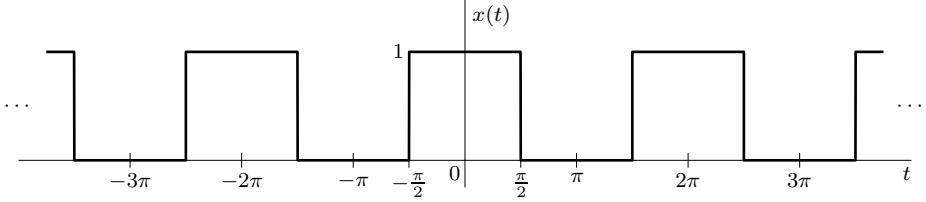


Figure 1.29: A 2π -periodic dc-offset square wave $x(t)$.

In this case, $T_0 = 2\pi$. Therefore, $\omega_0 = 1$. The trigonometric Fourier series representation for this periodic signal is

$$x(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos(kt) + b_k \sin(kt),$$

where

$$\begin{aligned} a_0 &= \frac{1}{2\pi} \int_{-\pi}^{\pi} x(t) dt = \frac{1}{2\pi} \int_{-\pi/2}^{\pi/2} (1) dt = \frac{1}{2}, \\ a_k &= \frac{2}{2\pi} \int_{-\pi/2}^{\pi/2} (1) \cos(kt) dt = \frac{2}{k\pi} \sin \left(\frac{k\pi}{2} \right) = \text{sinc} \left(\frac{k}{2} \right) \\ &= \begin{cases} 0 & k \text{ even} \\ \frac{2}{k\pi} & k = 1, 5, 9, 13, \dots \\ -\frac{2}{k\pi} & k = 3, 7, 11, 15, \dots \end{cases}, \\ \text{and } b_k &= \frac{1}{\pi} \int_{-\pi/2}^{\pi/2} (1) \sin(kt) dt = 0. \end{aligned} \quad (1.67)$$

Therefore,

$$x(t) = \frac{1}{2} + \frac{2}{\pi} \left(\cos(t) - \frac{1}{3} \cos(3t) + \frac{1}{5} \cos(5t) - \frac{1}{7} \cos(7t) + \dots \right). \quad (1.68)$$

Observe that since $b_k = 0$, only the cosine terms appear in the trigonometric series and complete spectral information is contained in the plot of a_k , as shown in Fig. 1.30a. As expected, the amplitudes of the nonzero harmonics alternate sign. MATLAB makes plotting Fig. 1.30a simple.

```
01 omega0 = 1; a0 = 0.5; k = 1:10; ak = sinc(k/2); k = [0,k]; ak = [a0,ak];
02 subplot(311); stem(k*omega0,ak); xlabel('k\omega_0'); ylabel('a_k');
```

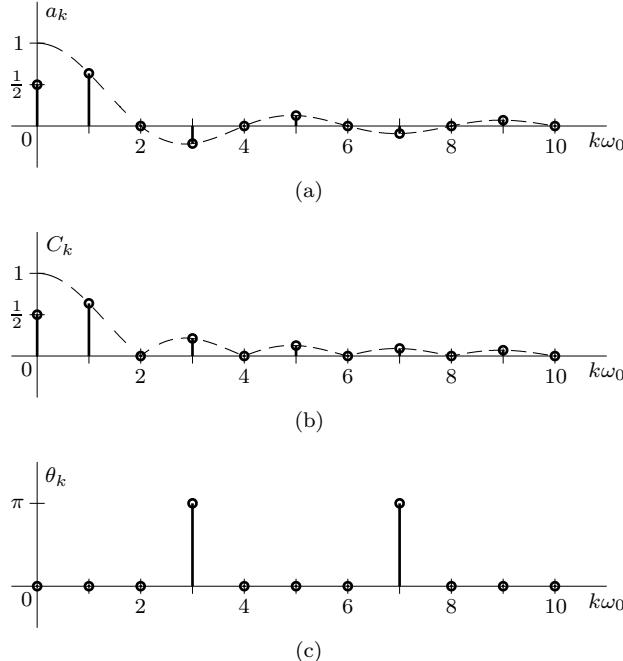


Figure 1.30: Trigonometric spectra for dc-offset square wave $x(t)$: (a) a_k , (b) C_k , and (c) θ_k .

The absolute values of amplitudes are the *magnitudes*. A negative amplitude can be converted to a positive amplitude by adjusting the phase by π rads using the trigonometric identity[†]

$$-\cos(x) = \cos(x + \pi).$$

Using this fact, we can express the series in Eq. (1.68) as

$$x(t) = \frac{1}{2} + \frac{2}{\pi} \left(\cos(t) + \frac{1}{3} \cos(3t + \pi) + \frac{1}{5} \cos(5t) + \frac{1}{7} \cos(7t + \pi) + \dots \right). \quad (1.69)$$

All the amplitudes are rendered positive. Equation (1.69) is precisely the compact trigonometric Fourier series; by inspection, the magnitude and phase spectra are

$$\begin{aligned} C_0 &= 0.5, \\ C_k &= \begin{cases} 0 & k \text{ even} \\ \frac{2}{\pi k} & k \text{ odd} \end{cases}, \\ \text{and } \theta_k &= \begin{cases} 0 & \text{for all } k \neq 3, 7, 11, 15, \dots \\ \pi & k = 3, 7, 11, 15, \dots \end{cases}. \end{aligned}$$

[†]Because $\cos(x \pm \pi) = -\cos(x)$, we could have chosen the phase π or $-\pi$. In fact, $\cos(x \pm m\pi) = -\cos(x)$ for any odd integral value of m . Therefore, the phase can be chosen as $\pm m\pi$ where m is any convenient odd integer.

The magnitude and phase spectra are shown in Figs. 1.30b and 1.30c, respectively. Observe that magnitude spectrum in Fig. 1.30b is the rectified version of the amplitude spectrum in Fig. 1.30a. Figures 1.30b and 1.30c are obtained by continuing our MATLAB code.

```
03 Ck = abs(ak); thetak = atan2(0,ak).*(abs(ak)>1e-10);
04 subplot(312); stem(k*omega0,Ck); xlabel('k\omega_0'); ylabel('C_k');
05 subplot(313); stem(k*omega0,thetak); xlabel('k\omega_0'); ylabel('\theta_k');
```

Notice that the $.*(abs(ak)>1e-10)$ portion at the end of line 03 is used to eliminate tiny a_k (which should be zero) that result from computer round-off; if not removed, MATLAB diligently computes phase values that are misleading and meaningless.

From Eq. (1.68), we observe that the series converges to 0.5 at all discontinuities. For example, at $t = \pi/2$, all the terms in the series except a_0 vanish, which yields $x(\frac{\pi}{2}) = 0.5$. This is the average of the values of $x(t)$ on either side of the discontinuity because, for a small ϵ , $x(\frac{\pi}{2} - \epsilon) = 1$ and $x(\frac{\pi}{2} + \epsilon) = 0$.

Example 1.6 ◀

▷ Example 1.7 (Fourier Series and Spectra of a Unit Impulse Train)

Find the exponential Fourier series and sketch the corresponding spectra for the T_0 -periodic unit impulse train $x(t) = \tilde{\delta}(t)$ shown in Fig 1.31. From this result, sketch the corresponding compact trigonometric spectrum and write the compact trigonometric Fourier series.

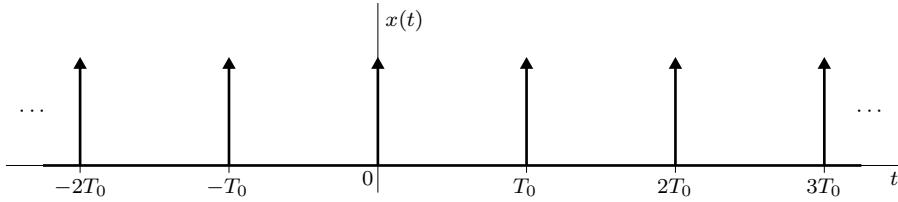


Figure 1.31: T_0 -periodic unit impulse train $x(t) = \tilde{\delta}(t)$.

Since $\tilde{\delta}(t)$ is T_0 -periodic, it has a fundamental frequency of $\omega_0 = \frac{2\pi}{T_0}$. The exponential Fourier series is thus given by

$$\tilde{\delta}(t) = \sum_{k=-\infty}^{\infty} X_k e^{j k \omega_0 t} = \sum_{k=-\infty}^{\infty} X_k e^{j 2\pi k t / T_0}, \quad (1.70)$$

where

$$X_k = \frac{1}{T_0} \int_{T_0} \tilde{\delta}(t) e^{-j k \omega_0 t} dt.$$

Choosing the interval of integration as $(-\frac{T_0}{2}, \frac{T_0}{2})$ and recognizing that over this interval $\tilde{\delta}(t) = \delta(t)$,

$$X_k = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} \delta(t) e^{-j k \omega_0 t} dt = \frac{1}{T_0}. \quad (1.71)$$

Substitution of this value into Eq. (1.70) yields the desired exponential Fourier series,

$$\tilde{\delta}(t) = \frac{1}{T_0} \sum_{k=-\infty}^{\infty} e^{j k \omega_0 t} = \frac{1}{T_0} \sum_{k=-\infty}^{\infty} e^{j 2\pi k t / T_0}.$$

Equation (1.71) shows that the exponential spectrum is uniform ($X_k = 1/T_0$) for all frequencies $k\omega_0$. The spectrum, being real and positive, requires only the magnitude plot $|X_k|$, shown in Fig. 1.32a, or C_k , shown in Fig. 1.32b. All phases ($\angle X_k$ and θ_k) are zero.

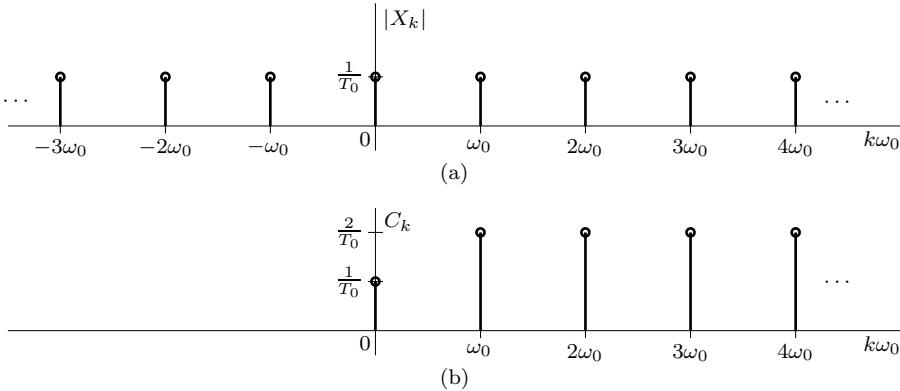


Figure 1.32: Fourier spectra for $x(t) = \tilde{\delta}(t)$: (a) $|X_k|$ and (b) C_k .

To sketch the compact trigonometric spectrum, we use Eq. (1.66) to obtain

$$\begin{aligned} C_0 &= 1/T_0, \\ C_k &= 2/T_0 \quad (k = 1, 2, 3, \dots), \\ \text{and } \theta_k &= 0. \end{aligned}$$

Figure 1.32b shows the compact trigonometric Fourier spectrum. From this spectrum we can express $\tilde{\delta}(t)$ using the compact trigonometric form as

$$\tilde{\delta}(t) = \frac{1}{T_0} + \frac{2}{T_0} (\cos(\omega_0 t) + \cos(2\omega_0 t) + \cos(3\omega_0 t) + \dots), \quad \text{where } \omega_0 = \frac{2\pi}{T_0}.$$

Example 1.7 ▶

1.7.3 Convergence of a Series

The key to many puzzles of the Fourier series lies in the nature of the convergence of the Fourier series. Convergence of infinite series is a challenging problem. It took mathematicians several decades to understand the convergence aspect of the Fourier series. We shall barely touch the surface here.

In any practical application, we can use only a finite number of terms in a series. If, using a fixed number of terms, the series guarantees convergence within an arbitrarily small error at every value of t over some interval $T_1 \leq t \leq T_2$, such a series is highly desirable and is called a *uniformly convergent* series over the interval $T_1 \leq t \leq T_2$. If a series converges at every value of t over the interval $T_1 \leq t \leq T_2$, but to guarantee convergence within a given error requires different number of terms at different t , then the series is still convergent but is less desirable. It goes under the name *point-wise convergent* series.

Finally, we have the case of a series which refuses to converge at some t , no matter how many terms are added. But the series may *converge in the mean*, that is, the energy of the difference between $x(t)$ and the corresponding finite term series approaches zero as the number of terms approaches infinity.[†] To explain this concept, let us consider representation of a function $x(t)$ by an

[†]The reason for calling this “convergence in the mean” is that minimizing the error energy over a certain interval is equivalent to minimizing the mean square value of the error over the same interval.

infinite series

$$x(t) = \sum_{k=-\infty}^{\infty} z_k(t).$$

Let the partial sum of the middle $2K + 1$ terms of the series on the right-hand side be denoted by $x_K(t)$, that is,

$$x_K(t) = \sum_{k=-K}^{K} z_k(t),$$

which we recognize as being identical in form to the truncated Fourier series in Eq. (1.61). If we approximate $x(t)$ by $x_K(t)$, the *error* in the approximation is the difference $x(t) - x_K(t)$. The series converges *in the mean* to $x(t)$ in the interval $(0, T_0)$ if

$$\int_0^{T_0} |x(t) - x_K(t)|^2 dt \rightarrow 0 \quad \text{as } K \rightarrow \infty.$$

That is, the energy of the error $x(t) - x_K(t)$ approaches zero as $K \rightarrow \infty$. This form of convergence does not require that the series be equal to $x(t)$ for all t . It just requires the energy of the difference (area under $|x(t) - x_K(t)|^2$) to vanish as $K \rightarrow \infty$. Superficially it may appear that if the energy of a signal over an interval is zero, the signal (error) must be zero everywhere. This is not true. Signal energy can be zero even if it has nonzero values at a finite number of isolated points. This is because although the signal is nonzero at a point (and zero everywhere else), the area under its square is still zero. Thus, a series that converges in the mean to $x(t)$ need not converge to $x(t)$ at a finite number of points. This is precisely what happens to the Fourier series when $x(t)$ has jump discontinuities. This is also what makes the Fourier series convergence compatible with the Gibbs phenomenon, which is discussed later in this section.

There is a simple criterion for ensuring that a periodic signal $x(t)$ has a Fourier series that converges in the mean. The Fourier series for $x(t)$ converges to $x(t)$ in the mean if $x(t)$ has a finite energy over one period, that is,

$$\int_{T_0}^{T_0} |x(t)|^2 dt < \infty. \quad (1.72)$$

In all the examples discussed so far, the condition of Eq. (1.72) is satisfied, and hence the corresponding Fourier series converge in the mean. Condition (1.72) also guarantees that the Fourier coefficients are finite.

We shall now discuss an alternate set of criteria due to Dirichlet for convergence of the Fourier series.

Dirichlet Conditions

Dirichlet has shown that if $x(t)$ satisfies certain conditions, its Fourier series is guaranteed to converge point-wise at all points where $x(t)$ is continuous. Moreover, at points of discontinuity, $x(t)$ converges to the value midway between the two values of $x(t)$ on either side of the discontinuity. The *Dirichlet conditions* are

1. $x(t)$ is absolutely integrable over one period, that is,

$$\int_{T_0}^{T_0} |x(t)| dt < \infty, \quad (1.73)$$

2. $x(t)$ has a finite number of finite discontinuities in one period, and

3. $x(t)$ contains a finite number of maxima and minima in one period.

Any periodic waveform that can be generated in a laboratory satisfies the Dirichlet conditions and hence possesses a convergent Fourier series. Thus, the physical possibility of a periodic waveform is a valid and sufficient condition for the existence of a convergent Fourier series.

Truncation of the Fourier Series: The Gibbs Phenomenon

The Fourier series of a signal $x(t)$ shows explicitly the sinusoidal components of $x(t)$. We can synthesize $x(t)$ by consecutively adding the sinusoids in the spectrum of $x(t)$. For a finite number of terms, the sum is the truncated Fourier series approximation $x_K(t)$. For smooth, well-behaved signals, $x_K(t)$ can closely approximate $x(t)$ even using a relatively few number of terms. Sometimes, such as in the case when discontinuities are present, $x_K(t)$ can exhibit unusual behavior regardless of the size of K .

Consider a periodic square-pulse signal $x(t)$ (Fig. 1.29) and its truncated Fourier series approximations $x_K(t)$ (Fig. 1.33), obtained for various K by using Eqs. (1.61), (1.62), and (1.67) as

$$x_K(t) = a_0 + \sum_{k=1}^K [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] = 0.5 + \sum_{k=1}^K \text{sinc}(k/2) \cos(kt).$$

We start the synthesis with only the first term in the series, $a_0 = 0.5$ ($K = 0$). This term is a constant (dc). Approximating $x(t)$ with just the dc term is a gross approximation of the square wave, as shown in Fig. 1.33a. In the next step, we add the first harmonic (fundamental) to the dc component, which results in a signal shown in Fig. 1.33b. Observe that this synthesized signal $x_1(t)$ somewhat resembles $x(t)$. It is a smoothed-out version of $x(t)$. The sharp corners in $x(t)$ are not reproduced in this signal because sharp corners indicate rapid changes, and their reproduction requires rapidly varying (i.e., higher frequency) components, which are yet to be included. Figure 1.33c shows the sum of dc and the first and third harmonics (in this case, even harmonics have zero magnitudes). As we increase the number of harmonics progressively, as illustrated in Fig. 1.33d (sum up to the fifth harmonic, $K = 5$) and Fig. 1.33e (sum up to the nineteenth harmonic, $K = 19$), the edges of the pulses become sharper, and $x_K(t)$ resembles $x(t)$ more closely.

The plot of the truncated series approximates closely the function $x(t)$ as K increases, and we expect that the series will converge exactly to $x(t)$ as the number of terms $K \rightarrow \infty$. The curious fact, as seen from Fig. 1.33, is that even for large K , the truncated series exhibits an oscillatory behavior and an overshoot with a peak value of about 9% in the vicinity of the discontinuity.[†] Regardless of the value of K , the overshoot remains at about 9%. This strange behavior appears to contradict our expectation that $x_K(t)$ approaches $x(t)$ as $K \rightarrow \infty$. In fact, this apparent contradiction puzzled many people at the turn of the century. Josiah Willard Gibbs gave a mathematical explanation of this behavior, which is now called the *Gibbs phenomenon*.

We can reconcile the two conflicting notions as follows. If f_0 is the fundamental frequency of the series, the highest frequency of oscillation of the synthesized signal $x_K(t)$ is Kf_0 ; thus, the spikes (oscillations) with 9% overshoot have a width of approximately $1/Kf_0$ and are spaced approximately $1/2Kf_0$ to either side of each discontinuity. For the case in Fig. 1.33, the fundamental frequency is $f_0 = 1/2\pi$, the spike width is about $2\pi/K$, and the 9% peaks occur at $\pm\pi/K$ from each discontinuity. As we increase the number of terms K , the frequency of oscillation increases, spike width $1/Kf_0$ diminishes, and spikes move closer toward their discontinuities. As $K \rightarrow \infty$, the error energy $\rightarrow 0$ because the error consists mostly of the spikes, whose widths $\rightarrow 0$. Therefore, as $K \rightarrow \infty$, the corresponding Fourier series differs from $x(t)$ by about 9% at the immediate left and right of the points of discontinuity, and yet the error energy $\rightarrow 0$. Thus the series converges in the mean. Where $x(t)$ is continuous, the series also converges point-wise. This is because at any instant, no matter how close to the discontinuity, we can increase K to push the 9% spike closer to the discontinuity. The ramifications and the cure for Gibbs phenomenon are discussed in Sec. 2.4.

Gibbs phenomenon is present when there is a jump discontinuity in $x(t)$. When a continuous function $x(t)$ is synthesized using the first K terms of the Fourier series, the synthesized function approaches $x(t)$ for all t as $K \rightarrow \infty$. No Gibbs phenomenon appears.

[†] In optics, the oscillations are called *diffraction fringes* and appear at every point for which the object shows large variations in illumination.

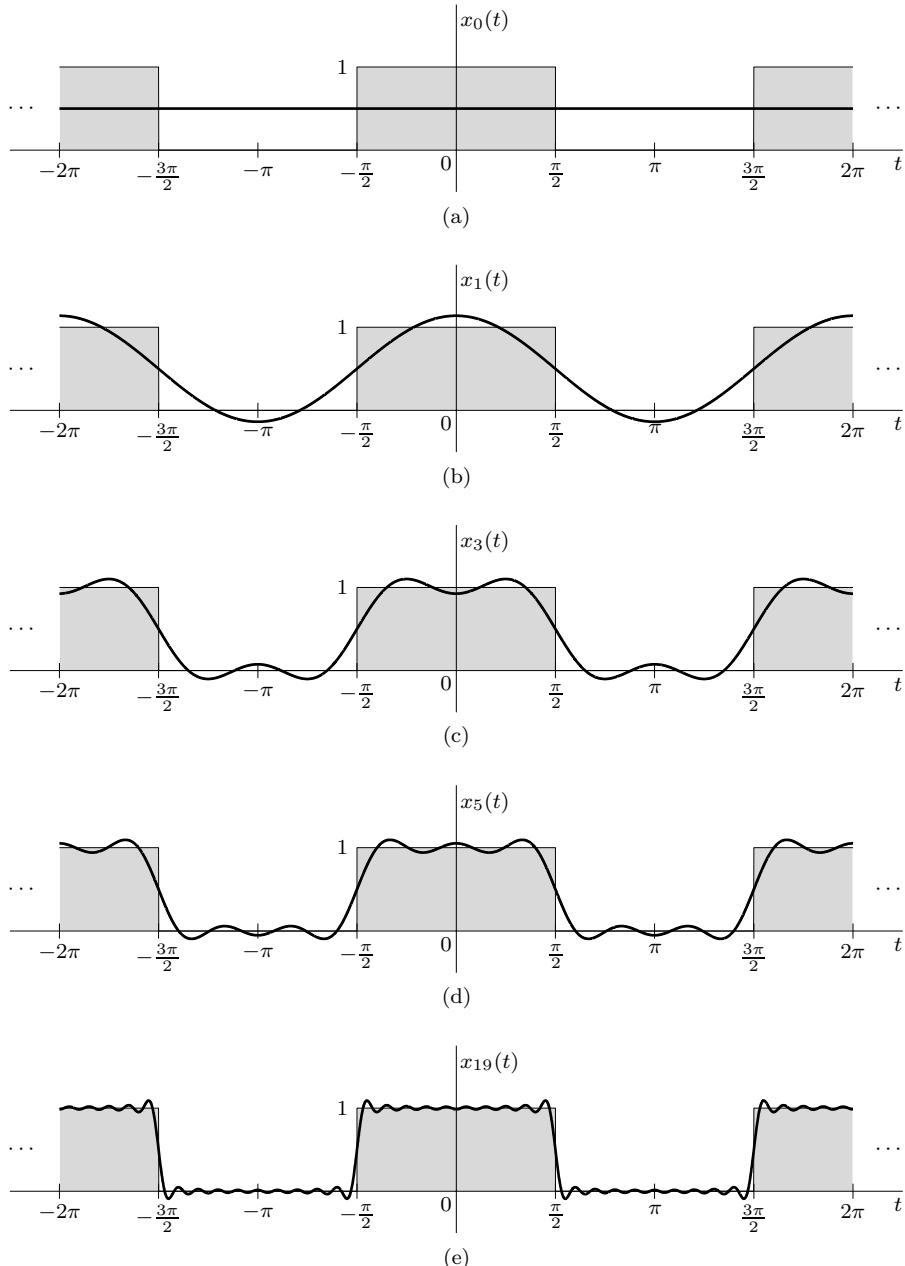


Figure 1.33: Synthesis $x_K(t)$ of a square-pulse periodic signal by successive addition of its harmonics: (a) $K = 0$, (b) $K = 1$, (c) $K = 3$, (d) $K = 5$, and (e) $K = 19$.

▷ **Example 1.8 (Signal Synthesis Using a Truncated Fourier Series)**

Use Eq. (1.61) to synthesize $K = 5$ and $K = 20$ truncated Fourier series approximations of the signal $x(t)$ shown in Fig. 1.26. Plot the results over $(-1.5\pi \leq t \leq 1.5\pi)$.

Using the analysis results of Ex. 1.5, MATLAB provides a direct means to compute and plot $x_5(t)$.

```

01 omega0 = 2; K = 5; t = linspace(-2.5*pi,2.5*pi,10001);
02 Xk = @(k) (2/pi)*(1-exp(-pi/2))./(1+1j*4*k); xK = zeros(size(t));
03 for k = -K:K,
04     xK = xK+Xk(k)*exp(11j*k*omega0*t);
05 end
06 plot(t,xK); xlabel('t'); ylabel('x_{5}(t)');

```

The $x_{20}(t)$ case is readily obtained by changing $K = 5$ to $K = 20$ in line 01. Figure 1.34 presents the results along with the original signal $x(t)$ for comparison (shaded). Regardless whether K is large or small, notice the persistent presence of the Gibbs phenomenon at all points of discontinuity.

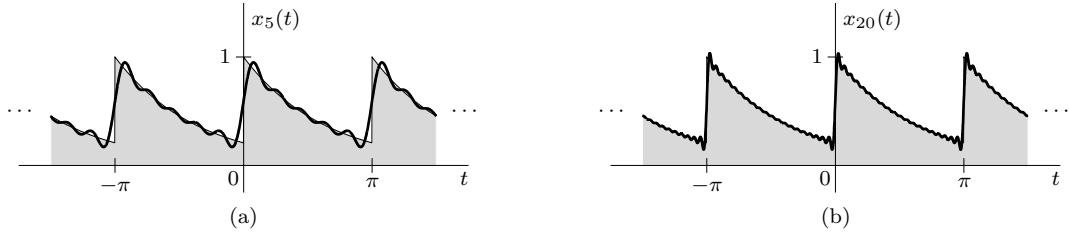


Figure 1.34: Synthesis of $x(t)$ using the truncated Fourier series: (a) $x_5(t)$ and (b) $x_{20}(t)$.

Example 1.8 \triangleleft

1.8 The Fourier Transform

A Fourier series represents a periodic signal as a sum of everlasting exponentials (or sinusoids). By applying a limiting process, we now extend this spectral representation to express an aperiodic signal as a continuous sum (integral) of everlasting exponentials (or sinusoids). This can be done by considering an aperiodic signal $x(t)$ as a periodic signal with period $T_0 \rightarrow \infty$. Since $x(t)$ repeats at infinite intervals ($T_0 \rightarrow \infty$), its fundamental frequency $\omega_0 = \frac{2\pi}{T_0} \rightarrow 0$, and the spectrum becomes continuous. The infinite sum, the Fourier series, reduces to an integral, the Fourier integral. Thus, we can show (see [1]) that an aperiodic signal $x(t)$ can be expressed as a Fourier integral (rather than a Fourier series) as

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega, \quad (1.74)$$

where

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt. \quad (1.75)$$

The equality in Eq. (1.74) is not in the ordinary sense but in the mean square sense. Thus, the right-hand side converges to $x(t)$ in the mean.[†] Equation (1.74), which is just Eq. (1.55) with $s = j\omega$, is the Fourier transform *synthesis equation*, and Eq. (1.75) is the Fourier transform *analysis equation*. Note that *the transform $X(\omega)$ is the frequency-domain specification of $x(t)$* .

We call $X(\omega)$ the *direct Fourier transform* of $x(t)$, or just the *Fourier transform* of $x(t)$, while $x(t)$ is the *inverse Fourier transform* of $X(\omega)$. Symbolically, these statements are expressed as

$$X(\omega) = \mathcal{F}\{x(t)\} \quad \text{and} \quad x(t) = \mathcal{F}^{-1}\{X(\omega)\}.$$

[†]This implies that

$$\lim_{W \rightarrow \infty} \int_{-\infty}^{\infty} \left| x(t) - \frac{1}{2\pi} \int_{-W}^W X(\omega) e^{j\omega t} d\omega \right|^2 dt = 0.$$

The transform relationship between $x(t)$ and $X(\omega)$ is represented in pair-wise form as

$$x(t) \iff X(\omega).$$

It is helpful to keep in mind that the Fourier integral in Eq. (1.74) is basically a Fourier series with fundamental frequency $\Delta\omega$ approaching zero. Therefore, most of the discussion and properties of the Fourier series apply to the Fourier transform and vice versa.

Magnitude and Phase Spectra

The spectrum $X(\omega)$ is generally complex and is represented in polar form as

$$X(\omega) = |X(\omega)|e^{j\angle X(\omega)}. \quad (1.76)$$

In Eq. (1.76), the quantities $|X(\omega)|$ and $\angle X(\omega)$ are both real functions of ω and represent the magnitude spectrum and the phase spectrum, respectively.

Equation (1.75) shows that for real $x(t)$, $X(\omega)$ is conjugate symmetric, that is, $X(\omega) = X^*(-\omega)$. Thus, for real $x(t)$,

$$|X(\omega)| = |X(-\omega)| \quad \text{and} \quad \angle X(\omega) = -\angle X(-\omega). \quad (1.77)$$

In other words, a real signal $x(t)$ has an even magnitude spectrum $|X(\omega)|$ and an odd phase spectrum $\angle X(\omega)$. Unsurprisingly, the result in Eq. (1.77) is identical to that for the Fourier series, as seen in Eq. (1.59).

▷ Example 1.9 (Fourier Transform of a Causal Exponential)

Assuming a is real, find the Fourier transform of the causal exponential $x(t) = e^{-at}u(t)$, an example of which is shown in Fig. 1.35, and plot the corresponding magnitude and phase spectra.

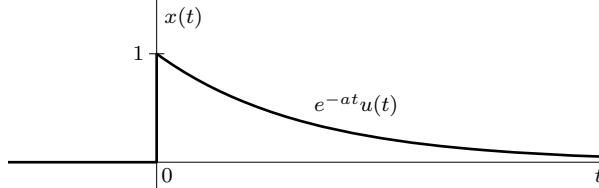


Figure 1.35: A causal exponential $x(t) = e^{-at}u(t)$.

Using Eq. (1.75),

$$X(\omega) = \int_{-\infty}^{\infty} e^{-at}u(t)e^{-j\omega t} dt = \int_0^{\infty} e^{-(a+j\omega)t} dt = \frac{-1}{a+j\omega} e^{-(a+j\omega)t} \Big|_0^{\infty}.$$

Since $|e^{-j\omega t}| = 1$, $e^{-(a+j\omega)t} = e^{-at}e^{-j\omega t} = 0$ as $t \rightarrow \infty$ as long as $a > 0$. Therefore,

$$X(\omega) = \frac{1}{a+j\omega} \quad \text{if } a > 0.$$

If $a < 0$, then $X(\omega)$ does not converge.

Expressing $a + j\omega$ in polar form as $\sqrt{a^2 + \omega^2} e^{j\tan^{-1}(\frac{\omega}{a})}$, we obtain, for $a > 0$,

$$X(\omega) = \frac{1}{\sqrt{a^2 + \omega^2}} e^{-j\tan^{-1}(\frac{\omega}{a})}.$$

Consequently,

$$|X(\omega)| = \frac{1}{\sqrt{a^2 + \omega^2}} \quad \text{and} \quad \angle X(\omega) = -\tan^{-1}\left(\frac{\omega}{a}\right).$$

The magnitude spectrum $|X(\omega)|$ and the phase spectrum $\angle X(\omega)$ are depicted in Fig. 1.36. Observe that $|X(\omega)|$ is an even function of ω , and $\angle X(\omega)$ is an odd function of ω , as expected.

Setting $a = 1$ for convenience, these plots are confirmed using MATLAB.

```
01 omega = linspace(-10,10,1001); a = 1; X = 1./(a+1j*omega);
02 subplot(121); plot(omega,abs(X)); xlabel('omega'); ylabel('|X(\omega)|');
03 subplot(122); plot(omega,angle(X)); xlabel('omega'); ylabel('angle X(\omega)');
```

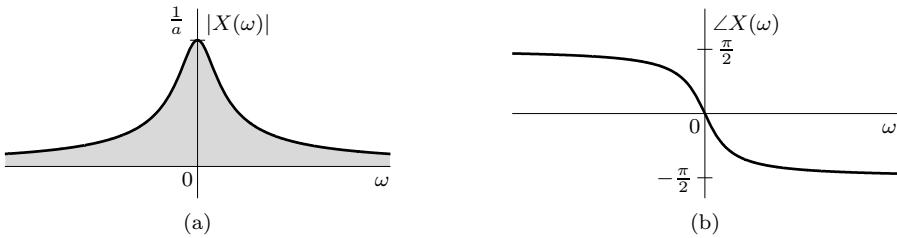


Figure 1.36: Fourier spectra for $x(t) = e^{-at}u(t)$ with $a > 0$: (a) $|X(\omega)|$ and (b) $\angle X(\omega)$.

Example 1.9 ◀

Existence of the Fourier Transform

In Ex. 1.9 we observed that when $a < 0$, the Fourier integral for $e^{-at}u(t)$ does not converge anywhere. Hence, the Fourier transform for $e^{-at}u(t)$ does not exist if $a < 0$ (growing exponential). Clearly, not all signals are Fourier-transformable. One can show that if the energy of $x(t)$ is finite, that is, if

$$\int_{-\infty}^{\infty} |x(t)|^2 dt < \infty, \quad (1.78)$$

then the existence of $X(\omega)$ is guaranteed.

Just as in the case of Fourier series, we also have an alternate set of conditions, the Dirichlet conditions, for the existence of the Fourier transform $X(\omega)$. The first of these conditions is

1. $x(t)$ is absolutely integrable, that is,

$$\int_{-\infty}^{\infty} |x(t)| dt < \infty. \quad (1.79)$$

Taking the absolute value of Eq. (1.75), it follows that the existence of $X(\omega)$ is guaranteed ($|X(\omega)| < \infty$) if Eq. (1.79) is satisfied because $|e^{-j\omega t}| = 1$.

For a convergent Fourier transform, in addition to Eq. (1.79), the function $x(t)$ must also satisfy the following conditions:

2. It may have only a finite number of maxima and minima over any finite interval.
3. It may have only a finite number of discontinuities over any finite interval, and each of these discontinuities must be finite.

$x(t)$	$X(\omega)$	
1. $e^{\lambda t}u(t)$	$\frac{1}{j\omega - \lambda}$	$\text{Re}\{\lambda\} < 0$
2. $e^{\lambda t}u(-t)$	$-\frac{1}{j\omega - \lambda}$	$\text{Re}\{\lambda\} > 0$
3. $e^{\lambda t }$	$\frac{-2\lambda}{\omega^2 + \lambda^2}$	$\text{Re}\{\lambda\} < 0$
4. $t^k e^{\lambda t}u(t)$	$\frac{k!}{(j\omega - \lambda)^{k+1}}$	$\text{Re}\{\lambda\} < 0$
5. $e^{-at} \cos(\omega_0 t)u(t)$	$\frac{a+j\omega}{(a+j\omega)^2 + \omega_0^2}$	$a > 0$
6. $e^{-at} \sin(\omega_0 t)u(t)$	$\frac{\omega_0}{(a+j\omega)^2 + \omega_0^2}$	$a > 0$
7. $\Pi\left(\frac{t}{\tau}\right)$	$\tau \text{sinc}\left(\frac{\pi\omega}{2\pi}\right)$	$\tau > 0$
8. $\frac{B}{\pi} \text{sinc}\left(\frac{Bt}{\pi}\right)$	$\Pi\left(\frac{\omega}{2B}\right)$	$B > 0$
9. $\Lambda\left(\frac{t}{\tau}\right)$	$\frac{\tau}{2} \text{sinc}^2\left(\frac{\pi\omega}{4\pi}\right)$	$\tau > 0$
10. $\frac{B}{2\pi} \text{sinc}^2\left(\frac{Bt}{2\pi}\right)$	$\Lambda\left(\frac{\omega}{2B}\right)$	$B > 0$
11. $e^{-t^2/2\sigma^2}$	$\sigma\sqrt{2\pi}e^{-\sigma^2\omega^2/2}$	$\sigma > 0$
12. $\delta(t)$	1	
13. 1	$2\pi\delta(\omega)$	
14. $u(t)$	$\pi\delta(\omega) + \frac{1}{j\omega}$	
15. $\text{sgn}(t)$	$\frac{2}{j\omega}$	
16. $e^{j\omega_0 t}$	$2\pi\delta(\omega - \omega_0)$	
17. $\cos(\omega_0 t)$	$\pi [\delta(\omega - \omega_0) + \delta(\omega + \omega_0)]$	
18. $\sin(\omega_0 t)$	$\frac{\pi}{j} [\delta(\omega - \omega_0) - \delta(\omega + \omega_0)]$	
19. $\cos(\omega_0 t)u(t)$	$\frac{\pi}{2} [\delta(\omega - \omega_0) + \delta(\omega + \omega_0)] + \frac{j\omega}{\omega_0^2 - \omega^2}$	
20. $\sin(\omega_0 t)u(t)$	$\frac{\pi}{2j} [\delta(\omega - \omega_0) - \delta(\omega + \omega_0)] + \frac{\omega_0}{\omega_0^2 - \omega^2}$	
21. $\sum_{k=-\infty}^{\infty} \delta(t - kT)$	$\omega_0 \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_0)$	$\omega_0 = \frac{2\pi}{T}$

Table 1.1: Selected Fourier transform pairs.

When these conditions are satisfied, the Fourier integral on the right-hand side of Eq. (1.74) converges to $x(t)$ at all points where $x(t)$ is continuous and converges to the average of the right-hand and left-hand limits of $x(t)$ at points where $x(t)$ is discontinuous.

We have seen in Ex. 1.9 that the Fourier transform does not exist for an exponentially growing signal, which violates both Eq. (1.78) and Eq. (1.79). On the other hand, the signal $\text{sinc}(t)$ violates Eq. (1.79) but does satisfy condition Eq. (1.78). Hence, its Fourier transform exists.

Both sets of conditions are sufficient, but not necessary, for the existence of the Fourier transform. There are signals that do not satisfy either of the sets yet have Fourier transforms, at least in the generalized sense. The signal $x(t) = u(t)$ is neither absolutely integrable nor has finite energy. The integral in Eq. (1.75) is nonconvergent, and its Fourier transform, which contains a term $\pi\delta(\omega)$ with infinite height (and zero width), exists not in the ordinary sense but only in the generalized sense.

Any signal that can be generated in practice satisfies the Dirichlet conditions and therefore has a Fourier transform. Thus, the physical existence of a signal is a sufficient condition for the existence of its transform. As engineers who are primarily interested in the physical world, we might conclude

that a physical existence criterion will ensure that any signal of our interest has a Fourier transform. Unfortunately, this conclusion is incorrect. Engineers rely heavily on many signals that are physically unrealizable but mathematically useful, such as the previously discussed $\text{sinc}(t)$ function, which has a Fourier transform, and the unit step function, which has a Fourier transform only in the generalized sense. Table 1.1 gives a short table of selected Fourier transform pairs. Pairs 1–11 involve ordinary Fourier transforms, while pairs 12–21 require the use of generalized functions.

▷ **Example 1.10 (Fourier Transform of $\Pi(t/\tau)$)**

Use direct integration to determine the Fourier transform of $x(t) = \Pi(t/\tau)$, which is illustrated in Fig. 1.9b. Plot the magnitude and phase spectra.

By definition,

$$X(\omega) = \int_{-\infty}^{\infty} \Pi(t/\tau) e^{-j\omega t} dt.$$

Since $\Pi(t/\tau) = 1$ for $|t| < \frac{\tau}{2}$ and is zero for $|t| > \frac{\tau}{2}$,

$$\begin{aligned} X(\omega) &= \int_{-\tau/2}^{\tau/2} e^{-j\omega t} dt = -\frac{1}{j\omega} \left(e^{-j\omega\tau/2} - e^{j\omega\tau/2} \right) \\ &= \frac{2 \sin\left(\frac{\omega\tau}{2}\right)}{\omega} = \tau \frac{\sin\left(\pi \frac{\omega\tau}{2\pi}\right)}{\left(\pi \frac{\omega\tau}{2\pi}\right)} = \tau \text{sinc}\left(\frac{\omega\tau}{2\pi}\right). \end{aligned}$$

Therefore,

$$\Pi(t/\tau) \iff \tau \text{sinc}\left(\frac{\omega\tau}{2\pi}\right).$$

This result matches pair 7 of Table 1.1.

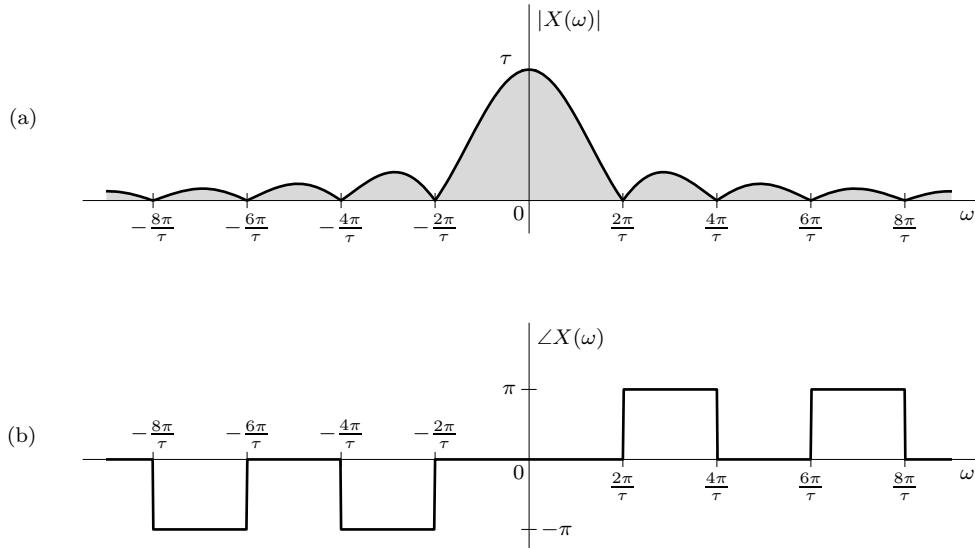


Figure 1.37: Fourier spectra for $x(t) = \Pi(t/\tau)$: (a) $|X(\omega)|$ and (b) $\angle X(\omega)$.

Although $X(\omega)$ is real in this case and can be plotted using a single graph (a simple sinc function), it is good habit to plot the magnitude and phase spectra separately. Recall that $\text{sinc}(k) = 0$ for all nonzero, integer k . Hence, $X(\omega) = \tau \text{sinc}\left(\frac{\omega\tau}{2\pi}\right) = 0$ when $\frac{\omega\tau}{2\pi} = k$ or $\omega = \frac{2\pi k}{\tau}$, as depicted in the Fig. 1.37a plot of the magnitude spectrum. Since $X(\omega)$ is real, these zero-crossings reflect when the

amplitude sign changes (from positive to negative and vice versa); thus the zero-crossings indicate when the phase transitions between 0 and $\pm\pi$, which is verified in the Fig. 1.37b plot of the phase spectrum.[†]

Bandwidth of $\Pi(t/\tau)$

The spectrum $X(\omega)$ in Fig. 1.37a peaks at $\omega = 0$ and decays at higher frequencies. Therefore, $\Pi(t/\tau)$ is a lowpass signal with most of the signal energy in lower frequency components. Strictly speaking, because the spectrum extends from 0 to ∞ , the bandwidth is ∞ . However, much of the spectrum is concentrated within the first lobe of the sinc function, from $\omega = 0$ to $\omega = \frac{2\pi}{\tau}$. Therefore, a rough estimate of the bandwidth of a rectangular pulse of width τ seconds is $\frac{2\pi}{\tau}$ rad/s, or $\frac{1}{\tau}$ Hz. To compute bandwidth, we must consider the spectrum only for positive values of frequency [1]. Note the reciprocal relationship of the pulse width with its bandwidth (spectral width). We shall observe later that this result is true in general.

Example 1.10 \blacktriangleleft

1.9 Fourier Transform Properties

We now cover the important properties of the Fourier transform and their implications as well as applications. The proofs can be found in any text on the Fourier transform (e.g., see [1]). The properties of the Fourier transform are useful not only in deriving the direct and inverse transforms of many functions but also in obtaining several valuable results in signal processing. Before embarking on this topic, however, it is illuminating to observe one very interesting and pervasive aspect of the Fourier transform: time-frequency duality.

Equations (1.74) and (1.75) show an interesting fact: the direct and the inverse transform operations are remarkably similar. These operations, required to go from $x(t)$ to $X(\omega)$ and then from $X(\omega)$ to $x(t)$, are depicted graphically in Fig. 1.38. There are only two minor differences in these

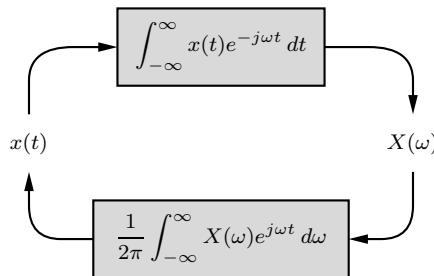


Figure 1.38: A near symmetry between the direct and inverse Fourier transforms.

operations: the factor $1/2\pi$ appears only in the inverse operator, and the exponential indices in the two operations have opposite signs. Otherwise the two operations are symmetrical.[‡] This obser-

[†]The phase spectrum, which is required in this case to be an odd function of ω , may be drawn in several other ways since a negative sign can be represented using phases of $\pi + 2\pi k$ for all integer k . All such representations are equivalent.

[‡]Of the two differences, the former can be eliminated by a change of variable from ω (in radians per second) to f (in hertz). In this case,

$$\omega = 2\pi f \quad \text{and} \quad d\omega = 2\pi df.$$

Therefore, the direct and the inverse transforms are given by

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft} dt \quad \text{and} \quad x(t) = \int_{-\infty}^{\infty} X(f)e^{j2\pi ft} df.$$

This leaves only one significant difference, that of sign change in the exponential index. Otherwise the two operations are symmetrical.

vation has far-reaching consequences in the study of the Fourier transform. It is the basis of the so-called duality of time and frequency. *The duality principle may be compared with a photograph and its negative. A photograph can be obtained from its negative, and by using an identical procedure, the negative can be obtained from the photograph.* For any result or relationship between $x(t)$ and $X(\omega)$, there exists a dual result or relationship that is obtained by interchanging the roles of $x(t)$ and $X(\omega)$. Consider, for example, transform 12 in Table 1.1:

$$\delta(t) \iff 1.$$

The dual of this transform states that

$$1 \iff 2\pi\delta(\omega),$$

which is just pair 13 in Table 1.1. Observe the role reversal of time and frequency in these two equations (as well as the 2π factor that results from choosing the radian frequency ω rather than the hertzian frequency f). The value of the duality principle lies in the fact that *whenever we derive any result, we can be sure that it has a dual*. This possibility can give valuable insights about many unsuspected properties and results in signal processing.

The reader should not fail to observe the ever-present duality in the upcoming discussions. We shall observe that for every property of the Fourier transform, there is a dual property.

1.9.1 Duality Property

The duality property states that

$$\text{if } x(t) \iff X(\omega), \text{ then } X(t) \iff 2\pi x(-\omega). \quad (1.80)$$

Observe that these two relationships are the dual of each other. From this equation, it follows that because

$$\underbrace{\Pi\left(\frac{t}{\tau}\right)}_{x(t)} \iff \underbrace{\tau \operatorname{sinc}\left(\frac{\tau\omega}{2\pi}\right)}_{X(\omega)}, \quad (1.81)$$

then

$$\underbrace{\tau \operatorname{sinc}\left(\frac{\tau t}{2\pi}\right)}_{X(t)} \iff \underbrace{2\pi \Pi\left(-\frac{\omega}{\tau}\right)}_{2\pi x(-\omega)} = 2\pi \Pi\left(\frac{\omega}{\tau}\right). \quad (1.82)$$

Equations (1.81) and (1.82) are found in Table 1.1 as pairs 7 and 8 (with $\tau = 2B$). With the duality property, knowing either pair ensures that we know the other. Figure 1.39 shows these dual pairs graphically.

The last step in Eq. (1.82) follows since $\Pi(\cdot)$ is an even function. This evenness results in a near-perfect symmetry between time and frequency, except for the minor factor of 2π . When such is not the case, the symmetry is not so perfect. For example, $e^{\lambda t}u(t) \iff \frac{1}{j\omega - \lambda}$ (Table 1.1, pair 1) yields the dual $\frac{1}{jt - \lambda} \iff 2\pi e^{-\lambda\omega}u(-\omega)$ (not in Table 1.1). In this case, the duality property's frequency reflection is apparent: the dual of a right-sided time signal involves a left-sided frequency signal.

▷ Drill 1.16 (Applying the Duality Property)

Apply the duality property to pairs 3 and 17 in Table 1.1 to show that

$$(a) \quad \frac{-2\lambda}{t^2 + \lambda^2} \iff 2\pi e^{\lambda|\omega|} \quad (b) \quad \delta(t - t_0) + \delta(t + t_0) \iff 2 \cos(t_0\omega)$$

△

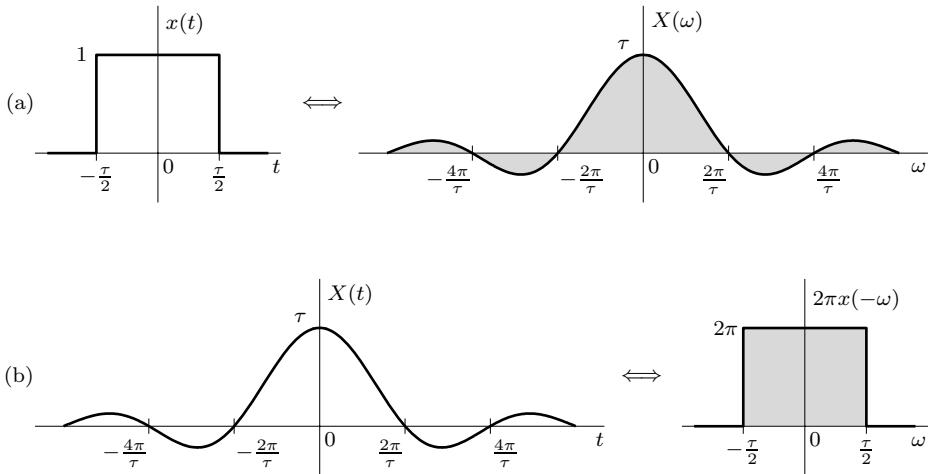


Figure 1.39: Illustrating duality with (a) $x(t) = \Pi(t/\tau) \iff X(\omega) = \tau \text{sinc}(\tau\omega/2\pi)$ and (b) its dual $X(t) = \tau \text{sinc}(\tau t/2) \iff 2\pi x(-\omega) = 2\pi \Pi(\omega/\tau)$.

1.9.2 Linearity Property

The Fourier transform is linear; that is, if

$$x(t) \iff X(\omega) \quad \text{and} \quad y(t) \iff Y(\omega),$$

then, for any constants a and b ,

$$ax(t) + by(t) \iff aX(\omega) + bY(\omega). \quad (1.83)$$

The proof is trivial and follows directly from Eq. (1.75). This result can be extended to any finite number of terms. Interestingly, applying duality to the linearity property is somewhat anticlimactic: the same property results (that a sum of scaled signals transforms into a sum of scaled signals). This will not be the case for many of the upcoming properties.

1.9.3 Complex-Conjugation Property

The complex-conjugation property states that

$$\text{if } x(t) \iff X(\omega), \text{ then } x^*(t) \iff X^*(-\omega). \quad (1.84)$$

As with the linearity property, the proof is trivial and follows directly from Eq. (1.75).

More interesting are some of the observations that arise from the complex-conjugation property:

1. If $x(t)$ is real, then $X(\omega)$ is conjugate symmetric. That is, if $x(t) = x^*(t)$, then $X(\omega) = X^*(-\omega)$.
2. If $x(t)$ is imaginary, then $X(\omega)$ is conjugate antisymmetric. That is, if $x(t) = -x^*(t)$, then $X(\omega) = -X^*(-\omega)$.
3. If $y(t)$ is the real part of some signal $x(t)$, then its transform $Y(\omega)$ is the conjugate-symmetric portion of $X(\omega)$. That is, if $y(t) = \text{Re}\{x(t)\} = \frac{1}{2}x(t) + \frac{1}{2}x^*(t)$, then $Y(\omega) = X_{cs}(\omega) = \frac{1}{2}X(\omega) + \frac{1}{2}X^*(-\omega)$.
4. If $y(t)$ is the imaginary part of some signal $x(t)$ (recall that the imaginary portion of a complex signal is a *real* quantity), then its transform $Y(\omega)$ is $1/j$ times the conjugate-antisymmetric portion of $X(\omega)$. That is, if $y(t) = \text{Im}\{x(t)\} = \frac{1}{2j}x(t) - \frac{1}{2j}x^*(t)$, then $Y(\omega) = \frac{1}{j}X_{ca}(\omega) = \frac{1}{2j}X(\omega) - \frac{1}{2j}X^*(-\omega)$.

These observations provide us better understanding of signals and their transforms. Knowing that a signal is real, for example, provides the powerful observation that the Fourier transform possesses conjugate symmetry; this symmetry, in turn, allows us to completely specify the transform using just half of the total frequencies (such as $\omega > 0$).

1.9.4 Scaling Property

The scaling property states that, for any real constant a ,

$$\text{if } x(t) \iff X(\omega), \text{ then } x(at) \iff \frac{1}{|a|}X\left(\frac{\omega}{a}\right). \quad (1.85)$$

The function $x(at)$ represents the function $x(t)$ compressed in time by a factor a . Similarly, the function $X(\frac{\omega}{a})$ represents the function $X(\omega)$ expanded in frequency by the same factor a . *The scaling property states that the time compression of a signal results in spectral expansion and that the time expansion of a signal results in spectral compression.* Intuitively, compression in time by a factor a means that the signal is varying more rapidly by factor a . To synthesize such a signal, the frequencies of its sinusoidal components must be increased by the same factor a , implying that its frequency spectrum is expanded by the factor a . Similarly, a signal expanded in time varies more slowly; hence the frequencies of its components are lowered, implying that its frequency spectrum is compressed. For instance, the signal $\cos(2\omega_0 t)$ is the same as the signal $\cos(\omega_0 t)$ time compressed by a factor of 2. Clearly, the spectrum of the former (impulses at $\pm 2\omega_0$) is an expanded version of the spectrum of the latter (impulses at $\pm \omega_0$). The effect of scaling is demonstrated by another example in Fig. 1.40. Note the time-frequency duality implicit in this property. More discussion of this interesting topic may be found in the literature [4].

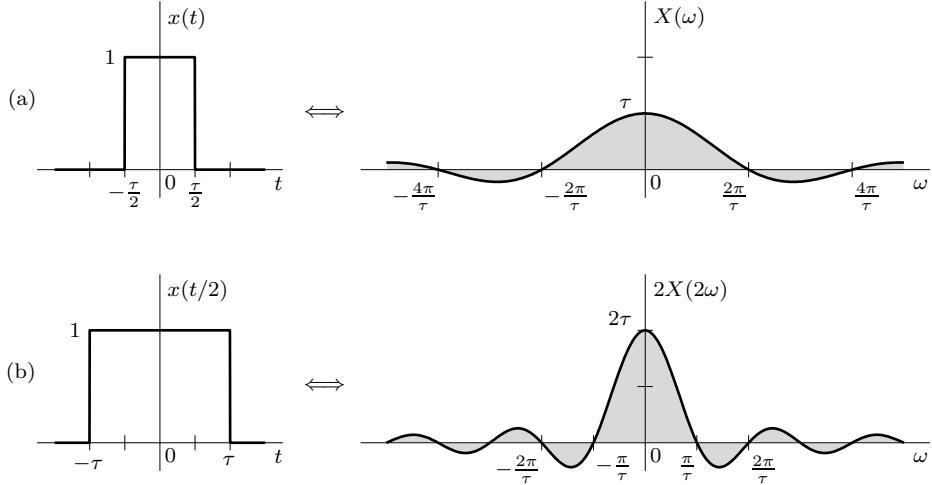


Figure 1.40: Illustrating the scaling property with (a) $x(t) = \Pi(t/\tau) \iff X(\omega) = \tau \text{sinc}(\tau\omega/2\pi)$ and (b) its scale $x(t/2) = \Pi(t/2\tau) \iff 2X(2\omega) = 2\tau \text{sinc}(2\omega\tau/\pi)$.

Time-Reversal Property

By letting $a = -1$ in Eq. (1.85), we obtain the *time-reversal property*:

$$\text{if } x(t) \iff X(\omega), \text{ then } x(-t) \iff X(-\omega). \quad (1.86)$$

As Eq. (1.86) makes clear, the time-reversal property also serves as a *frequency-reversal property*.

As with the complex-conjugation property, the time-reversal property leads to some interesting observations:

1. If $x(t)$ is even, then $X(\omega)$ is even. That is, if $x(t) = x(-t)$, then $X(\omega) = X(-\omega)$.
2. If $x(t)$ is odd, then $X(\omega)$ is odd. That is, if $x(t) = -x(-t)$, then $X(\omega) = -X(-\omega)$.
3. If $y(t)$ is the even part of some signal $x(t)$, then its transform $Y(\omega)$ is the even portion of $X(\omega)$. That is, if $y(t) = x_e(t) = \frac{1}{2}x(t) + \frac{1}{2}x(-t)$, then $Y(\omega) = X_e(\omega) = \frac{1}{2}X(\omega) + \frac{1}{2}X(-\omega)$.
4. If $y(t)$ is the odd part of some signal $x(t)$, then its transform $Y(\omega)$ is the odd portion of $X(\omega)$. That is, if $y(t) = x_o(t) = \frac{1}{2}x(t) - \frac{1}{2}x(-t)$, then $Y(\omega) = X_o(\omega) = \frac{1}{2}X(\omega) - \frac{1}{2}X(-\omega)$.

▷ Drill 1.17 (Combined Complex Conjugation and Time Reversal)

Use the combined properties of complex conjugation and time reversal to show the following *duals* of the complex-conjugation properties:

- (a) If $x(t)$ is conjugate symmetric, then $X(\omega)$ is real.
- (b) If $x(t)$ is conjugate antisymmetric, then $X(\omega)$ is imaginary.
- (c) If $y(t)$ is the conjugate-symmetric portion of $x(t)$, then $Y(\omega)$ is the real portion of $X(\omega)$.
- (d) If $y(t)$ is the conjugate-antisymmetric portion of $x(t)$, then $Y(\omega) = j\text{Im}\{X(\omega)\}$.

△

1.9.5 Time-Shifting Property

The time-shifting property states that

$$\text{if } x(t) \iff X(\omega), \text{ then } x(t - t_0) \iff X(\omega)e^{-j\omega t_0}. \quad (1.87)$$

This result shows that *delaying a signal by t_0 seconds does not change its magnitude spectrum. The phase spectrum, however, is changed by $-\omega t_0$* .

▷ Example 1.11 (A Time Shift Imparts a Linear Phase Shift)

Using real $a > 0$, find the Fourier transform of $x(t) = e^{-a|t-t_0|}$, shown in Fig. 1.41. Using $a = 2$ and $t_0 = 1$, sketch the corresponding magnitude and phase spectra.

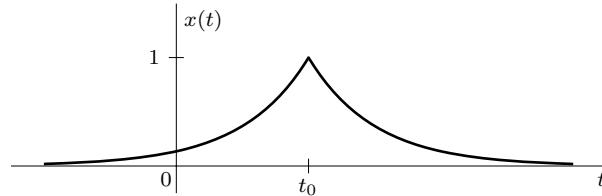


Figure 1.41: Time-shifted exponential $x(t) = e^{-a|t-t_0|}$.

This function is a time-shifted version of $e^{-a|t|}$. Using pair 3 in Table 1.1 (with $\lambda = -a$) and Eq. (1.87), we obtain

$$e^{-a|t-t_0|} \iff \frac{2a}{\omega^2 + a^2} e^{-j\omega t_0}.$$

Spectral plots are obtained using MATLAB.

```

01 omega = linspace(-10,10,1001); X = @ (omega) 4 ./ (omega.^2 + 4) .* exp (-1j * omega);
02 subplot(121); plot(omega, abs(X(omega))); xlabel(' \omega'); ylabel(' |X(\omega)| ');
03 subplot(122); plot(omega, angle(X(omega))); xlabel(' \omega'); ylabel(' \angle X(\omega)');

```

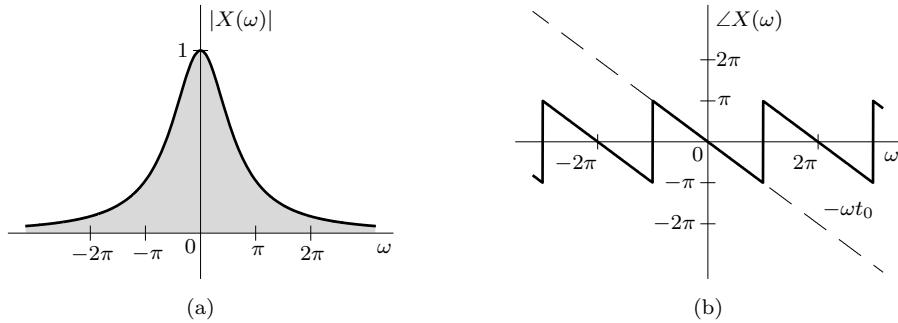


Figure 1.42: Spectra for $x(t) = e^{-a|t-t_0|}$ using $a = 2$ and $t_0 = 1$: (a) $|X(\omega)|$ and (b) $\angle X(\omega)$.

The spectrum of $e^{-a|t-t_0|}$, shown in Fig. 1.42, is the same as that of $e^{-a|t|}$ except for an added linear phase shift $-\omega t_0$ that results as a direct consequence of the time delay t_0 . As with most computer-calculated phases, the phase spectrum (solid line) in Fig. 1.42b assumes the principal value of the angle rather than simply $-\omega t_0$ (dashed line). The principal value differs from the actual value by $\pm 2\pi k$ radians (integer k) in such a way as to ensure that the principal value remains within the range $\pm\pi$. Thus, the principal value exhibits jump discontinuities of $\pm 2\pi$ whenever the original phase goes beyond the range $\pm\pi$. The two representations, however, are completely equivalent. As Ex. 1.12 will show, however, principal value plots are not always the most illuminating.

Example 1.11 ◁

▷ Example 1.12 (Unwrapping Phase for Clarity)

Find the Fourier transform of the shifted gate pulse $x(t) = \Pi\left(\frac{t}{\tau} - \frac{3}{4}\right)$, shown in Fig. 1.43. Plot the corresponding magnitude and phase spectra.

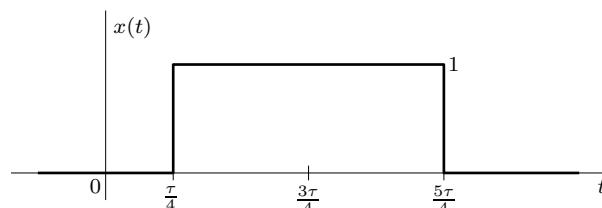


Figure 1.43: Time-shifted gate $x(t) = \Pi\left(\frac{t}{\tau} - \frac{3}{4}\right)$.

The pulse $x(t)$ is the gate pulse $\Pi(t/\tau)$ in Fig. 1.9b delayed by $t_0 = 3\tau/4$ seconds, that is, $x(t) = \Pi(\frac{t-0.75\tau}{\tau})$. Hence, according to Eq. (1.87), its Fourier transform is the Fourier transform of $\Pi(t/\tau)$ multiplied by $e^{-j\omega(\frac{3\tau}{4})}$. Therefore,

$$X(\omega) = \tau \text{sinc} \left(\frac{\omega\tau}{2\pi} \right) e^{-j\omega \frac{3\tau}{4}}.$$

The magnitude spectrum $|\tau \text{sinc}(\frac{\omega\tau}{2})|$, shown in Fig. 1.44a, is identical to the magnitude spectrum

for the unshifted rectangular pulse, as shown in Fig. 1.37a. Figure 1.44b shows the phase using principal values. Setting $\tau = 1$ for convenience, MATLAB easily computes and plots these spectra.

```
01 omega = linspace(-8.5*pi,8.5*pi,10001); X = sinc(omega/(2*pi)).*exp(-1j*omega*3/4);
02 subplot(311); plot(omega,abs(X)); xlabel('\omega'); ylabel('|X(\omega)|');
03 subplot(312); plot(omega,angle(X)); xlabel('\omega'); ylabel('angle X(\omega)');
```

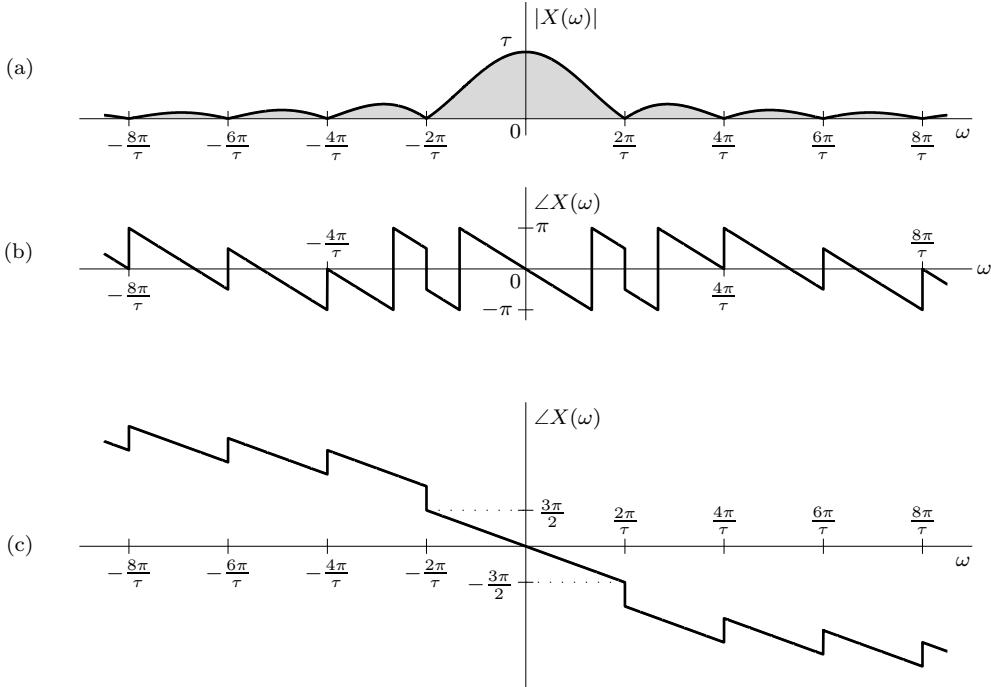


Figure 1.44: Spectra for $x(t) = \Pi\left(\frac{t}{\tau} - \frac{3}{4}\right)$: (a) $|X(\omega)|$, (b) $\angle X(\omega)$, and (c) unwrapped $\angle X(\omega)$.

Although the phase plot of Fig. 1.44b appears as the aimless frolick of a playful child, its madness has a method. The jumps of $\pm\pi$ at $\omega = 2\pi k / \tau$ (integer k) occur due to the sign alternations of the sinc function at those frequencies. The remaining jumps of $\pm 2\pi$ occur to keep the phase in the principal value range of $\pm\pi$.

Figure 1.44c shows a more informative presentation of the phase spectrum that is obtained by *unwrapping* the phase response found in Fig. 1.44b. The unwrapping process essentially removes the jumps of $\pm 2\pi$ used to keep angle values in the principal range. Now, the linear component of phase that is due to the time shift is clearly visible, only to be broken by occasional jumps of $\pm\pi$ resulting from alternations in sign of the sinc function. The time delay itself is easily ascertained by computing the slope near the origin as $t_0 = \frac{-3\pi}{4\pi/\tau} = -\frac{3\tau}{4}$. MATLAB's built-in `unwrap` command simplifies the unwrapping process.

```
04 Xunwrapang = unwrap(angle(X),1.5*pi);
05 Xunwrapang = Xunwrapang-(Xunwrapang(1)+Xunwrapang(end))/2;
06 subplot(313); plot(omega,Xunwrapang); xlabel('\omega'); ylabel('angle X(\omega)');
```

Line 05, while not absolutely necessary, compensates for the fact that MATLAB unwraps vectors from one side to the other, rather than from the middle out, and ensures that our phase plot has the desired odd symmetry.

Example 1.12 ◀

A Physical Explanation of Linear Phase

The time-shifting property tells us that a time delay in a signal causes a linear phase shift in its spectrum. This result can also be derived by heuristic reasoning. Imagine $x(t)$ being synthesized by its Fourier components, which are sinusoids of certain magnitudes and phases. The delayed signal $x(t - t_0)$ can be synthesized by the same sinusoidal components, each delayed by t_0 seconds. The magnitudes of the components remain unchanged. Therefore, the magnitude spectrum of $x(t - t_0)$ is identical to that of $x(t)$. The time delay of t_0 in each sinusoid, however, does change the phase of each component. Now, a sinusoid $\sin(\omega t)$ delayed by t_0 is given by

$$\sin(\omega[t - t_0]) = \sin(\omega t) \underbrace{-\omega t_0}_{\text{phase shift}}.$$

Therefore, a time delay t_0 in a sinusoid of frequency ω manifests as a phase delay of ωt_0 (or phase shift of $-\omega t_0$). This is a linear function of ω , meaning that higher-frequency components must undergo proportionately higher phase shifts to achieve the same time delay. This effect is depicted in Fig. 1.45 with two sinusoids, the frequency of the second sinusoid being twice that of the first. The same time delay t_0 amounts to a phase shift of $-\pi/2$ in the first sinusoid and a phase shift of $-\pi$ in the second sinusoid. This verifies the fact that *to achieve the same time delay, higher-frequency sinusoids must undergo proportionately higher phase shifts*. The concept of a linear phase shift is very important, and we shall encounter it again when discussing distortionless signal transmission and filtering applications.

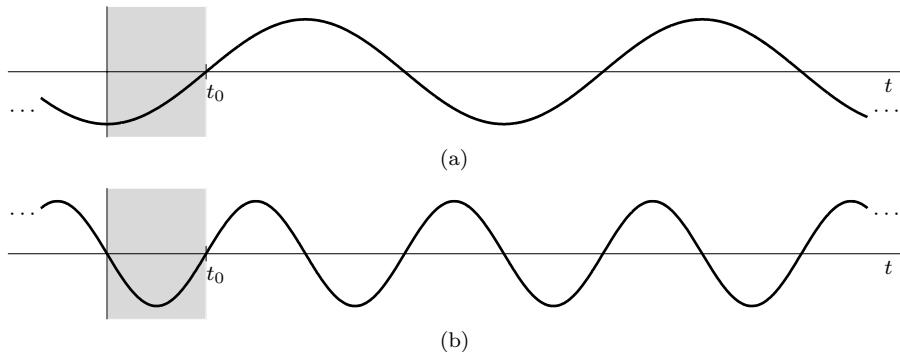


Figure 1.45: Illustrating linear phase: (a) the phase shift in $\sin(\omega[t - t_0])$ caused by delay t_0 is half as large as (b) the phase shift in $\sin(2\omega[t - t_0])$ caused by the same delay t_0 .

▷ Drill 1.18 (Using the Time-Shifting Property)

Using the time-shifting property and pair 8 in Table 1.1, show that

$$\operatorname{sinc}(\omega_0[t - t_0]) \iff \frac{1}{\omega_0} \Pi\left(\frac{\omega}{2\pi\omega_0}\right) e^{-j\omega t_0}.$$

Sketch the corresponding magnitude and phase spectra.

△

Frequency-Shifting Property

The frequency-shifting property, which is the dual of time-shifting property, states that

$$\text{if } x(t) \iff X(\omega), \text{ then } x(t)e^{j\omega_0 t} \iff X(\omega - \omega_0). \quad (1.88)$$

According to this property, the multiplication of a signal by a factor $e^{j\omega_0 t}$ shifts the spectrum of that signal by ω_0 . Changing ω_0 to $-\omega_0$ in Eq. (1.88) yields

$$x(t)e^{-j\omega_0 t} \iff X(\omega + \omega_0). \quad (1.89)$$

In most CT systems, frequency shifting is not achieved by multiplying signal $x(t)$ by the complex signal $e^{j\omega_0 t}$ (or $e^{-j\omega_0 t}$). Rather, $x(t)$ is typically multiplied by a real sinusoid as

$$x(t) \cos(\omega_0 t) = \frac{1}{2} (x(t)e^{j\omega_0 t} + x(t)e^{-j\omega_0 t}).$$

From Eqs. (1.88) and (1.89), it follows that

$$x(t) \cos(\omega_0 t) \iff \frac{1}{2} [X(\omega - \omega_0) + X(\omega + \omega_0)]. \quad (1.90)$$

This result, known as the *modulation property*, shows that the multiplication of a signal $x(t)$ by a sinusoid of frequency ω_0 shifts the spectrum $X(\omega)$ by $\pm\omega_0$, as depicted in Fig. 1.46.

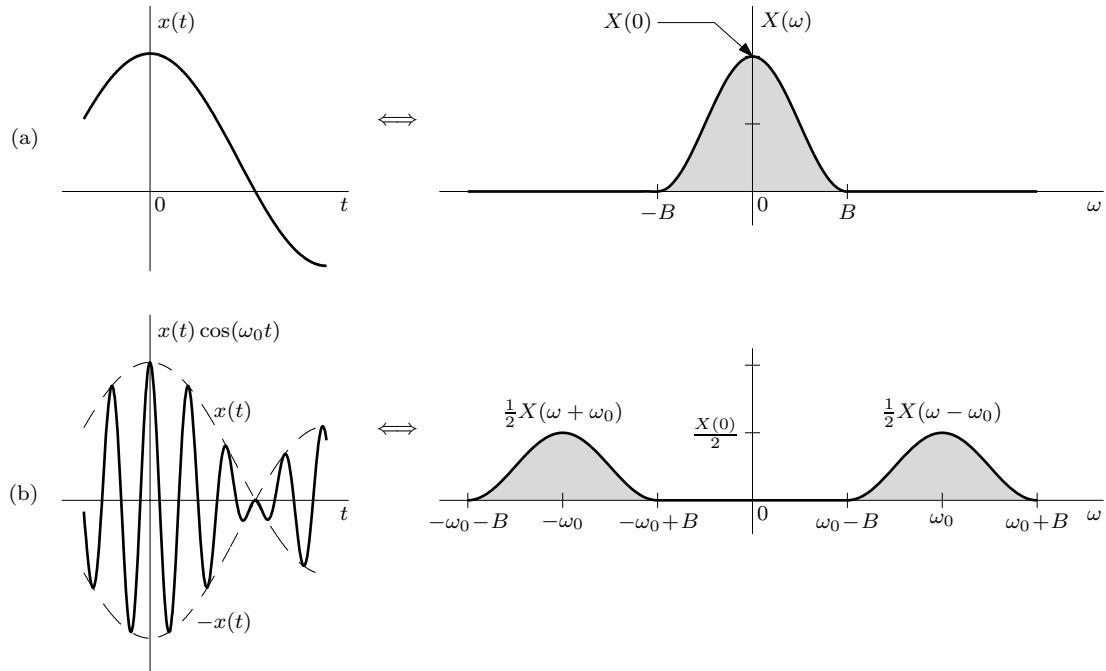


Figure 1.46: Multiplication by a sinusoid causes spectral shifting: (a) $x(t) \iff X(\omega)$ and (b) $x(t) \cos(\omega_0 t) \iff \frac{1}{2} [X(\omega - \omega_0) + X(\omega + \omega_0)]$.

Multiplication of a sinusoid $\cos(\omega_0 t)$ by $x(t)$ amounts to modulating the sinusoid's amplitude. This type of modulation is known as *amplitude modulation*. As long as ω_0 is sufficiently large, as is normally the case, notice that $\pm x(t)$ serves as the envelope of the sinusoid $\cos(\omega_0 t)$. The sinusoid, in this case $\cos(\omega_0 t)$, is called the *carrier*, the signal $x(t)$ is the *modulating signal*, and the signal $x(t) \cos(\omega_0 t)$ is the *modulated signal*. Figure 1.46 shows that the signal $x(t)$ has bandwidth B , but the bandwidth of the modulated signal is $2B$. Thus, modulation doubles the bandwidth.

Effecting a Constant Phase Shift in a Modulated Spectrum

From Eqs. (1.88) and (1.89), it follows that

$$x(t)e^{j(\omega_0 t+\theta)} \iff X(\omega - \omega_0)e^{j\theta} \quad \text{and} \quad x(t)e^{-j(\omega_0 t+\theta)} \iff X(\omega + \omega_0)e^{-j\theta}.$$

Addition of these two equation yields

$$x(t) \cos(\omega_0 t + \theta) \iff \frac{1}{2} (X(\omega - \omega_0)e^{j\theta} + X(\omega + \omega_0)e^{-j\theta}). \quad (1.91)$$

Comparison of Eq. (1.91) with Eq. (1.90) shows that merely by shifting the carrier phase by a constant θ , we can shift the phase of the entire modulated spectrum by θ . The right-hand side of Eq. (1.91) shows that the entire spectrum $X(\omega - \omega_0)$ (centered at ω_0) acquires a constant phase θ . Similarly, the entire spectrum $X(\omega + \omega_0)$ (centered at $-\omega_0$) acquires a constant phase $-\theta$. We shall later see how this principle allows us to relax the condition for linear phase characteristics in bandpass systems.

▷ Drill 1.19 (Using the Modulation Property)

Sketch the signal $x(t) = \Pi\left(\frac{t}{4\pi}\right)$ and the corresponding modulated signal $x(t) \cos(5t)$. Show that

$$\Pi\left(\frac{t}{4\pi}\right) \cos(5t) \iff 2\pi \text{sinc}(2[\omega - 5]) + 2\pi \text{sinc}(2[\omega + 5]).$$

△

1.9.6 Time-Differentiation and Time-Integration Properties

The time-differentiation property states that[†]

$$\text{if } x(t) \iff X(\omega), \text{ then } \frac{d}{dt}x(t) \iff j\omega X(\omega).$$

A generalization of the time-differentiation property states that

$$\text{if } x(t) \iff X(\omega), \text{ then } \frac{d^k}{dt^k}x(t) \iff (j\omega)^k X(\omega). \quad (1.92)$$

The time-differentiation property is easily proven by directly differentiating Eq. (1.74).

Similar to the time-differentiation property, the time-integration property states that

$$\text{if } x(t) \iff X(\omega), \text{ then } \int_{-\infty}^t x(\tau)d\tau \iff \frac{X(\omega)}{j\omega} + \pi X(0)\delta(\omega). \quad (1.93)$$

The proof of this property is considered in Drill 1.20.

Frequency-Differentiation Property

The dual of the time-differentiation property is the frequency-differentiation property, which states that

$$\text{if } x(t) \iff X(\omega), \text{ then } (-jt)^k x(t) \iff \frac{d^k}{d\omega^k} X(\omega). \quad (1.94)$$

The frequency-differentiation property is easily proven by differentiating Eq. (1.75) with respect to ω .

1.9.7 Time-Domain Convolution Property

The time-domain convolution property states that if

$$x(t) \iff X(\omega) \quad \text{and} \quad y(t) \iff Y(\omega),$$

[†]This property is valid only if the transform of $\frac{d}{dt}x(t)$ exists.

then

$$x(t) * y(t) \iff X(\omega)Y(\omega). \quad (1.95)$$

The convolution property allows us to replace a cumbersome time-domain convolution operation with a simple frequency-domain multiplication. The proof can be found in virtually any introductory signals and systems text, such as [1].

The utility of this apparently trivial result is profound, and it forms the basis for the frequency-domain analysis of LTIC systems. As we shall see in Ch. 2, the frequency domain provides an intuitive filtering perspective of system behavior and greatly aids in our ability to understand, analyze, and design systems.

▷ Drill 1.20 (Proving the Time-Integration Property with Convolution)

Use pair 14 in Table 1.1, the convolution property of Eq. (1.95), and the fact that $x(t) * u(t) = \int_{-\infty}^t x(\tau) d\tau$ to prove the time-integration property given in Eq. (1.93).

△

▷ Drill 1.21 (Using the Time-Domain Convolution Property)

Using the time-domain convolution property, show that for $\text{Re}\{\lambda_1\} < 0$ and $\text{Re}\{\lambda_2\} < 0$,

$$e^{\lambda_1 t} u(t) * e^{\lambda_2 t} u(t) = \frac{e^{\lambda_1 t} - e^{\lambda_2 t}}{\lambda_1 - \lambda_2} u(t).$$

Hint: Use the convolution property of Eq. (1.95) to find the Fourier transform of $e^{\lambda_1 t} u(t) * e^{\lambda_2 t} u(t)$, and then use a partial fraction expansion to find its inverse Fourier transform.

△

Frequency-Domain Convolution Property

The dual of the time-domain convolution property is the frequency-domain convolution property, which states that if

$$x(t) \iff X(\omega) \quad \text{and} \quad y(t) \iff Y(\omega),$$

then

$$x(t)y(t) \iff \frac{1}{2\pi} X(\omega) * Y(\omega). \quad (1.96)$$

Notice that the factor of $1/2\pi$ occurs as a result of using the *radian* frequency variable ω . If the transforms are expressed using the *hertzian* frequency variable f , then the property simplifies to $x(t)y(t) \iff X(f) * Y(f)$. Referencing the time-domain side of the relationship, Eq. (1.96) is sometimes called the *multiplication property* of the Fourier transform.

▷ Drill 1.22 (Using the Frequency-Domain Convolution Property)

Using the frequency-domain convolution property, show that

$$\text{sinc}(\tau_1\omega) * \text{sinc}(\tau_2\omega) = \begin{cases} \text{sinc}(\tau_1\omega)/\tau_2 & \tau_1 < \tau_2 \\ \text{sinc}(\tau_2\omega)/\tau_1 & \tau_1 > \tau_2 \end{cases}.$$

△

1.9.8 Correlation and the Correlation Property

The correlation between CT functions $x(\cdot)$ and $y(\cdot)$ is defined as

$$\rho_{x,y}(\tau) = \int_{-\infty}^{\infty} x(t + \tau)y^*(t) dt. \quad (1.97)$$

Correlation provides a measure of similarity between two signals as a function of the time lag τ between them. The more similar two signals are at a particular shift τ , the larger the corresponding correlation value becomes.[†] Correlation finds common use in engineering systems. For example, global positioning system (GPS) receivers utilize correlation. The receiver correlates known GPS satellite codes stored in memory with the signal received by the receiver's antenna. Large correlation values indicate that a particular satellite is in the receiver's field of view; the corresponding lag values for each detected satellite are used to triangulate the receiver's position using known satellite almanac data.

When the signals $x(t)$ and $y(t)$ are distinct from one another, the correlation function $\rho_{x,y}(\tau)$ is often termed the *cross-correlation function* between $x(t)$ and $y(t)$. If $y(t) = x(t)$, then the correlation function $\rho_{x,x}(\tau)$ is termed the *autocorrelation function*.

If the correlation function in Eq. (1.97) looks familiar, it should. It is nearly identical to the convolution operation given in Eq. (1.41). Only a little effort is required to demonstrate that

$$\rho_{x,y}(\tau) = x(\tau) * y^*(-\tau). \quad (1.98)$$

Equation (1.98) tells us that correlation is no more difficult, and no more easy, than convolution. Applying the complex-conjugation, time-reversal, and convolution properties yields the correlation property

$$\rho_{x,y}(\tau) = x(\tau) * y^*(-\tau) \iff X(\omega)Y^*(\omega). \quad (1.99)$$

▷ Drill 1.23 (Correlation Order Is Important)

Verify that the order of variables is important in the cross-correlation function by showing that

$$\rho_{x,y}(\tau) = \rho_{y,x}^*(-\tau) \neq \rho_{y,x}(\tau).$$

△

A Frequency-Domain Representation of Signal Energy

The autocorrelation $\rho_{x,x}(\tau)$ of a signal $x(t)$ at $\tau = 0$ should be large. After all, not much should be more similar to a signal $x(t)$ than an exact copy of itself. A reasonable question, still, is how to interpret the corresponding autocorrelation value $\rho_{x,x}(0)$. Simple substitution of $\tau = 0$ into Eq. (1.97) provides

$$\rho_{x,x}(0) = \int_{-\infty}^{\infty} x(t)x^*(t) dt = \int_{-\infty}^{\infty} |x(t)|^2 dt = E_x. \quad (1.100)$$

In other words, the autocorrelation function indicates strength of similarity in a manner consistent with the way we measure signal strength, namely, signal energy. Much like a correlation coefficient, the correlation function is often normalized by signal energy.

Additional insight is obtained by noting that since $\rho_{x,x}(\tau)$ has Fourier transform $X(\omega)X^*(\omega)$, it can be synthesized using Eq. (1.74) as

$$\rho_{x,x}(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)X^*(\omega)e^{j\omega\tau} d\omega = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(\omega)|^2 e^{j\omega\tau} d\omega.$$

[†]For further discussion of the physical significance of correlation and intuition on why it serves as a measure of similarity between signals, refer to [2].

Substituting $\tau = 0$ and combining with Eq. (1.100), we obtain

$$E_x = \int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(\omega)|^2 d\omega. \quad (1.101)$$

This statement is the well-known *Parseval's theorem* for the Fourier transform. This result allows us to determine the signal energy from either the time-domain specification $x(t)$ or the frequency-domain specification $X(\omega)$ of the same signal.

Equation (1.101) can be interpreted to mean that the energy of a signal $x(t)$ results from energies contributed by all the spectral components of the signal $x(t)$. The total signal energy is the area under $|X(\omega)|^2$ (divided by 2π). Similar to the approach shown in Fig. 1.12, it is helpful to consider this area as the infinite sum

$$E_x = \lim_{\Delta\omega \rightarrow 0} \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} |X(k\Delta\omega)|^2 \Delta\omega.$$

If we consider the small band of frequencies $\Delta\omega$ ($\Delta\omega \rightarrow 0$) located at $\omega = k\Delta\omega$, as illustrated in Fig. 1.47, the energy ΔE_x of these spectral components is the area of $|X(k\Delta\omega)|^2$ under this band (divided by 2π):

$$\Delta E_x = \frac{1}{2\pi} |X(k\Delta\omega)|^2 \Delta\omega = |X(k\Delta\omega)|^2 \Delta f.$$

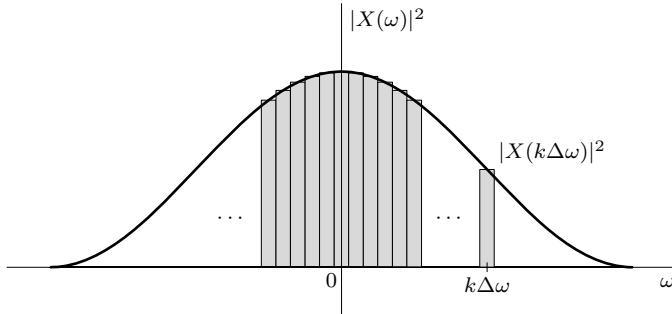


Figure 1.47: Interpretation of energy spectral density.

In general, then, $|X(\omega)|^2 \Delta f$ represents the energy contributed by the components in a band of Δf Hz.[†] Therefore, $|X(\omega)|^2$ is the *energy spectral density* (per unit bandwidth in Hz). To compute the energy over some range of frequencies $[\omega_1, \omega_2]$, the energy spectral density is simply integrated over those frequencies as

$$\Delta E_x = \frac{1}{2\pi} \int_{\omega_1}^{\omega_2} |X(\omega)|^2 d\omega. \quad (1.102)$$

For real signals, $X(\omega)$ and $X(-\omega)$ are conjugates, and $|X(\omega)|^2 = X(\omega)X^*(\omega) = X(\omega)X(-\omega)$ is an even function of ω . Therefore, Eq. (1.101) can be expressed as[‡]

$$E_x = \frac{1}{\pi} \int_0^{\infty} |X(\omega)|^2 d\omega. \quad (1.103)$$

[†]For real signals, the band of negative frequencies also contributes an equal amount of energy. Hence, for real signals, the energy contributed by a band Δf Hz at $\pm\omega$ is $2|X(\omega)|^2 \Delta f$. See Eq. (1.103).

[‡]This assumes that $X(\omega)$ does not contain an impulse at $\omega = 0$. If such an impulse exists, it should be integrated separately with a multiplying factor of $1/2\pi$ rather than $1/\pi$.

For real signals, the signal energy E_x , which results from contributions from all the frequency components from $\omega = -\infty$ to ∞ , is given by ($1/\pi$ times) the area under $|X(\omega)|^2$ from $\omega = 0$ to ∞ . For real signals, it is common to specify frequency bands using just positive frequencies; the implicit assumption is that positive- and negative-frequency components are both included.

▷ Drill 1.24 (Using Parseval's Theorem)

Use Parseval's theorem to show that the energy of the signal $y(t) = \frac{2a}{t^2+a^2}$ is, for $a > 0$, $E_y = \frac{2\pi}{a}$. Hint: Find $Y(\omega)$ using pair 3 from Table 1.1 and the duality property.

△

Essential Bandwidth of a Signal

The spectra of most signals extend to infinity. However, because the energy of any practical signal is finite, the signal spectrum must approach 0 as $\omega \rightarrow \infty$. Most of the signal energy is contained within a certain band of B rad/s (or Hz), and the energy contributed by the components beyond B is negligible. We can therefore suppress the signal spectrum beyond B with little effect on the signal shape and energy. The bandwidth B is called the *essential bandwidth* of the signal. The criterion for selecting B depends on the error tolerance in a particular application. We may, for example, select B to be that band which contains 95% of the signal energy.[†] This figure may be higher or lower than 95% depending on the precision needed. As Ex. 1.13 will show, Eq. (1.102) is often useful to determine the essential bandwidth of a signal.

Suppression of all the spectral components of $x(t)$ beyond the essential bandwidth results in a signal $\hat{x}(t)$ that is a close approximation of $x(t)$. If we use a 95% criterion for the essential bandwidth, the energy of the error $x(t) - \hat{x}(t)$ is 5% of E_x .

▷ Example 1.13 (Computing Essential Bandwidth)

Using real $a > 0$, find the energy of signal $x(t) = e^{-at}u(t)$. Using a 95% energy criterion, determine the essential bandwidth B of the signal. How does B change if a 99% energy criterion is used?

We have

$$E_x = \int_{-\infty}^{\infty} |x(t)|^2 dt = \int_0^{\infty} e^{-2at} dt = \frac{1}{2a}.$$

We can verify this result by Parseval's theorem. Using pair 1 in Table 1.1, the signal spectrum is

$$X(\omega) = \frac{1}{j\omega + a}.$$

Since $x(t)$ is real, we use the Eq. (1.103) form of Parseval's theorem to determine the signal energy as

$$E_x = \frac{1}{\pi} \int_0^{\infty} |X(\omega)|^2 d\omega = \frac{1}{\pi} \int_0^{\infty} \frac{1}{\omega^2 + a^2} d\omega = \frac{1}{\pi a} \tan^{-1} \left(\frac{\omega}{a} \right) \Big|_0^{\infty} = \frac{1}{2a}.$$

The band $(-B \leq \omega \leq B)$ contains 95% of the signal energy, that is, $0.95/2a$. Using Eq. (1.102), we obtain

$$\frac{0.95}{2a} = \frac{1}{2\pi} \int_{-B}^B \frac{d\omega}{\omega^2 + a^2} = \frac{1}{\pi} \int_0^B \frac{d\omega}{\omega^2 + a^2} = \frac{1}{\pi a} \tan^{-1} \left(\frac{\omega}{a} \right) \Big|_0^B = \frac{1}{\pi a} \tan^{-1} \left(\frac{B}{a} \right),$$

or

$$B = a \tan \left(\frac{0.95\pi}{2} \right) = 12.706a \text{ rad/s.}$$

[†]For lowpass signals, the essential bandwidth may also be defined as a frequency at which the value of the magnitude spectrum is a small fraction (about 1%) of its peak value.

This result indicates that the spectral components of $x(t)$ in the band from 0 (dc) to $12.706a$ rad/s ($2.02a$ Hz) contribute 95% of the total signal energy; all the remaining spectral components in the band from $2.02a$ Hz to ∞ contribute only 5% of the signal energy.

Following the same procedure, the 99% energy bandwidth is computed as

$$B = a \tan\left(\frac{0.99\pi}{2}\right) = 63.657a \text{ rad/s.}$$

Increasing the energy criterion from 95% to 99% results in a *five-fold* increase in essential bandwidth.

Example 1.13 ◀

▷ **Example 1.14 (Estimating Essential Bandwidth)**

Find the energy of the unit-duration pulse $x(t) = \Pi(t)$. Using a 95% energy criterion, determine the essential bandwidth B of the signal.

It is most convenient to find the energy of $x(t)$ in the time domain as

$$E_x = \int_{-0.5}^{0.5} (1)^2 dt = 1.$$

To determine the essential bandwidth, we write Eq. (1.102) in terms of B as

$$\Delta E_x = .95E_x = 0.95 = \frac{1}{2\pi} \int_{-B}^B |X(\omega)|^2 d\omega = \frac{1}{2\pi} \int_{-B}^B \operatorname{sinc}^2(\omega/2\pi) d\omega. \quad (1.104)$$

This is a difficult equation to solve and requires numerical integration techniques. To better understand how a solution is obtained, we rewrite Eq. (1.104) as

$$\left| 0.95 - \frac{1}{2\pi} \int_{-B}^B \operatorname{sinc}^2(\omega/2\pi) d\omega \right| = 0. \quad (1.105)$$

The left-hand side of Eq. (1.105) is minimized (zero) when B is the correct 95% essential bandwidth. Thus, our problem becomes one of minimizing the left-hand side of Eq. (1.105) with respect to B . In the context of such a minimization problem, the left-hand side of Eq. (1.105) is called the *objective function* for the minimization.

Once again, MATLAB possesses built-in functions that help solve the problem.

```
01 Xsquare = @(omega) (sinc(omega/(2*pi))).^2;
02 DeltaEx = @(B) quad(Xsquare,-B,B)/(2*pi);
03 ObjFun = @(B) abs(.95-DeltaEx(B)); B = fminsearch(ObjFun,1)
B = 13.0245
```

Lines 01 and 02 define the integrand and integral of Eq. (1.104), respectively. Notice that line 02 computes the integral numerically using MATLAB's `quad` function. Line 03 defines our objective function and uses the MATLAB command `fminsearch` to minimize this function beginning with an initial guess of 1. The result tells us that the 95% effective bandwidth of $x(t)$ is $B = 13.0245$ rad/s. To verify this result, we numerically evaluate the right-hand side of Eq. (1.104) and verify the result is 0.95.

```
04 DeltaEx(B)
ans = 0.9500
```

Example 1.14 ◀

A Frequency-Domain Representation of Signal Power

We can readily extend the idea of energy spectral density to power signals for which the signal power is given by

$$P_x = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |x(t)|^2 dt = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-\infty}^{\infty} |x_T(t)|^2 dt = \lim_{T \rightarrow \infty} \frac{E_{x_T}}{T}. \quad (1.106)$$

Here, $x_T(t)$ is the signal $x(t)$ truncated outside $|t| > \frac{T}{2}$, that is, $x_T(t) = x(t)\Pi\left(\frac{t}{T}\right)$ and E_{x_T} is the energy of this truncated signal $x_T(t)$. Using Eq. (1.101) in Eq. (1.106) yields

$$P_x = \lim_{T \rightarrow \infty} \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{|X_T(\omega)|^2}{T} d\omega = \frac{1}{2\pi} \int_{-\infty}^{\infty} S_x(\omega) d\omega. \quad (1.107)$$

Here, $S_x(\omega)$, the *power spectral density* of $x(t)$, is defined as

$$S_x(\omega) = \lim_{T \rightarrow \infty} \frac{|X_T(\omega)|^2}{T}. \quad (1.108)$$

The power spectral density has similar physical interpretation as the energy spectral density. Analogous to Eq. (1.102), the power ΔP_x contained in some nonzero range of frequencies $\Delta\omega$ is obtained by integrating the power spectral density over those frequencies as

$$\Delta P_x = \frac{1}{2\pi} \int_{\Delta\omega} S_x(\omega) d\omega. \quad (1.109)$$

Using Properties to Reduce Tedious Computations

For many signals, it can be quite tedious to directly compute the Fourier transform using Eq. (1.75). Properties often offer a simpler way, particularly for piecewise polynomial signals. As the next example shows, we can readily compute the Fourier transform of a piecewise polynomial signal without any integration by using the time-differentiation and time-shifting properties.

▷ **Example 1.15 (Using Properties to Determine the Fourier Transform of a Signal)**

Using only properties and the fact that $\delta(t) \iff 1$, determine the Fourier transform of $x(t) = \Lambda\left(\frac{t-2}{2}\right)$.

To begin, notice that we can express $x(t)$ as

$$x(t) = (t-1)u(t-1) - 2(t-2)u(t-2) + (t-3)u(t-3).$$

Thus,

$$\frac{d}{dt}x(t) = u(t-1) - 2u(t-2) + u(t-3)$$

and

$$\frac{d^2}{dt^2}x(t) = \delta(t-1) - 2\delta(t-2) + \delta(t-3). \quad (1.110)$$

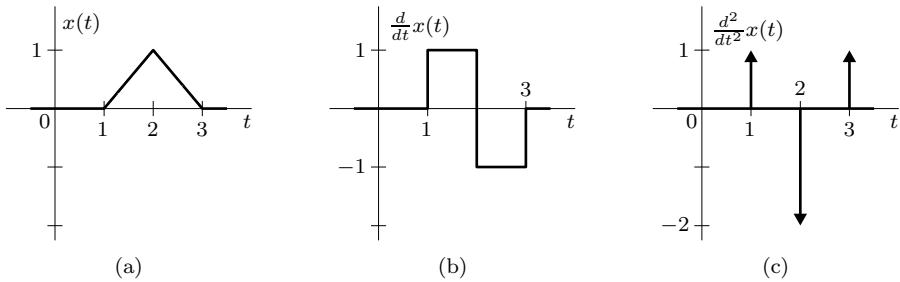
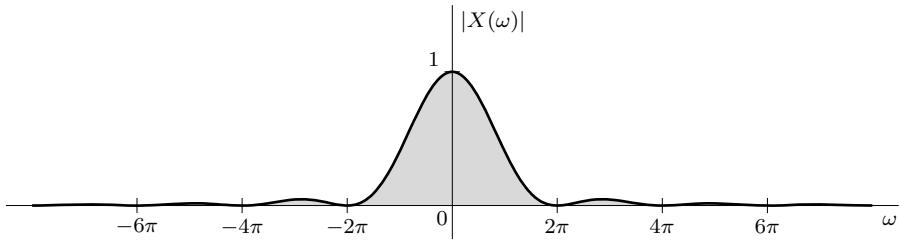
The signal $x(t)$ and its first two derivatives are shown in Fig. 1.48.

Using the time-differentiation and time-shifting properties, the Fourier transform of Eq. (1.110) is given as

$$-\omega^2 X(\omega) = e^{-j\omega} - 2e^{-j2\omega} + e^{-j3\omega} = 2e^{-j2\omega} (\cos(\omega) - 1),$$

which is easily solved for $X(\omega)$ as long as $\omega \neq 0$. For $\omega = 0$, we determine, by inspection, that $X(0) = 1$. Thus,

$$X(\omega) = \begin{cases} \frac{2e^{-j2\omega}}{\omega^2} [1 - \cos(\omega)] & \omega \neq 0 \\ 1 & \omega = 0 \end{cases}.$$

Figure 1.48: Signal $x(t)$ and its first two derivatives.Figure 1.49: Magnitude spectrum $|X(\omega)|$.

In this case as well as many others, $X(\omega)$ is obtained through properties with far less effort than direct computation using Eq. (1.75), which would require repeated integration by parts. Figure 1.49 shows the magnitude spectrum $|X(\omega)|$.

Example 1.15 ◀

1.9.9 Extending Fourier Transform Properties to the Fourier Series

Fourier analysis comes in many flavors, including the Fourier transform and the Fourier series. The similarities between these varieties greatly exceed their differences. It is little surprise, therefore, that most of the properties of the Fourier transform extend, if not exactly then at least in spirit, to the Fourier series. In a general sense, for example, both transforms share basic symmetry relations, as shown in Table 1.2. Notice that each symmetry relation possesses an equivalent dual. For example, this duality ensures that a real time-domain *or* a real frequency-domain signal has a conjugate-symmetric transform and vice versa.

Time (or Frequency)	\iff	Frequency (or Time)
Real	\iff	Conjugate symmetric
Imaginary	\iff	Conjugate antisymmetric
Even	\iff	Even
Odd	\iff	Odd

Table 1.2: Symmetries shared by the Fourier transform and the Fourier series.

Table 1.3 summarizes the Fourier transform properties already discussed and provides a side-by-side comparison with the corresponding properties, when they exist, of the Fourier series. When

talking about the Fourier series and its properties, it is implicit that the underlying signals are periodic with common fundamental frequency ω_0 . As Table 1.3 makes clear, the Fourier transform

Fourier Transform	Fourier Series
Synthesis: $x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega$ Analysis: $X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt$ Duality: if $x(t) \iff X(\omega)$, then $X(t) \iff 2\pi x(-\omega)$ Linearity: $ax(t) + by(t) \iff aX(\omega) + bY(\omega)$ Complex Conjugation: $x^*(t) \iff X^*(-\omega)$ Scaling and Reversal: $x(at) \iff \frac{1}{ a } X\left(\frac{\omega}{a}\right)$ $x(-t) \iff X(-\omega)$ Shifting: $x(t - t_0) \iff X(\omega) e^{-j\omega t_0}$ $x(t) e^{j\omega_0 t} \iff X(\omega - \omega_0)$ Differentiation: $\frac{d}{dt} x(t) \iff j\omega X(\omega)$ $-jtx(t) \iff \frac{d}{d\omega} X(\omega)$ Time Integration: $\int_{-\infty}^t x(\tau) d\tau \iff \frac{X(\omega)}{j\omega} + \pi X(0) \delta(\omega)$ Convolution: $x(t) * y(t) \iff X(\omega) Y(\omega)$ $x(t)y(t) \iff \frac{1}{2\pi} X(\omega) * Y(\omega)$ Correlation: $\rho_{x,y}(\tau) = x(\tau) * y^*(-\tau) \iff X(\omega) Y^*(\omega)$ Parseval's: $E_x = \int_{-\infty}^{\infty} x(t) ^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) ^2 d\omega$	Synthesis: $x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}$ Analysis: $X_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk\omega_0 t} dt$ Duality: Linearity: $ax(t) + by(t) \iff aX_k + bY_k$ Complex Conjugation: $x^*(t) \iff X_{-k}$ Scaling and Reversal: $x(-t) \iff X_{-k}$ Shifting: $x(t - t_0) \iff X_k e^{-jk\omega_0 t_0}$ $x(t) e^{jk_0 \omega_0 t} \iff X_{k-k_0}$ Differentiation: $\frac{d}{dt} x(t) \iff jk\omega_0 X_k$ Time Integration: Convolution: $\frac{1}{T_0} x(t) \circledast y(t) \iff X_k Y_k$ $x(t)y(t) \iff X_k * Y_k$ Correlation: $\rho_{x,y}(\tau) = \frac{1}{T_0} x(\tau) \circledast y^*(-\tau) \iff X_k Y_k^*$ Parseval's: $P_x = \frac{1}{T_0} \int_{T_0} x(t) ^2 dt = \sum_{k=-\infty}^{\infty} X_k ^2$

Table 1.3: Fourier transform properties and Fourier series properties.

and the Fourier series have essentially identical complex-conjugation and time-reversal properties, which provides mathematical justification of the shared symmetries presented in Table 1.2.

Despite the overall similarities found in Table 1.3, there are some noteworthy differences between the two sets of properties. To begin, we notice that the Fourier series does not possess the duality property of the Fourier transform. This is not to say the Fourier series lacks dual relations, because it certainly does. Rather, the Fourier series, which utilizes integration for analysis and summation for synthesis, lacks the Fourier transform's near-perfect symmetry (Fig. 1.38) that the duality property requires. Next, the Fourier series has no need of a general scaling property; time scaling a periodic signal $x(t)$ simply changes the fundamental frequency ω_0 and does not affect the Fourier series coefficients X_k . Since the Fourier series coefficients X_k are a function of a discrete frequency variable

k (rather than a continuous frequency variable), there is no frequency-differentiation property for the Fourier series. Further, since the integration of a periodic signal is unbounded, there is no time-integration property of the Fourier series.

The differences in the convolution and correlation properties are less superficial. In the Fourier series, where signals are periodic, traditional convolution has little meaning. Rather, *circular convolution*, which is also called *cyclic or periodic convolution*, is the appropriate tool. The circular convolution of two periodic signals $x(t)$ and $y(t)$ is defined as

$$x(t) \circledast y(t) = \int_{T_0} x(\tau) y(t - \tau) d\tau. \quad (1.111)$$

Compared with ordinary convolution, circular convolution integrates only over a single period T_0 (rather than from $-\infty$ to ∞). This difference should look familiar. Within the scale factor $1/T_0$, it is precisely the same difference found between the Eq. (1.32) expression for signal energy and the Eq. (1.34) power expression for periodic signals. Parseval's theorem, which follows from correlation, shares these same changes in going from the Fourier transform to the Fourier series. In the Fourier series, the quantity $|X_k|^2$ serves much the same role as the power spectral density function $S_x(\omega)$ of Eq. (1.108) discussed previously. Lastly, notice that the product rule (convolution in frequency) of the Fourier series involves a traditional convolution of two discrete sequences, which is a topic discussed in depth in Ch. 5.

Beyond the Fourier Transform and Fourier Series

Table 1.4 provides additional insight into Fourier analysis. We know a periodic continuous-time signal is analyzed with the Fourier series, which is discrete in frequency. Thus, being periodic in time leads to being discrete in frequency. Further, an aperiodic CT signal is analyzed with the Fourier transform, which is continuous in frequency. Thus, being aperiodic in time leads to being continuous in frequency. Although we delay rigorous proofs for later, duality also applies here. Thus, being continuous in time leads to being aperiodic in frequency, and being discrete in time leads to being periodic in frequency. Thus, we see that the spectra produced by the Fourier series or the Fourier transform, which both operate on continuous-time signals, must result in aperiodic spectra. Further, we learn the fundamental characteristics that exist in the Fourier analysis of discrete-time signals. An aperiodic DT sequence is analyzed with the discrete-time Fourier transform (DTFT), which is necessarily periodic and continuous in frequency. A periodic DT sequence is analyzed with the discrete-time Fourier series (DTFS), which is necessarily periodic and discrete in frequency. Periodic DT sequences are often analyzed with a variant of the DTFS called the discrete Fourier transform (DFT), of which the fast Fourier transform (FFT) is a computationally efficient version. Later chapters fully treat the DTFT, the DTFS, the DFT, and the FFT.

Time (or Frequency)	\iff	Frequency (or Time)
Periodic	\iff	Discrete
Aperiodic	\iff	Continuous

Table 1.4: Fundamental Fourier relations.

1.10 The Laplace Transform

The Fourier transform is a tool that allows us to represent a signal $x(t)$ as a continuous sum of exponentials of the form $e^{j\omega t}$, whose frequencies are restricted to the imaginary axis in the complex plane ($s = j\omega$). Such a representation is quite valuable in the analysis and processing of signals. In

the area of system analysis, however, the use of the Fourier transform leaves much to be desired. First, the Fourier transform exists only for a restricted class of signals and, therefore, cannot be used for inputs such as growing exponentials. Second, the Fourier transform cannot be used easily to analyze unstable or even marginally stable systems.

The basic reason for both these difficulties is that for some signals, such as $e^{at}u(t)$ ($a > 0$), the Fourier transform does not exist. This is because ordinary exponentials of the form $e^{j\omega t}$, on account of their constant amplitudes, are incapable of synthesizing exponentially growing signals. This problem could be resolved if it were possible to use basis signals of the form e^{st} rather than $e^{j\omega t}$, where the complex frequency s is not restricted to just the imaginary axis as it is in the Fourier transform. This is precisely what is done in the *bilateral Laplace transform*, which generalizes the frequency variable $s = j\omega$ to $s = \sigma + j\omega$. Such generalization permits us to use exponentially growing sinusoids to synthesize a signal $x(t)$. The (bilateral) Laplace transform of $x(t)$ is given by (see [1])

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt, \quad (1.112)$$

and the inverse relationship is given by[†]

$$x(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} X(s)e^{st} ds. \quad (1.113)$$

The bilateral Laplace transform and its inverse are denoted symbolically as

$$X(s) = \mathcal{L}\{x(t)\} \quad \text{and} \quad x(t) = \mathcal{L}^{-1}\{X(s)\}$$

or simply as

$$x(t) \xrightleftharpoons{\mathcal{L}} X(s).$$

When computing Eq. (1.112), it is necessary to specify where the transform converges in the s -plane. The *region of convergence* (ROC) R_x , sometimes called the *region of existence*, is a function of the variable s and describes the area in the complex plane where $X(s)$ converges (exists). Conveniently, left-sided signals have left-sided ROCs, right-sided signals have right-sided ROCs, and signals with both left- and right-sided components have ROCs that are the union of the individual left- and right-sided ROCs, normally a strip in the s -plane. The ROCs for finite-duration signals include the entire s -plane (except perhaps $s = 0$ and $|s| = \infty$).

Due to the complex (versus real) integration required, it is difficult to evaluate Eq. (1.113) directly. Rather, inverse Laplace transforms are typically performed by table look-up. If a particular transform $X(s)$ is not available in a table, partial fraction expansions and Laplace transform properties are normally helpful. See [1] for greater details on ROCs and computing inverse Laplace transforms.

The Unilateral Laplace Transform

In practice, we deal most often with causal signals. When we observe this restriction, the Laplace transform is called the *unilateral* Laplace transform. Mathematically, the unilateral Laplace transform of causal signal $x(t)$ is

$$X(s) = \int_{0^-}^{\infty} x(t)e^{-st} dt. \quad (1.114)$$

The unilateral Laplace transform is denoted symbolically as

$$X(s) = \mathcal{L}_u\{x(t)\}$$

[†]In Eq. (1.113), σ must be chosen in the ROC of $X(s)$.

or simply as

$$x(t) \xrightarrow{\mathcal{L}_u} X(s).$$

Since the unilateral Laplace transform is only used with causal signals, its region of convergence can be inferred and is thus typically not specified. The inverse of the unilateral Laplace transform does not differ from that of the bilateral Laplace transform and is thus represented by Eq. (1.113).

As comparing Eqs. (1.112) and (1.114) makes clear, the unilateral Laplace transform is just a special case of the bilateral transform where all signals are restricted to be causal. Since the bilateral Laplace transform is more general than the unilateral case, it is natural to question why the restrictive unilateral Laplace transform is ever considered. There are two primary reasons. First, the unilateral Laplace transform can be more convenient than the bilateral case, particularly since there is no need to specify any ROCs. Second, and perhaps more important, the unilateral transform better facilitates the analysis of linear constant-coefficient differential equations that possess nonzero initial conditions. This will become more clear in Sec. 1.10.2 when Laplace transform properties are reviewed.

Table 1.5 gives a short table of selected bilateral Laplace transform pairs, including their ROCs. Entries 1–10, which involve causal signals $x(t)$, double as a table of unilateral transforms, in which case specification of the ROC is unnecessary. Entries 11–19 involve anti-causal signals and therefore do not possess a unilateral Laplace transform.

1.10.1 Connection between the Fourier and Laplace Transforms

According to Eq. (1.112), the general (bilateral) Laplace transform of a signal $x(t)$ is

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt.$$

Setting $s = j\omega$ in this equation yields

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt,$$

where $X(j\omega) = X(s)|_{s=j\omega}$. Observe that the right-hand-side integral defines $X(\omega)$, the Fourier transform of $x(t)$. Does this mean that the Fourier transform can be obtained from the corresponding Laplace transform by setting $s = j\omega$? In other words, is it true that $X(j\omega) = X(\omega)$? Well, yes and no. Yes, it is true in most cases. For example, when $x(t) = e^{\lambda t}u(t)$ and $\text{Re}\{\lambda\} < 0$, its Laplace transform is $1/(s - \lambda)$, and $X(j\omega) = 1/(j\omega - \lambda)$, which is equal to its Fourier transform $X(\omega)$ (assuming $\text{Re}\{\lambda\} < 0$). However, for the unit step function $u(t)$, the Laplace transform is

$$u(t) \xrightarrow{\mathcal{L}} \frac{1}{s}, \quad \text{ROC: } \text{Re}\{s\} > 0.$$

The Fourier transform is given by

$$u(t) \xrightarrow{j\omega} \frac{1}{j\omega} + \pi\delta(\omega).$$

Clearly, $X(j\omega) \neq X(\omega)$ in this case.

To better understand this puzzle, consider the fact that both the Laplace and Fourier transforms synthesize $x(t)$ using everlasting exponentials of the form e^{st} . The frequency s can be anywhere in the ROC for the Laplace transform, but it is restricted to the ω -axis in the case of the Fourier transform. For example, the unit step function is readily synthesized in the Laplace transform by a relatively simple spectrum $X(s) = 1/s$, in which the frequencies s are chosen in the RHP $\text{Re}\{s\} > 0$. The Fourier transform can also synthesize $u(t)$ using the restricted frequencies along the ω -axis, but the spectrum is more complicated than when we are free to choose the frequencies in the RHP.

$x(t)$	$X(s)$	ROC
1. $\delta(t)$	1	All s
2. $u(t)$	$\frac{1}{s}$	$\text{Re}\{s\} > 0$
3. $t^k u(t)$	$\frac{k!}{s^{k+1}}$	$\text{Re}\{s\} > 0$
4. $e^{\lambda t} u(t)$	$\frac{1}{s-\lambda}$	$\text{Re}\{s\} > \text{Re}\{\lambda\}$
5. $t^k e^{\lambda t} u(t)$	$\frac{k!}{(s-\lambda)^{k+1}}$	$\text{Re}\{s\} > \text{Re}\{\lambda\}$
6. $\cos(\omega_0 t) u(t)$	$\frac{s}{s^2 + \omega_0^2}$	$\text{Re}\{s\} > 0$
7. $\sin(\omega_0 t) u(t)$	$\frac{\omega_0}{s^2 + \omega_0^2}$	$\text{Re}\{s\} > 0$
8. $e^{-at} \cos(\omega_0 t) u(t)$	$\frac{s+a}{(s+a)^2 + \omega_0^2}$	$\text{Re}\{s\} > -a$
9. $e^{-at} \sin(\omega_0 t) u(t)$	$\frac{\omega_0}{(s+a)^2 + \omega_0^2}$	$\text{Re}\{s\} > -a$
10. $e^{-at} \cos(\omega_0 t + \theta) u(t)$	$\begin{aligned} & \frac{\cos(\theta)s + a \cos(\theta) - \omega_0 \sin(\theta)}{s^2 + 2as + (a^2 + \omega_0^2)} \\ &= \frac{0.5e^{j\theta}}{s+a-j\omega_0} + \frac{0.5e^{-j\theta}}{s+a+j\omega_0} \end{aligned}$	$\text{Re}\{s\} > -a$
11. $u(-t)$	$-\frac{1}{s}$	$\text{Re}\{s\} < 0$
12. $t^k u(-t)$	$-\frac{k!}{s^{k+1}}$	$\text{Re}\{s\} < 0$
13. $e^{\lambda t} u(-t)$	$-\frac{1}{s-\lambda}$	$\text{Re}\{s\} < \text{Re}\{\lambda\}$
14. $t^k e^{\lambda t} u(-t)$	$-\frac{k!}{(s-\lambda)^{k+1}}$	$\text{Re}\{s\} < \text{Re}\{\lambda\}$
15. $\cos(\omega_0 t) u(-t)$	$-\frac{s}{s^2 + \omega_0^2}$	$\text{Re}\{s\} < 0$
16. $\sin(\omega_0 t) u(-t)$	$-\frac{\omega_0}{s^2 + \omega_0^2}$	$\text{Re}\{s\} < 0$
17. $e^{-at} \cos(\omega_0 t) u(-t)$	$-\frac{s+a}{(s+a)^2 + \omega_0^2}$	$\text{Re}\{s\} < -a$
18. $e^{-at} \sin(\omega_0 t) u(-t)$	$-\frac{\omega_0}{(s+a)^2 + \omega_0^2}$	$\text{Re}\{s\} < -a$
19. $e^{-at} \cos(\omega_0 t + \theta) u(-t)$	$\begin{aligned} & \frac{\cos(\theta)s + a \cos(\theta) - \omega_0 \sin(\theta)}{s^2 + 2as + (a^2 + \omega_0^2)} \\ &= -\frac{0.5e^{j\theta}}{s+a-j\omega_0} + \frac{0.5e^{-j\theta}}{s+a+j\omega_0} \end{aligned}$	$\text{Re}\{s\} < -a$

Table 1.5: Selected bilateral Laplace transform pairs.

When a signal's ROC includes the ω -axis, both the Laplace and Fourier transforms can synthesize the signal with the same spectrum and $X(s)|_{s=j\omega} = X(\omega)$. Absolutely integrable signals fall into this category. However, when a signal's ROC does not include the ω -axis, such as when a signal is not absolutely integrable, the Laplace spectrum will necessarily fail to properly synthesize the signal using only frequencies $s = j\omega$, which are outside its region of convergence. In such cases, $X(j\omega) \neq X(\omega)$.

Notice that the Fourier transform is able to do something the Laplace transform cannot: it can synthesize certain signals such as $u(t)$ that are not absolutely integrable using the restricted frequencies $s = j\omega$. Fourier can also accommodate periodic signals, a task the Laplace transform cannot manage with any choice of frequencies s . Before we feel too sorry for the poor Laplace transform, however, notice that the Laplace transform easily handles growing exponentials, which is something the Fourier transform cannot tolerate. This discussion shows that although the Fourier transform may be considered as a special case of the Laplace transform, we need to circumscribe

such a view.

1.10.2 Laplace Transform Properties

Because it is a generalization of the Fourier transform, the properties of the bilateral Laplace transform are similar to those of the Fourier transform. Table 1.6 summarizes the important properties of the bilateral Laplace transform. Table 1.6 also provides a side-by-side comparison with the corresponding properties, when they exist, of the unilateral Laplace transform. When talking about the unilateral Laplace transform and its properties, it is assumed that the underlying signals are causal.

Bilateral Laplace Transform	Unilateral Laplace Transform
<p>Synthesis: $x(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} X(s)e^{st} ds$</p> <p>Analysis: $X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt$, ROC: R_x</p> <p>Linearity: $ax(t) + by(t) \xleftrightarrow{\mathcal{L}} aX(s) + bY(s)$, ROC: At least $R_x \cap R_y$</p> <p>Complex Conjugation: $x^*(t) \xleftrightarrow{\mathcal{L}} X^*(s^*)$, ROC: R_x</p> <p>Scaling and Reversal: $x(at) \xleftrightarrow{\mathcal{L}} \frac{1}{ a } X\left(\frac{s}{a}\right)$, ROC: R_x scaled by $1/a$ $x(-t) \xleftrightarrow{\mathcal{L}} X(-s)$, ROC: R_x reflected</p> <p>Shifting: $x(t - t_0) \xleftrightarrow{\mathcal{L}} X(s)e^{-st_0}$, ROC: R_x $x(t)e^{s_0 t} \xleftrightarrow{\mathcal{L}} X(s - s_0)$, ROC: R_x shifted by $\text{Re}\{s_0\}$</p> <p>Differentiation: $\frac{d}{dt} x(t) \xleftrightarrow{\mathcal{L}} sX(s)$, ROC: At least R_x $-tx(t) \xleftrightarrow{\mathcal{L}} \frac{d}{ds} X(s)$, ROC: R_x</p> <p>Time Integration: $\int_{-\infty}^t x(\tau)d\tau \xleftrightarrow{\mathcal{L}} \frac{1}{s} X(s)$, ROC: At least $R_x \cap (\text{Re}\{s\} > 0)$</p> <p>Convolution: $x(t) * y(t) \xleftrightarrow{\mathcal{L}} X(s)Y(s)$, ROC: At least $R_x \cap R_y$</p>	<p>Synthesis: $x(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} X(s)e^{st} ds$</p> <p>Analysis: $X(s) = \int_{0-}^{\infty} x(t)e^{-st} dt$</p> <p>Linearity: $ax(t) + by(t) \xleftrightarrow{\mathcal{L}_u} aX(s) + bY(s)$</p> <p>Complex Conjugation: $x^*(t) \xleftrightarrow{\mathcal{L}_u} X^*(s^*)$</p> <p>Scaling and Reversal: If $a > 0$: $x(at) \xleftrightarrow{\mathcal{L}_u} \frac{1}{a} X\left(\frac{s}{a}\right)$</p> <p>Shifting: If $t_0 > 0$: $x(t - t_0) \xleftrightarrow{\mathcal{L}_u} X(s)e^{-st_0}$ $x(t)e^{s_0 t} \xleftrightarrow{\mathcal{L}_u} X(s - s_0)$</p> <p>Differentiation: $\frac{d}{dt} x(t) \xleftrightarrow{\mathcal{L}_u} sX(s) - x(0^-)$ (general case shown below) $-tx(t) \xleftrightarrow{\mathcal{L}_u} \frac{d}{ds} X(s)$</p> <p>Time Integration: $\int_{0-}^t x(\tau)d\tau \xleftrightarrow{\mathcal{L}_u} \frac{1}{s} X(s)$</p> <p>Convolution: $x(t) * y(t) \xleftrightarrow{\mathcal{L}_u} X(s)Y(s)$</p>
Unilateral Laplace Transform Time Differentiation, General Case $x^{(k)}(t) = \frac{d^k}{dt^k} x(t) \xleftrightarrow{\mathcal{L}_u} s^k X(s) - \sum_{i=0}^{k-1} s^{k-1-i} x^{(i)}(0^-)$	

Table 1.6: Properties of the bilateral and unilateral Laplace transforms.

The strong connection between the Fourier and Laplace transforms is clear when the properties of the Laplace transform in Table 1.6 are compared with those of the Fourier transform in Table 1.3. Furthermore, as the unilateral Laplace transform is a special case of the bilateral Laplace transform,

the majority of the properties between the two are identical. Only a few differences are to be found. First, note that an ROC need not be specified for the unilateral case. The scaling and shifting properties possess restrictions for the unilateral transform to ensure causal results, as required. Similarly, the time-reversal property finds no place among the unilateral Laplace transform properties since it always transforms a causal signal into a noncausal signal. The most interesting difference, as alluded to previously, is found in the differentiation property; initial conditions are explicitly visible for the unilateral Laplace transform. For additional details, refer to [1].

▷ **Example 1.16 (System Analysis Using the Unilateral Laplace Transform)**

Find the response $y(t)$ of a causal LTIC system with transfer function $H(s) = 1/(s + 5)$ if the input is $x(t) = u(t)$. Locate the system poles and zeros and determine whether or not the system is stable.

The output of an LTIC system is $y(t) = x(t) * h(t)$. Applying the convolution property of the unilateral Laplace transform (Table 1.6) yields

$$Y(s) = X(s)H(s).$$

From Table 1.5, we find that $x(t) = u(t) \xrightarrow{\mathcal{L}_u} X(s) = 1/s$. Hence,

$$Y(s) = \frac{1}{s(s + 5)} = \frac{1}{5} \left(\frac{1}{s} - \frac{1}{s + 5} \right).$$

Using Table 1.5 to determine the inverse transform yields

$$y(t) = \frac{1}{5} (1 - e^{-5t}) u(t).$$

By inspection, this system has a single finite pole at $s = -5$. This system does not have a finite zero, but since every pole is paired with a zero (and vice versa), we can establish that there is a system zero at $|s| = \infty$.

Since the only pole of $H(s)$ is LHP, the system is BIBO stable. Further, as long as the system is observable and controllable (see the footnote on page 32), we can also state the system is asymptotically stable.

Example 1.16 ◀

1.11 Summary

This chapter provides a review of fundamental continuous-time (CT) signals and systems concepts. Despite apparent differences, CT concepts are highly relevant to the study of digital signal processing.

Varieties of operators, signal models, and signal classifications assist in the characterization and manipulation of CT signals and systems. The CT time-shifting operation relocates a signal in time, while the time-scaling operation allows for signal compression, expansion, and reflection. The unit step, unit gate, unit triangle, and unit impulse functions are all useful CT signal models, as are the exponential and interpolation (sinc) functions. Exponential and unit impulse functions are both commonly used as building blocks (basis functions) to construct more complicated waveforms and are invaluable in the frequency-domain and time-domain analysis of LTIC systems, respectively. Signal classifications, such as those based on signal symmetries, provide helpful insights into signal character and behavior. Such classifications also identify which techniques are most appropriate for signal analysis.

It is difficult to mathematically characterize the behavior of a CT system in the most general case. By restricting our attention to CT systems that are linear and time-invariant (LTIC), however, the

mathematics are tractable, and the system output to an arbitrary input is given by the convolution of the input $x(t)$ with the system's unit impulse response $h(t)$, denoted by $x(t) * h(t)$. The impulse response provides a time-domain characterization of an LTIC system, and convolution provides a time-domain expression of system output.

The response of an LTIC system with transfer function $H(s)$ to an everlasting sinusoid of frequency ω is also an everlasting sinusoid of the same frequency. The output magnitude is $|H(j\omega)|$ times the input magnitude, and the output sinusoid is shifted in phase with respect to the input sinusoid by $\angle H(j\omega)$ radians. This unique behavior provides the foundation for frequency- and transform-domain analysis. In this context, three transforms are particularly appropriate: the Fourier series, the Fourier transform, and the Laplace transform.

A periodic signal of period T_0 can be represented by a sum of exponentials (or sinusoids) of frequencies $\omega_0 = 2\pi/T_0$ and all its harmonics. Such a representation is called the Fourier series. We can express a periodic signal using either the exponential or trigonometric form; the two forms are equivalent. A truncated Fourier series generally provides a good approximation of a signal, although Gibbs phenomenon will occur at any points of discontinuity.

An aperiodic signal can be expressed as a Fourier integral (instead of a Fourier series). The existence of the Fourier transform is guaranteed for signals that are absolutely integrable or those with finite energy. The Fourier transform of an exponentially growing signals does not exist. Moreover, the Fourier transform cannot be used to analyze unstable (or even marginally stable) systems. The Fourier transform exhibits a strong time-frequency duality. Both the Fourier transform and the Fourier series share similar properties that simplify frequency-domain operations.

Extension of the Fourier transform, where the frequency variable $j\omega$ is generalized to the variable $\sigma + j\omega$, is called the Laplace transform. Because of this generalization, the Laplace transform exists for a much wider variety of signals. The Laplace transform can handle all types of linear systems: stable, unstable, or marginally stable.

Armed with these concepts, we are prepared to tackle both the analysis and design of systems from a frequency-domain or filtering perspective. This is the focus of the next chapter.

References

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
2. Lathi, B. P., *Signal Processing and Linear Systems*, Oxford University Press, New York, 1998.
3. Kailath, T., *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.
4. Bracewell, R. N., *Fourier Transform and Its Applications*, Revised 2nd Ed., McGraw-Hill, New York, 1986.

Problems

1.1-1 What is the difference between a continuous-time signal and a discrete-time signal?

1.1-2 What is the difference between an analog signal and a digital signal?

1.2-1 Using signal $x(t)$ shown in Fig. P1.2-1, sketch

- (a) $x(-t)$
- (b) $x(t + 5)$
- (c) $x(2t/3)$
- (d) $x(2 - t/2)$

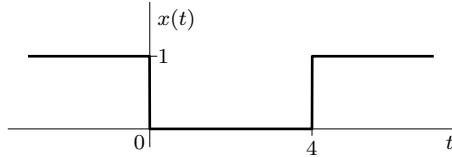


Figure P1.2-1

1.2-2 Repeat Prob. 1.2-1 using signal $x(t)$ shown in Fig. P1.2-2.

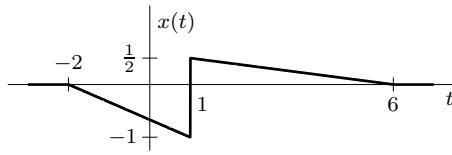


Figure P1.2-2

1.2-3 Define a signal as

$$2x(-3t + 1) = t[u(-t - 1) - u(-t + 1)],$$

where $u(t)$ is the unit step function.

(a) Plot $2x(-3t + 1)$ over a suitable range of t .

(b) Plot $x(t)$ over a suitable range of t .

1.2-4 Using $x(at + b)$ shown in Fig. P1.2-4, sketch the original signal $x(t)$ if

- (a) $a = 2$ and $b = 1$
- (b) $a = 2$ and $b = -1$
- (c) $a = -2$ and $b = 1$
- (d) $a = -2$ and $b = -1$

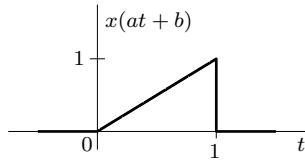


Figure P1.2-4

1.2-5 Repeat Prob. 1.2-4 using

- (a) $a = 1/3$ and $b = 1/2$
- (b) $a = 1/3$ and $b = -1/2$
- (c) $a = -1/3$ and $b = 1/2$
- (d) $a = -1/3$ and $b = -1/2$

1.2-6 Consider Fig. P1.2-6. Express signals $x_a(t)$, $x_b(t)$, $x_c(t)$, $x_d(t)$, and $x_e(t)$ in terms of signal $x(t)$ and its time-shifted, time-scaled or time-reversed versions.

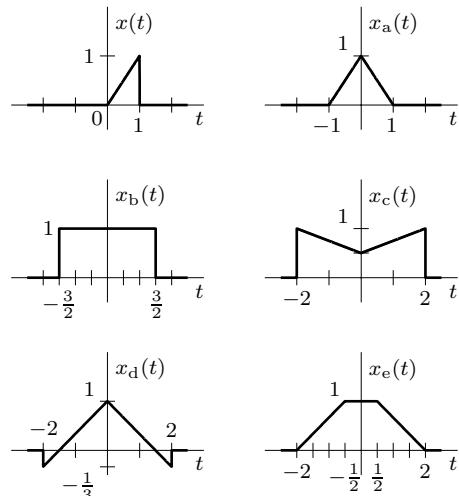


Figure P1.2-6

1.3-1 Sketch the signals

- (a) $u(t - 4) - u(t - 7)$
- (b) $u(t - 4) + u(t - 7)$
- (c) $t^2[u(t + 1) - u(t + 2)]$
- (d) $(t - 3)[u(t - 1) - u(t - 3)]$

1.3-2 Express the signal $x(t)$ in Fig. P1.2-2 by a single expression that is valid for all t .

1.3-3 Define $x(t) = t[u(t) - u(t-1)]$. Letting $a = 1/2$ and $b = 2$, sketch

- | | |
|---------------|---------------|
| (a) $x(at+b)$ | (b) $x(at)+b$ |
| (c) $ax(t+b)$ | (d) $ax(t)+b$ |

1.3-4 Repeat Prob. 1.3-3 using $a = -1/2$ and $b = 2$.

1.3-5 Repeat Prob. 1.3-3 using $a = 1/2$ and $b = -2$.

1.3-6 Repeat Prob. 1.3-3 using $a = -1/2$ and $b = -2$.

1.3-7 Sketch the following functions:

- | | |
|--|--|
| (a) $\Pi\left(\frac{t}{3}\right)$ | (b) $\Lambda\left(\frac{3\omega}{25}\right)$ |
| (c) $\Pi\left(\frac{t-3}{6}\right)$ | (d) $\Pi(t) - \Lambda(t)$ |
| (e) $\text{sinc}\left(\frac{\omega}{5}\right)$ | (f) $\text{sinc}\left(\frac{\omega-10}{5\pi}\right)$ |
| (g) $\text{sinc}\left(\frac{t}{5\pi}\right)\Pi\left(\frac{t}{30}\right)$ | |

1.3-8 Using Eq. (1.5), simplify the following expressions:

- | |
|---|
| (a) $\left(\frac{\cos(t)}{t^2+2}\right)\delta(t)$ |
| (b) $\left(\frac{j\omega+3}{\omega^2+4}\right)\delta(\omega)$ |
| (c) $e^{-t}\cos(3t-\pi/4)\delta(t)$ |
| (d) $\left(\frac{\sin(\pi t-\pi)}{t^2+4}\right)\delta(t+1)$ |
| (e) $\left(\frac{1}{j\omega+2}\right)\delta(\omega-3)$ |
| (f) $\left(\frac{\sin(k\omega)}{\omega}\right)\delta(\omega)$ |

Hint: For part (f), use L'Hôpital's rule.

1.3-9 Evaluate the following integrals:

- | |
|---|
| (a) $\int_{-\infty}^{\infty} \delta(\tau)x(t-\tau) d\tau$ |
| (b) $\int_{-\infty}^{\infty} x(\tau)\delta(t-\tau) d\tau$ |
| (c) $\int_{-\infty}^{\infty} \delta(t)e^{j(\omega t+\phi)} dt$ |
| (d) $\int_{-\infty}^{\infty} \cos(3t)\delta(t-\pi) dt$ |
| (e) $\int_{-\infty}^{\infty} [\delta(t+3)e^t + \delta(t)] dt$ |
| (f) $\int_{-\infty}^{\infty} (t^3 + 2t^2 + 2t + 1)\delta(1-t) dt$ |
| (g) $\int_{-\infty}^{\infty} x(2-t)\delta(t/2) dt$ |
| (h) $\int_{-\infty}^{\infty} e^{(t-1)}\delta(2t-6)\cos\left(\frac{\pi}{2}[t-5]\right) dt$ |

Caution: $\int_{-\infty}^{\infty} \delta(at) dt \neq 1$.

1.3-10 Using the signal $x(t)$ shown in Fig. P1.3-10, find and sketch

- | |
|-------------------------------------|
| (a) $x_a(t) = \frac{d}{dt}x(t)$ |
| (b) $x_b(t) = \frac{d^2}{dt^2}x(t)$ |

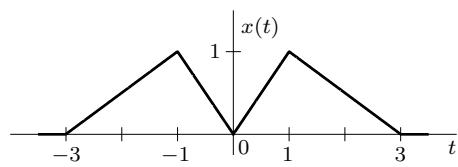


Figure P1.3-10

1.3-11 Using $x(t)$ illustrated in Fig. P1.3-11, find and sketch $y(t) = \int_{-\infty}^t x(\tau) d\tau$.

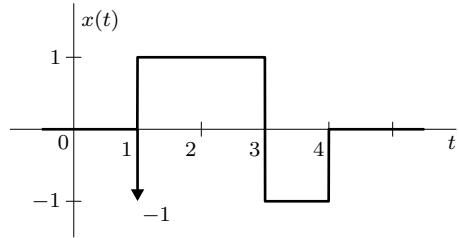


Figure P1.3-11

1.3-12 Using $x(t)$ illustrated in Fig. P1.3-12, find and sketch $y(t) = \int_{-\infty}^t x(\tau) d\tau$.

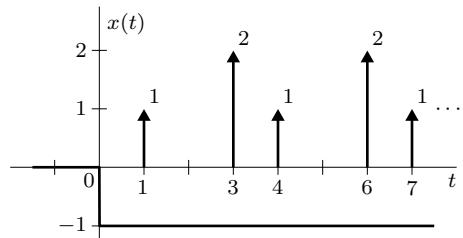


Figure P1.3-12

1.3-13 Sketch $\cos(\omega_1 t)\sin(\omega_2 t)$ when $\omega_1 \gg \omega_2$. What changes if $\omega_1 \ll \omega_2$? What happens if ω_1 and ω_2 are close in value?

1.3-14 Define $x(t) = \Pi(t/3) + \Lambda(t/4)$. Sketch $y(t) = \sum_{k=-3}^3 x(t+6k)$ over a suitable range of t .

1.3-15 Define $x(t) = \Lambda(t/2)$. Sketch $y(t) = \sum_{k=-\infty}^{\infty} (-1)^k x(3t+k)$ over a suitable range of t .

- 1.3-16** Define $x(t) = \frac{1}{2K+1} \sum_{k=-K}^K e^{jkt}$. Using $K = 20$, plot $x(t)$ over $(-2\pi \leq t \leq 2\pi)$. Compare $x(t)$ to a plot of $\text{sinc}(Kt/\pi)$. What might be a good use of $x(t)$?

- 1.4-1** Using Eqs. (1.12) and (1.18), show that a $x(t) = e^{\sigma t} \cos(\omega t)$ can be expressed, using complex frequency $s = \sigma + j\omega$, as a sum of the exponentials e^{st} and e^{-s^*t} . Locate in the complex s -plane the frequencies of

- | | |
|------------------------|------------------------|
| (a) $\cos(2t)$ | (b) $e^{-3t} \cos(2t)$ |
| (c) $3e^{2t} \cos(2t)$ | (d) $2e^{-3t}$ |
| (e) $3e^{2t}$ | (f) 3 |

- 1.4-2** Find the energies of the signals illustrated in Fig. P1.4-2. Comment how sign change, time shift, and halving affect signal energy. What is the effect on the energy if the signal is divided by factor k ?

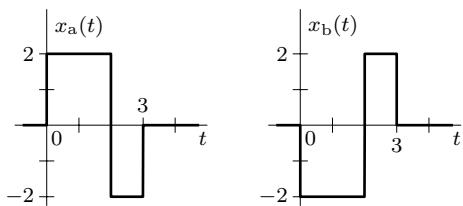
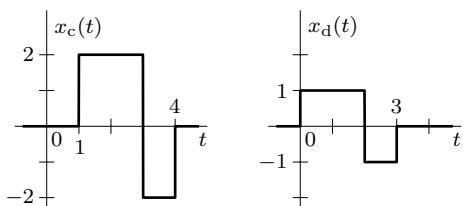


Figure P1.4-2



- 1.4-3** Sketch the even and odd portions of the signals shown in Fig. P1.4-2.

- 1.4-4** Find the energies of the signals illustrated in Fig. P1.4-4. Comment how time shifting, time reversal, and amplitude scaling affect signal energy.

- 1.4-5** Using the signals shown in Fig. P1.4-4, define $f(t) = x_a(t) + jx_b(t)$ and $g(t) = x_c(t) + jx_d(t)$. Further, let $f_{cs}(t)$ be the conjugate-symmetric portion of $f(t)$, and let $g_{ca}(t)$ be the conjugate-antisymmetric portion of $g(t)$. Sketch

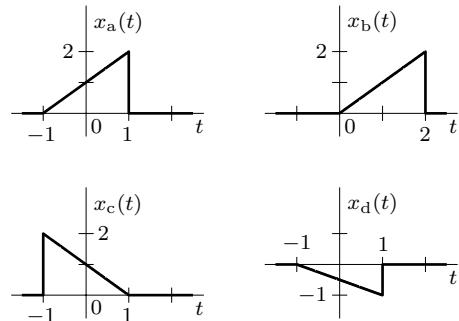


Figure P1.4-4

- | |
|--|
| (a) $y_a(t) = \text{Re}\{f_{cs}(t)\}$ |
| (b) $y_b(t) = \text{Im}\{f_{cs}(t)\}$ |
| (c) $y_c(t) = \text{Re}\{g_{ca}(t)\}$ |
| (d) $y_d(t) = \text{Im}\{g_{ca}(t)\}$ |
| (e) $y_e(t) = \text{Re}\{f(t)g(t)\}$ |
| (f) $y_f(t) = \text{Im}\{f_{cs}(t)g_{ca}(t)\}$ |

- 1.4-6** Find the energy and power of the signal in Fig. P1.2-1.

- 1.4-7** Find the energy and power of the signal in Fig. P1.2-2.

- 1.4-8** Find the power of the 4-periodic signal $x(t)$ shown in Fig. P1.4-8. Find also the powers and the rms values of: (a) $-x(t)$, (b) $\frac{1}{2}x(t)$, and (c) $kx(t)$. Comment.

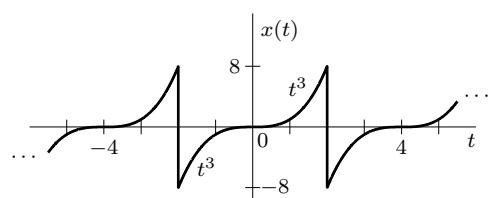


Figure P1.4-8

- 1.4-9** Determine the power and the rms value for each of the following signals:

- | |
|--|
| (a) $5 + 10 \cos(100t + \frac{\pi}{2})$ |
| (b) $7 \cos(100t + \frac{\pi}{3}) + 14 \sin(150t + \frac{\pi}{4})$ |
| (c) $7 \cos(5t) \cos(10t)$ |
| (d) $7 \cos(5t) \sin(10t)$ |
| (e) $\cos(10t)[10 + 2 \sin(3t)]$ |
| (f) $e^{jat} \sin(\omega_0 t)$ |

1.4-10 Consider the signal

$$x(t) = \sum_{k=K_1}^{K_2} X_k e^{j\omega_k t},$$

where all frequencies are distinct ($\omega_k \neq \omega_l$ for all $k \neq l$) but not necessarily harmonically related. Show that the power of this signal is

$$P_x = \sum_{k=K_1}^{K_2} |X_k|^2.$$

1.4-11 Determine whether the signal $e^{\lambda t} u(t)$ is an energy signal, a power signal, or none if

- (a) λ is real and positive
- (b) λ is real and negative
- (c) λ is purely imaginary

1.4-12 Consider a binary signal $x(t)$. For $t < 0$, $x(t) = 0$. For positive time, $x(t)$ toggles between one and zero as follows: one for 1 second, zero for 1 second, one for 1 second, zero for 2 seconds, one for 1 second, zero for 3 seconds, and so forth. That is, the “on” time is always one second but the “off” time increases by one second between each toggle. A portion of $x(t)$ is shown in Fig. P1.4-12. Determine the energy and power of $x(t)$.

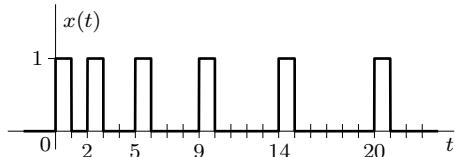


Figure P1.4-12

1.4-13 Show that the energy of $x(t - t_0)$ is still E_x , the energy for $x(t)$. That is, show that shifting a signal does not affect its energy.

1.4-14 Show that the energy of $x(at)$ is $\frac{1}{|a|} E_x$. That is, time scaling a signal by a reciprocally scales the original signal energy E_x by $\frac{1}{|a|}$.

1.4-15 The signal shown in Fig. P1.4-15 is defined as

$$x(t) = \begin{cases} t & 0 \leq t < 1 \\ 0.5 + 0.5 \cos(2\pi t) & 1 \leq t < 2 \\ 3 - t & 2 \leq t < 3 \\ 0 & \text{otherwise} \end{cases}.$$

The energy of $x(t)$ is $E_x \approx 1.0417$.

(a) What is the energy E_y of $y(t) = \frac{1}{3}x(2t)$?

(b) A periodic signal $g(t)$ is defined as

$$g(t) = \begin{cases} x(t) & 0 \leq t < 4 \\ g(t+4) & \forall t \end{cases}.$$

What is the power P_g of $g(t)$? What is the power of $h(t) = \frac{1}{3}g(2t)$?

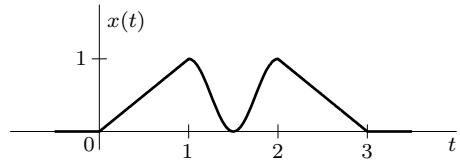


Figure P1.4-15

1.4-16 Let $x_a(t) = x_b(t) = t^2$ over $(0 \leq t \leq 1)$. Notice, this statement does not require $x_a(t) = x_b(t)$ for all t .

- (a)** Define $x_a(t)$ as an even, periodic signal with period $T_a = 2$. Sketch $x_a(t)$ and determine its power.
- (b)** Design an odd, periodic signal $x_b(t)$ with period $T_b = 3$ and power equal to unity. Fully describe $x_b(t)$ and sketch the signal over at least one full period. Hint: There are an infinite number of possible solutions to this problem – you only need to find one of them!
- (c)** Using your results from (a) and (b), create a complex-valued function $x_c(t) = x_a(t) + jx_b(t)$. Determine whether or not this signal is periodic. If yes, determine the period T_c . If no, justify why the signal is not periodic. In either case, compute the power of $x_c(t)$.

1.4-17 Consider three signals: $x_a(t) = \cos(t)$, $x_b(t) = \sin(\pi t)$, and $x_c(t) = x_a(t) + x_b(t)$.

- (a)** For signal $x_a(t)$, determine its fundamental period T_a as well as its power.
- (b)** For signal $x_b(t)$, determine its fundamental period T_b as well as its power.
- (c)** Show that signal $x_c(t)$ is not periodic. Determine the power of $x_c(t)$.

1.4-18 Figure P1.4-18 plots a complex signal $w(t)$ in the complex plane over the time range $(0 \leq t \leq 1)$. The time $t = 0$ corresponds with the origin, while the time $t = 1$ corresponds with the point $(2,1)$. In the complex plane, plot $w(t)$ over $(-1 \leq t \leq 1)$ if

- (a) $w(t)$ is an even signal
- (b) $w(t)$ is an odd signal
- (c) $w(t)$ is a conjugate-symmetric signal
- (d) $w(t)$ is a conjugate-antisymmetric signal

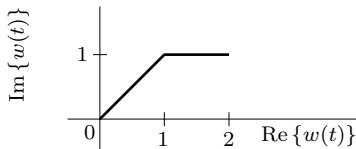


Figure P1.4-18

1.5-1 Supporting your answer mathematically, determine whether the following systems, which have input $x(t)$ and output $y(t)$, are linear or nonlinear:

- (a) $\frac{d}{dt}y(t) + 3y(t) = x^2(t)$
- (b) $\frac{d}{dt}y(t) + 2ty(t) = t^2x(t)$
- (c) $2y(t) + [\frac{d}{dt}y(t)]^2 = 4x(t)$
- (d) $\frac{d}{dt}y(t) + y^2(t) = x(t)$
- (e) $3y(t) + 2 = x(t)$
- (f) $\frac{d}{dt}y(t) + \sin(t)y(t) = \frac{d}{dt}x(t) + 2x(t)$
- (g) $\frac{d}{dt}y(t) + 2y(t) = x(t)\frac{d}{dt}x(t)$
- (h) $y(t) = \int_{-\infty}^t x(\tau) d\tau$

1.5-2 Supporting your answer mathematically, determine whether or not the following systems, which have input $x(t)$ and output $y(t)$, are time invariant:

- (a) $y(t) = x(t - 2)$
- (b) $y(t) = x(-t)$
- (c) $y(t) = x(at)$
- (d) $y(t) = t x(t - 2)$
- (e) $y(t) = \int_{-5}^5 x(\tau) d\tau$
- (f) $y(t) = [\frac{d}{dt}x(t)]^2$

1.5-3 A system is specified by its input-output relationship as

$$y(t) = x^2(t) / [\frac{d}{dt}x(t)].$$

Show that the system satisfies the homogeneity property but not the additivity property.

1.5-4 Suppose a certain LTIC system with input $x(t)$, output $y(t)$, and two initial conditions $x_1(0)$ and $x_2(0)$ produces the following observations:

$x(t)$	$x_1(0)$	$x_2(0)$	$y(t)$
0	1	-1	$e^{-t}u(t)$
0	2	1	$e^{-t}(3t + 2)u(t)$
$u(t)$	-1	-1	$2u(t)$

Determine $y(t)$ when both the initial conditions are zero and the input $x(t)$ is as shown in Fig. P1.5-4.

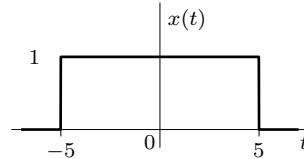


Figure P1.5-4

Hint: There are three causes: the input and each of the two initial conditions. Because of the linearity property, if a cause is increased by a factor k , the response to that cause also increases by the same factor k . Moreover, if causes are added, the corresponding responses add.

1.5-5 The unit impulse response of an LTIC system is $h(t) = e^{-t}u(t)$. Find this system's zero-state response $y(t)$ if the input $x(t)$ is

- | | |
|-------------------|--------------------|
| (a) $u(t)$ | (b) $e^{-t}u(t)$ |
| (c) $e^{-2t}u(t)$ | (d) $\sin(3t)u(t)$ |

1.5-6 Repeat Prob. 1.5-5 using

$$h(t) = e^{-t}u(t - 2).$$

1.5-7 Repeat Prob. 1.5-5 using

$$h(t) = (2e^{-3t} - e^{-2t})u(t).$$

- 1.5-8** Repeat Prob. 1.5-5 for a *complex* system that has impulse response

$$h(t) = e^{(j-t)t} u(t).$$

- 1.5-9** Determine the unit step response of a system that has impulse response

$$h(t) = (1 - 2t)e^{-2t} u(t).$$

That is, determine the output $y(t)$ in response to input $x(t) = u(t)$.

- 1.5-10** Supporting your answer mathematically, determine whether or not the following systems, which have input $x(t)$ and output $y(t)$, are causal:

- (a) $y(t) = x(t - 2)$
- (b) $y(t) = x(-t)$
- (c) $y(t) = x(at)$, where $a > 1$
- (d) $y(t) = x(at)$, where $a < 1$
- (e) $y(t) = \int_{t-5}^{t+5} x(\tau) d\tau$

- 1.5-11** A system is given by $y(t) = \frac{d}{dt}x(t - 1)$.

- (a) Is the system linear? Justify your answer.
- (b) Is the system time-invariant? Justify your answer.
- (c) Is the system causal? Justify your answer.
- (d) Is the system BIBO stable? Hint: Let system input $x(t)$ be a square wave.

- 1.6-1** Design a periodic signal $x(t)$ that oscillates 20 times per second, reaches a peak amplitude of 2 at $t = \pi$, and will pass through *any* LTIC system undistorted in shape.

- 1.6-2** Find the response of an LTIC system with transfer function $H(s) = \frac{s+2}{s^2+5s+4}$ to the following everlasting sinusoidal inputs:

- (a) $5 \cos(2t + \pi/6)$
- (b) $10 \sin(2t + \pi/4)$
- (c) $10 \cos(3t + 40^\circ)$

- 1.6-3** Consider a system with transfer function $H(s) = \frac{-(s-10)}{s+10}$. Determine the system

response to the following everlasting sinusoids:

- | | |
|---------------------|-------------------------------|
| (a) $\cos(t)$ | (b) $\sin(2t)$ |
| (c) $\cos(10t)$ | (d) $\cos(100t)$ |
| (e) $e^{j\omega t}$ | (f) $\cos(\omega t + \theta)$ |

Does this system affect the magnitude of the inputs? Does the system affect the phase of the inputs? Comment.

- 1.7-1** Find the exponential Fourier series and sketch the corresponding spectra for the 2-periodic signal $x(t)$ shown in Fig. P1.7-1.

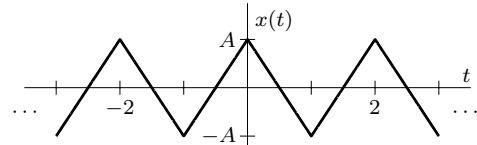


Figure P1.7-1

- 1.7-2** Repeat Prob. 1.7-1 for the π -periodic signal $x(t)$ shown in Fig. P1.7-2.

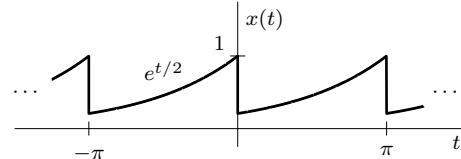


Figure P1.7-2

- 1.7-3** If the two halves of one period of a periodic signal are of identical shape except that the one is the negative of the other, the periodic signal is said to have *half-wave symmetry*. If a periodic signal $x(t)$ with a period T_0 satisfies the half-wave symmetry condition, then

$$x\left(t - \frac{T_0}{2}\right) = -x(t).$$

In this case, show that all the even-numbered harmonics vanish and that the odd-numbered harmonic coefficients are given by

$$X_k = \frac{2}{T_0} \int_0^{T_0/2} x(t) e^{-jk\omega_0 t} dt.$$

Using these results, find the exponential Fourier series for the periodic signals shown in Fig. P1.7-3.

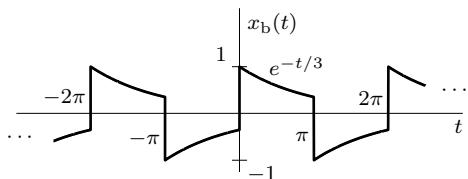
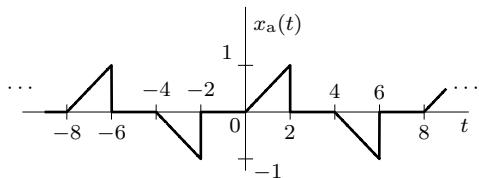


Figure P1.7-3

- 1.7-4** Determine the complex Fourier series coefficients X_k for the *complex* signal $x(t) = e^{j2t} + e^{j3t}$. What is the fundamental period T_0 and fundamental frequency ω_0 of $x(t)$?

- 1.7-5** The exponential Fourier series of a certain periodic function is given as $x(t) = (2 + j2)e^{-j3t} + j2e^{-jt} + 3 - j2e^{jt} + (2 - j2)e^{j3t}$.

- (a) Sketch the exponential Fourier spectra.
- (b) By inspection of the spectra in part (a), sketch the trigonometric Fourier spectra for $x(t)$.
- (c) Find the compact trigonometric Fourier series from the spectra found in either part (a) or part (b).
- (d) Determine the bandwidth of $x(t)$.

- 1.7-6** Show that

$$\begin{aligned}\frac{\pi}{4} &= 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots \\ &= \cos(1) - \frac{1}{3} \cos(3) + \frac{1}{5} \cos(5) - \dots \\ &= \sin(1) + \frac{1}{3} \sin(3) + \frac{1}{5} \sin(5) + \dots\end{aligned}$$

- 1.7-7** Find the trigonometric Fourier series and sketch the corresponding Fourier spectra for the periodic signal $x(t)$ shown in Fig. P1.7-1.

- 1.7-8** Repeat Prob. 1.7-7 for the periodic signal $x(t)$ shown in Fig. P1.7-2.

- 1.7-9** Show that if a periodic signal $x(t)$ is an even function of t , then all the sine terms in its trigonometric Fourier series vanish,

and if $x(t)$ is an odd function of t , all the cosine terms vanish. In either case, show that to compute the Fourier coefficients, we need to evaluate integrals over only half the period.

- 1.7-10** A certain periodic signal is given by $x(t) = 3 + \sqrt{3} \cos(2t) + \sin(2t) + \sin(3t) - \frac{1}{2} \cos\left(5t + \frac{\pi}{3}\right)$.

- (a) Sketch the trigonometric Fourier spectra.
- (b) By inspection of the spectra in part (a), sketch the exponential Fourier series spectra for $x(t)$.
- (c) By inspection of the spectra in part (b), write the exponential Fourier series for $x(t)$.

- 1.7-11** Consider the 2π -periodic signal $x(t)$ defined as

$$x(t) = \begin{cases} t/A & 0 \leq t < A \\ 1 & A \leq t < \pi \\ 0 & \pi \leq t < 2\pi \\ x(t + 2\pi) & \forall t \end{cases}.$$

For small A , $x(t)$ looks like a square wave; as $A \rightarrow \pi$, $x(t)$ resembles a type of sawtooth wave.

- (a) Determine the Fourier series X_k for signal $x(t)$. Sketch the corresponding magnitude and phase spectra.

Next, synthesize $x(t)$ using a truncated Fourier series $x_K(t)$, as given in Eq. (1.61). Using the following parameters, accurately plot $x_K(t)$ and comment on the approximation, with particular attention to Gibbs phenomenon.

- (a) $A = \pi/2$ and $K = 20$
- (b) $A = \pi/2$ and $K = 100$
- (c) $A = \pi/64$ and $K = 20$
- (d) $A = \pi/64$ and $K = 100$

- 1.8-1** Using the definition of Eq. (1.75), find the Fourier transforms of the signals depicted in Fig. P1.8-1.

- 1.8-2** Using the definition of Eq. (1.74), find the inverse Fourier transforms of the spectra in Fig. P1.8-2.

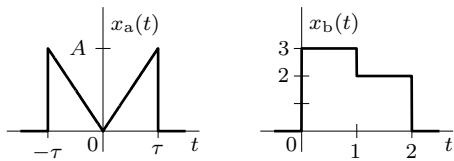


Figure P1.8-1

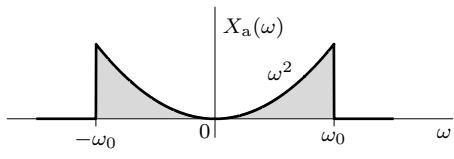


Figure P1.8-2

- 1.8-3** Using the definition of Eq. (1.74), show that the inverse Fourier transform of $\Pi\left(\frac{\omega-10}{2\pi}\right)$ is $\text{sinc}(t)e^{j10t}$.

- 1.8-4** Using the spectra shown in Fig. P1.8-4, find the inverse Fourier transform of $X_a(\omega)$ and $X_b(\omega)$. This problem illustrates how just a difference in phase spectra can produce entirely different time-domain signals.

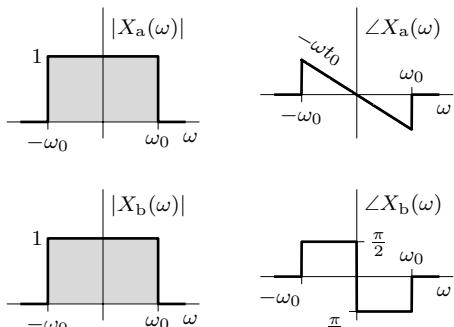


Figure P1.8-4

- 1.8-5** Determine the Fourier transform of a Gaussian pulse defined as $x(t) = e^{-t^2}$. Plot both $x(t)$ and $X(\omega)$. How do the two curves compare? Hint:

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{(t-a)^2}{2}} dt = 1.$$

- 1.9-1** The Fourier transform of the triangular pulse $x(t)$ in Fig. P1.2-6 is expressed as

$$X(\omega) = \frac{1}{\omega^2} (e^{-j\omega} + j\omega e^{-j\omega} - 1).$$

Using this information and the time-shifting and time-scaling properties, find the Fourier transforms of the signals $x_a(t)$, $x_b(t)$, $x_c(t)$, $x_d(t)$, and $x_e(t)$, also shown in Fig. P1.2-6.

- 1.9-2** Using only the time-shifting property and Table 1.1, find the Fourier transforms of the signals depicted in Fig. P1.9-2.

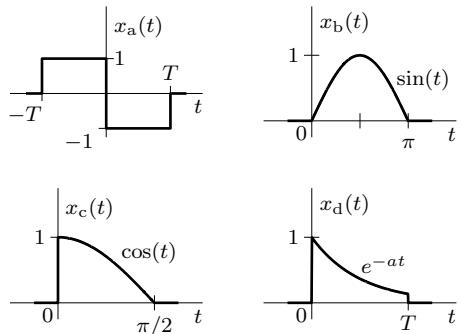


Figure P1.9-2

- 1.9-3** The signals in Fig. P1.9-3 are modulated signals with carrier $\cos(10t)$. Find the Fourier transforms of these signals using the appropriate properties of the Fourier transform and Table 1.1. Sketch the magnitude and phase spectra.

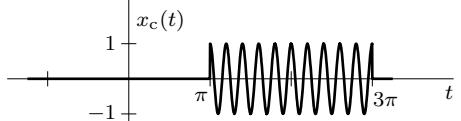
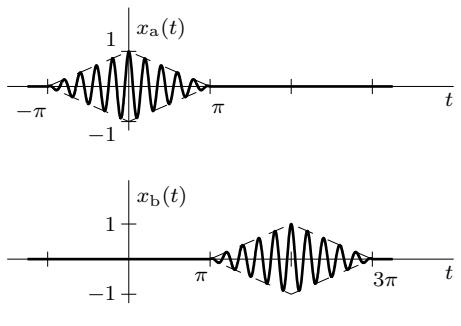


Figure P1.9-3

1.9-4 Show that

$$\int_{-\infty}^{\infty} \text{sinc}^2(kx) dx = \frac{1}{k}.$$

Hint: Identify the integral as the energy of an appropriate signal.

1.9-5 Apply the duality property to the appropriate pairs in Table 1.1 to show that

- (a) $\frac{1}{2}(\delta(t) + j/\pi t) \iff u(\omega)$
- (b) $\delta(t+T) + \delta(t-T) \iff 2\cos(T\omega)$
- (c) $\delta(t+T) - \delta(t-T) \iff 2j\sin(T\omega)$

1.9-6 Suppose that a signal $x(t)$ is bandlimited to B Hz. Show that the signal $x^k(t)$ is bandlimited to kB Hz.

1.9-7 Consider a signal $x(t)$ with Fourier transform $X(\omega) = \Pi\left(\frac{\omega-3}{2}\right) - \Pi\left(\frac{\omega+3}{2}\right)$. Use the frequency-shifting property and Table 1.1 to determine $x(t) = \mathcal{F}^{-1}\{X(\omega)\}$.

1.9-8 Consider a signal $x(t)$ with Fourier transform $X(\omega) = \Lambda\left(\frac{\omega+3}{2}\right) - \Lambda\left(\frac{\omega-3}{2}\right)$. Use the frequency-shifting property and Table 1.1 to determine $x(t) = \mathcal{F}^{-1}\{X(\omega)\}$.

1.9-9 Recall that the unit-amplitude gate function with duration τ is denoted as $\Pi(t/\tau)$.

- (a) Determine τ_a that results in a 95% essential bandwidth of 5 Hz.
- (b) Determine τ_b that results in a 90% essential bandwidth of 10 Hz.
- (c) Determine τ_c that results in a 75% essential bandwidth 20 Hz.

1.9-10 Consider the signal $x(t) = e^{-at}u(t)$, where a is real and > 0 .

- (a) Determine a_a that results in a 95% essential bandwidth of 5 Hz.
- (b) Determine a_b that results in a 90% essential bandwidth of 10 Hz.
- (c) Determine a_c that results in a 75% essential bandwidth 20 Hz.

1.9-11 Using MATLAB, determine the essential bandwidth of the unit triangle function $\Lambda(t)$ using energy criteria of

- (a) 99%
- (b) 95%
- (c) 90%
- (d) 75%

1.10-1 By direct integration, determine the bilateral Laplace transforms, including ROCs, of the following functions:

- (a) $x_a(t) = u(1-t) - u(-t)$
- (b) $x_b(t) = te^{-2t}u(t)$
- (c) $x_c(t) = e^{-j\omega_0 t}u(t)$
- (d) $x_d(t) = (e^{2t} + 2e^{-t})u(-t)$
- (e) $x_e(t) = \cosh(t)u(-t)$
- (f) $x_f(t) = e^{-t} \cos(10t)u(t)$
- (g) $x_g(t) = e^{-|t|}$
- (h) $x_h(t) = e^{-2tu(t)}$

1.10-2 By direct integration, determine the bilateral Laplace transforms, including ROCs, of the signals shown in Fig. P1.9-2.

1.10-3 Using the definition, determine the bilateral Laplace transforms, including ROCs, of the following complex-valued functions:

- (a) $x_a(t) = (j + e^{jt})u(t)$
- (b) $x_b(t) = j \cosh(t)u(-t)$
- (c) $x_c(t) = e^{j(\frac{\pi}{4})}u(-t+1) + j\delta(t-5)$
- (d) $x_d(t) = j^t u(-t) + \delta(t-\pi)$

1.10-4 Determine the inverse Laplace transforms of the following functions:

- (a) $X_a(s) = \frac{2s+5}{s^2+5s+6}$,
ROC: $(\text{Re}\{s\} > -2)$
- (b) $X_b(s) = \frac{2s+5}{s^2+5s+6}$,
ROC: $(-3 < \text{Re}\{s\} < -2)$
- (c) $X_c(s) = \frac{2s+1}{(s+1)(s^2+2s+2)}$,
ROC: $(\text{Re}\{s\} > -1)$
- (d) $X_d(s) = \frac{2s+1}{(s+1)(s^2+2s+2)}$,
ROC: $(\text{Re}\{s\} < -1)$
- (e) $X_e(s) = \frac{(s+1)^2}{s^2-s-6}$,
ROC: $(-3 < \text{Re}\{s\} < 2)$
- (f) $X_f(s) = \frac{s+2}{(s-1)(s+1)}$,
ROC minimizes the energy of $x_f(t)$

1.10-5 Using Table 1.5 and the properties found in Table 1.6, determine the unilateral Laplace transforms of the following functions:

- (a) $x_a(t) = u(1-t) - u(-t)$
- (b) $x_b(t) = e^{-(t-\tau)}u(t-\tau)$

(c) $x_c(t) = e^{-(t+\tau)}u(t - \tau)$

(d) $x_d(t) = te^{-t}u(t - \tau)$

(e) $x_e(t) = \sin(\omega_0 t)u(t - \tau)$

1.10-6 Using Table 1.5 and the properties found in Table 1.6, determine the bilateral Laplace transforms, including ROCs, of the signals shown in Fig. P1.9-2.

1.10-7 Determine the inverse unilateral Laplace transform of

$$X(s) = \frac{1}{e^{s+3}} \frac{s^2}{(s+1)(s+2)}.$$

1.10-8 Consider a LTIC system with causal impulse response $h(t)$.

- (a) Using the convolution property, show that the transfer function of an LTIC system is given by a transform-domain ratio of output to input, $H(s) = Y(s)/X(s)$.
- (b) Using Laplace transform properties, determine the transfer function of a system described by $y(t) + \frac{d}{dt}y(t) = x(t)$.
- (c) Using Table 1.5, determine the impulse response $h(t)$ of a system described by $y(t) + \frac{d}{dt}y(t) = x(t)$.
- (d) Determine the output $y(t)$ of a system described by $y(t) + \frac{d}{dt}y(t) = x(t)$ to the everlasting sinusoidal input $x(t) = \cos(t)$.

1.10-9 Two students, Amy and Jeff, disagree about an analog system function given by $H_a(s) = s$. Sensible Jeff claims the system has a zero at $s = 0$. Rebellious Amy, however, notes that the system function can be rewritten as $H_a(s) = \frac{1}{s-1}$, and claims that this implies a system pole at $s = \infty$. Who is correct? Why? What are the poles and zeros of the system $H_b(s) = \frac{1}{s}$?

Chapter 2

Continuous-Time Analog Filters

This chapter completes our review of continuous-time signals and systems by investigating the topic of frequency response. A proper understanding of frequency response is necessary to develop effective analysis and design tools for continuous-time analog filters. Initially, we consider the frequency-domain analysis and characterization of LTIC systems. Ideal and practical filter characteristics are reviewed, and filter implementation issues are considered. Next, we discuss the specification of practical filters and detail analog filter transformation techniques. We conclude with the development of traditional CT analog filter design methods, including several detailed design examples.

2.1 Frequency Response of an LTIC System

As discovered in the previous chapter, the impulse response $h(t)$ of an LTIC system completely characterizes its zero-state behavior. Applying the Laplace transform to the impulse response yields the system transfer function $H(s) = \mathcal{L}\{h(t)\}$. Clearly, the system transfer function shares a close kinship with the unit impulse response, and together they form a unique transform pair. Knowing one ensures the other is known as well.

From Eq. (1.47), we know that we can find $H(s)$ as the *time-domain* ratio of system output to input as long as the input is an everlasting exponential e^{st} . Consequently, $H(s)$ is fundamentally linked to sinusoidal (exponential) steady-state (everlasting) analysis. Section 1.6 demonstrates that the LTIC system response to an everlasting exponential input $x(t) = e^{st}$ is also an everlasting exponential, $H(s)e^{st}$. We represent this input-output pair using an arrow directed from the input to the output as

$$e^{st} \implies H(s)e^{st}. \quad (2.1)$$

Setting $s = \pm j\omega$ in this relationship yields[†]

$$e^{j\omega t} \implies H(j\omega)e^{j\omega t} \quad \text{and} \quad e^{-j\omega t} \implies H(-j\omega)e^{-j\omega t}.$$

The addition of these two relationships yields

$$2 \cos(\omega t) \implies H(j\omega)e^{j\omega t} + H(-j\omega)e^{-j\omega t}. \quad (2.2)$$

As long as the system is real[‡], $H(-j\omega) = H^*(j\omega)$, and Eq. (2.2) simplifies to

$$\cos(\omega t) \implies \operatorname{Re}\{H(j\omega)e^{j\omega t}\}. \quad (2.3)$$

[†]The results that follow, where we have let $s = j\omega$, are valid for asymptotically stable systems where the region of convergence of $H(s)$ includes the ω -axis, although useful insights are possible for marginally stable systems as well.

[‡]A system is called real if real-valued inputs always generate real-valued outputs. Real systems have real-valued impulse responses and, therefore, conjugate-symmetric frequency responses.

This equation also follows from a property of real linear systems that

$$\text{if } x(t) \implies y(t), \text{ then } \operatorname{Re}\{x(t)\} \implies \operatorname{Re}\{y(t)\}. \quad (2.4)$$

In the present case, $e^{j\omega t} \implies H(j\omega)e^{j\omega t}$, so $\cos(\omega t) = \operatorname{Re}\{e^{j\omega t}\} \implies \operatorname{Re}\{H(j\omega)e^{j\omega t}\}$.

We can express $H(j\omega)$ in the polar form as

$$H(j\omega) = |H(j\omega)| e^{j\angle H(j\omega)}.$$

When substituted into Eq. (2.3), the relationship becomes

$$\cos(\omega t) \implies |H(j\omega)| \cos[\omega t + \angle H(j\omega)].$$

This result is readily generalized to

$$A \cos(\omega t + \theta) \implies |H(j\omega)| A \cos[\omega t + \theta + \angle H(j\omega)]. \quad (2.5)$$

Equation (2.5) shows that a sinusoid of radian frequency ω input to a real system yields a sinusoid of the same frequency ω . *The magnitude of the output sinusoid is $|H(j\omega)|$ times the input magnitude, and the phase of the output sinusoid is shifted by $\angle H(j\omega)$ with respect to the input phase.* For instance, if a certain system has $|H(j10)| = 3$ and $\angle H(j10) = -\pi/6$, then the system amplifies a sinusoid of frequency $\omega = 10$ by a factor of 3 and shifts its phase by $\pi/6$ radians. The system response to an input $4 \cos(10t + \pi/4)$ is $3 \times 4 \cos(10t + \pi/4 - \pi/6) = 12 \cos(10t + \pi/12)$.

Clearly, $|H(j\omega)|$ is the system *gain*, and a plot of $|H(j\omega)|$ versus ω shows the system gain as a function of frequency ω . This function is known as the *magnitude response* and is frequently expressed in a decibel scale as $20 \log_{10} |H(j\omega)|$. Similarly, $\angle H(j\omega)$ is the *phase response*, and a plot of $\angle H(j\omega)$ versus ω shows how the system modifies or changes the phase of an input sinusoid. These two plots together, as functions of ω , are called the *frequency response of the system*. Observe that $H(j\omega)$ has the information of both $|H(j\omega)|$ and $\angle H(j\omega)$. For this reason, $H(j\omega)$ is also called the frequency response of the system. Frequency response plots show at a glance how a system responds to sinusoids of various frequencies. Thus, the frequency response of a system portrays the system's filtering characteristics.

Needless to say, this discussion also applies to $H(\omega)$, the Fourier transform of a system's impulse response $h(t)$. If $h(t)$ is absolutely integrable, then the ROC for $H(s)$ contains the ω -axis, the system is BIBO stable, and

$$H(\omega) = H(s)|_{s=j\omega} = H(j\omega).$$

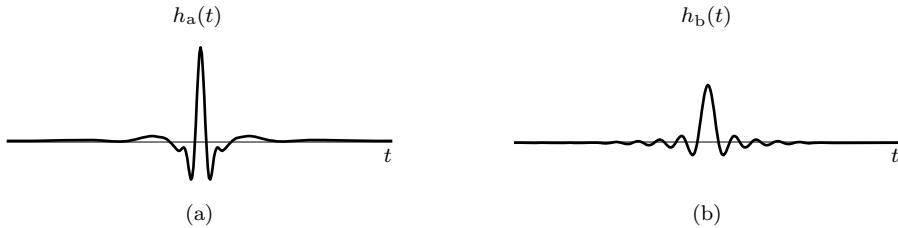
In other words, $H(j\omega)$ and the Fourier transform $H(\omega)$ are equally suited to serve as the frequency response of a system. When units of hertz are preferred, frequency response is expressed as $H(f) = H(\omega)|_{\omega=2\pi f}$. In any case, modern computing technologies make it simple to generate accurate magnitude and phase response plots.

The Better Orator?

Since $h(t)$ and $H(f)$ form a unique transform pair, we know that either is fully qualified to speak on behalf of a system. It is therefore natural to ask which, if either, is the better orator. To develop insight into this question, we consider the familiar topic of human hearing.

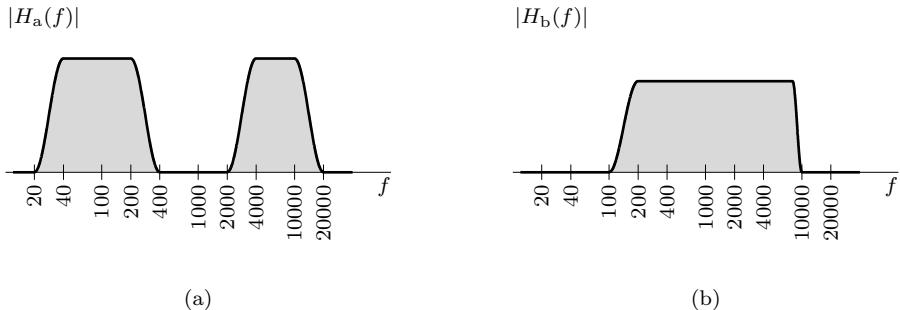
If we model a person's hearing as an LTIC system, the corresponding impulse response and frequency response functions both characterize how input sound is heard. One is a time-domain characterization, and the other is a frequency-domain characterization. Which function, $h(t)$ or $H(f)$, provides the most intuitive understanding of the person's hearing?

Figure 2.1 illustrates two possible impulse responses $h_a(t)$ and $h_b(t)$ that describe the respective hearing of two people named, for convenience, Abe and Beatrice. Each response plot is at the same scale over approximately 2 ms. Although $h_a(t)$ and $h_b(t)$ completely characterize the hearing of Abe and Beatrice, neither provides a particularly intuitive description. Still, some useful information is

Figure 2.1: Hearing impulse responses: (a) $h_a(t)$ and (b) $h_b(t)$.

present. Since $h_a(t) \neq h_b(t)$, we know Abe and Beatrice will hear the same input differently. Neither possesses perfect hearing, which would require $h(t) = \delta(t)$. We might surmise that the person with the most delta-like impulse response has the best hearing, but it is not obvious from Fig. 2.1 which response is most like $\delta(t)$. Figure 2.1 simply does not make clear which person hears the best.

Using a logarithmic frequency scale, Fig. 2.2 displays the magnitude responses $|H_a(f)|$ and $|H_b(f)|$ for Abe and Beatrice, respectively. Once again, we can quickly establish that Abe and Beatrice hear differently (since $|H_a(f)| \neq |H_b(f)|$), and neither possesses perfect hearing (since neither $|H_a(f)|$ nor $|H_b(f)|$ equals 1). Many additional characteristics, however, are also easy to establish. In the case of Abe, we see that his hearing range spans from 20 Hz to 20 kHz, although he has significant hearing loss between 200 Hz and 4 kHz. This loss is particularly important since it overlaps the primary frequencies that constitute human speech. When you speak to Abe, he's not going to hear you well. Beatrice, on the other hand, hears over a smaller range from about 100 Hz to 10 kHz, but she does not suffer from significant hearing loss in the frequency range of human speech. When you speak, Beatrice will hear you much better than Abe. Clearly in this case, *the frequency response description provides a more intuitive understanding of system behavior than the impulse response description*. Put another way, $H(f)$ is a better orator than $h(t)$. This observation holds true for most systems in general.

Figure 2.2: Hearing magnitude responses: (a) $|H_a(f)|$ and (b) $|H_b(f)|$.

▷ **Example 2.1 (Frequency Responses of Common Systems)**

Determine and plot the frequency responses (magnitude and phase) for

- | | |
|---|-----------------------------|
| (a) an ideal time delay of T seconds
(c) an ideal integrator | (b) an ideal differentiator |
|---|-----------------------------|

(a) An Ideal Time Delay of T Seconds

As shown in Ex. 1.4 on page 31, the transfer function of an ideal T -second delay is $H(s) = e^{-sT}$.

Therefore, $H(j\omega) = e^{-j\omega T}$. The corresponding magnitude and phase responses are

$$|H(j\omega)| = 1 \quad \text{and} \quad \angle H(j\omega) = -\omega T. \quad (2.6)$$

Plots of the magnitude and phase responses are shown in Fig. 2.3. The magnitude response is constant (unity) for all frequencies, and the phase response increases linearly with frequency with a slope of $-T$ (linear phase shift).

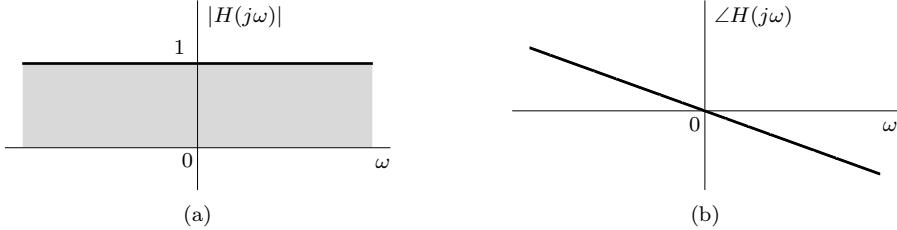


Figure 2.3: An ideal T -second delay: (a) magnitude response and (b) phase response.

An ideal time delay of T seconds is a distortionless system, which merely delays the input signal by T seconds. The magnitude spectrum of the output of this device is identical to that of the input, but the phase spectrum of the output has an additional term $-\omega T$. This added phase is a linear function of ω with slope $-T$ (the negative of the delay). This result is consistent with the time-shifting property of Eq. (1.87), which shows that delaying a signal by T seconds leaves its magnitude spectrum unaltered but changes its phase spectrum by $-\omega T$.

(b) An Ideal Differentiator

As shown in Ex. 1.4, the transfer function of an ideal differentiator is $H(s) = s$. Therefore, $H(j\omega) = j\omega = \omega e^{j\pi/2}$. Consequently,

$$|H(j\omega)| = |\omega| \quad \text{and} \quad \angle H(j\omega) = \frac{\pi}{2} \operatorname{sgn}(\omega). \quad (2.7)$$

The magnitude and phase responses are shown in Fig. 2.4. The magnitude response increases linearly with frequency, and the phase response is a constant $\pi/2$ for all positive frequencies ($-\pi/2$ for negative frequencies). This result can be explained physically by recognizing that if a sinusoid $\cos(\omega t)$ is passed through an ideal differentiator, the output is $-\omega \sin(\omega t) = \omega \cos(\omega t + \frac{\pi}{2})$. Therefore, the output sinusoid amplitude is ω times the input amplitude; that is, the magnitude response (gain) increases linearly with frequency ω . Moreover, the output sinusoid undergoes a constant phase shift $\frac{\pi}{2}$ with respect to the input $\cos(\omega t)$. Therefore, the phase response is a constant $(\pi/2)$ for $\omega \geq 0$.

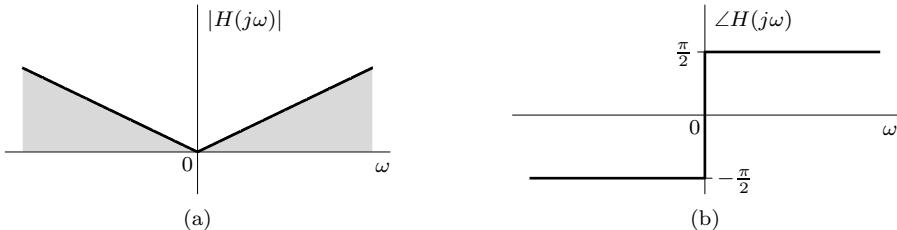


Figure 2.4: An ideal differentiator: (a) magnitude response and (b) phase response.

In an ideal differentiator, the magnitude response (gain) is proportional to frequency ($|H(j\omega)| = |\omega|$) so that the higher-frequency components are enhanced (see Fig. 2.4). Practical signals are contaminated with noise, which, by its nature, is a broad-band (rapidly varying) signal containing

components of very high frequencies. A differentiator therefore amplifies higher frequency components strongly, which can increase the noise disproportionately to the point of drowning out the desired signal. This is the reason why ideal differentiators are typically avoided in practice.

(c) An Ideal Integrator

As shown in Ex. 1.4, the transfer function of an ideal integrator is $H(s) = 1/s$. Therefore, $H(j\omega) = \frac{1}{j\omega} = \frac{-j}{\omega} = \frac{1}{\omega}e^{-j\pi/2}$. Correspondingly,

$$|H(j\omega)| = \frac{1}{|\omega|} \quad \text{and} \quad \angle H(j\omega) = -\frac{\pi}{2} \operatorname{sgn}(\omega). \quad (2.8)$$

The magnitude and phase responses are illustrated in Fig. 2.5. The magnitude response is inversely proportional to frequency, and the phase shift is a constant $-\pi/2$ for all positive frequencies ($\pi/2$ for negative frequencies). This result can be explained physically by recognizing that if a sinusoid $\cos(\omega t)$ is passed through an ideal integrator, the output is $\frac{1}{\omega} \sin(\omega t) = \frac{1}{\omega} \cos(\omega t - \frac{\pi}{2})$. Therefore, the magnitude response is inversely proportional to ω , and the phase response is constant ($-\pi/2$) for $\omega \geq 0$.

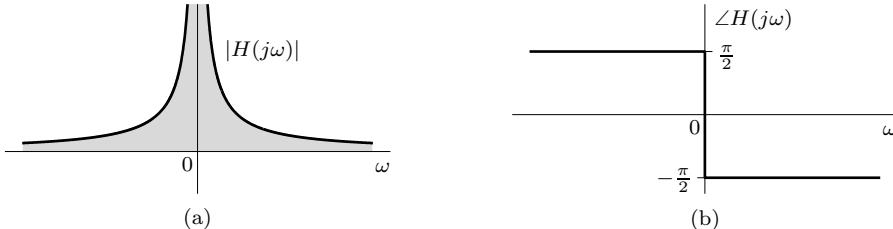


Figure 2.5: An ideal integrator: (a) magnitude response and (b) phase response.

Because its gain is $1/|\omega|$, the ideal integrator suppresses higher-frequency components but enhances lower-frequency components with $\omega < 1$. Consequently, noise signals (if they do not contain an appreciable amount of very low-frequency components) are suppressed (smoothed out) by an integrator.

Example 2.1 \triangleleft

2.1.1 Pole-Zero Plots

Although frequency response plots provide an excellent snapshot of system behavior, the transfer function itself provides additional insight into system behavior and character. Consider the somewhat restricted case where an LTIC system is described by the constant-coefficient linear differential equation[†]

$$a_K \frac{d^K y}{dt^K} + a_{K-1} \frac{d^{K-1} y}{dt^{K-1}} + \cdots + a_1 \frac{dy}{dt} + a_0 y = b_L \frac{d^L x}{dt^L} + b_{L-1} \frac{d^{L-1} x}{dt^{L-1}} + \cdots + b_1 \frac{dx}{dt} + b_0 x.$$

As shown in Ex. 1.4 on page 31, the transfer function $H(s)$ follows directly from the differential equation as

$$H(s) = \frac{B(s)}{A(s)} = \frac{b_L s^L + b_{L-1} s^{L-1} + \cdots + b_1 s + b_0}{a_K s^K + a_{K-1} s^{K-1} + \cdots + a_1 s + a_0} = \left(\frac{b_L}{a_K} \right) \frac{\prod_{l=1}^L (s - z_l)}{\prod_{k=1}^K (s - p_k)}. \quad (2.9)$$

[†]Although restricted in the sense that not all LTIC systems are described by constant-coefficient linear differential equations, this model still accommodates a very broad range of practical systems.

Here, z_l (the system zeros) are the roots of the polynomial $(b_L s^L + b_{L-1} s^{L-1} + \dots + b_1 s + b_0)$, and p_k (the system poles) are the roots of the polynomial $(a_K s^K + a_{K-1} s^{K-1} + \dots + a_1 s + a_0)$.

As Eq. (2.9) makes clear, the frequency response $H(\omega) = H(s)|_{s=j\omega}$ is suppressed at frequencies near a system zero where $(j\omega - z_l)$ is small. Conversely, the frequency response $H(\omega)$ is amplified at frequencies near a system pole where $(j\omega - p_k)$ is small and $1/(j\omega - p_k)$ is large. A *pole-zero plot* locates a system's poles and zeros in the complex s -plane using (traditionally) x's and o's, respectively. Such plots help establish the relative location, impact, and importance of each pole and zero on overall system behavior. Further, as we shall learn later, pole-zero plots are quite useful in guiding proper decisions for system implementation.

▷ **Example 2.2 (Pole-Zero and $H(j\omega)$ Plots to Characterize System Behavior)**

Consider a system whose transfer function is

$$H(s) = \frac{2s}{s^2 + 2s + 5}.$$

Determine and plot the poles and zeros of $H(s)$, and use the pole-zero information to predict overall system behavior. Confirm this predicted behavior by computing and graphing the system's frequency response (magnitude and phase). Lastly, find and plot the system response $y(t)$ for the inputs

$$(a) \quad x_a(t) = \cos(2t) \quad (b) \quad x_b(t) = 2 \sin(4t - \pi/3)$$

By inspection, we note that there is a single finite zero at $s = 0$ (and a second zero at ∞). Although the system poles can be determined using the quadratic formula, MATLAB determines the system poles with less effort.

```
01 z = 0; p = roots([1 2 5])
p = -1+2i -1-2i
```

The pole-zero plot, shown in Fig. 2.6a, is readily constructed using MATLAB.

```
02 subplot(131); plot(real(z),imag(z),'o',real(p),imag(p),'x');
03 xlabel('Re(s)'), ylabel('Im(s)');
```

With a zero at $s = 0$, we expect low-frequency inputs (ω near 0) to be suppressed. Similarly, we expect high-frequency inputs ($|\omega| \rightarrow \infty$) to be suppressed due to the zero at ∞ . The response is amplified at frequencies in close proximity to the poles. In the present case, the frequencies $\omega = \pm 2$ are closest to the system poles, and it is in this region that we expect a relatively strong response.

To verify this behavior, we obtain the frequency response function by substituting $s = j\omega$ into $H(s)$. The magnitude and phase responses are thus

$$|H(j\omega)| = \frac{\sqrt{(2\omega)^2}}{\sqrt{(5 - \omega^2)^2 + (2\omega)^2}} \quad \text{and} \quad \angle H(j\omega) = \operatorname{sgn}(\omega) \frac{\pi}{2} - \tan^{-1} \left(\frac{2\omega}{5 - \omega^2} \right). \quad (2.10)$$

MATLAB plots of the magnitude and phase responses are shown in Figs. 2.6b and 2.6c.

```
04 H = @(s) 2*s./(s.^2+2*s+5); omega = linspace(0,9,1001);
05 subplot(132); plot(omega,abs(H(1j*omega)));
06 xlabel('\omega'); ylabel('|H(j\omega)|');
07 subplot(133); plot(omega,angle(H(1j*omega)));
08 xlabel('\omega'); ylabel('angle H(j\omega)');
```

As expected, the frequency response confirms the behavior predicted from the pole-zero plot. Specifically, in Fig. 2.6b we see bandpass behavior with a peak gain near $|\omega| = 2$ (where the poles are closest) and a gain of zero at $\omega = 0$ and $|\omega| \rightarrow \infty$ (where the zeros are present). Taken together,

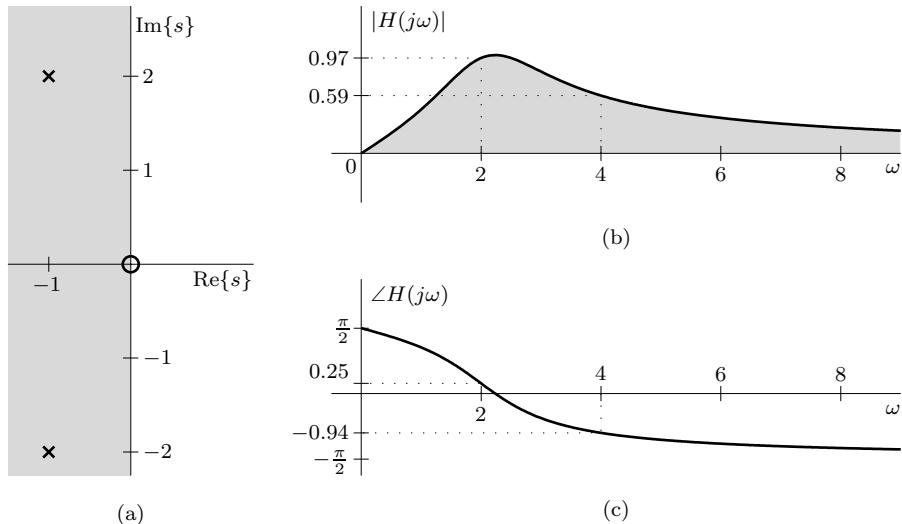


Figure 2.6: Plots for $H(s) = \frac{2s}{s^2 + 2s + 5}$: (a) pole-zero, (b) $|H(j\omega)|$, and (c) $\angle H(j\omega)$.

the magnitude and phase plots of Figs. 2.6b and 2.6c furnish complete information about the response of the system to sinusoidal inputs. The pole-zero plot is nearly as complete in its description of system behavior, failing only to capture the overall gain of the system.

(a) Response to $x_a(t) = \cos(2t)$

In this case, the input frequency is $\omega = 2$, and

$$|H(j2)| = \frac{4}{\sqrt{17}} \approx 0.9701 \quad \text{and} \quad \angle H(j2) = \frac{\pi}{2} - \tan^{-1}\left(\frac{4}{1}\right) \approx 0.2450.$$

Of course, it is also possible to attain these values directly from the magnitude and phase response plots shown in Figs. 2.6a and 2.6b, respectively. This result means that for a sinusoidal input with frequency $\omega = 2$, the magnitude gain of the system is 0.9701, and the phase shift is 0.2450 radians. In other words, the output magnitude is 0.9701 times the input magnitude, and the phase of the output is shifted with respect to that of the input by 0.2450 radians. Using Eq. (2.5), the response to $x_a(t) = \cos(2t)$ is thus

$$y_a(t) = 0.9701 \cos(2t + 0.2450).$$

Figure 2.7a plots the response $y_a(t)$ along with the input $x_a(t)$ for reference. In this case, the output is nearly identical to the input.

```
09 t = linspace(-pi,pi,1001); xa = cos(2*t); ya = abs(H(1j*t2))*cos(2*t+angle(H(1j*t2)));
10 subplot(121); plot(t,ya,t,xa); xlabel('t'); ylabel('y_a(t)');
```

(b) Response to $x_b(t) = 2 \sin(4t - \pi/3)$

Here, the input frequency is $\omega = 4$, and

$$|H(j4)| = \frac{8}{\sqrt{185}} \approx 0.5882 \quad \text{and} \quad \angle H(j4) = \frac{\pi}{2} - \tan^{-1} \left(\frac{8}{-11} \right) \approx -0.9420.$$

Using Eq. (2.5), the response to $x_b(t) = 2 \sin(4t - \pi/3)$ is thus

$$y_b(t) = 2(0.5882) \sin(4t - \pi/3 - 0.9420) = 1.1763 \sin(4t - 1.9892).$$

Figure 2.7b plots $y_b(t)$ with $x_b(t)$ added for reference. Unlike the first case, the input $x_b(t)$ is significantly altered by the system, both in gain and in phase.

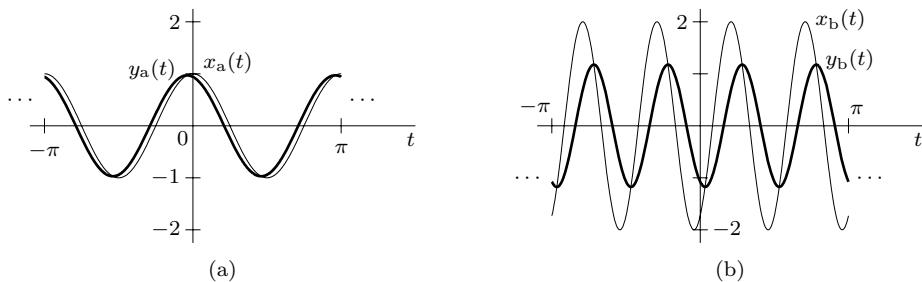


Figure 2.7: $H(s)$ sinusoidal responses: (a) $x_a(t) \Rightarrow y_a(t)$ and (b) $x_b(t) \Rightarrow y_b(t)$.

```

11 xb = 2*sin(4*t-pi/3); yb = abs(H(1j*4))*2*sin(4*t-pi/3+angle(H(1j*4)));
12 subplot(122); plot(t,yb,t,xb); xlabel('t'); ylabel('y_b(t)');

```

Example 2.2 ◁

▷ Drill 2.1 (Characterizing Behavior of a Real LTIC System)

Consider an LTIC system specified by

$$\frac{d^2}{dt^2}y(t) + 3\frac{d}{dt}y(t) + 2y(t) = \frac{d}{dt}x(t) + 5x(t).$$

Determine and plot the poles and zeros of the system, and use the pole-zero information to predict overall system behavior. Confirm this predicted system behavior by computing and graphing the system's frequency response (magnitude and phase). Lastly, find and plot the system response $y(t)$ for the input $x(t) = 20 \sin(3t - \pi/4)$.

► Drill 2.2 (Characterizing Behavior of a Complex LTIC System)

Consider a *complex* LTIC system specified by system function

$$H(s) = \frac{s + 2j}{s^2 + (2 - 4j)s + (-3 - 4j)}.$$

Determine and plot the poles and zeros of the system, and use the pole-zero information to predict overall system behavior. Confirm this predicted behavior by computing and graphing the system's frequency response (magnitude and phase). Lastly, determine the system output $y(t)$ in response to the *real* input $x(t) = 10 \sin(2t + \pi/4)$.

2.2 Signal Transmission through LTIC Systems

As seen in Eq. (1.42), an LTIC system's output is related to its impulse response and input through convolution as

$$y(t) = x(t) * h(t).$$

Assuming that both $x(t)$ and $h(t)$ are Fourier transformable, the time-domain convolution property yields

$$Y(\omega) \equiv X(\omega)H(\omega). \quad (2.11)$$

Here, the cumbersome time-domain convolution operation is transformed into a simple frequency-domain multiplication. Figure 2.8 summarizes these equivalent relationships and emphasizes that, except for unstable systems, $H(\omega)$ serves every bit as well as $h(t)$ to characterize system behavior.

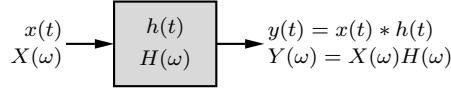


Figure 2.8: Basic block diagram of an LTIC system.

Notice that Eq. (2.11) applies only to asymptotically (and marginally) stable systems because $h(t)$ is not Fourier transformable for unstable systems. Moreover, even for marginally stable systems, the Fourier integral does not converge in the ordinary sense. In addition, $x(t)$ has to be Fourier transformable. Consequently, exponentially growing inputs cannot be handled by this method.

It is important to emphasize that the primary utility of Eq. (2.11) is to gain an *intuitive filtering perspective* of system behavior; it is typically not the most appropriate form for systems analysis. The Laplace transform, which is a generalized Fourier transform, is more versatile and capable of analyzing all kinds of LTIC systems whether stable, unstable, or marginally stable. The Laplace transform can also handle exponentially growing inputs. Compared with the Laplace transform, the Fourier transform in system analysis is clumsier. Hence, the Laplace transform is generally preferable to the Fourier transform for LTIC system analysis. Still, as the next example illustrates, LTIC system analysis is often possible using Fourier transform methods.

▷ Example 2.3 (System Analysis Using the Fourier Transform)

For input $x(t) = e^{-t}u(t)$, find the zero-state response of the stable LTIC system with frequency response

$$H(\omega) = \frac{1}{j\omega + 2}.$$

In this case,

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt = \int_0^{\infty} e^{-t(j\omega+1)} dt = \frac{1}{j\omega + 1}.$$

Therefore,

$$Y(\omega) = X(\omega)H(\omega) = \frac{1}{(j\omega + 1)(j\omega + 2)}.$$

Expanding the right-hand side in partial fractions yields

$$Y(\omega) = \frac{1}{j\omega + 1} - \frac{1}{j\omega + 2}.$$

Thus, the desired zero-state response is

$$y(t) = (e^{-t} - e^{-2t})u(t).$$

Example 2.3 ◁

▷ Drill 2.3 (System Analysis Using the Fourier Transform)

For the system in Ex. 2.3 ($H(\omega) = \frac{1}{j\omega+2}$), show that the zero-state response to the input $x(t) = e^t u(-t)$ is $y(t) = \frac{1}{3} [e^t u(-t) + e^{-2t} u(t)]$.

◁

Spectral Modification Caused by a System

The transmission of an input signal $x(t)$ through a system changes it into an output signal $y(t)$. During transmission through the system, some frequency components may be boosted in amplitude, while others may be attenuated. The relative phases of the various components may also change. In general, the output waveform is different from the input waveform.

Equation (2.11) shows the nature of these changes and provides a frequency-domain or filtering perspective of system behavior. It states that the spectrum of a system's output $Y(\omega)$ is just the input spectrum $X(\omega)$ multiplied by the system frequency response $H(\omega)$. This relationship clearly highlights the spectral shaping (or modification) of the signal by the system, and it can be expressed in polar form as

$$|Y(\omega)|e^{j\angle Y(\omega)} = |X(\omega)| |H(\omega)| e^{j[\angle X(\omega) + \angle H(\omega)]}.$$

Therefore,

$$|Y(\omega)| = |X(\omega)| |H(\omega)| \quad \text{and} \quad \angle Y(\omega) = \angle X(\omega) + \angle H(\omega). \quad (2.12)$$

An input spectral component of frequency ω is modified in magnitude by a factor $|H(\omega)|$ (the magnitude response) and is shifted in phase by an angle $\angle H(\omega)$ (the phase response). Thus, the input's magnitude spectrum $|X(\omega)|$ is changed during transmission to $|X(\omega)| |H(\omega)|$. Similarly, the input's phase spectrum $\angle X(\omega)$ is changed to $\angle X(\omega) + \angle H(\omega)$. Plots of $|H(\omega)|$ and $\angle H(\omega)$ as functions of ω show at a glance how the system modifies the magnitudes and phases of the input.

2.2.1 Distortionless Transmission

In several applications, such as signal amplification or message transmission over a communication channel, we desire that the output waveform be a replica of the input waveform. In such cases we need to minimize the distortion caused by the amplifier or the communication channel. It is therefore of practical interest to determine the characteristics of a system that allows a signal to pass without distortion (*distortionless transmission*).

Transmission is said to be distortionless if, within a multiplicative constant, the input and the output have identical shapes. If a system merely delays the input without changing its waveform, the transmission is also considered distortionless. Thus, in distortionless transmission, the input $x(t)$ and the output $y(t)$ satisfy the condition

$$y(t) = ax(t - t_g). \quad (2.13)$$

The Fourier transform of this equation yields

$$Y(\omega) = aX(\omega)e^{-j\omega t_g}.$$

Since $Y(\omega) = X(\omega)H(\omega)$, the frequency response of a distortionless system is therefore

$$H(\omega) = a e^{-j\omega t_g}.$$

From this equation it follows that[†]

$$|H(\omega)| = a \quad \text{and} \quad \angle H(\omega) = -\omega t_g. \quad (2.14)$$

Thus, distortionless transmission requires that the magnitude response $|H(\omega)|$ is constant and that the phase response $\angle H(\omega)$ is a linear function of ω with slope $-t_g$, where t_g is the delay of the output with respect to input. Figure 2.9 illustrates these characteristics for positive delay t_g . Recall from Ex. 2.1 that these are also the characteristics of an ideal delay of t_g seconds with gain a .

[†]For simplicity, Eq. (2.14) assumes that a is real and positive. If this is not the case, minor adjustments of $|H(\omega)| = |a|$ $\angle H(\omega) = -\omega t_g + \angle a$ are required.

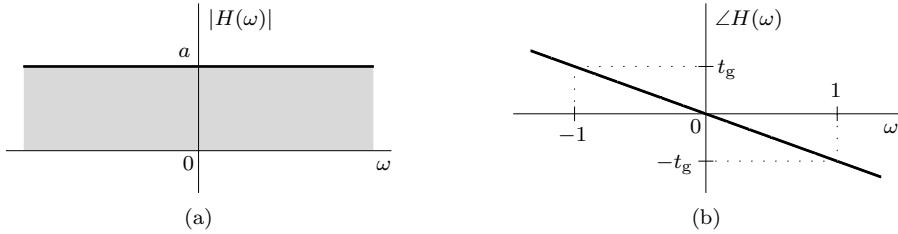


Figure 2.9: Distortionless transmission: (a) magnitude response and (b) phase response.

Measure of Delay Variation

For distortionless transmission, we require that the system possess *linear phase* characteristics. For real systems, the phase is not only a linear function of ω but also passes through the origin (or $\pm\pi$ if a is negative) at $\omega = 0$. In practice, many systems have phase characteristics that are only approximately linear. A convenient way of judging phase linearity is to plot the slope of $\angle H(\omega)$ as a function of frequency. This slope, which is a constant for an ideal linear phase (ILP) system but a function of ω in the general case, is expressed as

$$t_g(\omega) = -\frac{d}{d\omega}\angle H(\omega). \quad (2.15)$$

If $t_g(\omega)$ is constant, then all the components are delayed by the same amount. If the slope is not constant, however, the time delay t_g varies with frequency.[†] This variation means that different frequency components undergo different amounts of time delay, and consequently, the output waveform will not be a replica of the input waveform. Referred to as *group delay* or *envelope delay*, we shall soon see that $t_g(\omega)$ plays an important role in bandpass systems.

It is often thought (erroneously) that a system with a flat magnitude response (an *allpass system*) is distortionless. However, as the next example demonstrates, a system with a flat magnitude response may significantly distort a signal if the phase response is not linear ($t_g(\omega)$ not constant).

▷ Example 2.4 (Flat Magnitude Response Does Not Imply a Distortionless System)

Hilbert transformers are LTIC systems that, among other things, are useful in the study of communication systems. Ideally, a Hilbert transformer is described by

$$h(t) = \frac{1}{\pi t} \iff H(\omega) = -j\text{sgn}(\omega). \quad (2.16)$$

Compute and plot the magnitude response $|H(\omega)|$, the phase response $\angle H(\omega)$, and the group delay $t_g(\omega)$ of a Hilbert transformer. Compute and plot the output $y(t)$ of this system in response to the pulse input $x(t) = \Pi(t)$, and comment on the results.

By inspection, we establish[‡]

$$|H(\omega)| = 1 \quad \text{and} \quad \angle H(\omega) = -\frac{\pi}{2}\text{sgn}(\omega).$$

Differentiating $\angle H(\omega)$, the group delay $t_g(\omega)$ is

$$t_g(\omega) = \pi\delta(\omega).$$

[†]Recall that the phase response may have jump discontinuities when the amplitude response goes negative (see Fig. 1.44 on page 56). Jump discontinuities also appear in principal value representations of phase. Such jump discontinuities should be ignored when computing group delay.

[‡]Technically, $|H(\omega)| = 1$ everywhere except at $\omega = 0$ where $|H(0)| = 0$. This subtle distinction does not affect our overall discussion.

As shown in Fig. 2.10, the behavior of a Hilbert transformer looks simple: the magnitude response is unity across frequency, the phase response exhibits a single jump discontinuity at $\omega = 0$ where it transitions from $\pi/2$ to $-\pi/2$, and the group delay is zero everywhere except at $\omega = 0$. Such boring frequency characteristics might lead us to underestimate the impact of this system on an input signal. Our conditions for distortionless transmission, for example, are largely violated due to a single point $\omega = 0$.

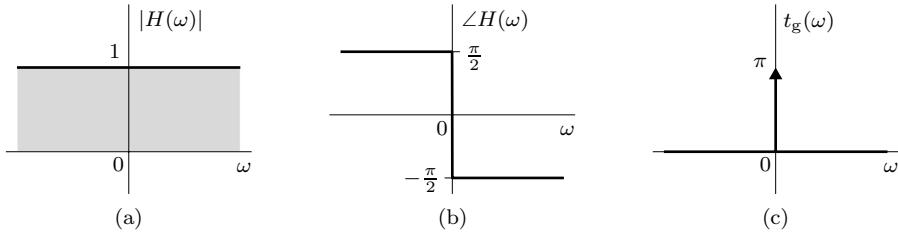


Figure 2.10: Hilbert transformer: (a) $|H(\omega)|$, (b) $\angle H(\omega)$, and (c) $t_g(\omega)$.

The output $y(t)$ is computed using time-domain convolution,

$$y(t) = \int_{-\infty}^{\infty} h(\tau)x(t - \tau) d\tau = \int_{t-1/2}^{t+1/2} \frac{1}{\pi(\tau)} d\tau = \frac{1}{\pi} \ln |\tau| \Big|_{t-1/2}^{t+1/2} = \frac{1}{\pi} \ln \frac{|t + 1/2|}{|t - 1/2|}.$$

Figure 2.11 illustrates the severe distortion that occurs when input $x(t) = \Pi(t)$ is passed through a Hilbert transformer. As this example demonstrates, a flat magnitude response is by no means a guarantee of distortionless transmission. Although $x(t)$ and $y(t)$ have radically different appearances, both are unity-energy signals with the same magnitude spectra.

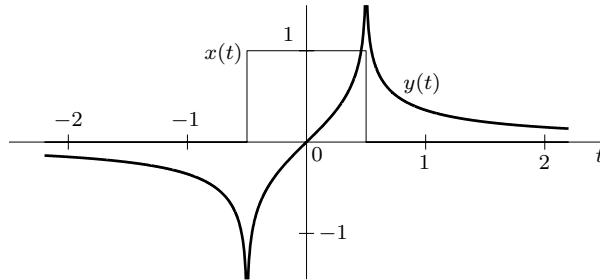


Figure 2.11: Hilbert transformer output $y(t)$ in response to $x(t) = \Pi(t)$.

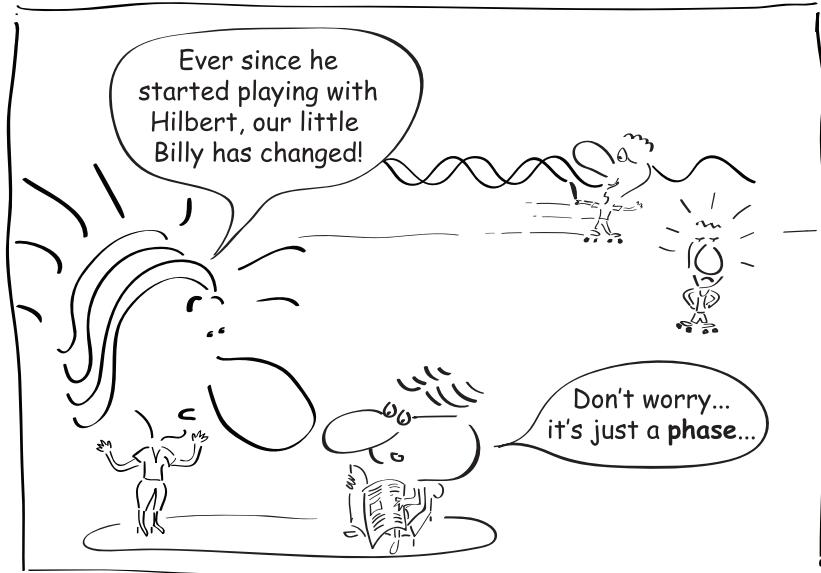
Example 2.4 ◀

The Nature of Distortion in Audio and Video Signals

Generally speaking, a human ear readily perceives amplitude distortion but is relatively insensitive to phase distortion. For phase distortion to become noticeable, the variation in delay (variation in the slope of $\angle H(\omega)$) should be comparable to the signal duration or, in the case the signal itself is long, the physically perceptible duration. In the case of audio signals, each spoken syllable can be considered as an individual signal. The average duration of a spoken syllable is on the order of 0.01 to 0.1 seconds. Practical audio systems may have nonlinear phase, yet no signal distortion is perceived because the maximum variation in the slope of $\angle H(\omega)$ is only a small fraction of a millisecond. This is the real truth underlying the statement that “the human ear is relatively insensitive to phase

distortion” [2]. As a result, the manufacturers of audio equipment make available only $|H(\omega)|$, the magnitude response characteristic of their systems.

For video signals, in contrast, the situation is exactly the opposite. A human eye is sensitive to phase distortion but is relatively insensitive to amplitude distortion. Amplitude distortion in television signals appears as a partial destruction of the relative half-tone values of the picture, which is not readily apparent to the human eye. Phase distortion (nonlinear phase), on the other hand, causes different time delays in different picture elements. The result is a smeared picture, which is readily perceived by the human eye. Phase distortion is also very important in digital communication systems because the nonlinear phase characteristic of a channel causes pulse dispersion (spreading), which, in turn, causes pulses to interfere with neighboring pulses. This interference can cause errors at the receiver: a logic one may read as a logic zero and vice versa.



With Hilbert, it's just a phase.

2.2.2 Real Bandpass Systems and Group Delay

The distortionless transmission conditions of Eq. (2.14) can be relaxed slightly for real bandpass systems, which are commonly encountered in engineering. For real lowpass systems to be distortionless, the phase characteristics should be not only linear over the band of interest but should also pass through the origin (or $\pm\pi$ for a negative dc gain). For real bandpass systems, the phase characteristics should also be linear over the band of interest, but it need not pass through the origin.

Consider an LTIC system with magnitude and phase characteristics as shown in Fig. 2.12, where the magnitude spectrum is a constant $|a|$ and the phase is $\phi_0 - \omega t_g$ over a band $2B$ centered at frequency ω_c . The phase is linear but does not pass through the origin. Over the bands of interest, we thus describe $H(\omega)$ as

$$H(\omega) = \begin{cases} |a|e^{j(\phi_0 - \omega t_g)} & \omega_c - B \leq \omega \leq \omega_c + B \\ |a|e^{j(-\phi_0 - \omega t_g)} & -\omega_c - B \leq \omega \leq -\omega_c + B \end{cases} . \quad (2.17)$$

Most practical systems satisfy the conditions of Eq. (2.17), at least over a small band. Such systems are said to have *generalized linear phase* (GLP) in contrast to the ideal linear phase (ILP) characteristics in Fig. 2.9. For distortionless transmission of bandpass signals, the system need satisfy Eq. (2.17) only over the bandwidth of the bandpass signal (see Ex. 2.5).

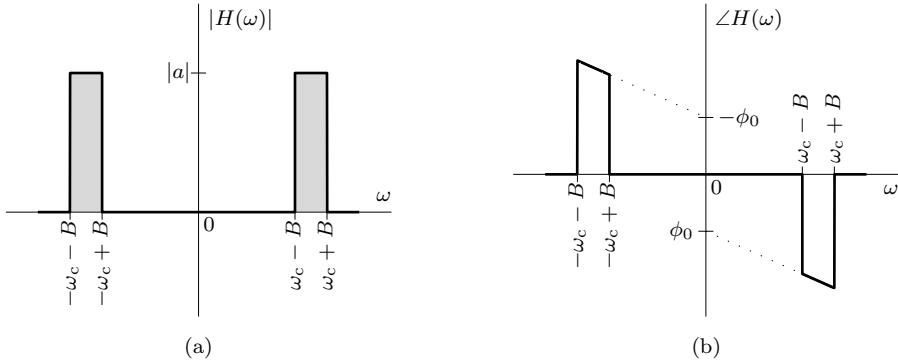


Figure 2.12: Generalized linear phase characteristics: (a) $|H(\omega)|$ and (b) $\angle H(\omega)$.

To understand how the system shown in Fig. 2.12 qualifies as a distortionless system, consider a modulated input signal $x_{\text{bp}}(t) = x(t) \cos(\omega_c t)$. Now, $x_{\text{bp}}(t)$ is a bandpass signal whose spectrum is centered at $\omega = \omega_c$. The signal $\cos \omega_c t$ is the carrier, and the signal $x(t)$, which is a real lowpass signal of bandwidth B , is the time-domain envelope of $x_{\text{bp}}(t)$ (see Fig. 1.46).[†] We shall now show that the transmission of $x_{\text{bp}}(t)$ through $H(\omega)$ results in distortionless transmission of the envelope $x(t)$ while the carrier phase changes by ϕ_0 relative to this envelope.

To simplify our discussion, we first write $x_{\text{bp}}(t)$ as

$$x_{\text{bp}}(t) = \operatorname{Re} \{ x(t) e^{j\omega_c t} \}.$$

Since the system is real, the output is given by Eq. (2.4) as

$$y_{\text{bp}}(t) = \operatorname{Re} \{ H \{ x(t) e^{j\omega_c t} \} \}.$$

To find $y_{\text{bp}}(t)$, we only need to determine $H \{ x(t) e^{j\omega_c t} \}$. Using the frequency-shifting property of Eq. (1.88), the Fourier transform yields

$$H \{ x(t) e^{j\omega_c t} \} \iff H(\omega) X(\omega - \omega_c).$$

Because $X(\omega - \omega_c)$ is zero for $\omega < 0$, we express this result using Eq. (2.17) as

$$H(\omega) X(\omega - \omega_c) = |a| e^{j(\phi_0 - \omega t_g)} X(\omega - \omega_c).$$

Using both the time-shifting and frequency-shifting properties of Eqs. (1.87) and (1.88), the inverse Fourier transform yields

$$|a| e^{j\phi_0} X(\omega - \omega_c) e^{-j\omega t_g} \iff |a| e^{j\phi_0} x(t - t_g) e^{j\omega_c(t - t_g)}.$$

Taking the real portion, the bandpass output $y_{\text{bp}}(t)$ is thus[‡]

$$y_{\text{bp}}(t) = |a| x(t - t_g) \cos [\omega_c(t - t_g) + \phi_0]. \quad (2.18)$$

As Eq. (2.18) makes clear, the envelope and carrier are both delayed by t_g , although the carrier acquires an additional phase shift of ϕ_0 . The delay t_g is termed the *group* (or *envelope*) delay and

[†]The envelope of a bandpass signal is well defined only when the bandwidth of the envelope is well below the carrier ω_c ($B \ll \omega_c$).

[‡]A general bandpass signal with spectrum centered at $\omega = \omega_c$ can be represented as $x_{\text{bp}}(t) = x(t) \cos[\omega_c t + \theta(t)]$, where $x(t)$ is the envelope of the signal and $\theta(t)$ is a time-varying phase. When this signal is passed through the system in Eq. (2.17), the system output is given by (see Prob. 2.2-4) $y_{\text{bp}}(t) = |a| x(t - t_g) \cos[\omega_c(t - t_g) + \phi_0 + \theta(t - t_g)]$.

is the negative slope of $\angle H(\omega)$ at ω_c . Observe that the output $y_{\text{bp}}(t)$ is basically the delayed input $x_{\text{bp}}(t - t_g)$, except that the the output carrier acquires an extra phase ϕ_0 .

In order to better understand the relaxed condition for distortionless transmission of bandpass signals through a GLP system, remember that a bandpass signal is basically a modulated signal with carrier frequency ω_c , which is the center frequency of the band. The phase response of a GLP system, as shown in Fig. 2.12b, is composed of a linear phase component $-\omega t_g$ and a constant phase component ϕ_0 . When a bandpass signal is passed through this system, the constant phase component causes the entire spectrum of the output signal to acquire a phase ϕ_0 . We know from Eq. (1.91) that such a constant phase can also be acquired by using the same bandpass input with a carrier having a constant phase ϕ_0 . Thus, the output of a GLP system for bandpass input $x(t) \cos(\omega_c t)$ is identical to the output of an ideal linear phase (ILP) system for input $x(t) \cos(\omega_c t + \phi_0)$. Since an ILP system merely delays by t_g , the output (within a gain constant) is $x(t - t_g) \cos[\omega_c(t - t_g) + \phi_0]$. Clearly, a GLP system simply delays $x(t)$ without distortion.

▷ Example 2.5 (Generalized Linear Phase Behavior for Practical Systems)

Consider the real bandpass input $x_{\text{bp}}(t) = x(t) \cos(\omega_c t)$ transmitted through the bandpass system of Ex. 2.2, $H(s) = 2s/(s^2 + 2s + 5)$. Determine a suitable ω_c that helps maximize the bandwidth of GLP behavior and distortionless transmission. Compute the corresponding group delay t_g and carrier phase offset ϕ_0 .

To achieve GLP behavior, we require a flat magnitude response and linear phase response. As seen from Fig. 2.13, the magnitude response is most flat over the widest range of frequencies near its peak. From Eq. (2.10) we see that this magnitude peak occurs when the denominator term $(5 - \omega^2)$ is zero, or at $\omega = \pm\sqrt{5}$ (the phase is relatively linear in this region as well). Thus,

$$\omega_c = \sqrt{5}.$$

The slope of the phase response is

$$\frac{d}{d\omega} \angle H(\omega) = -\frac{d}{d\omega} \tan^{-1} \left(\frac{2\omega}{5 - \omega^2} \right) = -\frac{1}{1 + \left(\frac{2\omega}{5 - \omega^2} \right)^2} \left[\frac{2(5 - \omega^2) + 4\omega^2}{(5 - \omega^2)^2} \right] = -\frac{4\omega^2 + 2(5 - \omega^2)}{4\omega^2 + (5 - \omega^2)^2}.$$

Setting $\omega = \omega_c = \sqrt{5}$, we see that $\angle H(\sqrt{5}) = 0$ and $\frac{d}{d\omega} \angle H(\sqrt{5}) = -1$. A straight-line approximation to the phase response is thus

$$\angle H(\omega) \approx \sqrt{5} - \omega = \phi_0 - t_g\omega.$$

Correspondingly,

$$t_g = 1 \quad \text{and} \quad \phi_0 = \sqrt{5}.$$

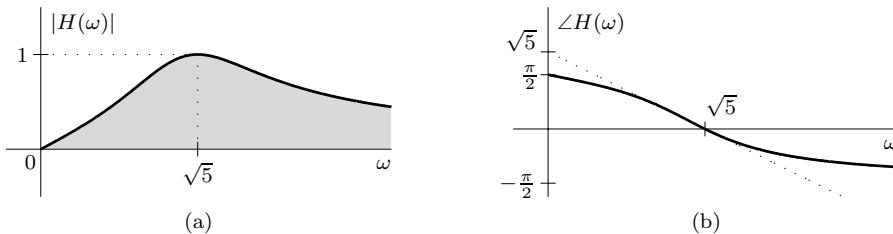


Figure 2.13: GLP behavior for $H(s) = \frac{2s}{s^2 + 2s + 5}$: (a) $|H(j\omega)|$ and (b) $\angle H(j\omega)$.

2.3 Ideal and Realizable Filters

A *filter* is a device or rule that allows selective output. A coffee filter, for example, is a mechanical filter that blocks large particulate matter (the coffee grounds). In a signal processing context, a filter selectively passes or blocks the frequency (complex exponential) components of an input signal. Ideal filters allow distortionless transmission over a certain band (or bands) of frequencies, the passband (or passbands), and suppress the remaining band (or bands) of frequencies, the stopband (or stopbands). Since filters are frequency-selective devices, filter behavior is conveniently summarized using magnitude and phase response plots.

Figure 2.14a, for example, shows the magnitude response of an ideal real continuous-time lowpass filter. Frequencies below $|\omega| = \omega_1$ rad/s are passed, while those above $|\omega| = \omega_1$ are suppressed. Here, ω_1 identifies the filter's *cutoff (or corner) frequency*. Figure 2.14 also displays the magnitude responses of three other common real filters: the bandpass filter, the highpass filter, and the bandstop filter. Each filter has unit gain in its passband(s) and zero gain in its stopband(s). By combining the four basic filter types, various multi-band filters are also possible.

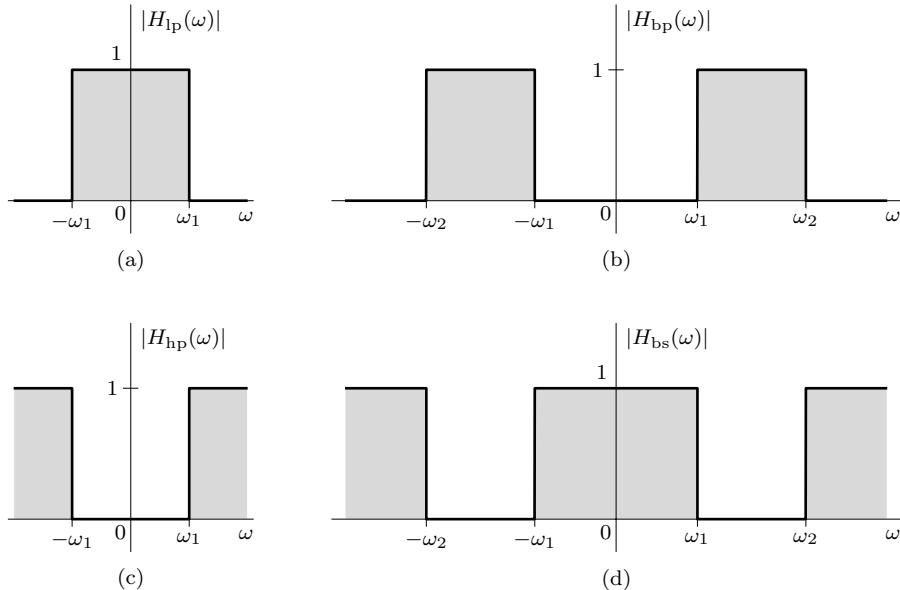


Figure 2.14: Filter magnitude responses: (a) lowpass, (b) bandpass, (c) highpass, and (d) bandstop.

As we shall see later in this chapter, it is a simple matter to transform a lowpass filter to any of the other three basic types. Thus, for the time being, we shall concentrate solely on lowpass filters. Most of our subsequent discussions and observations about lowpass filters apply to highpass, bandpass, and bandstop filters as well.

▷ Drill 2.4 (Relating Magnitude Response and Filter Characteristics)

Using Fig. 2.14a, identify the magnitude response features that correspond to a filter that is (a) ideal, (b) real, (c) continuous-time, and (d) lowpass.



Quest for the Holy Grail

The holy grail of lowpass filters has zero gain in the stopband, unity gain in the passband, and zero phase (no delay in the output), as shown in Fig. 2.15a.[†] This elusive response is sought by intrepid filter designers the world over, but success has never been achieved. To help understand why, we turn our attention to the filter's impulse response. Since $H(\omega) = \Pi\left(\frac{\omega}{2B}\right)$, the impulse response is $h(t) = \frac{B}{\pi} \text{sinc}\left(\frac{Bt}{\pi}\right)$, as shown in Fig. 2.15b.

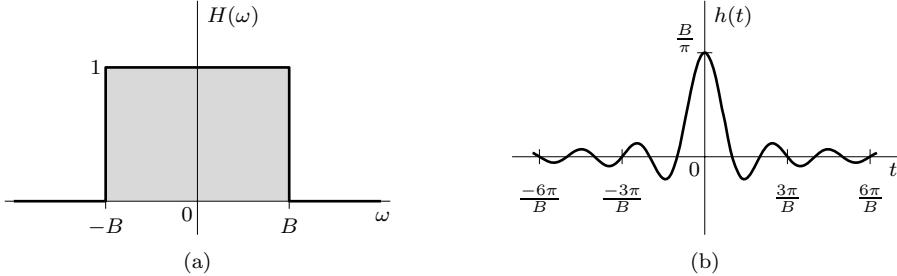


Figure 2.15: Ideal lowpass filter with zero output delay: (a) $H(\omega)$ and (b) $h(t)$.

Recall that $h(t)$ is the system response to an impulse input $\delta(t)$, which is applied at $t = 0$. Figure 2.15b shows a curious fact: the response $h(t)$ begins even before the input is applied. Clearly, the system is noncausal because it anticipates the input. It is impossible to build such a prophetic system in practice. So this filter is physically unrealizable. Similarly, one can show that other ideal filters (highpass, bandpass, and bandstop filters depicted in Fig. 2.14) are also noncausal and therefore physically unrealizable.

For a physically realizable system, the system must be *causal*; that is, $h(t)$ must be causal ($h(t) = 0$ for $t < 0$). In the frequency domain, this condition is equivalent to the well-known *Paley-Wiener criterion*, which states that a necessary and sufficient condition for the magnitude response $|H(\omega)|$ to be realizable is

$$\int_{-\infty}^{\infty} \frac{|\ln |H(\omega)||}{1 + \omega^2} d\omega < \infty. \quad (2.19)$$

If $H(\omega)$ does not satisfy this condition, it is unrealizable. Note that if $|H(\omega)| = 0$ over any finite band, $|\ln |H(\omega)|| = \infty$ over that band, and the condition of Eq. (2.19) is violated. If, however, $H(\omega) = 0$ at a single frequency (or a set of discrete frequencies), the integral in Eq. (2.19) may still be finite even though the integrand is infinite. Therefore, for a physically realizable system, $H(\omega)$ may be zero at some discrete frequencies, but it cannot be zero over any finite band. According to this criterion, ideal filter characteristics (Fig. 2.14) are clearly unrealizable.[‡] The holy grail of lowpass filters is beyond our grasp.

One method to make the filter causal (and therefore realizable) is to simply multiply $h(t)$ by $u(t)$ (Fig. 2.16a). Cut down the middle, however, a cup cannot hold water, and as shown in Fig. 2.16b, our causal filter's magnitude response only poorly approximates the desired ideal (shaded). To further understand this poor performance, notice that $\hat{h}(t) = h(t)u(t)$ only has half the energy of $h(t) = \frac{B}{\pi} \text{sinc}\left(\frac{Bt}{\pi}\right)$. Consequently, the frequency response $\hat{H}(\omega)$ has only half the energy of the

[†]Figure 2.15a is just Fig. 2.14a with $\omega_1 = B$ and zero phase. In the case of a lowpass filter, the cutoff frequency is equal to the bandwidth of the filter.

[‡] $|H(\omega)|$ is assumed to be square-integrable, that is,

$$\int_{-\infty}^{\infty} |H(\omega)|^2 d\omega$$

is finite. Note that the Paley-Wiener criterion is a test for the realizability of the magnitude response $|H(\omega)|$. If $|H(\omega)|$ satisfies the Paley-Wiener criterion, it does not follow that $h(t)$ is causal. All it says is that a suitable phase function can be assigned to $|H(\omega)|$ so that the resulting function has a causal $h(t)$.

desired response $H(\omega)$. It is of little surprise that $\hat{H}(\omega)$ cannot attain the lofty ideals of $H(\omega)$. *While causality is a necessary condition for our filter to be realizable, it is not sufficient to ensure good filter behavior.*

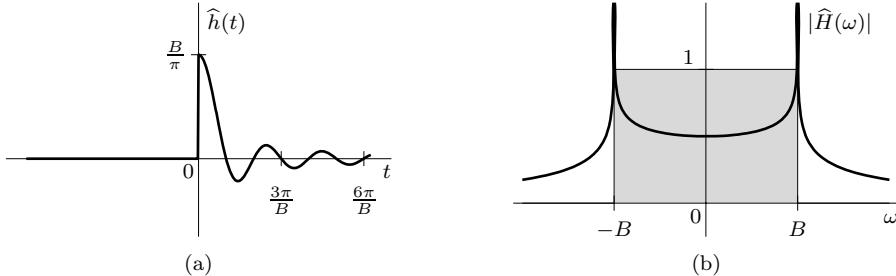


Figure 2.16: A causal filter: (a) $\hat{h}(t) = h(t)u(t) = \frac{B}{\pi}\text{sinc}\left(\frac{Bt}{\pi}\right)u(t)$ and (b) $|\hat{H}(\omega)|$.

Good Things Come to Those Who Wait

A primary reason that our truncation $\hat{h}(t) = h(t)u(t)$ performs so poorly is that the filter, in addition to being causal, is still trying to achieve zero phase (and consequently zero delay). Our impatient demand for an immediate output has a devastating impact on filter performance. As shown in Sec. 2.2.1, however, the stringent requirement of zero phase is not required for distortionless transmission. A time delay has no effect on the ideal magnitude characteristics of Fig. 2.14.

In the case of our ideal lowpass filter, allowing the output to be delayed by t_d delays the impulse response as

$$h(t - t_d) = \frac{B}{\pi}\text{sinc}\left[\frac{B(t - t_d)}{\pi}\right].$$

The corresponding frequency response is $\Pi\left(\frac{\omega}{2B}\right)e^{-j\omega t_d}$. Although the phase response is now linear (with slope $-t_d$), the magnitude response remains as shown in Fig. 2.14a.

The utility of allowing a delay t_d in our output becomes apparent when we force, as realizable filters require, the response to be causal,

$$\hat{h}(t) = h(t - t_d)u(t) = \frac{B}{\pi}\text{sinc}\left[\frac{B(t - t_d)}{\pi}\right]u(t).$$

As the delay t_d increases, ever-increasing amounts of the original $h(t)$ are likewise preserved, which provides an ever-increasing improvement in the filter's behavior. Figure 2.17 illustrates the idea for various choices of t_d .

Theoretically, a delay $t_d = \infty$ is needed to realize ideal characteristics. As shown in Fig. 2.17a, however, even small delays (less than $\frac{\pi}{B}$) can provide impressive improvements in filter performance. If t_d is sufficiently large, say, three or four times $\frac{\pi}{B}$, $\hat{h}(t)$ will be a close approximation of $h(t - t_d)$, and the resulting filter $\hat{H}(\omega)$ will be a good approximation of the ideal. Figure. 2.17c shows the case where t_d is six times $\frac{\pi}{B}$. A close realization of the ideal filter is achieved at the expense of an increased time-delay t_d in the output; this situation is common for causal realizations of noncausal systems. When it comes to filter performance, good things come to those who wait.

Figure 2.17 demonstrates that the truncation operation (cutting the $t < 0$ tail of $h(t - t_d)$ to make it causal) creates some problems: undesirable ripples appear, particularly near the transition frequency B . We discuss these problems and their cure in Sec. 2.4. Further, while the filters shown in Fig. 2.17 are causal and therefore theoretically realizable, it is not clear how we would physically build a filter that would possess these exact characteristics. For example, the large spikes in $|\hat{H}(\omega)|$ at $\omega = B$, besides being undesirable, prove difficult to realize with a physical system. Practical

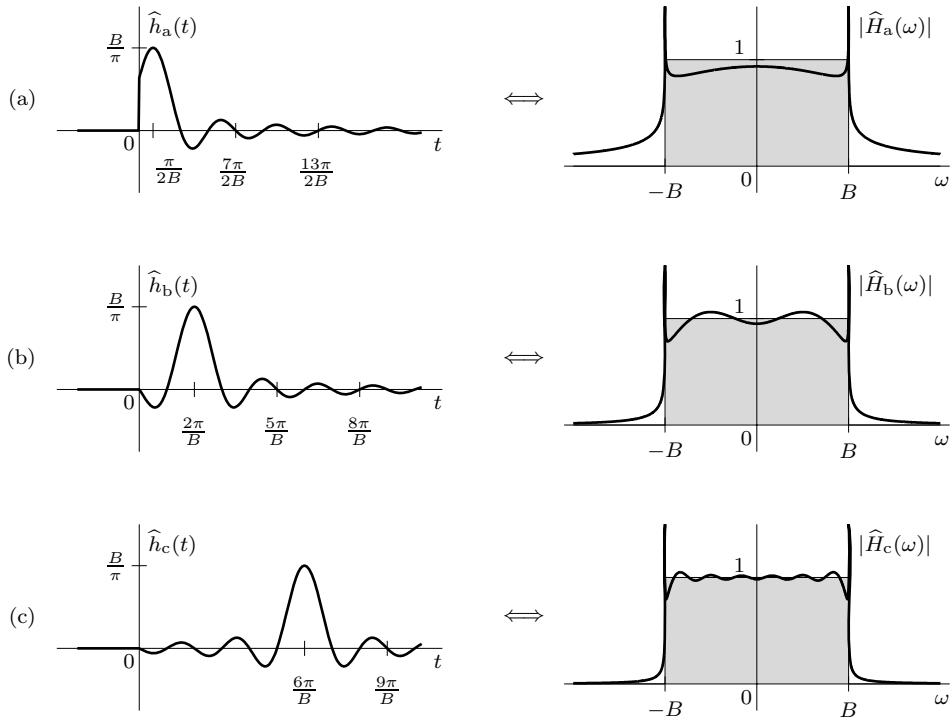


Figure 2.17: The beneficial effect of delay: (a) $t_d = \frac{\pi}{2B}$, (b) $t_d = \frac{2\pi}{B}$, and (c) $t_d = \frac{6\pi}{B}$.

(realizable) filter characteristics are gradual, without jump discontinuities in the magnitude response. In Sec. 2.7, we shall study families of causal filters, such as Butterworth and Chebyshev filters, that approach ideal characteristics and are also readily realized with physical circuits.

▷ Example 2.6 (Delay in Audio Systems)

Determine a suitable output delay t_d for a practical audio filter designed to pass frequencies audible to humans.

A basic audio filter is lowpass in nature and rejects frequencies out of the audible band (20 kHz). Thus, $B = 40000\pi$ rad/s. Taking t_d to be four times $\frac{\pi}{B}$ yields

$$t_d = \frac{4\pi}{40000\pi} \text{ sec} = \frac{1}{10000} \text{ sec} = 0.1 \text{ ms.}$$

This delay is quite small compared with phoneme lengths, which are typically in the tens to hundreds of milliseconds.

Example 2.6 ◀

▷ **Drill 2.5 (A Gaussian Frequency Response Is Unrealizable)**

Assuming that α is real and positive, show that a filter with Gaussian transfer function $H(\omega) = e^{-\alpha\omega^2}$ is unrealizable. Demonstrate this fact in two ways by showing (a) that its impulse response is noncausal and (b) that $|H(\omega)|$ violates the Paley-Wiener criterion.

△

2.4 Data Truncation by Windows

We often need to truncate data in diverse situations from numerical computations to filter design. For example, if we need to compute numerically the Fourier transform of some signal, say, $e^{-t}u(t)$, on a computer, we will have to truncate the signal $e^{-t}u(t)$ for $t > T$ (T is typically four to five time constants and above). The reason is that in numerical computations we can deal with data of finite duration only. Similarly, the impulse response $h(t)$ of an ideal lowpass filter is noncausal and has infinite duration. For a practical design, we may want to truncate $h(t)$ to make $h(t)$ causal with finite duration. To eliminate aliasing during signal sampling, we need to truncate, using an anti-aliasing filter, the signal spectrum beyond the half sampling frequency $\omega_s/2$ (see Sec. 3.3). Or we may want to synthesize a periodic signal by adding the first n harmonics and truncating all the higher harmonics. These examples show that data truncation can occur in the time domain as well as the frequency domain. On the surface, truncation appears to be a simple problem of cutting off the data at a point where it is deemed to be sufficiently small. Unfortunately, this is not the case. Simple truncation in the time domain can cause some unsuspected problems in the frequency domain and vice versa.

Data truncation produces either a finite-duration result, such as when a signal is multiplied by a rectangular window $\Pi(t/\tau)$, or a semi-infinite-duration result, such as when a signal is multiplied by the unit step $u(t)$. The former case is generally categorized as a *window operation*, and we often speak of such operations as *data windowing*. Although the remainder of this section concentrates on window operations where the results of truncation are finite in duration, our observations and conclusions also apply to cases where the truncation results are semi-infinite in duration, such as those shown in Fig. 2.17.

2.4.1 Impairments Caused by Windowing

Let us start with time-domain windowing. Similar conclusions hold for frequency-domain windowing. A time-domain window operation may be regarded as multiplying a signal $x(t)$ of large (possibly infinite) width by a window function $w(t)$ of smaller (finite) width. Simple truncation amounts to using a *rectangular window* $w_{rec}(t) = \Pi\left(\frac{t}{T}\right)$ (Fig. 2.18a) in which we assign unit weight to all the data within the window width ($|t| < \frac{T}{2}$) and assign zero weight to all the data lying outside the window ($|t| > \frac{T}{2}$). It is also possible to use a window in which the weight assigned to the data within the window may not be constant. In a *triangular window* $w_{tri}(t) = \Lambda\left(\frac{t}{T}\right)$, for example, the weight assigned to the data decreases linearly over the window width (Fig. 2.18b).

Consider a signal $x(t)$ and a window function $w(t)$. The windowed signal $x_w(t)$ is

$$x_w(t) = x(t)w(t). \quad (2.20)$$

Application of the frequency-domain convolution property (Eq. (1.96)) yields

$$X_w(\omega) = \frac{1}{2\pi} X(\omega) * W(\omega). \quad (2.21)$$

According to the width property of convolution (see [1]), it follows that the width of $X_w(\omega)$ equals the sum of the widths of $X(\omega)$ and $W(\omega)$.[†] Thus, truncation (windowing) causes the signal spectrum

[†] $W(\omega)$, the Fourier transform of the window function $w(t)$, is called the *Dirichlet kernel* of the window.

to spread out by B_w , the bandwidth of the window function $w(t)$. Clearly, windowing a signal causes distortion in the form of its spectrum spreading out (or smearing) by B_w . Recall that signal bandwidth is inversely proportional to signal duration (width). Hence, wider windows possess smaller bandwidths B_w and produce less *spectral spreading*. This result is expected because a wider window allows more data (closer approximation), which produces smaller distortion (less spectral spreading). A narrow window width accommodates less data (poorer approximation), which produces more distortion (greater spectral spreading).

Spectral spreading by B_w would be the only impairment caused by truncation if $w(t)$ were truly bandlimited to B_w . Unfortunately, $w(t)$, being timelimited, cannot be strictly bandlimited.[†] Its spectrum approaches zero asymptotically at a rate depending on the smoothness of the window. Smoother (more rounded) windows provide faster decay rates. The rectangular window, because of its jump discontinuities, has poor behavior, with its spectrum decaying slowly at a rate proportional to $1/\omega$. Because of Eq. (2.21), the behavior of a window spectrum is directly mirrored in the behavior of $X_w(\omega)$. Thus, if $X(\omega)$ is bandlimited to B_x , the truncated signal spectrum will spread out at the edges by B_w . Beyond this band of $B_x + B_w$, the spectrum does not vanish but lingers on, decaying asymptotically at the same rate as that of the window spectrum. While the effective bandwidth of a window produces an effect of spectral spreading, the tails of a window spectrum produce the effect of *spectral leakage*. Spectral leakage decays asymptotically in the stopband and can produce undesired ripples in the passband. We have, therefore, two impairments from window operations: spectral smearing and spectral leakage. These twin effects will soon be clarified by an example of a lowpass filter design.

Rectangular and Triangular Windows

Before investigating an example that highlights the impairments of windowing, let us examine the behavior of two windows: the rectangular window $w_{\text{rec}}(t)$ and the triangular (or *Bartlett*) window $w_{\text{tri}}(t)$. As shown in Fig. 2.18a, the width (duration) of either of the windows is T seconds. The spectra of these windows, shown in Fig. 2.18b, are each dominated by a single main lobe (shaded). Figure 2.18c shows the normalized spectra of these windows using a decibel (dB) scale. The logarithmic plots of Fig. 2.18c visually exaggerate small amplitudes because a gain approaching zero approaches $-\infty$ on a logarithmic (dB) scale. Such an exaggeration helps to emphasize the behavior of the side lobes (unshaded), whose amplitudes are generally small.

The main spectral lobe (shaded), which provides an approximation of a window's effective bandwidth, causes spectral smearing. The main lobe of $w_{\text{rec}}(t)$ is $4\pi/T$ rad/s ($2/T$ Hz) wide, and the main lobe of $w_{\text{tri}}(t)$ is twice as wide ($8\pi/T$ rad/s or $4/T$ Hz). Windows with narrow main lobes are preferred to reduce the effects of spectral spreading. Ideally, a window should have a main lobe with zero width (and no side lobes). Only the impulse spectrum satisfies this ideal condition, whose inverse transform is a rectangular window of infinite width. This is a trivial result because an infinite-width window leads to no distortion. Unfortunately, infinite width is precisely what we want to avoid. In other words, practical windows always have a main lobe of nonzero width (as well as side lobes).

A window's side lobes (unshaded) cause spectral leakage. To minimize leakage, side lobes with small amplitudes (compared with the main lobe) and fast decay rates are desired. For convenience, we show the reference level of the main lobe peak at 0 dB (unity gain). For the rectangular window, the peak side lobe level is -13.3 dB, and that for the triangular window is -26.5 dB. The asymptotic decay of $W_{\text{rec}}(\omega)$ is slower (proportional to $1/\omega$) than that of $W_{\text{tri}}(\omega)$ (proportional to $1/\omega^2$). This is expected because the latter is a smoother function with no discontinuities. In contrast, the former has jump discontinuities requiring larger amounts of high-frequency components for its synthesis (spectrum decays more slowly). The rectangular window's rate of spectral decay is -20 dB/decade

[†]This follows from the property that a signal cannot be simultaneously timelimited and bandlimited. If it is timelimited, it is not bandlimited, and if it is bandlimited, it cannot be timelimited. It can, however, be simultaneously non-bandlimited and non-timelimited [1].

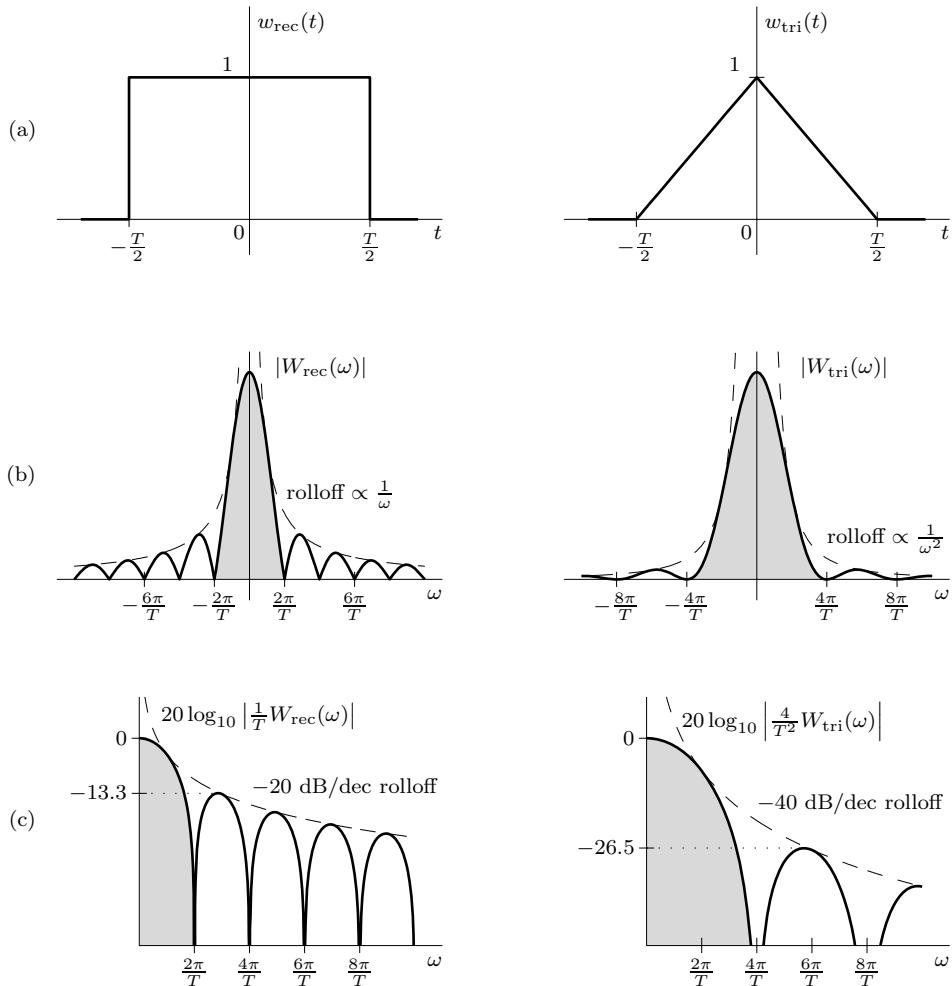


Figure 2.18: Rectangular and triangular windows: (a) functions, (b) spectra, and (c) normalized spectra in dB.

(-6 dB/octave), and that for the triangular window is -40 dB/decade (-12 dB/octave).[†]

From the viewpoint of the spectral leakage, a triangular window is superior (small side lobes with fast decay) to a rectangular window. However, the tradeoff is that the main lobe width of a triangular window is twice that of the corresponding rectangular window. From the perspective of spectral spreading, a rectangular window is preferable to a triangular window. As we shall later see, such quandaries in window choice are typical: windows with desirable main lobe characteristics tend to suffer undesirable side lobe characteristics and vice versa. We must try to find the best compromise to suit our particular needs.

2.4.2 Lowpass Filter Design Using Windows

Let us next use window functions to assist us in the construction of a realizable filter. Consider an ideal lowpass filter of bandwidth B rad/s, which has transfer function $H(\omega) = \Pi(\frac{\omega}{2B})$ (see

[†]Recall that decay as $1/\omega$ reduces the amplitude by a ratio of $1/10$ over one decade of frequency. The ratio $1/10$ corresponds to -20 dB. Similarly, decay at $1/\omega^2$ reduces the amplitude by a ratio $1/100$ over a decade of frequency. Thus, the amplitude decays at -40 dB/decade.

Fig. 2.15a). As shown in Fig. 2.15b, its impulse response $h(t) = \frac{B}{\pi} \text{sinc}(\frac{Bt}{\pi})$ is noncausal with infinite duration. To realize a practical design, we truncate $h(t)$ with a width- T window so that the impulse response has finite duration.[†] Such a truncation, due to smearing and leakage effects, distorts the realized response from our desired ideal. We shall use a rectangular window $w_{\text{rec}}(t)$ and a triangular window $w_{\text{tri}}(t)$ to truncate $h(t)$ and then examine the resulting filters. The truncated impulse responses are given as

$$h_{\text{rec}}(t) = h(t)w_{\text{rec}}(t) \quad \text{and} \quad h_{\text{tri}}(t) = h(t)w_{\text{tri}}(t). \quad (2.22)$$

Applying the frequency-convolution property to Eq. (2.22) yields

$$\begin{aligned} H_{\text{rec}}(\omega) &= H(\omega) * W_{\text{rec}}(\omega) = \int_{-\infty}^{\infty} H(\lambda)W_{\text{rec}}(\omega - \lambda) d\lambda \\ \text{and} \quad H_{\text{tri}}(\omega) &= H(\omega) * W_{\text{tri}}(\omega) = \int_{-\infty}^{\infty} H(\lambda)W_{\text{tri}}(\omega - \lambda) d\lambda. \end{aligned} \quad (2.23)$$

Since $w_{\text{rec}}(t) = \Pi(\frac{t}{T})$ and $w_{\text{tri}}(t) = \Lambda(\frac{t}{T})$, we know $W_{\text{rec}}(\omega) = T \text{sinc}(\frac{T\omega}{2\pi})$ and $W_{\text{tri}}(\omega) = \frac{T}{2} \text{sinc}^2(\frac{T\omega}{4\pi})$.

Figure 2.19 illustrates the idea using $T = \frac{7\pi}{B}$. The truncations of Eq. (2.22) are shown in Fig. 2.19a, and the results of the convolutions of Eq. (2.23) are shown in Fig. 2.19b. Both filter responses are approximations of the ideal lowpass filter (shaded). The lower portion of Fig. 2.19b displays the window spectra from Eq. (2.23) aligned at the transition frequency $\omega = B$; these curves emphasize the relation between the width of the window's main lobe and the resulting spectral spread in the filter's frequency response. Figure 2.19c presents the filter magnitude responses using a decibel scale.

From the plots in Fig. 2.19, we make the following observations:

1. Windowed (practical) filters show spectral spreading at the edges. Instead of a sudden transition, practical filters exhibit a gradual transition from passband to stopband. As a rough but intuitive estimate, the width of this *transition band* is equal to half the width of the main lobe of the window (see [3] for more accurate estimates). From a somewhat simplified perspective, the factor half occurs because the spectral spread (main lobe width) is divided over two transition frequencies ($\omega = \pm B$). Using a rectangular window, each transition band is $2\pi/T$ rad/s (1/T Hz), for a total spectral spread equal to the window spectrum's main lobe width ($4\pi/T$ rad/s). Using a triangular window, each transition band is $4\pi/T$ rad/s (2/T Hz), which is twice that of the rectangular window case. Since we prefer spectral spreading (main lobe width) to be as small as possible, rectangular windows provide superior behavior to triangular windows in terms of the spectral spread at band edges.
2. Although $H(\omega)$ is bandlimited with a distinct passband and stopband, the windowed filters are not bandlimited. The gain of $H_{\text{rec}}(\omega)$ in the stopband is not zero but goes to zero asymptotically as $1/\omega$ (-20 dB/dec). Similarly, the stopband gain of $H_{\text{tri}}(\omega)$ decreases as $1/\omega^2$ (-40 dB/dec). This *stopband rolloff*, as well as portions of the passband ripple, is a consequence of spectral leakage. Stopband oscillations (leakage) have higher amplitudes for the rectangular case than for the triangular case. Thus, the stopband behavior for the triangular case is superior to the rectangular case.

Plots of $20 \log_{10} |H_{\text{rec}}(\omega)|$ and $20 \log_{10} |H_{\text{tri}}(\omega)|$, as depicted in Fig. 2.19c, show stopband ripple (side lobe) behavior that is consistent with the side lobes shown in Fig. 2.18c. That is, a window spectrum's rolloff rates and peak side lobe levels directly impact a windowed filter's spectral characteristics. For example, a large peak side lobe, such as the -13.3-dB peak of

[†]To be truly realizable, we also need to delay the truncated function by $\frac{T}{2}$ in order to render it causal. However, this time delay only adds a linear phase component to the spectrum without changing the magnitude spectrum. For this reason, we shall ignore the delay in order to simplify our discussion.

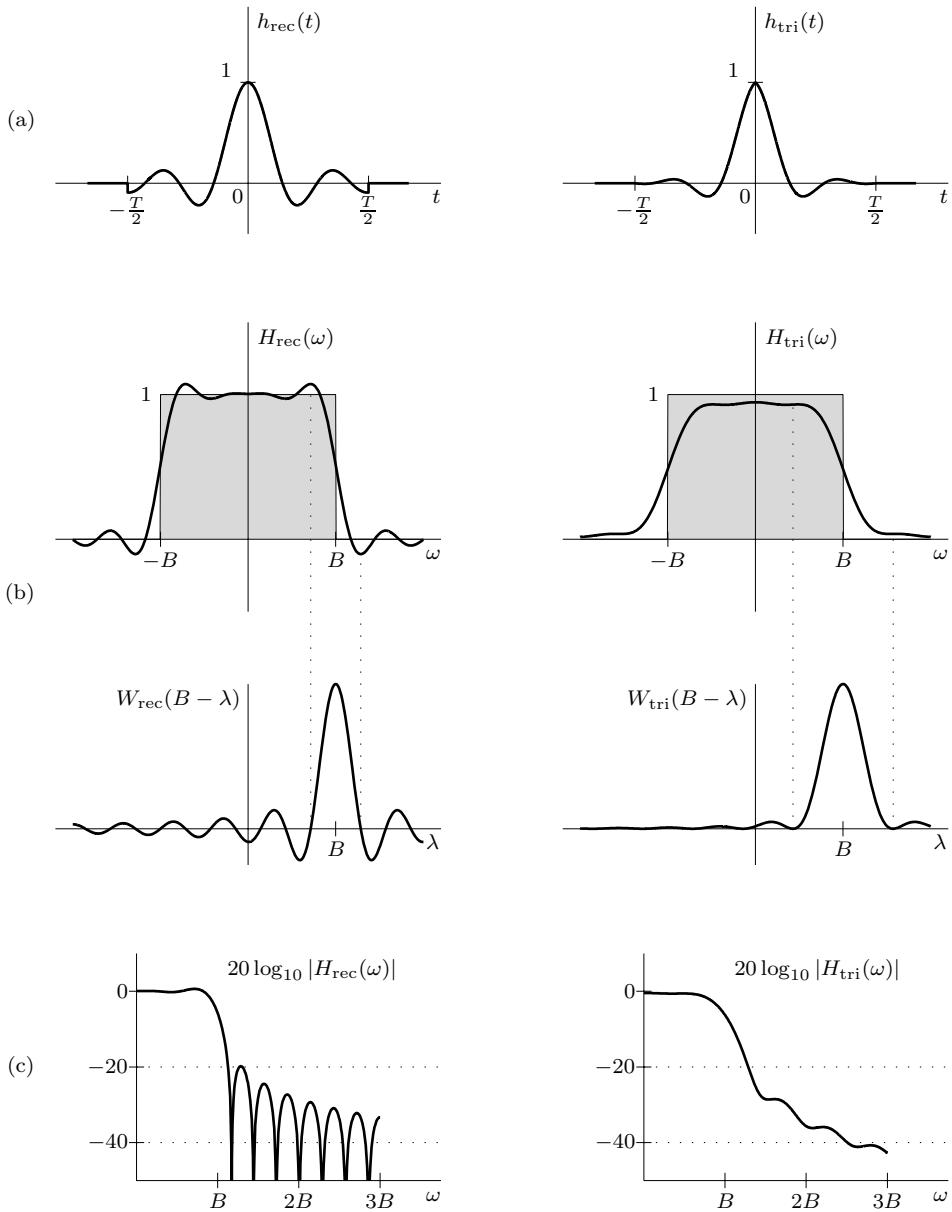


Figure 2.19: Lowpass filter design using rectangular and triangular windows ($T = \frac{7\pi}{B}$): (a) truncated impulse responses, (b) frequency responses, and (c) magnitude responses in dB.

a rectangular window, translates to a relatively large peak in the filter stopband (≈ -20 dB for the rectangular case). Further, the slow rolloff (-20 dB/dec) of a rectangular window spectrum produces slow rolloff (-20 dB/dec) in the filter stopband. To minimize leakage effects, we desire window spectra that possess small side lobes with a fast rates of decay (high rolloff rates).

Although this section focuses on the effects of windows on filter behavior, we emphasize that spectral smearing and leakage are significant concerns when windows are applied to any signal of interest. For example, if a width- T rectangular window is applied to a signal $x(t)$ that has two

spectral components differing in frequency by less than about $1/T$ Hz, then the two spectral components smear together and become indistinguishable. Similarly, triangular windowing causes signal components separated by less than about $2/T$ Hz to become indistinguishable. Thus, windowing $x(t)$ results in a loss of *spectral resolution*, an aspect critical in spectral analysis applications. Furthermore, so far we have discussed time-domain truncation and its effect on a signal's spectrum. Because of time-frequency duality, spectral (frequency-domain) truncation has a similar effect on the time-domain signal shape.

2.4.3 Remedies for Truncation Impairments

For better results, we must try to minimize truncation's twin side effects of spectral spreading (main lobe width) and leakage (side lobe levels and rolloff). Let us consider each of these ills.

1. The spectral spread (main lobe width) of a truncated signal is equal to the bandwidth of the window function $w(t)$. We know that signal bandwidth is inversely proportional to signal width (duration). Hence, to reduce the spectral spread (main lobe width), we need to increase the window width.
2. To improve leakage behavior, we must search for the cause of slow side lobe decay. It can be shown (see [1]) that the Fourier spectrum decays as $1/\omega$ for a signal with a jump discontinuity (such as a rectangular window), decays as $1/\omega^2$ for a continuous signal whose first derivative is discontinuous (such as a triangular window), decays as $1/\omega^3$ for a continuous signal with continuous first derivative but with discontinuous second derivative, and so on. The smoothness of a signal is measured by the number of continuous derivatives it possesses. The smoother the signal, the faster is the decay of its spectrum. Thus, we can achieve a given leakage behavior by selecting a suitably smooth window.

For a given window width, the remedies for the two effects conflict. If we try to improve one, the other deteriorates. For instance, among all the windows of a given width, the rectangular window has the smallest spectral spread (main lobe width) but has high level side lobes that decay slowly. A tapered (smooth) window of the same width has smaller and faster decaying side lobes but has a wider main lobe, which increases the spectral spread [1]. We can, however, compensate for the increased spectral spread (main lobe width) by widening the window. Thus, we can remedy both the side effects of truncation by selecting a suitably smooth window of wider width.

2.4.4 Common Window Functions

As Fig. 2.18 demonstrates, the tapered triangular window provides better leakage performance than the rectangular window, although at a cost of increased spectral spread. This increased spectral spread, however, can be reduced by increasing the duration of the window. As a result, tapered window functions are often preferred to rectangular (non tapered) windows. There are, however, countless ways to taper data, and different tapers produce different results.

In addition to the triangular window, there are many commonly used and well-known tapered window functions such as the Hann (von Hann), Hamming, Blackman, and Kaiser windows. Table 2.1 summarizes the most common windows and their characteristics [3,4]. Notice that the Kaiser window is expressed in terms of $I_0(\alpha)$, a 0th-order modified Bessel function of the first kind with (real) argument α . Upcoming Ex. 2.7 investigates the Kaiser window in more detail.

Excluding the rectangular window, each of the windows in Table 2.1 truncates data gradually, although they offer different tradeoffs with respect to spectral spread (main lobe width), peak side lobe amplitude, and side lobe rolloff rate. Figure 2.20, for example, illustrates the popular Hann and Hamming windows; both of these windows possess the gradual taper that is necessary to achieve desirable leakage characteristics.

Further, all the window functions in Table 2.1 are symmetrical about the origin and are thus even functions of t . Due to this even symmetry, $W(\omega)$ is a real function of ω , and $\angle W(\omega)$ is either

Window $w(t)$	Main Lobe Width	Rolloff Rate [dB/dec]	Peak Side Lobe Level [dB]
1. Rectangular: $\Pi\left(\frac{t}{T}\right)$	$\frac{4\pi}{T}$	-20	-13.3
2. Triangular (Bartlett): $\Lambda\left(\frac{t}{T}\right)$	$\frac{8\pi}{T}$	-40	-26.5
3. Hann: $\frac{1}{2} \left[1 + \cos\left(\frac{2\pi t}{T}\right)\right] \Pi\left(\frac{t}{T}\right)$	$\frac{8\pi}{T}$	-60	-31.5
4. Hamming: $[0.54 + 0.46 \cos\left(\frac{2\pi t}{T}\right)] \Pi\left(\frac{t}{T}\right)$	$\frac{8\pi}{T}$	-20	-42.7
5. Blackman: $[0.42 + 0.5 \cos\left(\frac{2\pi t}{T}\right) + 0.08 \cos\left(\frac{4\pi t}{T}\right)] \Pi\left(\frac{t}{T}\right)$	$\frac{12\pi}{T}$	-60	-58.1
6. Kaiser: $\frac{I_0\left(\alpha\sqrt{1-4\left(\frac{t}{T}\right)^2}\right)}{I_0(\alpha)} \Pi\left(\frac{t}{T}\right)$	varies with α	-20	varies with α

Table 2.1: Common window functions and their characteristics.

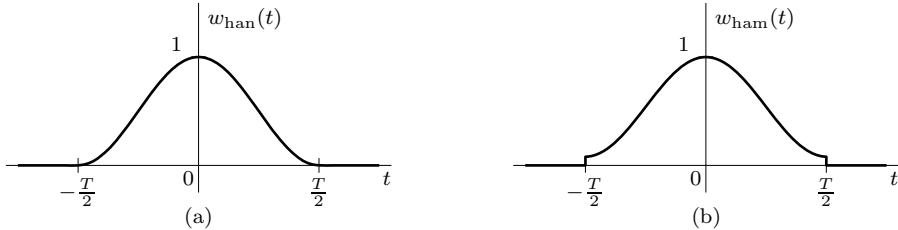


Figure 2.20: Two popular window functions: (a) Hann and (b) Hamming.

0 or π . Hence, the phase of the truncated signal has a minimal amount of distortion. If a window is shifted, as is sometimes necessary, an additional linear phase component is present that, like the case of distortionless transmission, is relatively unimportant. Although we specify the independent variable as t , we can easily change t to ω to accommodate applications that require windowing in the frequency domain.

Window choice depends on the particular application at hand. The rectangular window has the narrowest main lobe. The triangular (Bartlett) window (also called the *Fejer* or *Cesaro*) is inferior in nearly all respects to the Hann window. For this reason, it is rarely used in practice except for situations where the computational simplicity of the triangular window is required. A Hann window is preferred over a Hamming window in spectral analysis because it has faster side lobe decay. For filtering applications, on the other hand, the Hamming window is the better choice because it has the smallest side lobe amplitude for a given main lobe width. The Hamming window is the most widely used general purpose window. The Kaiser window is more versatile and adjustable. The parameter α controls the main lobe and side lobe tradeoff, and its proper selection allows a designer to tailor the window to suit a particular application.

▷ **Example 2.7 (The Kaiser Window)**

Compute and plot the Kaiser window for (a) $\alpha = 0$, (b) $\alpha = 5.4414$, and (c) $\alpha = 8.885$. Compare these cases with the rectangular, Hamming, and Blackman windows, respectively. What is the general effect of increasing the parameter α ?

Using the expressions given in Table 2.1, the required window functions are created in MATLAB along with a suitably wide time vector (normalized by T).

```
01 tbyT = linspace(-.75,.75,2000);
02 wrec = @(tbyT) 1.0*(abs(tbyT)<=0.5);
03 wham = @(tbyT) (0.54+0.46*cos(2*pi*tbyT)).*wrec(tbyT);
04 wbla = @(tbyT) (0.42+0.5*cos(2*pi*tbyT)+0.08*cos(4*pi*tbyT)).*wrec(tbyT);
05 wkai = @(tbyT,alpha) besseli(0,alpha*sqrt(1-4*tbyT.^2))/besseli(0,alpha).*wrec(tbyT);
```

Plots of the Kaiser windows, their respective comparison windows, and the absolute differences between each pair follow directly.

```
06 subplot(331); plot(tbyT,wkai(tbyT,0));
07 subplot(332); plot(tbyT,wrec(tbyT));
08 subplot(333); plot(tbyT,abs(wkai(tbyT,0)-wrec(tbyT)));
09 subplot(334); plot(tbyT,wkai(tbyT,5.4414));
10 subplot(335); plot(tbyT,wham(tbyT));
11 subplot(336); plot(tbyT,abs(wkai(tbyT,5.4414)-wham(tbyT)));
12 subplot(337); plot(tbyT,wkai(tbyT,8.885));
13 subplot(338); plot(tbyT,wbla(tbyT));
14 subplot(339); plot(tbyT,abs(wkai(tbyT,8.885)-wbla(tbyT)));
```

The resulting plots, following axis labeling and scaling, are shown in Fig. 2.21.

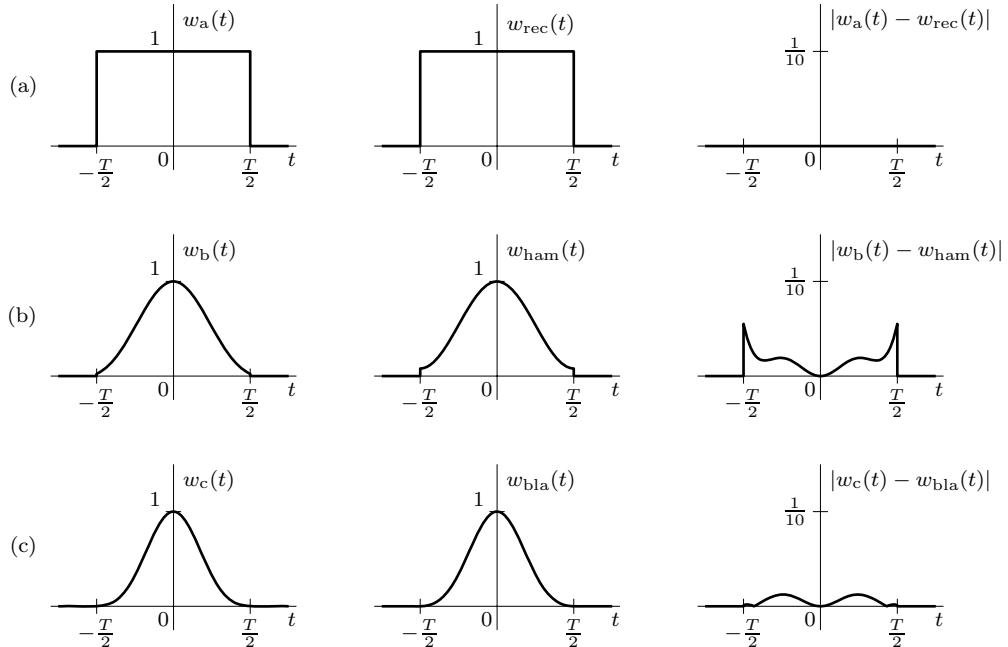


Figure 2.21: Kaiser windows and comparisons: (a) $\alpha = 0$ vs. rectangular, (b) $\alpha = 5.4414$ vs. Hamming, and (c) $\alpha = 8.885$ vs. Blackman.

As Fig 2.21a makes clear, a Kaiser window with $\alpha = 0$ is identical to the rectangular window.

Further, a Kaiser window with $\alpha = 5.4414$ closely approximates the Hamming window (Fig. 2.21b), and a Kaiser window with $\alpha = 8.885$ is nearly identical to the Blackman window (Fig. 2.21c).

Figure 2.21 also emphasizes the versatility that parameter α affords to the Kaiser window. Increasing α has two primary effects. First, it tends to make the window smoother. Second, it tends to reduce the *effective width* of the window. In terms of the Kaiser window's spectrum, this means that, as α increases, the side lobe levels decrease and the main lobe width increases.

Example 2.7 \triangleleft

▷ Drill 2.6 (Computing and Plotting Kaiser Windows)

Compute and plot the Kaiser window for α equal to 0, 1, 2, 3, 5, 10, and 20. Using a 95% energy criterion, determine the effective window width W for each case of α .

\triangleleft

2.5 Specification of Practical Filters

Section 2.3 introduces ideal filter characteristics and explains why such filters cannot be realized in practice. In this section, we introduce the specification of practical filters. As will be seen in Sec. 2.7, such specifications provide a foundation for the design of all common analog filters.

For ideal filters everything is black and white; the gains are either zero or unity over certain bands. As we saw earlier, real life does not permit such a world view. Things have to be gray or shades of gray. In practice, we can only hope to realize filters that approach ideal characteristics.

An ideal filter instantaneously jumps from passband (unity gain) to stopband (zero gain). There is no transition band between the two. Practical filters, on the other hand, transition from passband to stopband (or vice versa) gradually over a finite *transition band* of frequencies. Moreover, for realizable filters, the gain cannot be zero over a finite band (Paley-Wiener condition). As a result, practical filters have no true stopband. We therefore define a *stopband* as a frequency band over which the gain is below some small number δ_s (the stopband *ripple parameter*), as illustrated in Fig. 2.22 for the four basic (real) filter types.[†] Similarly, we define a *passband* as a frequency band over which the gain is between $1 - \delta_p$ and 1, also shown in Fig. 2.22.

Practical filters require specification of passband edges (ω_p), passband ripple (δ_p), stopband edges (ω_s), and stopband ripple (δ_s). As shown in Fig. 2.22 by the shaded regions, these specifications define the out-of-bounds zones for a particular design. If a candidate filter's magnitude response enters an out-of-bounds area, however briefly, then the filter fails to meet specifications, and a better filter is required. Each of the four filters shown in Fig. 2.22 touches, but does not cross, the out-of-bounds regions and therefore meets (just barely) the given specifications.

Ripple requirements are sometimes expressed as *minimum passband gain* ($1 - \delta_p$) and *maximum stopband gain* (δ_s). Some design methods, such as those of Ch. 8, offset the passband gain boundaries to $1 - \frac{1}{2}\delta_p$ and $1 + \frac{1}{2}\delta_p$; when defining the passband, just about any offset is acceptable. For the bandpass and bandstop filters shown in Fig. 2.22, the lower transition bands (bounded by ω_{s_1} and ω_{p_1}) are shown narrower than the upper transition bands (bounded by ω_{s_2} and ω_{p_2}). This behavior, while not required, is typical for common CT filter families.

Frequently, it is more convenient to work with filter attenuation rather than filter gain. Filter *attenuation*, expressed in dB, is the negative of filter gain, also expressed in dB. The half-power gain of $\frac{1}{\sqrt{2}}$, for example, is $20 \log_{10} \left(\frac{1}{\sqrt{2}} \right) \approx -3.01$ dB, which corresponds to 3.01 dB of attenuation. Figure 2.23 represents Fig. 2.22 in terms of attenuation rather than gain. The maximum passband attenuation (α_p) and the minimum stopband attenuation (α_s) are expressed in terms of the minimum

[†]While Fig. 2.22 considers real filters, complex filters are treated in an analogous manner.

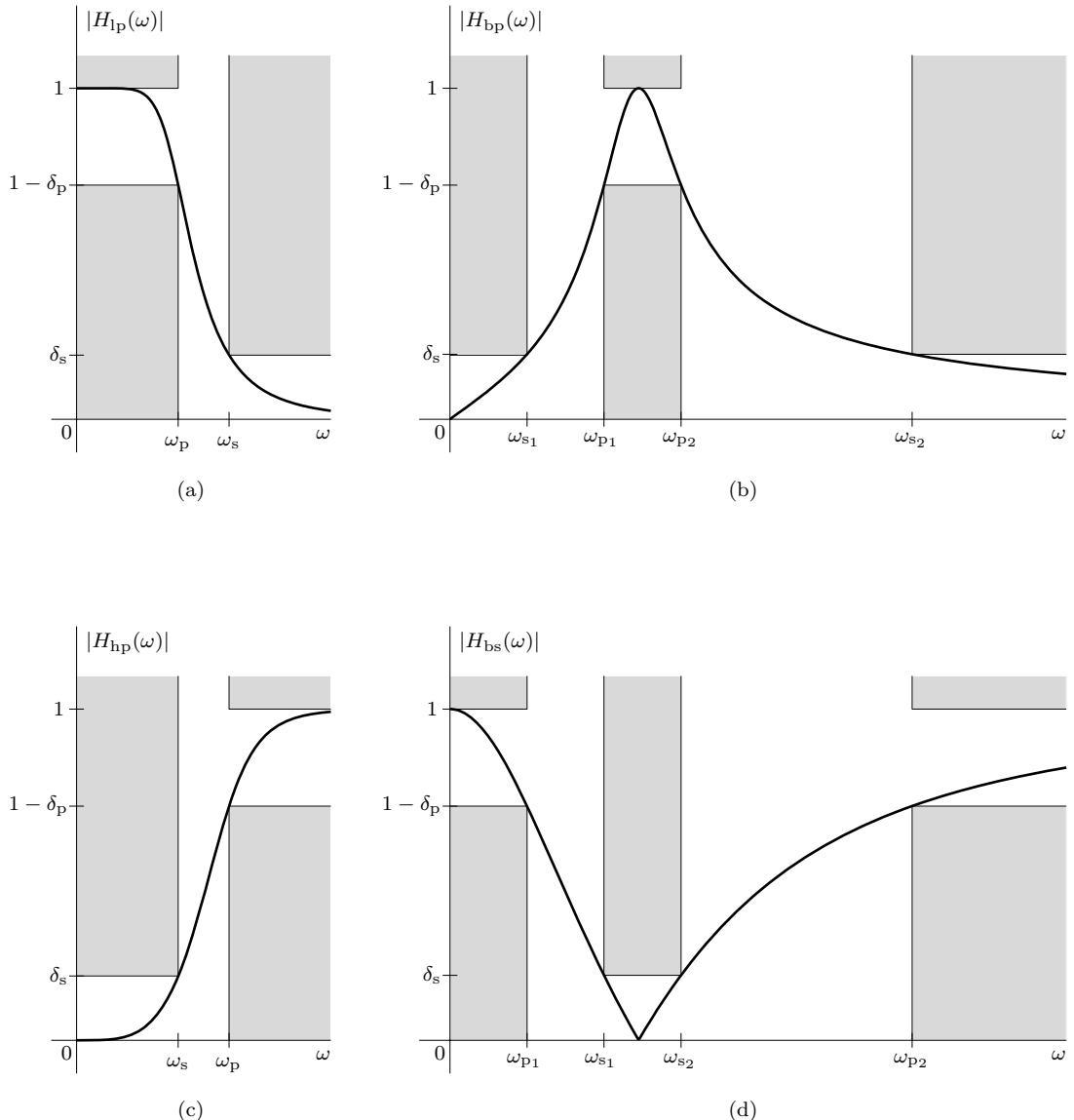


Figure 2.22: Practical filter specifications: (a) lowpass, (b) bandpass, (c) highpass, and (d) bandstop filters.

passband gain and maximum stopband gain as

$$\alpha_p = -20 \log_{10} (1 - \delta_p) \quad \text{and} \quad \alpha_s = -20 \log_{10} (\delta_s). \quad (2.24)$$

2.6 Analog Filter Transformations

The four basic filter types shown in Fig. 2.14 share remarkably similar characteristics: each is comprised of passband and stopband regions. Not surprisingly, it is relatively simple to transform from one type to another. The utility of such transformations is clear: they simplify the design process

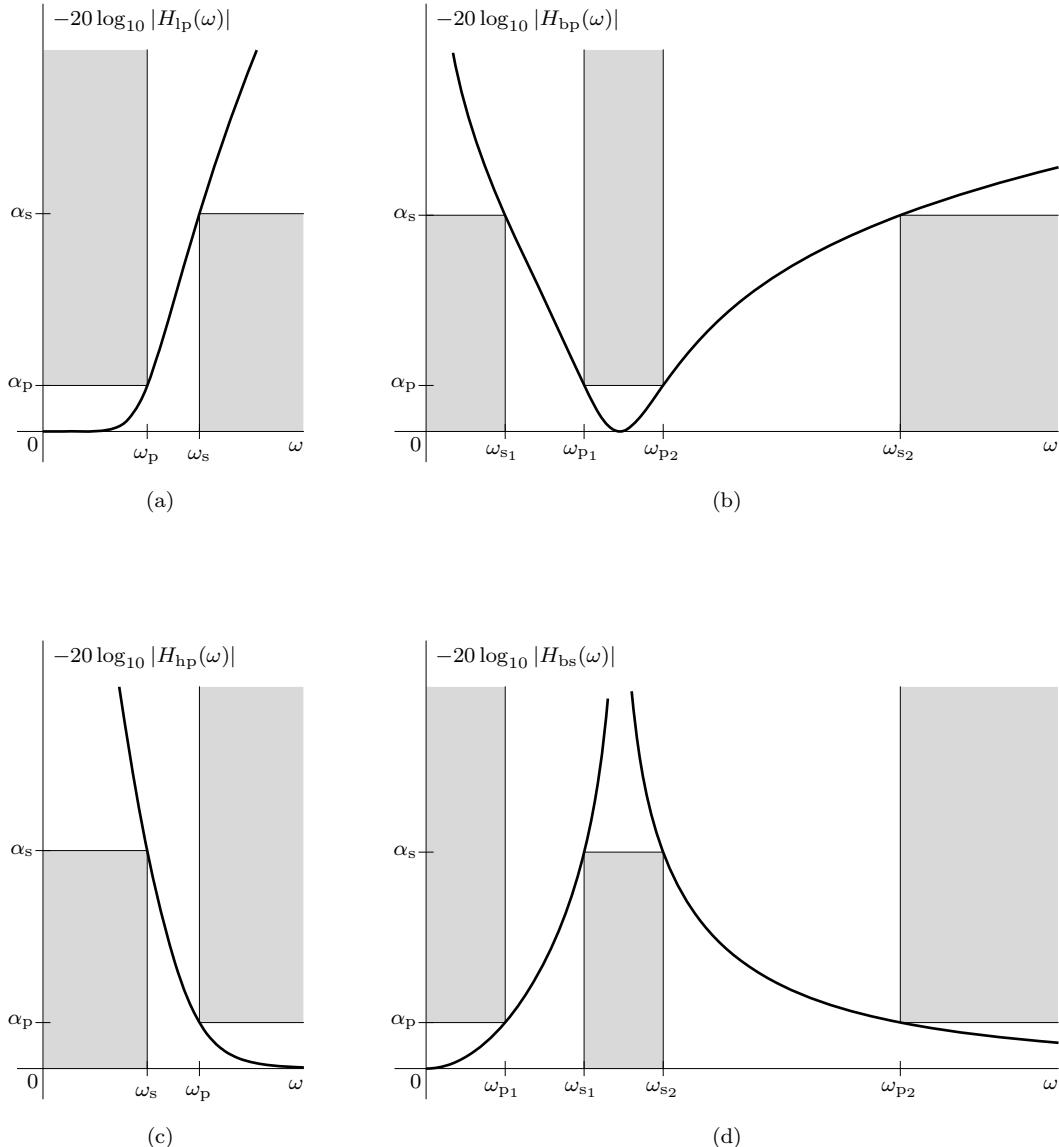


Figure 2.23: Practical filter specifications using attenuation: (a) lowpass, (b) bandpass, (c) highpass, and (d) bandstop filters.

by separating filter type (lowpass, highpass, bandpass, or bandstop) from filter family (Butterworth, Chebyshev, inverse Chebyshev, etc.). Consequently, we concentrate our design efforts on a *prototype filter* (usually lowpass) and then transform it to the final desired filter type.

In this section, we present four common analog filter transformations. Commonly referred to as *frequency transformations*, each begins with a lowpass response to produce, respectively, a lowpass, highpass, bandpass, or bandstop response. The latter three transforms ensure that a good lowpass filter design (prototype filter) automatically provides good highpass, bandpass, and bandstop filter designs. The lowpass-to-lowpass transform encourages the use of a *normalized* lowpass prototype filter, where a critical frequency parameter, such as ω_p or ω_s , is set to unity. Each transformation involves a simple substitution for s or ω in the prototype's system function $H_p(s)$ or frequency

response $H_p(\omega)$. Design examples that utilize frequency transformations are presented in this section and later in Sec. 2.7.

2.6.1 Lowpass-to-Lowpass Transformation

The lowpass-to-lowpass transformation is the most direct and simple and is given by

$$s \rightarrow \frac{\omega_0}{\omega_1} s \quad \text{or} \quad \omega \rightarrow \frac{\omega_0}{\omega_1} \omega. \quad (2.25)$$

Equation (2.25) scales (see Sec. 1.2.2) the original response by a factor $\frac{\omega_0}{\omega_1}$. Consequently, the lowpass-to-lowpass transformation is sometimes simply referred to as *frequency scaling*. By scaling frequencies, Eq. (2.25) maps the prototype's response at ω_0 to a new frequency ω_1 . Thus, a new lowpass response with critical frequency ω_1 is obtained from a prototype lowpass response with critical frequency ω_0 according to

$$H_{lp}(s) = H_p(s)|_{s \rightarrow \frac{\omega_0}{\omega_1} s} = H_p\left(\frac{\omega_0}{\omega_1} s\right) \quad \text{or} \quad H_{lp}(\omega) = H_p(\omega)|_{\omega \rightarrow \frac{\omega_0}{\omega_1} \omega} = H_p\left(\frac{\omega_0}{\omega_1} \omega\right). \quad (2.26)$$

Figure 2.24 illustrates the lowpass-to-lowpass transformation graphically. The prototype response $|H_p(\omega)|^2$ (note the reverse orientation of the vertical axis) is shown in the lower right, the transformation rule is shown in the upper right, and the transformed response $|H_{lp}(\omega)|^2$ is shown in the upper left. Dotted lines show how the prototype's frequency ω_0 , in this case set to the half-power point, maps to the new frequency ω_1 . Since the transformed filter in Eq. (2.26) equals, except in argument, the original prototype filter, the transformed filter retains the essential nature and character of the prototype; the smooth prototype filter shown in Fig. 2.24 must transform into a similar smooth filter.

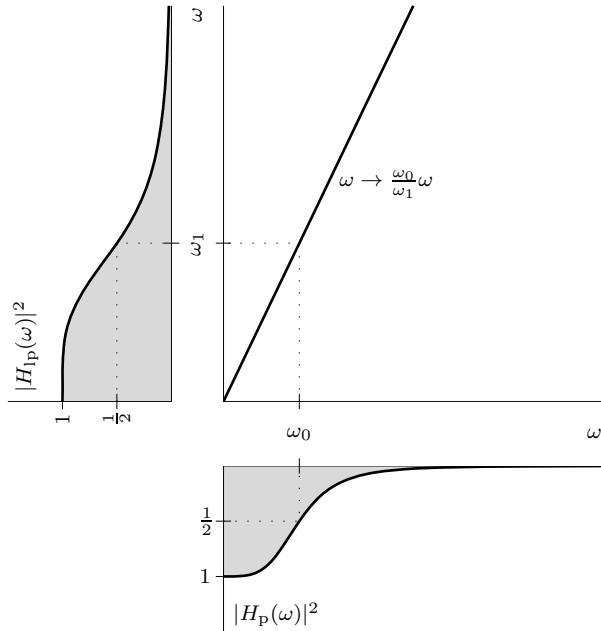


Figure 2.24: Lowpass-to-lowpass frequency transformation.

▷ **Drill 2.7 (Lowpass-to-Lowpass Transformation)**

A lowpass prototype filter is given as $H_p(s) = \frac{2}{s+2}$. Determine the lowpass-to-lowpass transformation rule to produce a lowpass filter $H_{lp}(s)$ with a half-power (3 dB) frequency of $\omega_1 = 3$ rad/s. Plot the magnitude response of the resulting lowpass filter.

△

2.6.2 Lowpass-to-Highpass Transformation

Frequencies undergo role reversals in the lowpass-to-highpass transformation, which is given by

$$s \rightarrow \frac{\omega_0\omega_1}{s} \quad \text{or} \quad \omega \rightarrow \frac{\omega_0\omega_1}{-\omega}. \quad (2.27)$$

As with the lowpass-to-lowpass transformation, Eq. (2.27) maps the prototype's response at $\pm\omega_0$ to new frequencies $\mp\omega_1$. Further, the reciprocal nature of the transformation causes high frequencies to exchange roles with low frequencies and vice versa. Thus, a highpass response with critical frequency ω_1 is obtained from a prototype lowpass response with critical frequency ω_0 according to

$$H_{hp}(s) = H_p\left(\frac{\omega_0\omega_1}{s}\right) \quad \text{or} \quad H_{hp}(\omega) = H_p\left(\frac{\omega_0\omega_1}{-\omega}\right). \quad (2.28)$$

Omitting, for convenience, the sign reversal in the transformation rule, Fig. 2.25 illustrates the lowpass-to-highpass transformation graphically.[†] As Eq. (2.27) requires, the transformed highpass filter retains the essential character of the lowpass prototype.

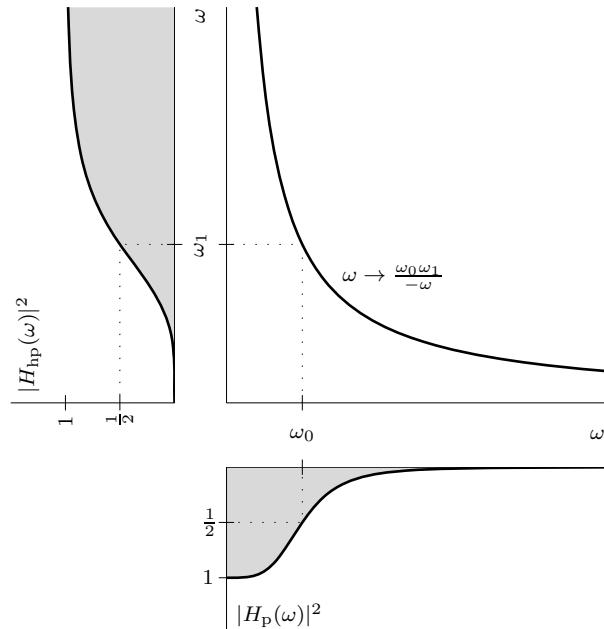


Figure 2.25: Lowpass-to-highpass frequency transformation.

[†]The sign reversal is unimportant for real filters; care should be taken, however, if the prototype is a complex filter. See Prob. 2.6-3.

▷ **Example 2.8 (Lowpass-to-Highpass Transformation)**

Apply the lowpass-to-highpass transformation $\omega \rightarrow \frac{2}{-\omega}$ to the lowpass prototype $H_p(\omega) = \frac{1}{j\omega+1}$. Plot the magnitude response, and determine the 3-dB frequency of the resulting filter.

The lowpass-to-highpass transformation yields

$$H_{hp}(\omega) = H_p(\omega)|_{\omega \rightarrow \frac{2}{-\omega}} = \frac{1}{-j\frac{2}{\omega} + 1} = \frac{-\omega}{\omega - j2}.$$

The prototype filter has 3-dB point $\omega_0 = 1$ rad/s. Since $\frac{\omega_0 \omega_1}{\omega} = \frac{2}{\omega}$, the highpass filter's 3-dB point is

$$\omega_1 = \frac{2}{\omega_0} = 2.$$

Figure 2.26, generated using MATLAB, shows the resulting highpass magnitude response and confirms that the 3-dB frequency ω_1 is equal to 2 rad/s.

```
01 omega = 0:.01:6; Hhp = @(omega) -omega./(omega-1j*2);
02 plot(omega,abs(Hhp(omega)));
03 xlabel('omega'); ylabel('|H_hp(\omega)|');
```

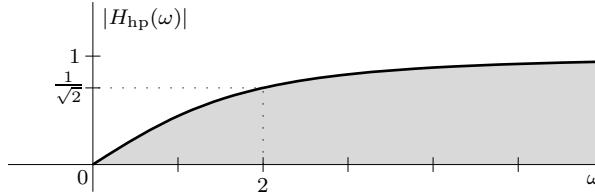


Figure 2.26: A plot of $|H_{hp}(\omega)| = \left| \frac{-\omega}{\omega - j2} \right|$.

Example 2.8 ▷

2.6.3 Lowpass-to-Bandpass Transformation

The lowpass-to-bandpass transformation utilizes a one-to-two mapping: each point of the original lowpass response maps to two points of the bandpass response. In this way, the lowpass filter's single cutoff frequency becomes both the upper and lower cutoff frequencies required of the bandpass response. Mathematically, the lowpass-to-bandpass transformation is given by

$$s \rightarrow \omega_0 \frac{s^2 + \omega_1 \omega_2}{s(\omega_2 - \omega_1)} \quad \text{or} \quad \omega \rightarrow \omega_0 \frac{\omega^2 - \omega_1 \omega_2}{\omega(\omega_2 - \omega_1)}, \quad (2.29)$$

where it is generally assumed that $\omega_2 > \omega_1$.

The nature of Eq. (2.29) ensures that the order, and therefore the complexity, of the bandpass filter is double that of the prototype. It is not difficult to show that the response of the prototype at frequencies $\pm\omega_0$ maps to the frequencies $\mp\omega_1$ and $\pm\omega_2$; readers are encouraged to work out the details. A bandpass response with critical frequencies ω_1 and ω_2 is therefore obtained from a prototype lowpass response with critical frequency ω_0 according to

$$H_{bp}(s) = H_p \left(\omega_0 \frac{s^2 + \omega_1 \omega_2}{s(\omega_2 - \omega_1)} \right) \quad \text{or} \quad H_{bp}(\omega) = H_p \left(\omega_0 \frac{\omega^2 - \omega_1 \omega_2}{\omega(\omega_2 - \omega_1)} \right). \quad (2.30)$$

Again omitting any sign reversals in the transformation rule, Fig. 2.27 graphically illustrates the lowpass-to-bandpass transformation (see footnote on page 116). As Eq. (2.29) requires, the transformed bandpass filter retains the essential features, such as smoothness and ripple levels, of the lowpass prototype.

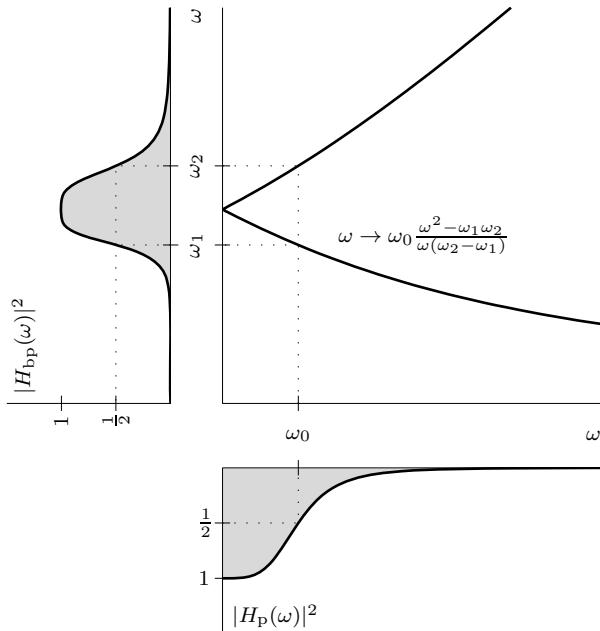


Figure 2.27: Lowpass-to-bandpass frequency transformation.

▷ Drill 2.8 (Lowpass-to-Bandpass Transformation)

A lowpass prototype filter is given as $H_p(s) = \frac{2}{s+2}$. Determine the lowpass-to-bandpass transformation rule to produce a bandpass filter $H_{bp}(s)$ with half-power (3 dB) cutoff frequencies $\omega_1 = 1$ rad/s and $\omega_2 = 3$ rad/s. Plot the magnitude response of the resulting bandpass filter.

△

2.6.4 Lowpass-to-Bandstop Transformation

Much as the lowpass-to-highpass transformation reciprocally relates to the lowpass-to-lowpass transformation, the lowpass-to-bandstop transformation reciprocally relates to the lowpass-to-bandpass transformation. Mathematically, the lowpass-to-bandstop transformation is thus given by

$$s \rightarrow \omega_0 \frac{s(\omega_2 - \omega_1)}{s^2 + \omega_1\omega_2} \quad \text{or} \quad \omega \rightarrow \omega_0 \frac{\omega(\omega_2 - \omega_1)}{-\omega^2 + \omega_1\omega_2}, \quad (2.31)$$

where it is generally assumed that $\omega_2 > \omega_1$. Aside from the ω_0 term, the left-hand expression in Eq. (2.31) is indeed the mathematical reciprocal of the corresponding expression in Eq. (2.29).

As with the lowpass-to-bandpass transformation, the nature of Eq. (2.31) ensures that the order of the bandstop filter is double that of the prototype and that the response of the prototype at frequencies $\pm\omega_0$ maps to the frequencies $\mp\omega_1$ and $\pm\omega_2$. A bandstop response with critical frequencies ω_1 and ω_2 is therefore obtained from a prototype lowpass response with critical frequency ω_0 .

according to

$$H_{\text{bs}}(s) = H_p \left(\omega_0 \frac{s(\omega_2 - \omega_1)}{s^2 + \omega_1 \omega_2} \right) \quad \text{or} \quad H_{\text{bs}}(\omega) = H_p \left(\omega_0 \frac{\omega(\omega_2 - \omega_1)}{-\omega^2 + \omega_1 \omega_2} \right). \quad (2.32)$$

Again omitting any sign reversals in the transformation rule, Fig. 2.28 graphically illustrates the lowpass-to-bandstop transformation (see footnote on page 116). As Eq. (2.31) requires, the transformed bandpass filter retains the essential features, such as smoothness and ripple levels, of the lowpass prototype.

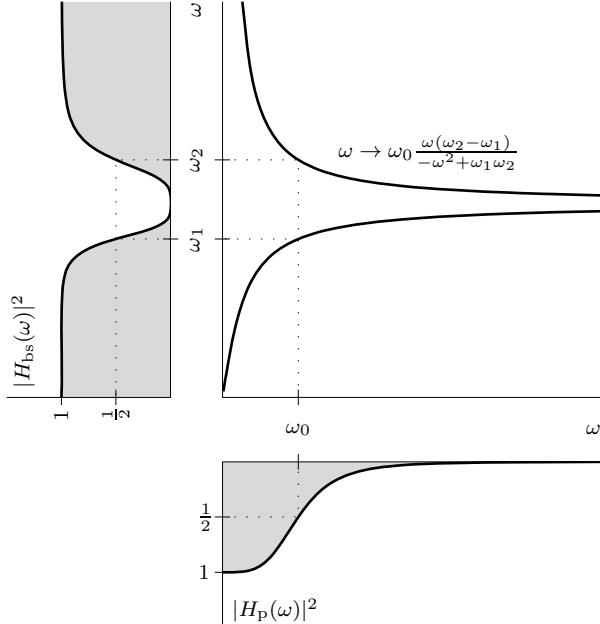


Figure 2.28: Lowpass-to-bandstop frequency transformation.

▷ Example 2.9 (Lowpass-to-Bandstop Transformation)

Apply a lowpass-to-bandstop transformation to the lowpass prototype $H_p(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$. The resulting bandstop filter should have 3-dB cutoff frequencies of 1 and 3 rad/s. Plot the magnitude response of the resulting filter.

The prototype filter has 3-dB point $\omega_0 = 1$ rad/s. Using the target 3-dB frequencies of $\omega_1 = 1$ and $\omega_2 = 3$ rad/s and Eq. (2.31), the lowpass-to-bandstop transformation is

$$s \rightarrow \omega_0 \frac{s(3-1)}{s^2 + 1(3)} = \frac{2s}{s^2 + 3}.$$

Since $H_p(s)$ is relatively simple, direct substitution is reasonable and yields

$$H_{\text{bs}}(s) = H_p(s) \Big|_{s \rightarrow \frac{2s}{s^2+3}} = \frac{1}{\left(\frac{2s}{s^2+3}\right)^2 + \sqrt{2}\frac{2s}{s^2+3} + 1} = \frac{(s^2 + 3)^2}{s^4 + 2\sqrt{2}s^3 + 10s^2 + 6\sqrt{2}s + 9}.$$

For higher-order systems, it is often more simple to perform the substitution of Eq. (2.31) using the *factored form* of the transfer function as given, for example, in Eq. (1.52).

The corresponding frequency response is easily obtained by evaluating $H_{\text{bs}}(s)$ for $s = j\omega$. Figure 2.29, generated using MATLAB, shows the resulting bandstop magnitude response and confirms that the 3-dB frequencies are 1 and 3 rad/s. Notice that $|H_{\text{bs}}(j\omega)| = 0$ at the geometric mean of the 3-dB frequencies, $\sqrt{\omega_1\omega_2} = \sqrt{3}$, not the arithmetic mean.

```
01 Hbs = @(s) (s.^2+3).^2./(s.^4+2*sqrt(2)*s.^3+10*s.^2+6*sqrt(2)*s+9);
02 omega = 0:.01:6; plot(omega,abs(Hbs(1j*omega)));
03 xlabel('omega'); ylabel('|H_bs(j\omega)|');
```

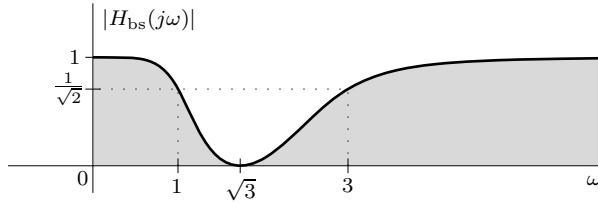


Figure 2.29: A plot of $|H_{\text{bs}}(j\omega)| = \left| \frac{(s^2+3)^2}{s^4+2\sqrt{2}s^3+10s^2+6\sqrt{2}s+9} \right|_{s=j\omega}$.

Example 2.9 ▶

2.7 Practical Filter Families

While various truncations of the impulse response can render a filter theoretically realizable, such methods do not provide a clear path to implement a continuous-time filter in hardware.[†] The truncations of Secs. 2.3 and 2.4, for example, do not produce a rational transfer function $H(s)$. A rational transfer function is desirable as it corresponds directly to a constant-coefficient linear differential equation (see Eq. (1.37) on page 28), and such equations are readily realized with electric circuits.

Fortunately, there are several families of practical filters that are realizable (causal) and that possess rational transfer functions, including Butterworth, Chebyshev, and inverse Chebyshev filters. This section covers these three families of filters, providing a variety of design examples along the way, and briefly introduces the elliptic and Bessel-Thomson filter families as well. For each filter family, a lowpass prototype is presented; highpass, bandpass, and bandstop responses are obtained by applying the transformations of Sec. 2.6. The discussion here is not comprehensive but is basically a summary of the design techniques. More discussion can be found elsewhere in the literature [5]. We begin with Butterworth filters.

2.7.1 Butterworth Filters

The magnitude response $|H(j\omega)|$ of an K th-order Butterworth lowpass filter, depicted in Fig. 2.30 for various orders, is given by

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \left(\frac{\omega}{\omega_c}\right)^{2K}}} \quad (2.33)$$

[†]Although window-based filter design is not particularly useful for continuous-time filters, it is, as we shall see in Ch. 8, very useful for the design of discrete-time finite-impulse response filters.

Observe that at $\omega = 0$, the gain $|H(j0)|$ is unity. At $\omega = \omega_c$, the gain $|H(j\omega_c)| = 1/\sqrt{2}$ or -3 dB. Thus, the Butterworth design parameter ω_c corresponds to the filter's half-power cutoff frequency, also called the 3-dB cutoff frequency. For large K , the Butterworth magnitude response approaches the ideal characteristic (shaded).

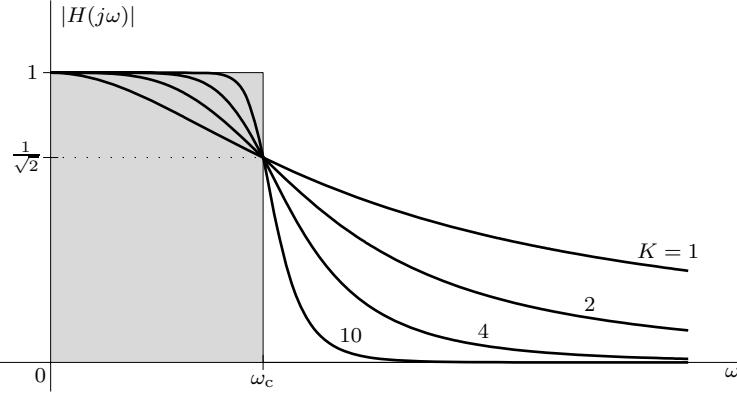


Figure 2.30: Butterworth magnitude responses for $K = 1, 2, 4$, and 10 .

The Butterworth magnitude response decreases monotonically, and the first $2K - 1$ derivatives of the magnitude response are zero at $\omega = 0$. For this reason, the Butterworth response is called *maximally flat* at $\omega = 0$. Observe that a constant characteristic (ideal) is flat for all $\omega < 1$. In the Butterworth filter we try to retain this property at least at the origin.[†]

To determine the corresponding transfer function $H(s)$, recall that, for real filters at least, $H(-j\omega)$ is the complex conjugate of $H(j\omega)$. Thus,

$$|H(j\omega)|^2 = H(j\omega)H^*(j\omega) = H(j\omega)H(-j\omega).$$

Substituting $\omega = s/j$ into this equation and using Eq. (2.33), we obtain

$$H(s)H(-s) = \frac{1}{1 + \left(\frac{s}{j\omega_c}\right)^{2K}}.$$

Consequently, the Butterworth filter is an all-pole transfer function of the form

$$H(s) = \frac{\omega_c^K}{\prod_{k=1}^K (s - p_k)}, \quad (2.34)$$

where p_k designate the system poles. Fortunately, the poles of a Butterworth lowpass filter are quite simple to find.

The poles of $H(s)H(-s)$ satisfy

$$s^{2K} = -(j\omega_c)^{2K} = e^{j\pi(2k-1)} (j\omega_c)^{2K}.$$

Solving this equation yields $2K$ poles

$$p_k = j\omega_c e^{\frac{j\pi}{2K}(2k-1)} \quad k = 1, 2, 3, \dots, 2K.$$

In words, the $2K$ poles of $H(s)H(-s)$ are spaced equally by π/K radians around a circle of radius ω_c centered at the origin in the complex plane, as shown in Fig. 2.31 for even ($K = 4$) and odd ($K = 5$) cases.

[†]Butterworth filters also exhibit a maximally flat characteristic at $\omega = \infty$.

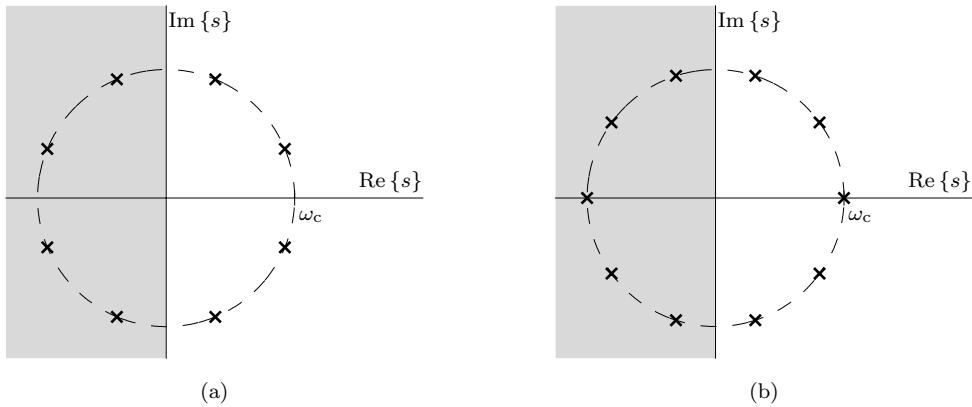


Figure 2.31: Butterworth poles and mirrors: (a) even order ($K = 4$) and (b) odd order ($K = 5$).

Of the $2K$ poles of $H(s)H(-s)$, K belong to our desired filter $H(s)$ and K belong to $H(-s)$, what we might consider the evil twin of $H(s)$. Since our desired filter is constrained to be both causal and stable, the poles of $H(s)$ must be the K left half-plane poles of $H(s)H(-s)$, such as those shown in the shaded regions of Fig. 2.31. The remaining K right half-plane poles (unshaded regions) belong to $H(-s)$, which forces this twin filter to be either noncausal (unrealizable) or unstable (unbounded output behavior). The virtuous $H(s)$ can do no wrong, while the wicked $H(-s)$ can do no right. Conveniently, the K left half-plane poles corresponding to $H(s)$ are given by

$$\begin{aligned} p_k &= j\omega_c e^{\frac{j\pi}{2K}(2k-1)} \\ &= -\omega_c \sin\left[\frac{\pi(2k-1)}{2K}\right] + j\omega_c \cos\left[\frac{\pi(2k-1)}{2K}\right] \quad k = 1, 2, 3, \dots, K. \end{aligned} \quad (2.35)$$

When the poles of Eq. (2.35) are inserted into Eq. (2.34) for the normalized case of $\omega_c = 1$, the resulting denominator is a *normalized Butterworth polynomial*. Analog filter design books, particularly older ones, almost always include tables of normalized Butterworth polynomials such as those given in Tables 2.2 and 2.3.

K	a_7	a_6	a_5	a_4	a_3	a_2	a_1
2							1.414214
3						2.000000	2.000000
4					2.613126	3.414214	2.613126
5				3.236068	5.236068	5.236068	3.236068
6			3.863703	7.464102	9.141620	7.464102	3.863703
7		4.493959	10.097835	14.591794	14.591794	10.097835	4.493959
8	5.125831	13.137071	21.846151	25.688356	21.846151	13.137071	5.125831

Table 2.2: Coefficients of normalized Butterworth polynomials $s^K + a_{K-1}s^{K-1} + \dots + a_1s + 1$.

Tables 2.2 and 2.3 can simplify filter design by eliminating potentially cumbersome calculations. Arbitrary cutoff frequencies ($\omega_c \neq 1$) are easily obtained by frequency scaling the normalized design. While effective for low-order designs common for analog implementation, tables tend to lack the necessary precision for high-order filter design that is currently common in digital implementations. Thus, direct computer calculation is usually preferable to the use of tables.

K	Normalized Butterworth Polynomials, factored form
1	$s + 1$
2	$s^2 + 1.414214s + 1$
3	$(s + 1)(s^2 + s + 1)$
4	$(s^2 + 0.765367s + 1)(s^2 + 1.847759s + 1)$
5	$(s + 1)(s^2 + 0.618034s + 1)(s^2 + 1.618034s + 1)$
6	$(s^2 + 0.517638s + 1)(s^2 + 1.414214s + 1)(s^2 + 1.931852s + 1)$
7	$(s + 1)(s^2 + 0.445042s + 1)(s^2 + 1.246980s + 1)(s^2 + 1.801938s + 1)$
8	$(s^2 + 0.390181s + 1)(s^2 + 1.111140s + 1)(s^2 + 1.662939s + 1)(s^2 + 1.961571s + 1)$

Table 2.3: Normalized Butterworth polynomials in factored form.

▷ **Example 2.10 (Butterworth Design Using Tables and Direct Calculation)**

Find the transfer function $H(s)$ of a fourth-order lowpass Butterworth filter with a 3-dB cutoff frequency $\omega_c = 10$ by (a) frequency scaling the appropriate Table 2.2 entry and (b) direct calculation. Plot the magnitude response to verify filter characteristics.

(a) Butterworth Design Using Tables and Frequency Scaling

Using the $K = 4$ entry of Table 2.2, the transfer function of the fourth-order normalized ($\omega_c = 1$) Butterworth lowpass prototype filter is

$$H_p(s) = \frac{1}{s^4 + 2.613126s^3 + 3.414214s^2 + 2.613126s + 1}.$$

The lowpass-to-lowpass transformation $s \rightarrow \frac{s}{10}$, which scales the cutoff frequency to the desired value of 10 rad/s, yields

$$H(s) = \frac{10^4}{s^4 + 26.13126s^3 + 341.4214s^2 + 2613.126s + 10^4}.$$

(b) Butterworth Design Using Direct Calculation

Using Eq. (2.35), we use MATLAB to compute the Butterworth pole locations p_k .

```
01 K = 4; k=1:K; omegac = 10;
02 p = 1j*omegac*exp(1j*pi/(2*K)*(2*k-1))
   p = -3.8268+9.2388i -9.2388+3.8268i -9.2388-3.8268i -3.8268-9.2388i
```

Next, these complex poles are expanded into polynomial form.

```
03 A = poly(p)
A = 1.0000 26.1313 341.4214 2613.1259 10000.0000
```

Substituting these polynomial coefficients into Eq. (2.34) yields the desired transfer function

$$H(s) = \frac{10^4}{s^4 + 26.1313s^3 + 341.4214s^2 + 2613.1259s + 10^4}.$$

This result agrees with the result of part (a).

With the transfer function coefficients in hand, it is simple to generate the magnitude response.

```
04 omega = linspace(0,35,1001); H = 10^4./polyval(A,1j*omega);
05 plot(omega,abs(H)); xlabel('omega'); ylabel('|H(j\omega)|');
```

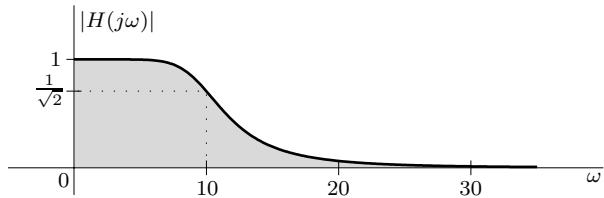


Figure 2.32: Magnitude response of lowpass Butterworth filter with $\omega_c = 10$.

The result, shown in Fig. 2.32, confirms that the response is Butterworth in nature and that the half-power cutoff frequency is correctly located at $\omega_c = 10$ rad/s.

Example 2.10 ◀

Determination of Butterworth Filter Order and Half-Power Frequency

One of the problems with Ex. 2.10 is that filter order K and half-power cutoff frequency ω_c are rarely known up front. Rather, filters tend to be specified in terms of passband and stopband parameters such as those presented in Sec. 2.5. Thus, we need a mechanism to convert these general filter specifications into the Butterworth design parameters of K and ω_c .

Expressing the maximum passband attenuation α_p and minimum stopband attenuation α_s , both in dB, in terms of the Butterworth response given by Eq. (2.33) yields

$$\alpha_p = -20 \log_{10} |H(j\omega_p)| = 10 \log_{10} \left[1 + \left(\frac{\omega_p}{\omega_c} \right)^{2K} \right] \quad (2.36)$$

and

$$\alpha_s = -20 \log_{10} |H(j\omega_s)| = 10 \log_{10} \left[1 + \left(\frac{\omega_s}{\omega_c} \right)^{2K} \right]. \quad (2.37)$$

Combining Eqs. (2.36) and (2.37) and solving for integer K yield

$$K = \left\lceil \frac{\log \sqrt{(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)}}{\log(\omega_s/\omega_p)} \right\rceil = \left\lceil \frac{\log [(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)]}{2 \log(\omega_s/\omega_p)} \right\rceil. \quad (2.38)$$

In Eq. (2.38), the ceiling function $\lceil \cdot \rceil$ ensures that an integer value is returned for K , as required.[†] Since K is (almost always) rounded up, the resulting filter will (almost always) exceed design requirements. Exactly how the filter exceeds specifications depends on our choice of ω_c . Solving Eq. (2.36) for ω_c yields

$$\omega_c = \frac{\omega_p}{(10^{\alpha_p/10} - 1)^{1/(2K)}}.$$

In this case, the filter will exactly meet passband specifications and will exceed stopband specifications. Alternatively, solving Eq. (2.37) for ω_c yields

$$\omega_c = \frac{\omega_s}{(10^{\alpha_s/10} - 1)^{1/(2K)}}.$$

[†]Equation (2.38) provides the smallest integer K that ensures filter specifications are met. Although larger values of K can also be used, the added complexity is not usually justified.

With this choice, the filter will exactly meet stopband specifications and will exceed passband specifications. Any choice of ω_c between these two extremes will work. Thus, assuming $\omega_p < \omega_s$, the half-power frequency ω_c must satisfy

$$\frac{\omega_p}{(10^{\alpha_p/10} - 1)^{1/2K}} \leq \omega_c \leq \frac{\omega_s}{(10^{\alpha_s/10} - 1)^{1/2K}}. \quad (2.39)$$

A One-Bumper Car?

If ω_c is chosen at either extreme, either the passband or the stopband specifications are met exactly with no margin for error. This is somewhat akin to driving a car with only one bumper. As long as nothing goes wrong, you never notice the missing bumper. However, if an accident occurs on the side without a bumper, you're bound to encounter some damage. We prefer cars with both bumpers. The buffer zones created by choosing ω_c at some convenient middle point of Eq. (2.39) serve as our filter's bumpers. With a buffer on either side, we help protect our filter from the damage of minor accidents, such as component variations or coefficient truncations. The next example clarifies these ideas.

▷ Example 2.11 (Butterworth Lowpass Filter Design)

Design the lowest-order Butterworth lowpass filter that meets the specifications $\omega_p \geq 10$ rad/s, $\alpha_p \leq 2$ dB, $\omega_s \leq 30$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

Step 1: Determine Order K

Using $\omega_p = 10$ rad/s, $\alpha_p = 2$ dB, $\omega_s = 30$ rad/s, and $\alpha_s = 20$ dB, MATLAB evaluation of Eq. (2.38) yields the necessary order K .

```
01 omegap = 10; alphap = 2; omegas = 30; alphas = 20;
02 K = ceil(log((10^(alphas/10)-1)/(10^(alphap/10)-1))/(2*log(omegas/omegap)))
K = 3
```

Step 2: Determine Half-Power Cutoff Frequency ω_c

Next, the range of possible ω_c is determined using Eq. (2.39).

```
03 omegac = [omegap/(10^(alphap/10)-1).^(1/(2*K)), omegas/(10^(alphas/10)-1).^(1/(2*K))]
omegac = 10.9350 13.9481
```

Choosing $\omega_c = 10.9350$ rad/s, which exactly meets passband specifications and exceeds stopband specifications, provides no margin for error on the passband side. Similarly, choosing $\omega_c = 13.9481$ rad/s, which just meets stopband requirements and surpasses passband requirements, provides no margin for error on the stopband side. A buffer zone is provided on both the passband and stopband sides by choosing some convenient middle value, such as

$$\omega_c = 12.5 \text{ rad/s.}$$

Step 3: Determine Transfer Function $H(s)$

Similar to Ex. 2.10, MATLAB conveniently computes the desired transfer function coefficients.

```
04 omegac = 12.5; k = 1:K; A = poly(1j*omegac*exp(1j*pi/(2*K)*(2*k-1)))
A = 1.0000 25.0000 312.5000 1953.1250
```

Alternatively, the lowpass-to-lowpass transformation $s \rightarrow s/\omega_c = s/12.5$ applied to the appropriate coefficients of Table 2.2 yields the same denominator.

```
05 A = [1 2 2 1].*[1 omegac omegac^2 omegac^3]
A = 1.0000 25.0000 312.5000 1953.1250
```

Computed either way, the transfer function is

$$H(s) = \frac{1953.125}{s^3 + 25s^2 + 312.5s + 1953.125}.$$

The corresponding magnitude response, plotted using MATLAB, is shown in Fig. 2.33.

```
06 omega = linspace(0,35,1001); H = omegac^K./(polyval(A,1j*omega));
07 plot(omega,abs(H));
```

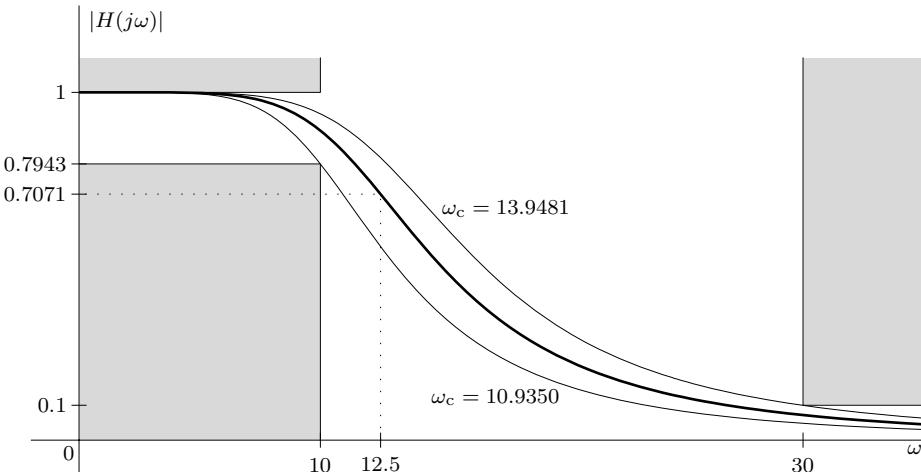


Figure 2.33: Lowpass Butterworth magnitude response with $K = 3$ and $\omega_c = 12.5$.

Several observations regarding Fig. 2.33 are in order. Since our plot is not in dB, the attenuation parameters are converted, which facilitates verification that the filter does indeed meet specification. Following Eq. (2.24), the passband floor is $1 - \delta_p = 10^{-\alpha_p/20} = 0.7943$, and the stopband ceiling is $\delta_s = 10^{-\alpha_s/20} = 0.1$. For comparison purposes, the figure also shows the Butterworth responses corresponding to the limiting values of ω_c as computed in line 03. As predicted, the $\omega_c = 10.9350$ curve exactly meets passband requirements and exceeds stopband requirements. Similarly, the $\omega_c = 13.9481$ curve exactly meets stopband requirements and exceeds passband requirements. Answering the call for safety, $\omega_c = 12.5$ equips our filter with buffer zones on both the passband and stopband sides.

Example 2.11 ◇

▷ Example 2.12 (Butterworth Bandpass Filter Design)

Design the lowest-order Butterworth bandpass filter that meets the specifications $\omega_{p_1} \leq 10$ rad/s, $\omega_{p_2} \geq 20$ rad/s, $\alpha_p \leq 2$ dB, $\omega_{s_1} \geq 5$ rad/s, $\omega_{s_2} \leq 35$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

The solution to this problem is executed in two steps. In the first step and similar to Ex. 2.11, we determine a suitable lowpass prototype. In the second step, a lowpass-to-bandpass transformation is applied to the lowpass prototype to obtain the desired Butterworth bandpass filter transfer function.

Step 1: Determine the Lowpass Prototype

Before we can determine the order K of the lowpass prototype, we need to determine how the bandpass characteristics relate to the lowpass prototype's stopband-to-passband ratio ω_s/ω_p . To this end, we normalize the passband frequency $\omega_p = 1$ and then use Eq. (2.29) to solve for the stopband

frequency, ω_s . Since there are two transition bands in the bandpass filter, two candidate values of ω_s are computed.

```
01 omegap1 = 10; omegap2 = 20; omegas1 = 5; omegas2 = 35; omegap = 1;
02 omegas = abs([omegap*(omegas1^2-omegap1*omegap2)/(omegas1*(omegap2-omegap1)),...
03           omegap*(omegas2^2-omegap1*omegap2)/(omegas2*(omegap2-omegap1))])
omegas = 3.5000 2.9286
```

These two values demonstrate an interesting fact: even though the bandpass filter's lower transition band is physically narrower than its upper transition band (5-rad/s width versus 15-rad/s width), it is the upper transition band that is most restrictive to the design. That is, the upper transition band in this particular example maps to a narrower interval for the lowpass prototype than does the lower transition band.

Since both transition bands need to be satisfied, the smallest (most restrictive) value is selected, $\omega_s = 2.9286$. Substituting this value into Eq. (2.38) yields the necessary order of the prototype filter.

```
04 omegas = min(omegas); alphap = 2; alphas = 20;
05 K = ceil(log((10^(alphas/10)-1)/(10^(alphap/10)-1))/(2*log(omegas/omegap)))
K = 3
```

The limiting values of ω_c are next calculated.

```
06 omegac = [omegap/(10^(alphap/10)-1).^(1/(2*K)),omegas/(10^(alphas/10)-1).^(1/(2*K))]
omegac = 1.0935 1.3616
```

Setting ω_c to the middle of these values, the coefficients of the lowpass prototype are next computed.

```
07 omegac = mean(omegac); k = 1:K;
08 pk = (1j*omegac*exp(1j*pi/(2*K)*(2*k-1))); A = poly(pk)
A = 1.000000 2.455106 3.013772 1.849782
```

Thus, the transfer function of the lowpass prototype is

$$H_{lp}(s) = \frac{1.849782}{s^3 + 2.455106s^2 + 3.013772s + 1.849782}. \quad (2.40)$$

Step 2: Determine the Bandpass Filter

Using Eq. (2.29), a simple transformation converts our lowpass prototype to the desired bandpass filter,

$$s \rightarrow \omega_p \frac{s^2 + \omega_{p_1}\omega_{p_2}}{s(\omega_{p_2} - \omega_{p_1})} = 1 \frac{s^2 + 10(20)}{s(20 - 10)} = \frac{s^2 + 200}{10s}.$$

This transformation doubles our prototype's order. Although it can be applied directly and with modest manual effort to Eq. (2.40), it is informative to develop MATLAB code that automates the task and extends to more difficult, higher-order cases.

Since our prototype is an all-pole filter, it is sufficient to consider what happens when our transformation is applied to a single pole p_k ,

$$\frac{1}{s - p_k} \rightarrow \frac{1}{\frac{s^2+200}{10s} - p_k} = \frac{10s}{s^2 - 10p_k s + 200}.$$

Thus, our single pole becomes a pair of poles along with the addition of one zero at $s = 0$.

Designating $as^2 + bs + c = s^2 - p_k(\omega_{p_2} - \omega_{p_1})s + \omega_{p_1}\omega_{p_2} = s^2 - 10p_k s + 200$, the quadratic formula $(-b \pm \sqrt{b^2 - 4ac})/(2a)$ allows us to compute the two poles that result from the transformation of each pole p_k . The roots for both numerator and denominator are then expanded into polynomial form.

```

09 a = 1; b = -pk*(omegap2-omegap1); c = omegap1*omegap2;
10 pk = [(-b+sqrt(b.^2-4*a*c))./(2*a),(-b-sqrt(b.^2-4*a*c))./(2*a)];
11 B = (omegap1^K)*((omegap2-omegap1)^K)*poly(zeros(K,1)), A = poly(pk)
B = 1849.7824 0.0000 0.0000 0.0000
A = 1.0000 24.5511 901.3772 11670.2056 180275.4443 982042.3150 8000000.0000

```

Notice that the numerator requires an ω_c^K term, as shown in Eq. (2.34).

Using these results, the transfer function of our order-6 bandpass filter is thus

$$H(s) = \frac{1849.78s^3}{s^6 + 24.55s^5 + 901.38s^4 + 11670.21s^3 + 180275.44s^2 + 982042.32s + 8000000}.$$

The filter's magnitude response, plotted using MATLAB, is shown in Fig. 2.34.

```

12 omega = linspace(0,40,1001); H = polyval(B,1j*omega)./polyval(A,1j*omega);
13 plot(omega,abs(H));

```

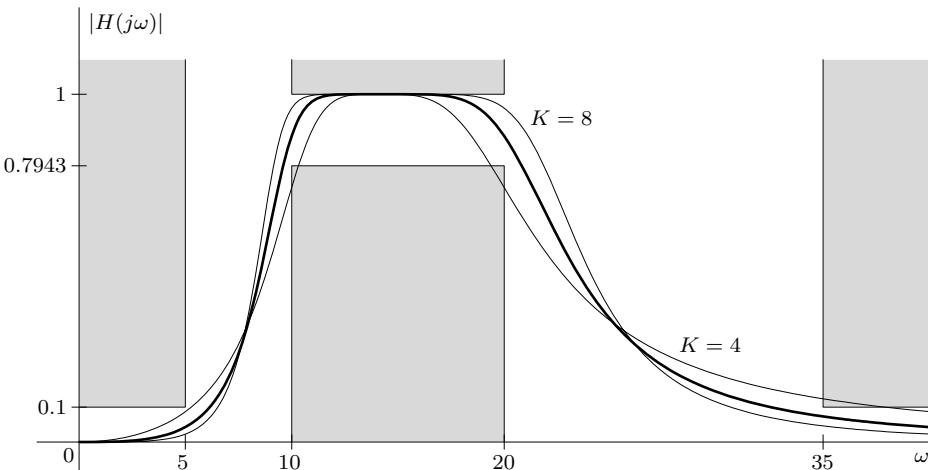


Figure 2.34: Order-6 Butterworth bandpass filter magnitude response $|H(j\omega)|$.

Several comments regarding Fig. 2.34 are relevant. Since our magnitude plot is not in dB, the attenuation parameters are converted as in Ex. 2.11 to simplify verification of filter performance. Since a compromise value of ω_c is chosen for the prototype, the resulting filter exceeds both passband and stopband requirements. However, as found in lines 02–03, differences in how the two transition bands map cause differences in the various buffer zones. In the present case, for example, the buffer near $\omega_{s_1} = 5$ rad/s is larger than the others. Particularly for bandpass and bandstop filters, more balanced buffer zones are difficult, although not impossible (see Ex. 2.15), to achieve.

For comparison purposes, Fig. 2.34 also shows the $K = 4$ and $K = 8$ bandpass responses that result using $K = 2$ and $K = 4$ lowpass prototypes. By reducing the prototype order below the minimum required, the resulting filter does not meet specifications. By increasing the prototype order beyond the minimum, the filter still meets specification but at an undesirable cost of added complexity.

Example 2.12 ▶

▷ **Drill 2.9 (Determining Butterworth Lowpass Filter Order)**

Determine the order K of a lowpass Butterworth filter to meet the specifications $\omega_p \geq 100$ rad/s, $\delta_p \leq 0.05$, $\omega_s \leq 200$ rad/s, and $\delta_s \leq 0.01$.

△

▷ **Drill 2.10 (Determining Butterworth Bandpass Filter Order)**

Determine the order K of a bandpass Butterworth filter to meet the specifications $\omega_{p1} \leq 100$ rad/s, $\omega_{p2} \geq 200$ rad/s, $\alpha_p \leq 0.5$ dB, $\omega_{s1} \geq 50$ rad/s, $\omega_{s2} \leq 250$ rad/s, and $\alpha_s \geq 40$ dB.

△

2.7.2 Chebyshev Filters

In this section, we cover Chebyshev filters, also called *Chebyshev type 1 filters*, in sufficient detail to allow competent design. Deeper treatments of this fascinating family of filters are available, such as those provided in [5] and [6].

The magnitude response of a Chebyshev lowpass filter is given by

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \epsilon^2 C_K^2 \left(\frac{\omega}{\omega_p}\right)}}, \quad (2.41)$$

where $C_K(x)$, the K th-order *Chebyshev polynomial*, is given by

$$C_K(x) = \cos[K \cos^{-1}(x)] \quad \text{or} \quad C_K(x) = \cosh[K \cosh^{-1}(x)]. \quad (2.42)$$

Traditionally, the cosine form of Eq. (2.42) is used for $|x| \leq 1$, and the hyperbolic cosine form is used for $|x| > 1$. With modern computing devices that support complex arithmetic, however, either form can be used without restriction. Since $C_K(x)$ oscillates back and forth between -1 and 1 over $-1 \leq x \leq 1$, Chebyshev polynomials are known as *equiripple functions*.

While Eq. (2.42) offers a compact form suitable for computation, the *polynomial* nature of $C_K(x)$, needed to produce a rational (realizable) system function $H(s)$, is not at all clear. As it turns out (see Prob. 2.7-8), $C_K(x)$ satisfies the recursion relation

$$C_K(x) = 2x C_{K-1}(x) - C_{K-2}(x) \quad \text{for } K > 1. \quad (2.43)$$

Noting from Eq. (2.42) that $C_0(x) = 1$ and $C_1(x) = x$, which are zeroth-order and first-order polynomials, respectively, it follows from Eq. (2.43) that $C_K(x)$ must be a K th-order polynomial. For example, $C_2(x) = 2xC_1(x) - C_0(x) = 2x^2 - 1$, $C_3(x) = 2xC_2(x) - C_1(x) = 4x^3 - 3x$, and so on.

From Eq. (2.42), it follows that $C_K(0)$ is either 1 (even K) or 0 (odd K) and that $C_K(1) = 1$ for all K . Use of these results in Eq. (2.41) leads to the conclusions that

$$|H(j0)| = \begin{cases} 1 & K \text{ odd} \\ \frac{1}{\sqrt{1+\epsilon^2}} & K \text{ even} \end{cases} \quad \text{and} \quad |H(j\omega_p)| = \frac{1}{\sqrt{1+\epsilon^2}}. \quad (2.44)$$

Using Eq. (2.41), the Chebyshev lowpass magnitude response is depicted in Fig. 2.35 for $K = 6$ and $K = 7$. Notice the different dc gain between the even- and odd-order cases. The Chebyshev magnitude response has ripples in the passband ($0 \leq \omega \leq \omega_p$) and is smooth (monotonic) outside the passband ($\omega > \omega_p$). As consequence of the equiripple nature of $C_K(\omega/\omega_p)$, the K maxima and minima over the passband are of equal height, oscillating between a maximum of unity and a minimum of $1/\sqrt{1+\epsilon^2}$.

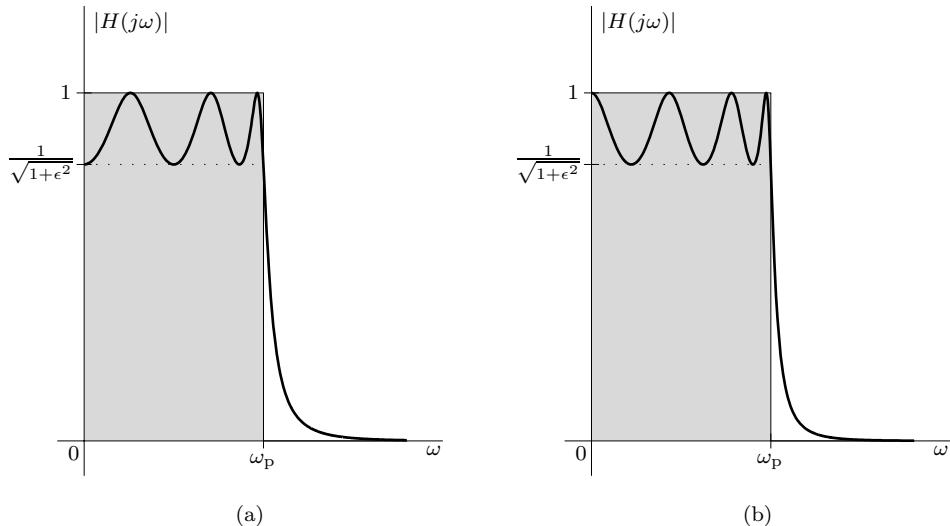


Figure 2.35: Chebyshev magnitude responses: (a) $K = 6$ and (b) $K = 7$.

The ratio of the maximum gain to the minimum gain in the passband is called the *peak-to-peak ripple*. For filters that have maximum passband gain of unity, such as the Chebyshev response of Eq. (2.41), this ratio (in dB) is identical to the maximum passband attenuation α_p and is given as

$$\alpha_p = 20 \log_{10} \sqrt{1 + \epsilon^2} = 10 \log_{10}(1 + \epsilon^2).$$

Consequently, the Chebyshev parameter ϵ in Eq. (2.41) controls the height of the filter's passband ripple and is given by

$$\epsilon^2 = 10^{\alpha_p/10} - 1. \quad (2.45)$$

Larger ϵ leads to greater ripple and vice versa.

Comparing Chebyshev and Butterworth Responses

The Chebyshev filter has a sharper cutoff (smaller transition band) than the same-order Butterworth filter, but this is achieved at the expense of inferior passband behavior (rippling instead of maximally flat).[†] Alternatively, as Ex. 2.13 demonstrates, Chebyshev filters typically require lower order K than Butterworth filters to meet a given set of specifications.

A subtle but important difference also exists in how Chebyshev and Butterworth filters are normalized: a normalized Butterworth filter has a half-power frequency of unity ($\omega_c = 1$), while a normalized Chebyshev filter has a unity passband cutoff frequency ($\omega_p = 1$). Since ω_c is rarely equal to ω_p , it follows that a transformation that is applied to a normalized Butterworth filter is not the same as that needed for a normalized Chebyshev filter.

Determination of Chebyshev Filter Order, Pole Locations, and Transfer Function

To determine Chebyshev filter order, we proceed in much the same way as we did for the Butterworth case. Expressing the minimum stopband attenuation α_s , in dB, in terms of the Chebyshev response

[†]We can show (see [7]) that, at higher frequencies in the stopband, the Chebyshev filter gain is smaller than the comparable Butterworth filter gain by about $6(K - 1)$ dB. This fact, however, is of little consequence as long as the filter satisfies stopband requirements.

given by Eq. (2.41) yields

$$\alpha_s = -20 \log_{10} |H(j\omega_s)| = 10 \log_{10} \left[1 + \epsilon^2 C_K^2 \left(\frac{\omega_s}{\omega_p} \right) \right].$$

Substituting Eqs. (2.42) and (2.45) into this result and solving for integer K yield

$$K = \left\lceil \frac{\cosh^{-1} \sqrt{(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)}}{\cosh^{-1}(\omega_s/\omega_p)} \right\rceil. \quad (2.46)$$

Comparing Eq. (2.46) with Eq. (2.38), we see that computing Chebyshev filter order is identical to the Butterworth case except that logarithms are replaced with inverse hyperbolic cosines. As we shall see next in determining Chebyshev pole locations, there are significant connections between Chebyshev and Butterworth filters.

As done in the Butterworth case, the Chebyshev filter poles, and thus its transfer function, are found by squaring the magnitude response of Eq. (2.41), substituting $\omega = s/j$ to yield $H(s)H(-s)$, and then selecting the left half-plane denominator roots. The procedure is relatively straightforward but tedious to complete analytically.[†] The end result, however, is elegant. Similar to how Butterworth poles lie on a semicircle, the poles of a Chebyshev lowpass filter lie on a semi-ellipse [7,5]. Mathematically, the Chebyshev poles are

$$p_k = -\omega_p \sinh \left[\frac{1}{K} \sinh^{-1} \left(\frac{1}{\epsilon} \right) \right] \sin \left[\frac{\pi(2k-1)}{2K} \right] + j\omega_p \cosh \left[\frac{1}{K} \sinh^{-1} \left(\frac{1}{\epsilon} \right) \right] \cos \left[\frac{\pi(2k-1)}{2K} \right] \quad k = 1, 2, \dots, K. \quad (2.47)$$

In scaling our poles by the passband frequency ω_p , our Chebyshev design will exactly meet passband requirements and, due to rounding up the order K , will exceed stopband requirements. As shown in upcoming Ex. 2.13, however, it is a simple matter to scale ω_p to provide, if desired, both passband and stopband buffer zones.

Careful comparison of Eq. (2.47) with Eq. (2.35) confirms a close relationship between pole locations on a Chebyshev ellipse and those on Butterworth circles: the real part of the Chebyshev poles coincide with those of an equivalent-order Butterworth filter with radius $\omega_p \sinh [\frac{1}{K} \sinh^{-1} (\frac{1}{\epsilon})]$, and the imaginary part of the Chebyshev poles are equal to those of a Butterworth filter with radius $\omega_p \cosh [\frac{1}{K} \sinh^{-1} (\frac{1}{\epsilon})]$. Figure 2.36 displays the poles of a third-order Chebyshev filter, which lie on a semi-ellipse, and emphasizes the relationship of these poles to the appropriately scaled Butterworth poles, which lie on semicircles.

The transfer function $H(s)$ of a K th-order lowpass Chebyshev filter is

$$H(s) = |H(j0)| \frac{(-1)^K p_1 p_2 \cdots p_K}{(s - p_1)(s - p_2) \cdots (s - p_K)} = |H(j0)| \prod_{k=1}^K \left(\frac{-p_k}{s - p_k} \right). \quad (2.48)$$

Here, $|H(j0)|$ ensures the proper dc gain and is determined according to Eq. (2.44).

As with Butterworth filters, the design procedure for Chebyshev filters can be achieved through the use of ready-made tables. Table 2.4, for example, provides the coefficients of the denominator polynomial for normalized ($\omega_p = 1$) Chebyshev filters with various passband ripple levels. Since separate tables are needed for each value of α_p and modern computer technologies easily compute Chebyshev filter parameters, the popularity of such tables has waned significantly in recent years.

[†]It is not recommended to perform this procedure numerically. Even with modern calculators or computers, it is difficult and sometimes impossible to determine with sufficient accuracy the roots of the polynomial $1 + \epsilon^2 C_K^2(s/j\omega_p)$, particularly for large K .

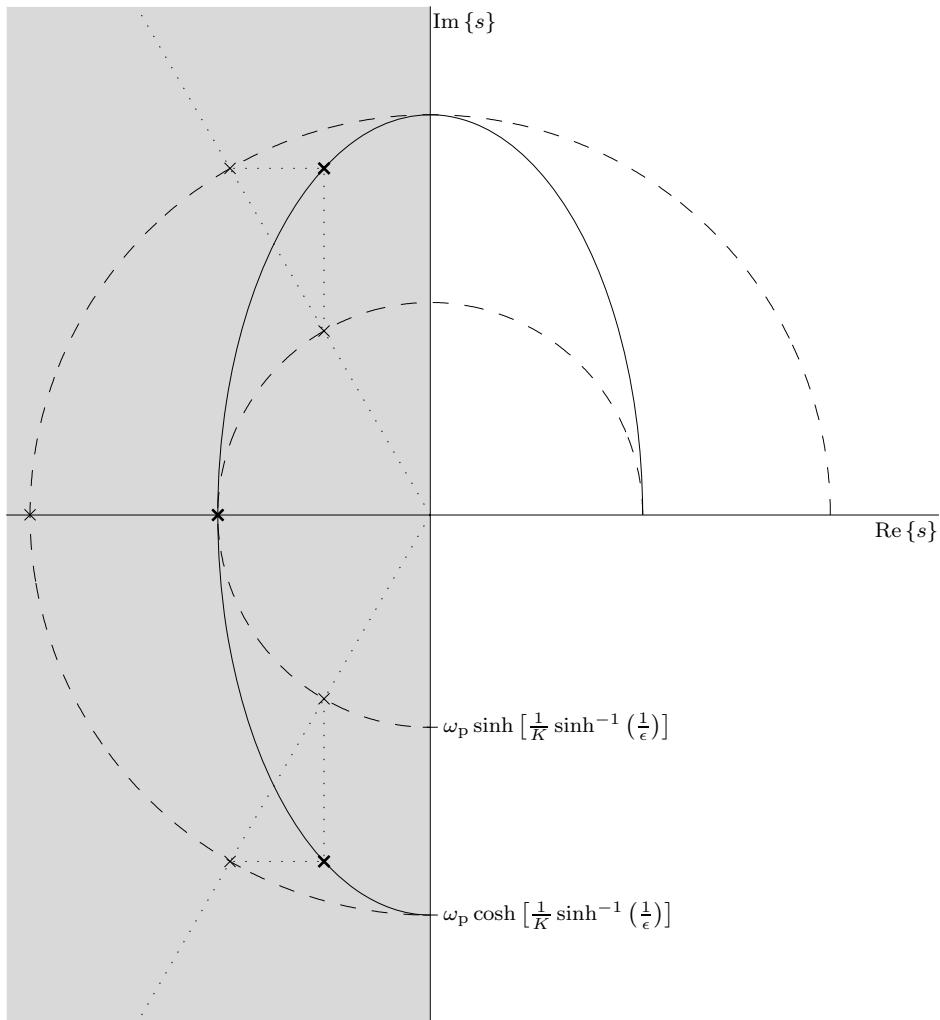


Figure 2.36: Third-order Chebyshev filter pole locations and relationship to Butterworth poles.

▷ **Drill 2.11 (Comparing Chebyshev and Butterworth Filter Orders)**

Using $\alpha_p \leq 0.5$ dB and $\alpha_s \geq 20$ dB, determine and compare the order K for both Chebyshev and Butterworth lowpass filters for $\omega_s/\omega_p = 1.01, 1.1, 2, 5$, and 10 .

△

▷ **Drill 2.12 (Effect of Passband Ripple on Chebyshev Pole Locations)**

Using Butterworth poles as in Fig. 2.36, determine and plot the poles for normalized ($\omega_p = 1$) $K = 3$ Chebyshev filters with (a) $\alpha_p = 3$ dB, (b) $\alpha_p = 0.3$ dB, and (c) $\alpha_p = 0.03$ dB. What is the general effect of passband ripple on a Chebyshev filter's pole locations?

△

K	a_6	a_5	a_4	a_3	a_2	a_1	a_0
1	0.1 dB of ripple						6.552203
2	($\alpha_p = 0.1$)					2.372356	3.314037
3				1.938811	2.629495	1.638051	
4			1.803773	2.626798	2.025501	0.828509	
5		1.743963	2.770704	2.396959	1.435558	0.409513	
6		1.712166	2.965756	2.779050	2.047841	0.901760	0.207127
7	1.693224	3.183504	3.169246	2.705144	1.482934	0.561786	0.102378
1	0.5 dB of ripple						2.862775
2	($\alpha_p = 0.5$)					1.425625	1.516203
3				1.252913	1.534895	0.715694	
4			1.197386	1.716866	1.025455	0.379051	
5		1.172491	1.937367	1.309575	0.752518	0.178923	
6		1.159176	2.171845	1.589764	1.171861	0.432367	0.094763
7	1.151218	2.412651	1.869408	1.647903	0.755651	0.282072	0.044731
1	1 dB of ripple						1.965227
2	($\alpha_p = 1$)					1.097734	1.102510
3				0.988341	1.238409	0.491307	
4			0.952811	1.453925	0.742619	0.275628	
5		0.936820	1.688816	0.974396	0.580534	0.122827	
6		0.928251	1.930825	1.202140	0.939346	0.307081	0.068907
7	0.923123	2.176078	1.428794	1.357545	0.548620	0.213671	0.030707
1	2 dB of ripple						1.307560
2	($\alpha_p = 2$)					0.803816	0.823060
3				0.737822	1.022190	0.326890	
4			0.716215	1.256482	0.516798	0.205765	
5		0.706461	1.499543	0.693477	0.459349	0.081723	
6		0.701226	1.745859	0.867015	0.771462	0.210271	0.051441
7	0.698091	1.993665	1.039546	1.144597	0.382638	0.166126	0.020431
1	3 dB of ripple						1.002377
2	($\alpha_p = 3$)					0.644900	0.707948
3				0.597240	0.928348	0.250594	
4			0.581580	1.169118	0.404768	0.176987	
5		0.574500	1.415025	0.548937	0.407966	0.062649	
6		0.570698	1.662848	0.690610	0.699098	0.163430	0.044247
7	0.568420	1.911551	0.831441	1.051845	0.300017	0.146153	0.015662

Table 2.4: Coefficients of normalized Chebyshev denominator polynomials $s^K + a_{K-1}s^{K-1} + \dots + a_1s + a_0$.

▷ Example 2.13 (Chebyshev Lowpass Filter Design)

Design the lowest-order Chebyshev lowpass filter that meets the specifications $\omega_p \geq 10$ rad/s, $\alpha_p \leq 2$ dB, $\omega_s \leq 30$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

Although the specifications are the same as those in Ex. 2.11, we shall find that the Chebyshev filter requires a lower order than the corresponding Butterworth filter.

Step 1: Determine Order K

Using $\omega_p = 10$ rad/s, $\alpha_p = 2$ dB, $\omega_s = 30$ rad/s, and $\alpha_s = 20$ dB, MATLAB evaluation of Eq. (2.46) yields the necessary order K .

```
01 omegap = 10; alphap = 2; omegas = 30; alphas = 20;
02 K = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/acosh(omegas/omegap))
K = 2
```

Observe that the Chebyshev filter requires lower order ($K = 2$) than the Butterworth filter ($K = 3$). The passband behavior of the Butterworth filter, however, is superior (maximally flat at $\omega = 0$) compared with that of the Chebyshev, which has rippled passband characteristics.

Step 2: Determine Passband Cutoff Frequency ω_p

Although, as is commonly done, we can simply set the passband frequency to the smallest permissible value ($\omega_p = 10$), this will result in exactly meeting passband specifications with no buffer zone and exceeding stopband specifications. Rather, we seek to set ω_p to a value that provides both passband and stopband buffer zones.

The choice $\omega_p = 10$ represents one extreme. The other is found setting $\omega_s = 30$ and solving Eq. (2.46) for ω_p .

```
03 omegap = [omegap,omegas*cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K)]
omegap = 10.0000 11.3349
```

Our compromise value of ω_p is chosen as the average of these two extremes.

```
04 omegap = mean(omegap)
omegap = 10.6674
```

Step 3: Determine Transfer Function $H(s)$

Next, we use Eqs. (2.44), (2.45), (2.47), and (2.48) to determine the desired transfer function coefficients.

```
05 epsilon = sqrt(10^(alphap/10)-1); k = 1:K;
06 H0 = (mod(K,2)==1)+(mod(K,2)==0)/sqrt(1+epsilon^2);
07 pk = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
08 1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
09 B = H0*prod(-pk), A = poly(pk)
B = 74.3963
A = 1.0000 8.5747 93.6594
```

Alternatively, the lowpass-to-lowpass transformation $s \rightarrow s/\omega_p = s/10.6674$ applied to the appropriate coefficients of Table 2.4 yields the same denominator.

```
10 A = [1 0.803816 0.823060].*[1 omegap omegap^2]
A = 1.0000 8.5747 93.6594
```

Thus,

$$H(s) = \frac{74.3963}{s^2 + 8.5747s + 93.6594}.$$

The corresponding magnitude response, plotted using MATLAB, is shown in Fig. 2.37.

```
11 omega = linspace(0,35,1001); H = B./(polyval(A,1j*omega));
12 plot(omega,abs(H));
```

Exactly as in Ex. 2.11, the dB attenuation parameters are converted to facilitate filter verification. For comparison purposes, the figure shows the Chebyshev responses corresponding to the limiting values of ω_p as computed in line 03. As predicted, the $\omega_p = 10$ curve exactly meets passband requirements and exceeds stopband requirements. Similarly, the $\omega_p = 11.3349$ curve exactly meets

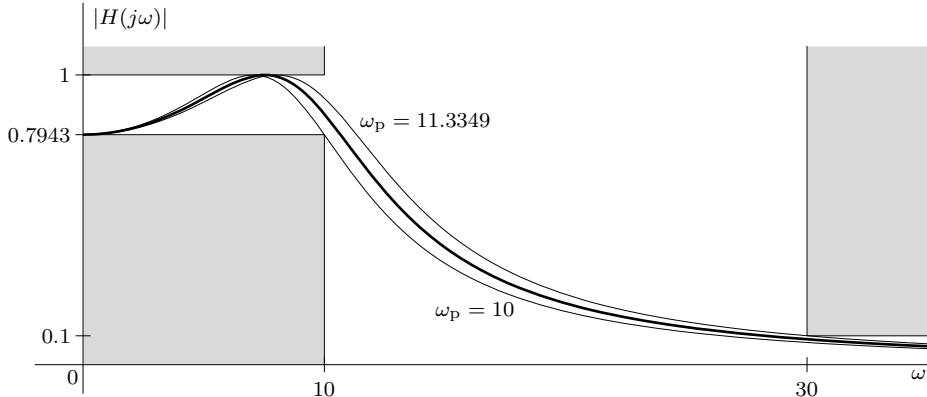


Figure 2.37: Lowpass Chebyshev magnitude response with $K = 2$ and $\omega_p = 10.6674$.

stopband requirements and exceeds passband requirements.

Example 2.13 ◀

▷ **Example 2.14 (Chebyshev Bandstop Filter Design)**

Design the lowest-order Chebyshev bandstop filter that meets the specifications $\omega_{p_1} \geq 5$ rad/s, $\omega_{p_2} \leq 35$ rad/s, $\alpha_p \leq 2$ dB, $\omega_{s_1} \leq 10$ rad/s, $\omega_{s_2} \geq 20$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

As in Ex. 2.12, the solution to this problem is executed in two steps. In the first step, we determine a suitable lowpass prototype. In the second step, a lowpass-to-bandstop transformation is applied to the lowpass prototype to obtain the desired Chebyshev bandstop filter transfer function.

Step 1: Determine the Lowpass Prototype

Before we can determine the order K of the lowpass prototype, we need to determine how the bandstop characteristics relate to the lowpass prototype's stopband-to-passband ratio ω_s/ω_p . We normalize $\omega_s = 1$ and then use Eq. (2.31) to solve for the other frequency, ω_p . Since there are two transition bands in the bandstop filter, two candidate values of ω_p are computed.

```

01 omegap1 = 5; omegap2 = 35; omegas1 = 10; omegas2 = 20; omegas = 1;
02 omegap = abs([omegas*(omegap1*(omegas2-omegas1))/(-omegap1^2+omegas1*omegas2),...
03           omegas*(omegap2*(omegas2-omegas1))/(-omegap2^2+omegas1*omegas2)])
omegap = 0.2857  0.3415

```

As in Ex. 2.12, it is the upper transition band that is most restrictive. Thus, to ensure that both transition bands are satisfied, we substitute $\omega_p = 0.3415$ into Eq. (2.46) and compute the prototype filter order.

```

04 omegap = max(omegap); alphap = 2; alphas = 20;
05 K = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/acosh(omegas/omegap))
K = 2

```

Similar to Ex. 2.13, the prototype stopband frequency is adjusted to provide both passband and stopband buffer zones.

```

06 omegas = mean([omegas, ...
07           omegap*cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K)])
omegas = 0.9519

```

In this case, the adjustment is rather small since $\omega_s = 0.9519$ is close to 1.

Next, we solve Eq. (2.46) for ω_p so that we can use Eqs. (2.44), (2.45), (2.47), and (2.48) to compute the coefficients of the lowpass prototype.

```

08 omegap = omegas/cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K);
09 epsilon = sqrt(10^(alphap/10)-1); k = 1:K;
10 pk = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
11     1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
12 H0 = mod(K,2)+mod(K+1,2)/sqrt(1+epsilon^2); B = H0*prod(-pk), A = poly(pk)
B = 0.0846
A = 1.0000  0.2891  0.1065

```

Thus, the transfer function of the lowpass prototype is

$$H_{lp}(s) = \frac{0.0846}{s^2 + 0.2891s + 0.1065}. \quad (2.49)$$

Step 2: Determine the Bandstop Filter

Using Eq. (2.31), a simple transformation converts our lowpass prototype to the desired bandstop filter,

$$s \rightarrow \omega_0 \frac{s(\omega_{s2} - \omega_{s1})}{s^2 + \omega_{s1}\omega_{s2}} = 1 \frac{(20 - 10)s}{s^2 + 10(20)} = \frac{10s}{s^2 + 200}. \quad (2.50)$$

Notice that ω_0 is set to unity and not $\omega_s = 0.9519$. Had we set $\omega_0 = \omega_s$, we would map our prototype's compromise stopband frequency to the stopband edges, causing the bandstop filter to exactly meet stopband specifications with no buffer zone. This would undo all our efforts from lines 06–07.

As with Ex. 2.12, we develop MATLAB code to automate the task of transformation rather than manually substituting Eq. (2.50) into Eq. (2.49). Since our prototype is an all-pole filter, it is sufficient to consider what happens when our transformation is applied to a single pole p_k ,

$$\frac{1}{s - p_k} \rightarrow \frac{1}{\frac{10s}{s^2 + 200} - p_k} = \frac{s^2 + 200}{-p_k \left(s^2 - \frac{10}{p_k}s + 200 \right)}.$$

Thus, our single pole becomes a pair of poles along with the addition of a pair of conjugate zeros at $s = \pm j\sqrt{200}$.

Designating $as^2 + bs + c = s^2 - \frac{\omega_{s2} - \omega_{s1}}{p_k}s + \omega_{s1}\omega_{s2} = s^2 - \frac{10}{p_k}s + 200$, the quadratic formula $(-b \pm \sqrt{b^2 - 4ac})/(2a)$ allows us to compute the two poles that result from the transformation of each pole p_k . The roots for both numerator and denominator are then expanded into polynomial form.

```

13 a = 1; b = -(omegas2-omegas1)./pk; c = omegas1*omegas2;
14 B = B/prod(-pk)*poly([1j*sqrt(c)*ones(K,1);-1j*sqrt(c)*ones(K,1)])
B = 0.7943  0.0000  317.7313  0.0000  31773.1293
15 A = poly([-b+sqrt(b.^2-4*a.*c))./(2*a),(-b-sqrt(b.^2-4*a.*c))./(2*a)])
A = 1.0000  27.1550  1339.3288  5431.0003  40000.0000

```

Using these results, the transfer function of our order-4 bandstop filter is thus

$$H(s) = \frac{0.7943s^4 + 317.7313s^2 + 31773.1293}{s^4 + 27.1550s^3 + 1339.3288s^2 + 5431.0003s + 40000}.$$

The filter's magnitude response, plotted using MATLAB, is shown in Fig. 2.38.

```

16 omega = linspace(0,100,1001); H = polyval(B,1j*omega)./polyval(A,1j*omega);
17 plot(omega,abs(H));

```

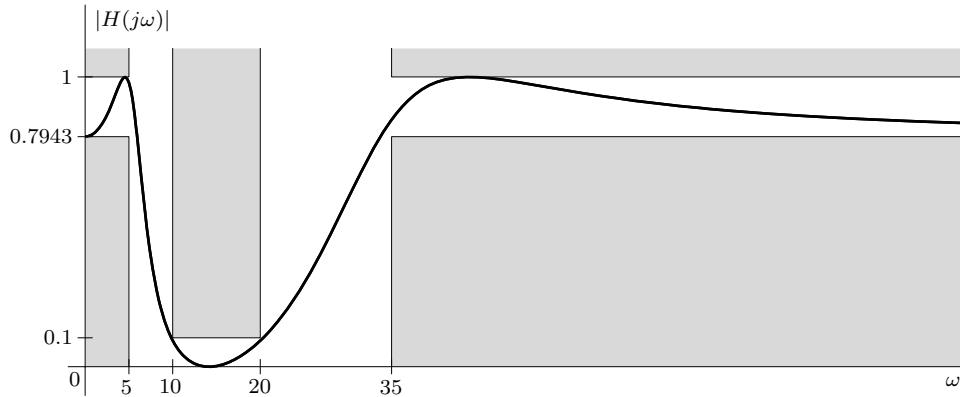


Figure 2.38: Order-4 Chebyshev bandstop filter magnitude response $|H(j\omega)|$.

Although somewhat difficult to see, buffer zones are present, and the filter exceeds both passband and stopband specifications. Despite selecting ω_s midway between extremes, the nonlinear nature of the lowpass-to-bandstop transformation causes the buffer zones to display less balance. Beyond the mathematics, there is an element of art to good filter design.

Example 2.14 ◀

▷ **Example 2.15 (Bandstop Filter Design with Balanced Buffer Zones)**

Using balanced buffer zones in both the upper and lower transition bands, design the lowest-order Chebyshev bandstop filter that meets the specifications $\omega_{p1} \geq 5$ rad/s, $\omega_{p2} \leq 50$ rad/s, $\alpha_p \leq 2$ dB, $\omega_{s1} \leq 10$ rad/s, $\omega_{s2} \geq 20$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

Given the nearly identical specifications, an initial design is readily achieved with simple modification of the code in Ex. 2.14. Changing ω_{p2} from 35 to 50 in line 01 and re-executing lines 01–17 result in the fourth-order bandstop response shown in Fig. 2.39. In this case, it is now the lower transition band that is most restrictive.

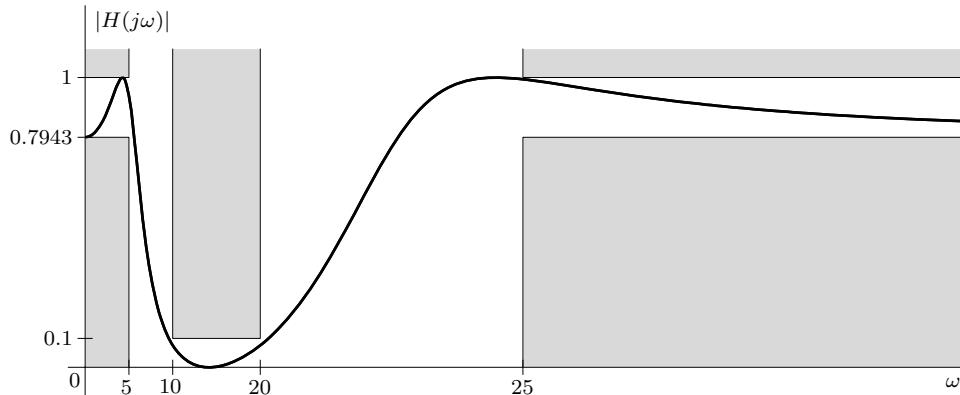


Figure 2.39: Order-4 Chebyshev bandstop filter with unbalanced buffer zones.

Although the filter shown in Fig. 2.39 meets specifications, the buffer zones in the upper transition

band are not balanced. By adjusting the stopband cutoff ω_s of the lowpass prototype (lines 06–07), the code of Ex. 2.14 provides balanced buffer zones in the more restrictive lower transition band. This adjustment is too small to balance the upper transition band. Further decreasing ω_s improves the upper transition band performance but degrades the lower transition region.

To develop one of the many possible solutions to this dilemma, it is worth understanding why the upper transition in Fig. 2.39 is crowded toward the stopband rather than the passband. The reason is that the lowpass-to-bandstop transformation is written in terms of the stopband parameters ω_{s_1} and ω_{s_2} . Thus, the lowpass prototype stopband cutoff ω_s , adjusted to deliver appropriately small buffers for the restrictive upper transition band, is mapped (crowded) toward ω_{s_2} . By adjusting ω_{s_2} in our lowpass-to-bandstop transformation rule, we can better balance the upper transition band's buffer zones.

We know that $\omega_{s_2} = 20$ rad/s causes our response to crowd toward the stopband side. The value ω_{s_2} that crowds our response toward the passband side at $\omega = \omega_{p_2} = 50$ rad/s is determined by solving Eq. (2.31) for ω_{s_2} ,

$$\omega_{s_2} = \frac{\omega^2 \omega_p + \omega \omega_{s_1}}{\omega_{s_1} \omega_p + \omega} \Big|_{\omega=\omega_{p_2}} = \frac{50^2(0.3318) + 500}{10(0.3318) + 50} = 24.9341.$$

To compromise, we set ω_{s_2} midway between the extremes at

$$\omega_{s_2} = (20 + 24.9341)/2 = 22.4670.$$

Changing line 13, shown below, to reflect this change and re-executing the code produce the result shown in Fig. 2.40. This well-balanced response comfortably satisfies all requirements.

```
13 omegas2 = 22.4670; a = 1; b = -(omegas2-omegas1)./pk; c = omegas1*omegas2;
```

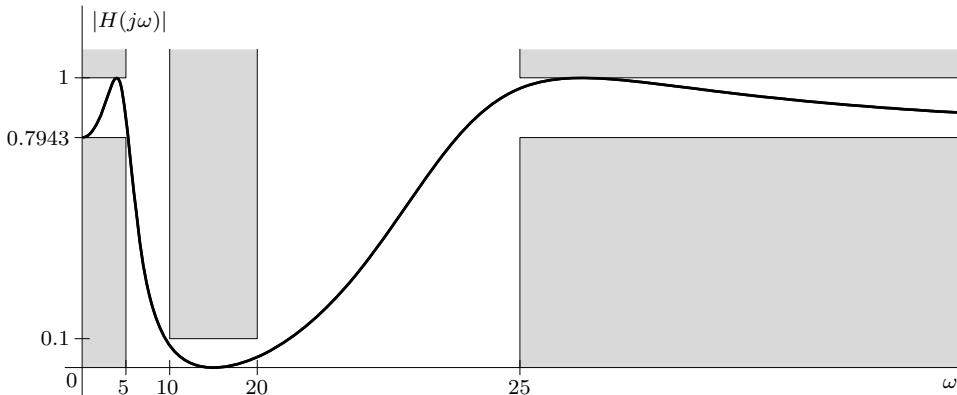


Figure 2.40: Order-8 Chebyshev bandstop filter with balanced buffer zones.

One caution is in order. It is possible to ask more than a particular filter can deliver. While adjusting parameters (ω_{s_2} in this case) may improve one transition region, the other transition region will also be impacted, sometimes to the point that specifications are no longer met. It is crucial, therefore, to always check the final design to verify filter behavior.

Example 2.15 □

▷ Drill 2.13 (Chebyshev Lowpass Filter Design)

Design a sixth-order Chebyshev lowpass filter for the specifications $\omega_p \geq 10$ rad/s, $\delta_p \leq 0.1$, $\omega_s \leq 30$ rad/s, and $\delta_s \leq 0.01$. Plot the corresponding magnitude response to verify whether design requirements are met.

□

2.7.3 Inverse Chebyshev Filters

The passband behavior of a Chebyshev filter exhibits ripples, and the stopband is smooth. Generally, passband behavior is more important, and we would prefer that the passband have a smooth response. However, ripples can be tolerated in the stopband as long as they meet α_s , the minimum allowable stopband attenuation. The *inverse Chebyshev* filter, also called a *Chebyshev type 2 filter*, does both: its magnitude response exhibits a maximally flat passband and an equiripple stopband. While Butterworth and Chebyshev type 1 filters have finite poles and no finite zeros, the inverse Chebyshev has both finite poles and zeros.

The inverse Chebyshev response can be obtained from a Chebyshev response in two steps. To begin, let $|H_c(\omega)|$ be a lowpass Chebyshev magnitude response with passband edge ω_p and ripple parameter ϵ , as shown in Fig. 2.41a. In the first step, we subtract $|H_c(\omega)|^2$ from 1 to obtain the response shown in Fig. 2.41b. Although highpass, the desired characteristics of stopband ripple and smooth passband are present. In the second step, we interchange the stopband and passband with the lowpass-to-highpass transformation of Eq. (2.27), which serves equally well as a highpass-to-lowpass transformation. Noting that frequency ω_p corresponds to our *stopband* frequency ω_s , the substitution $\omega \rightarrow -\omega_p \omega_s / \omega$ yields the inverse Chebyshev response, shown in Fig. 2.41c.

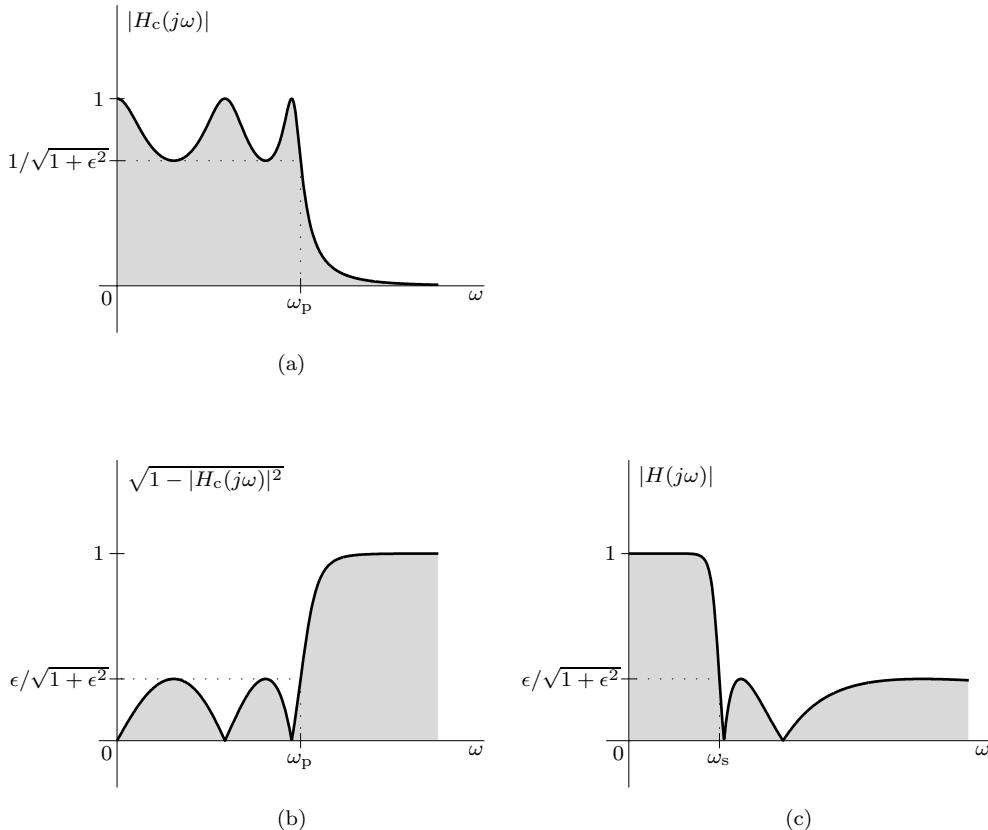


Figure 2.41: Obtaining an inverse Chebyshev response: (a) Chebyshev response $|H_c(j\omega)|$, (b) $\sqrt{1 - H_c^2(j\omega)}$, and (c) inverse Chebyshev response $|H(j\omega)|$.

Mathematically, the inverse Chebyshev magnitude response is given as

$$|H(j\omega)| = \sqrt{1 - |H_c(-j\omega_p \omega_s / \omega)|^2} = \sqrt{\frac{\epsilon^2 C_K^2(\omega_s / \omega)}{1 + \epsilon^2 C_K^2(\omega_s / \omega)}}, \quad (2.51)$$

where $C_K(\cdot)$ are the K th-order Chebyshev polynomials in Eq. (2.42). In Eq. (2.51), the negative sign in the lowpass-to-highpass transformation rule is eliminated since $|H(j\omega)|$ is an even function of ω .

Much like the Chebyshev response, the inverse Chebyshev response requires three parameters: order K , ripple parameter ϵ , and stopband edge ω_s . Following the same procedure as in the Chebyshev case, the required order for an inverse Chebyshev filter is

$$K = \left\lceil \frac{\cosh^{-1} \sqrt{(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)}}{\cosh^{-1}(\omega_s / \omega_p)} \right\rceil, \quad (2.52)$$

which is exactly the same as for the Chebyshev type 1 case. Relating the inverse Chebyshev stopband gain $\epsilon / \sqrt{1 + \epsilon^2}$ to α_s and solving for ϵ^2 yield

$$\epsilon^2 = \frac{1}{10^{\alpha_s/10} - 1}. \quad (2.53)$$

Comparing Eqs. (2.53) and (2.45) highlights that the ripple parameter ϵ is computed differently for Chebyshev type 1 and type 2 filters. Another difference is that the inverse Chebyshev response is expressed in terms of stopband edge ω_s rather than passband edge ω_p , as in the Chebyshev case. Consequently, a normalized inverse Chebyshev filter has $\omega_s = 1$.

The transfer function $H(s)$ of a K th-order lowpass inverse Chebyshev filter is

$$H(s) = \frac{\prod_{k=1}^K (p_k / z_k) \prod_{k=1}^K (s - z_k)}{\prod_{k=1}^K (s - p_k)}, \quad (2.54)$$

where z_k and p_k designate the filter zeros and poles, respectively.

Comparing the denominators of Eqs. (2.51) and (2.41) and using $\omega = s/j$, we see that the poles of the inverse Chebyshev filter are reciprocally related to the poles of a Chebyshev filter. More precisely, the inverse Chebyshev poles p_k are determined according to

$$p_k = \frac{\omega_p \omega_s}{p'_k}, \quad (2.55)$$

where p'_k are the poles, as given in Eq. (2.47), of a Chebyshev filter with passband edge ω_p and ripple parameter ϵ^2 set using Eq. (2.53), as required for the inverse Chebyshev response. Solving where the numerator of Eq. (2.51) equals zero, the inverse Chebyshev zeros z_k are computed as (see Prob. 2.7-16)

$$z_k = j\omega_s \sec \left(\frac{\pi(2k-1)}{2K} \right) \quad k = 1, 2, \dots, K. \quad (2.56)$$

Figure 2.42 shows the pole-zero plots of a sixth-order normalized ($\omega_s = 1$) inverse Chebyshev filter for three choices of α_s . Although the exact pole and zero locations vary for different parameters K , α_s , and ω_s , the plots represent general inverse Chebyshev character.

Inverse Chebyshev filters are preferable to the Chebyshev filters in many ways. In particular, the passband behavior, especially for small ω , is better for the inverse Chebyshev than for the Chebyshev or even for the Butterworth filter of the same order. The inverse Chebyshev also has the smallest transition band of the three filters. Moreover, the phase function (or time-delay) characteristic of the inverse Chebyshev filter is better than that of the Chebyshev filter [5]. Although Chebyshev

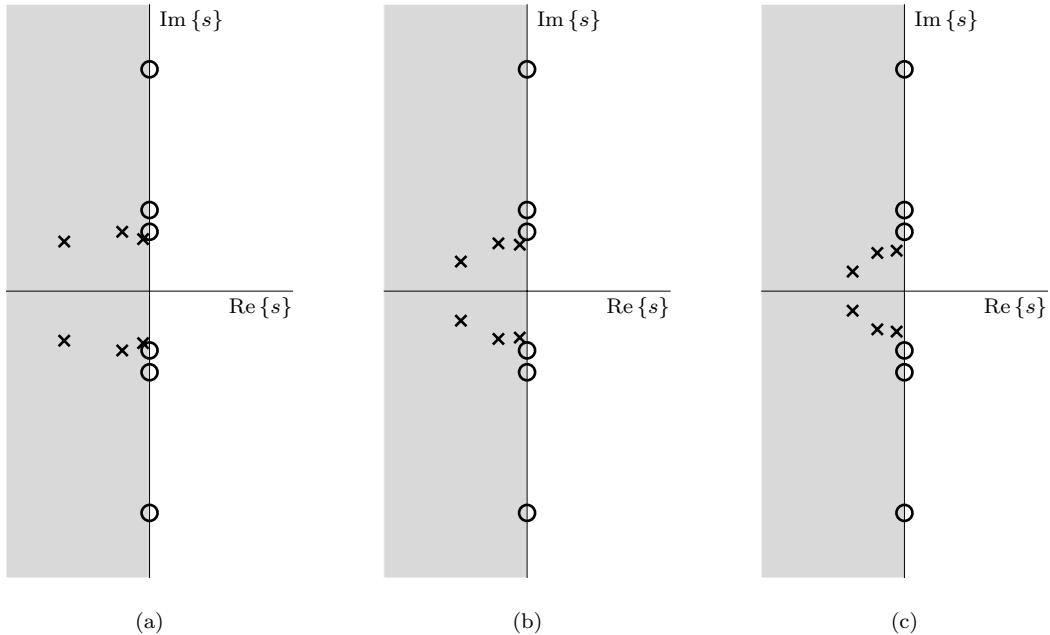


Figure 2.42: Pole-zero plots of a normalized $K = 6$ inverse Chebyshev lowpass filter: (a) $\alpha_s = 20$, (b) $\alpha_s = 30$, and (c) $\alpha_s = 40$.

and inverse Chebyshev filters require the same order K to meet a given set of specifications, the inverse Chebyshev realization requires more elements and thus is less economical than the Chebyshev filter. Compared with a comparable performance Butterworth filter, however, the inverse Chebyshev typically requires fewer elements.

▷ **Example 2.16 (Inverse Chebyshev Lowpass Filter Design)**

Design the lowest-order inverse Chebyshev lowpass filter that meets the specifications $\omega_p \geq 10$ rad/s, $\alpha_p \leq 2$ dB, $\omega_s \leq 20$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

Step 1: Determine Order K

Using $\omega_p = 10$ rad/s, $\alpha_p = 2$ dB, $\omega_s = 20$ rad/s, and $\alpha_s = 20$ dB, MATLAB evaluation of Eq. (2.52) yields the necessary order K .

```
01 omegap = 10; alphap = 2; omegas = 20; alphas = 20;
02 K = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/acosh(omegas/omegap))
K = 3
```

Step 2: Determine Stopband Cutoff Frequency ω_s

Although, as is commonly done, we can simply set the stopband frequency to the largest permissible value ($\omega_s = 20$), this will result in exactly meeting stopband specifications with no buffer zone and exceeding passband specifications. Rather, we seek to set ω_s to a value that provides both passband and stopband buffer zones.

The choice $\omega_s = 20$ represents one extreme. The other is found setting $\omega_p = 10$ and solving Eq. (2.52) for ω_s .

```
03 omegas = [omegap*cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1))/K),omegas]
omegas = 16.4972 20.0000
```

Our compromise value of ω_s is chosen as the average of these two extremes.

```
04 omegas = mean(omegas)
omegas = 18.2486
```

Step 3: Determine Transfer Function $H(s)$

To begin, we use Eqs. (2.53) and (2.47) to determine the pole locations of an appropriate Chebyshev filter.

```
05 epsilon = 1/sqrt(10^(alphas/10)-1); k = 1:K;
06 pk = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
07 1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
```

Next, we use Eqs. (2.55) and (2.56) to determine the pole and zero locations of our filter.

```
08 pk = omegap*omegas./pk; zk = 1j*omegas.*sec(pi*(2*k-1)/(2*K));
```

Expanding these polynomials and using Eq. (2.48) to determine the numerator's scaling constant, we obtain the desired transfer function coefficients.

```
09 B = prod(pk./zk)*poly(zk), A = poly(pk)
B = -0.0000 5.5022 -0.0000 2443.0336
A = 1.0000 25.6463 313.7283 2443.0336
```

Thus,

$$H(s) = \frac{5.5022s^2 + 2443.0336}{s^3 + 25.6463s^2 + 313.7283s + 2443.0336}.$$

The corresponding magnitude response, plotted using MATLAB, is shown in Fig. 2.43.

```
10 omega = linspace(0,70,1001); H = (polyval(B,1j*omega))./(polyval(A,1j*omega));
11 plot(omega,abs(H));
```

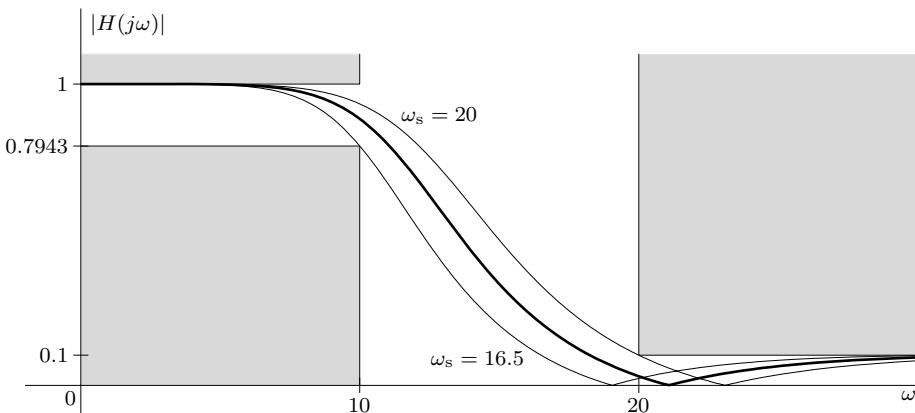


Figure 2.43: Lowpass inverse Chebyshev magnitude response with $K = 3$ and $\omega_s = 18.2486$.

For comparison purposes, Fig. 2.43 also shows the inverse Chebyshev responses corresponding to the limiting values of ω_s as computed in line 03. As predicted, the $\omega_s = 20$ curve exactly meets stopband requirements and exceeds passband requirements. Similarly, the $\omega_s = 16.4972$ curve exactly meets passband requirements and exceeds stopband requirements.

Example 2.16 ◀

▷ **Example 2.17 (Inverse Chebyshev Highpass Filter Design)**

Design a sixth-order inverse Chebyshev highpass filter with $\omega_p = 10 \text{ rad/s}$, $\alpha_p \leq 2 \text{ dB}$, and $\alpha_s \geq 40 \text{ dB}$. Determine the filter's stopband frequency ω_s , and plot the magnitude response to verify that design requirements are met.

In the first step, we determine a suitable lowpass prototype. In the second step, a lowpass-to-highpass transformation is applied to the lowpass prototype to obtain the desired inverse Chebyshev highpass filter transfer function.

Step 1: Determine the Lowpass Prototype

In this design, the filter order is specified. To simplify our later lowpass-to-highpass transformation, we normalize our prototype's passband frequency $\omega_p = 1$ and solve for the corresponding stopband frequency ω_s .

```
01 alphap = 2; alphas = 40; K = 6; omegap = 1;
02 omegas = omegap*cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K)
omegas = 1.4621
```

Next, we use Eqs. (2.53) and (2.47) to determine the pole locations of an appropriate Chebyshev filter. Then, we use Eqs. (2.55) and (2.56) to determine the pole and zero locations of our inverse Chebyshev lowpass prototype.

```
03 epsilon = 1/sqrt(10^(alphas/10)-1); k = 1:K;
04 pk = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
05 1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
06 pk = omegap*omegas./pk; zk = 1j*omegas.*sec(pi*(2*k-1)/(2*K));
07 B = prod(pk./zk)*poly(zk), A = poly(pk)
B = 0.0100 -0.0000 0.3848 -0.0000 2.1936 -0.0000 3.1263
A = 1.0000 4.4107 9.7309 13.6964 13.1611 8.2811 3.1263
```

Thus, the transfer function of our inverse Chebyshev lowpass prototype is given as

$$H_{lp}(s) = \frac{0.0100s^6 + 0.3848s^4 + 2.1936s^2 + 3.1263}{s^6 + 4.4107s^5 + 9.7309s^4 + 13.6964s^3 + 13.1611s^2 + 8.2811s + 3.1263}.$$

Step 2: Determine the Highpass Filter

To convert our lowpass prototype to the required highpass response, we need to map the prototype's normalized passband frequency $\omega_p = 1 \text{ rad/s}$ to the desired passband frequency of 10 rad/s . The lowpass-to-highpass transformation of Eq. (2.27) is thus $s \rightarrow \frac{1(10)}{s} = \frac{10}{s}$. Applying this transformation to Eq. (2.54) results in

$$H(s) = H_{lp}(10/s) = \frac{\prod_{k=1}^K (p_k/z_k) \prod_{k=1}^K (10/s - z_k)}{\prod_{k=1}^K (10/s - p_k)} = \frac{\prod_{k=1}^K (s - 10/z_k)}{\prod_{k=1}^K (s - 10/p_k)}.$$

In other words, the zeros and poles of the lowpass prototype reciprocate and scale according to $z_k \rightarrow 10/z_k$ and $p_k \rightarrow 10/p_k$. The resulting filter coefficients as well as the final stopband frequency ω_s are computed with MATLAB.

```
08 pk = 10./pk; zk = 10./zk; B = poly(zk), A = poly(pk), omegas = 10/omegas
B = 1.00 0.00 70.17 0.00 1230.83 0.00 3198.64
A = 1.00 26.49 420.98 4380.98 31125.62 141081.67 319863.65
omegas = 6.8394
```

The filter's magnitude response, plotted in dB using MATLAB, is shown in Fig. 2.44.

```
09 omega = linspace(0,20,1001); H = (polyval(B,1j*omega))./(polyval(A,1j*omega));
10 plot(omega,20*log10(abs(H)));
```

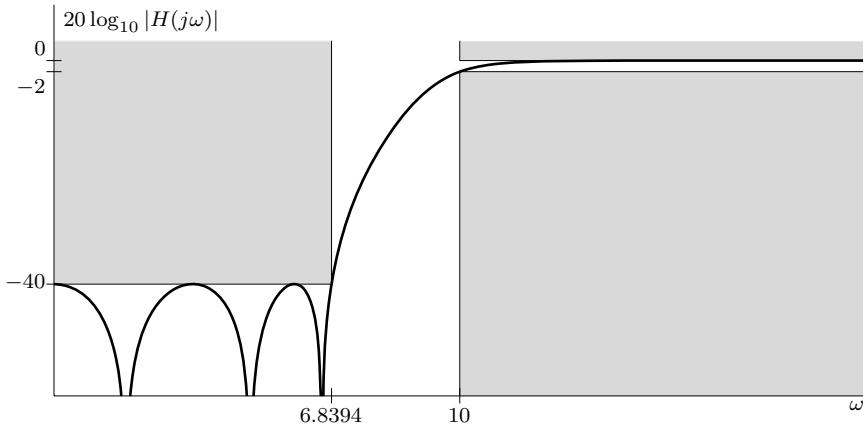


Figure 2.44: Highpass inverse Chebyshev magnitude response with $K = 6$.

Since the stopband attenuation is a significant 1/100 (40 dB), a dB plot is preferred to clearly view the stopband characteristics.

Example 2.17 ◀

2.7.4 Elliptic Filters

As we saw with inverse Chebyshev filters, placing a zero on the imaginary axis at $s = j\omega$ results in $|H(j\omega)| = 0$ (infinite attenuation). We can realize a sharper cutoff characteristic by placing a zero, or zeros, near $\omega = \omega_s$. While Butterworth and Chebyshev filters do not make use of such zeros in $H(s)$, both inverse Chebyshev and elliptic filters do. This is part of the reason for their superior response characteristics. In compensation, because of zeros in the numerator of $H(s)$, inverse Chebyshev and elliptic filter responses decay for $\omega > \omega_s$ at a slower rate than equivalent Butterworth or Chebyshev type 1 filters. For odd orders, inverse Chebyshev and elliptic filters decay at -20 dB/decade at very high frequencies; the responses do not decay to zero at all in the even-order cases. Butterworth and Chebyshev type 1 filters of order K , on the other hand, have stopband decay rates of $-20K$ dB/decade. The presence of zeros in the numerator of $H(s)$ in an elliptic filter causes the response to decay at a slower rate at very high frequencies since the effective rate of decay of a filter response at high frequencies is $-20(K - L)$ dB/decade, where $(K - L)$ is the difference between the number of poles K and number of zeros L . Usually, however, the magnitude response decay rate is unimportant as long as we meet our stopband attenuation specification of α_s .

Chebyshev filters have a smaller transition bands compared with that of a Butterworth filter because Chebyshev filters allow rippling in either the passband or stopband. If we allow ripple in both the passband and the stopband, we can achieve a further reduction in the transition band. Such is the case with elliptic, also called *Cauer*, filters. For a given transition band, an elliptic filter provides the largest ratio of the passband gain to stopband gain, or for a given ratio of passband to stopband gain, it requires the smallest transition band. The equiripple passband and stopband characteristics of an elliptic filter help it achieve the minimum order possible for given specifications.

The magnitude response of an elliptic filter is given by

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \epsilon^2 R_K^2(\omega/\omega_p, \xi)}}. \quad (2.57)$$

In Eq. (2.57), $R_K(\omega/\omega_p, \xi)$ is a K th-order *elliptic rational function* with discrimination factor ξ .[†] Using the properties of R_K (see [5]), we see that an elliptic filter's equiripple passband has magnitude limits of 1 and $1/\sqrt{1+\epsilon^2}$. The parameter ϵ controls the passband ripple, and the gain at ω_p is $1/\sqrt{1+\epsilon^2}$. Similarly, the filter's equiripple stopband has magnitude limits of $1/\sqrt{1+\epsilon^2\xi^2}$ and 0. Thus, the discrimination factor ξ is directly related to the filter's maximum stopband gain. Like Chebyshev type 1 filters, elliptic filters are normalized in terms of the passband edge ω_p . Using $K = 3$ and $K = 4$, Fig. 2.45 illustrates the equiripple passband and stopband character, as well as the narrow transition band, typical of elliptic filters.

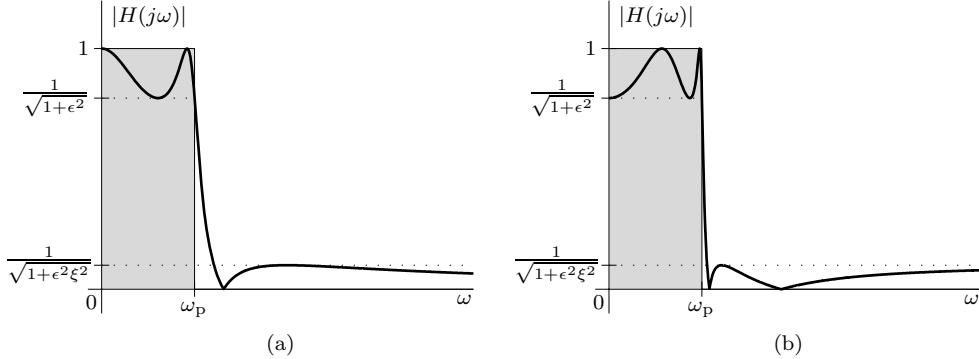


Figure 2.45: Lowpass elliptic magnitude responses: (a) $K = 3$ and (b) $K = 4$.

The transfer function $H(s)$ of a K th-order lowpass elliptic filter is

$$H(s) = |H(j0)| \frac{(s - z_1)(s - z_2) \cdots (s - z_K)}{(s - p_1)(s - p_2) \cdots (s - p_K)} = |H(j0)| \prod_{k=1}^K \left(\frac{s - z_k}{s - p_k} \right), \quad (2.58)$$

where $|H(j0)|$ is 1 or $1/\sqrt{1+\epsilon^2}$ for the odd- and even-order cases, respectively. Using expanded rather than factored form, the transfer function is equivalently expressed as $H(s) = B(s)/A(s)$, where numerator polynomial $B(s)$ and denominator polynomial $A(s)$ are both K th order and follow the form of Eq. (1.51).

Calculation of pole and zero locations of elliptic filters is much more complicated than that for Butterworth or even Chebyshev filters and is not covered here. Fortunately, elliptic filter design, including the determination of system poles and zeros, is greatly simplified by computer programs and extensive ready-made design tables available in the literature. As the next example demonstrates, elliptic filter design is nearly effortless using functions from MATLAB's Signal Processing Toolbox. Conveniently, the Signal Processing Toolbox also includes functions for Butterworth, Chebyshev, and other filter families.

▷ Example 2.18 (Elliptic Bandstop Filter Design)

Using functions from MATLAB's Signal Processing Toolbox, design an elliptic bandstop filter that meets the specifications $\omega_{p1} \geq 5$ rad/s, $\omega_{p2} \leq 25$ rad/s, $\alpha_p \leq 2$ dB, $\omega_{s1} \leq 10$ rad/s, $\omega_{s2} \geq 20$ rad/s, and $\alpha_s \geq 20$ dB. Plot the corresponding magnitude response to verify that design requirements are met.

The specifications for this filter are identical to those for the Chebyshev filter of Ex. 2.15. The

[†]Elliptic rational functions are sometimes referred to as *Chebyshev rational functions*, a name also given to an entirely different class of functions as well. We use the former convention to minimize potential confusion. Additionally, some sources express R_K in terms of a selectivity factor rather than the discrimination factor ξ ; the discrimination factor provides more insight for filter design applications.

MATLAB functions `ellipord` and `ellip` are used to determine the needed order K and filter coefficients, respectively. Although used to construct a bandstop filter in this case, these functions are general and support each of the four basic filter types. Function syntax is detailed in MATLAB's help facilities.

To begin, we determine the necessary order.

```
01 omegap = [5 25]; alphap = 2; omegas = [10 20]; alphas = 20;
02 K = ellipord(omegap, omegas, alphap, alphas, 's')
K = 3
```

In line 02, the designation '`s`' specifies the design as analog rather than digital. Not surprisingly, the computed filter order is smaller for the elliptic filter ($K = 3$) than for the corresponding Chebyshev filter ($K = 4$).

Next, the coefficients of the transfer function are computed, the frequency response is calculated, and the magnitude response plot is generated.

```
03 [B,A] = ellip(K,alphap,alphas,omegap, 'stop','s');
04 omega = linspace(0,100,1001); H = polyval(B,1j*omega)./polyval(A,1j*omega);
05 plot(omega,abs(H));
```

The resulting magnitude response plot of Fig. 2.46 confirms that design specifications are met.

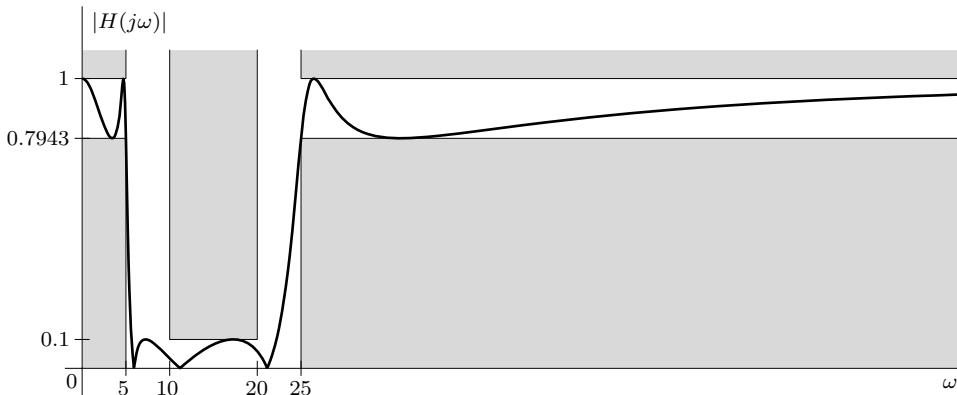


Figure 2.46: Order-3 elliptic bandstop filter magnitude response $|H(j\omega)|$.

Careful inspection of Fig. 2.46 highlights one deficiency of the MATLAB functions: the resulting elliptic filter is scaled to exactly meet passband requirements without a passband buffer zone. This is easily corrected by adjusting the passband frequencies ω_{p_1} and ω_{p_2} that are passed to the `ellip` function.

Again referencing Fig. 2.46, we see that the lower transition band occurs approximately over $5 \leq \omega \leq 5.5$, which is 4.5 rad/s narrower than required. The upper transition band is approximately 2 rad/s narrower than required. Adjusting the passband frequencies by half these surpluses, $\omega_{p_1} = 5 + 4.5/2 = 7.25$ and $\omega_{p_2} = 25 - 2/2 = 24$, provides reasonable buffer zones on both the passband and stopband sides. Replacing `omegap = [5 25]` in line 01 with `omegap = [5+4.5/2, 25-2/2]` and re-executing lines 01–05 produce the improved result shown in Fig. 2.47. Although not perfectly balanced, our eyeball approximations are certainly better than the original MATLAB result.

While balanced passband and stopband buffer zones are desirable, it is also possible to create buffer zones with regard to our attenuation, or ripple, specifications. In many cases, we can strengthen our attenuation specifications with only minor impact on our passband and stopband buffer zones. For example, increasing our stopband attenuation to 25 dB and decreasing our passband ripple to 1.5 dB produce the result shown in Fig. 2.48. This filter is no more complex than the original filter, shown in Fig. 2.46, but it provides comfortable margins for error nearly everywhere.

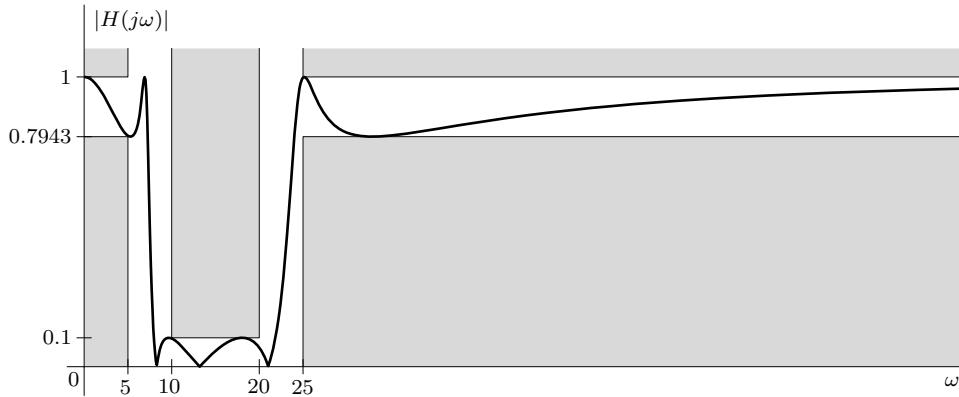


Figure 2.47: Order-3 elliptic bandstop filter with balanced buffer zones.

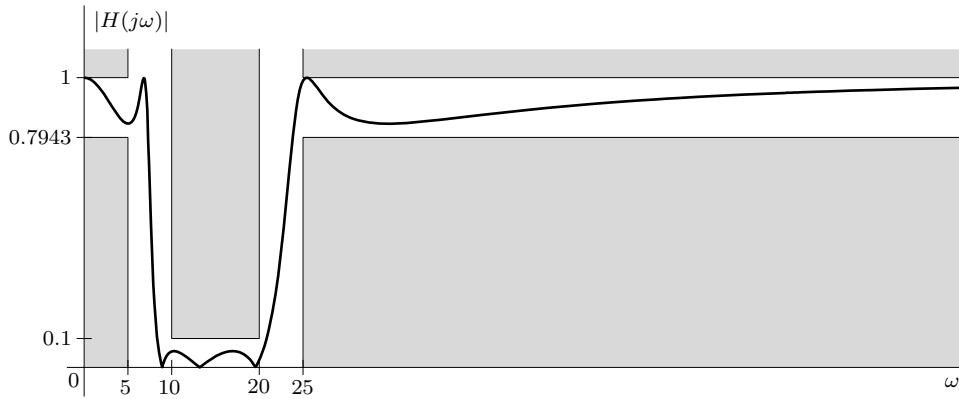


Figure 2.48: Order-3 elliptic bandstop filter with passband, stopband, and ripple buffer zones.

On a final note, it is worth pointing out that buffer zones, while sensible for almost any filter design, are usually most pronounced for low-order filters. Higher-order, stringent designs tend to have less wiggle room than low-order designs. Further, buffer zones are more important for analog filters, which are often low order, than digital filters, which are often higher order. The primary reason is that analog filters suffer from component variations that degrade system performance in unpredictable ways; degradations in digital filter performance, caused, for example, by coefficient truncations, are more controllable and predictable. As a result, the extra effort to create buffer zones is often not taken for digital or high-order filter designs.

Example 2.18 ◀

2.7.5 Bessel-Thomson Filters

Unlike the previous filters, where we approximate the magnitude response without paying attention to the phase response, the Bessel-Thomson filter is designed for maximally flat time delay over a bandwidth of interest. This means that the phase response is designed to be as linear as possible over a given bandwidth. Recall from Eq. (2.15) that time delay $t_g(\omega)$ acquired by a signal during its transmission through an LTI system is $-\frac{d}{d\omega} \angle H(\omega)$. We showed that an ideal delay system (flat or constant delay) with unit gain has a transfer function $H(s) = e^{-st_g}$. Over the bandwidth of interest,

Bessel-Thomson filters approximate this transfer function with a rational function in s so that the resulting frequency response has a maximally flat delay. The resulting transfer function coefficients are closely related to Bessel polynomials, and Thomson was one of the first to use them for this approximation. The K th-order Bessel-Thomson filter transfer function is given by

$$H(s) = \frac{\mathcal{B}_K(0)}{\mathcal{B}_K(s/\omega_1)}, \quad (2.59)$$

where ω_1 , essentially functioning as a lowpass-to-lowpass transformation, simply scales the normalized response by an amount ω_1 . In Eq. (2.59), the K th-order *reverse Bessel polynomials* $\mathcal{B}_K(s)$ can be expressed by the recursion formula

$$\mathcal{B}_k(s) = (2k - 1)\mathcal{B}_{k-1}(s) + s^2\mathcal{B}_{k-2}(s). \quad (2.60)$$

Knowing $\mathcal{B}_0(s) = 1$ and $\mathcal{B}_1(s) = s + 1$, we can determine $\mathcal{B}_K(s)$ for any value of K . For example, $\mathcal{B}_2(s) = 3\mathcal{B}_1(s) + s^2\mathcal{B}_0(s) = 3(s + 1) + s^2(1) = s^2 + 3s + 3$, and so on. For modest values of K , the $K + 1$ coefficients of the reverse Bessel polynomial $\mathcal{B}_K(s) = s^K + b_{K-1}s^{K-1} + \dots + b_1s + b_0$ can also be found directly using the formula

$$b_k = \frac{(2K - k)!}{k!(K - k)!2^{K-k}}. \quad (2.61)$$

Table 2.5 provides a short list of normalized ($\omega_1 = 1$) reverse Bessel polynomials.

K	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1									1
2								3	3
3							6	15	15
4						10	45	105	105
5					15	105	420	945	945
6				21	210	1260	4725	10395	10395
7			28	378	3150	17325	62370	135135	135135
8		36	630	6930	51975	270270	945945	2027025	2027025
9	45	990	13860	135135	945945	4729725	16216200	34459425	34459425

Table 2.5: Coefficients of normalized reverse Bessel polynomials $\mathcal{B}_K(s) = s^K + b_{K-1}s^{K-1} + \dots + b_1s + b_0$.

Bessel-Thomson filters are superior to the filters discussed so far in terms of the phase linearity or flatness of the time delay. However, they do not fair so well in terms of the flatness of the magnitude response. The Bessel-Thomson magnitude response, while lowpass, is inferior even to the Butterworth filter of the same order. For low frequencies, the group delay of a Bessel-Thomson filter is nearly constant and equals $1/\omega_1$. Thus, a normalized ($\omega_1 = 1$) Bessel-Thomson filter delivers a constant group delay of $t_g = 1$ s, at least for low frequencies. As the order K is increased, the bandwidth with constant group delay increases, as does the passband edge. As large order analog filters are difficult to realize, it is also possible to increase the passband and constant-delay bandwidths by increasing ω_1 . Notice, however, that increasing ω_1 produces the corollary effect of reducing the delay by a factor $1/\omega_1$. These characteristics are illustrated in Fig. 2.49. To design Bessel filters, we use plots for magnitude and delay response to determine the filter order K to meet given specifications. The details can be found in the literature [5,7].

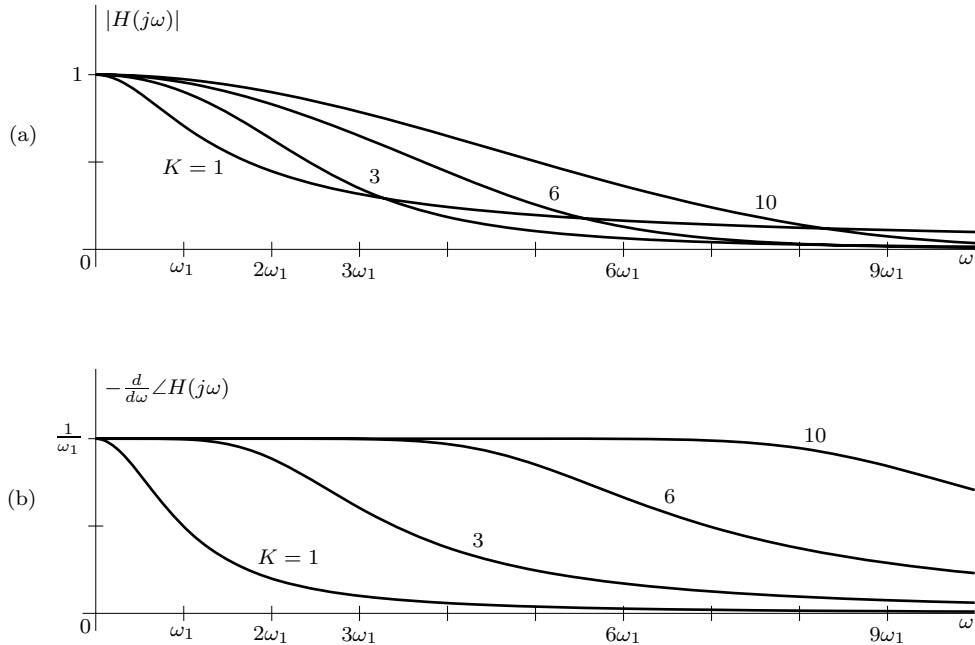


Figure 2.49: Normalized Bessel-Thomson filters: (a) magnitude response and (b) delay response.

▷ Drill 2.14 (Investigating Bessel-Thomson Filter Characteristics)

Design a fifth-order Bessel-Thomson filter that provides low-frequency delay of $100 \mu\text{s}$. Plot the filter's magnitude response, and identify the filter's 3-dB cutoff frequency. Plot the filter's delay response, and determine the bandwidth for which this delay is approximately constant.

△

2.8 Summary

Using the material reviewed in Ch. 1, this chapter introduces and develops the analysis and design of continuous-time systems from a frequency-domain or filtering perspective. The key to a frequency-based perspective is the crucial concept of frequency response $H(\omega)$. In general, frequency response is complex valued. Thus, $H(\omega)$ is typically presented in terms of the magnitude response $|H(\omega)|$ and the phase response $\angle H(\omega)$.

A sinusoidal input to a LTIC system produces an identical frequency sinusoid output, and the system frequency response dictates how the magnitude and phase of the input sinusoid change. A plot of $|H(j\omega)|$ versus ω indicates the magnitude gain for input sinusoids of various frequencies. A plot of $\angle H(j\omega)$ versus ω indicates the phase shift for input sinusoids of various frequencies. Together, the magnitude and phase responses completely capture a system's filtering characteristics.

Frequency response is intimately connected to the locations in the complex plane of the poles and zeros of the system transfer function. Placing a pole (a zero) near a frequency $j\omega_0$ in the complex plane enhances (suppresses) the frequency response at the frequency $\omega = \omega_0$. A proper combination of poles and zeros at suitable locations yields desired and predictable filter characteristics. Thus, pole-zero plots are useful to both the system analysis and design.

Signal transmission through an LTIC system is conveniently represented as the product of the input signal spectrum with the system frequency response, $Y(\omega) = X(\omega)H(\omega)$. For a distortionless

transmission of signals through an LTIC system, the magnitude response must be constant, and the phase response must be linear with ω over the band of interest. The negative of the slope of the phase response (with respect to ω) is the system time or group delay, also called envelope delay in the context of bandpass signals.

Standard filters are classified as lowpass, highpass, bandpass, or bandstop. Ideal filters, with infinitely sharp transition bands and no delay (zero phase response), possess noncausal impulse responses and are thus unrealizable. Proper delay and truncation of the impulse response produce a causal, and thus realizable, filter that closely approximates the desired ideal. Delay and truncation can produce near distortionless transmission over the filter passband. Time-domain truncations, including those caused by windowing operations, result in the impairments of spectral spreading and leakage. Wide windows that taper gradually toward zero produce the smallest spreading and leakage. Dual problems arise for data truncations in the frequency domain.

Practical filters are typically specified in terms of passband and stopband frequencies (ω_p and ω_s) and either passband and stopband ripple (δ_p and δ_s) or maximum passband and minimum stopband attenuation (α_p and α_s). Transformation rules convert a lowpass prototype filter to a highpass, bandpass, bandstop, or another lowpass response. Filter transformations simplify the design process by allowing us to concentrate our design efforts on a single type of filter, typically lowpass.

To facilitate the design of practical and realizable continuous-time analog filters, several filter families are considered: Butterworth, Chebyshev, inverse Chebyshev, elliptic, and Bessel-Thomson. A Butterworth filter has a maximally flat magnitude response over the passband. A Chebyshev filter has equiripple passband magnitude response and monotonic stopband response; the inverse Chebyshev response reverses these roles and provides monotonic passband and equiripple stopband. An elliptic filter provides both passband and stopband ripple. Bessel-Thomson filters, unlike the others, strive to provide a flat delay response (constant group delay) over the passband.

As we shall see in the upcoming chapters, all the topics of this chapter are relevant to the study of digital signal processing. To see how, we need first understand the process of sampling, which is the focus of the next chapter.

References

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
2. Guillemin, E. A., *Theory of Linear Physical Systems*, Wiley, New York, 1963.
3. Mitra, S. K., and Kaiser, J. F., *Handbook for Digital Signal Processing*, Wiley, New York, 1993.
4. Harris, F. J., “On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform,” *Proc. IEEE*, Vol. 66, No. 1, January 1978, pp. 51–83.
5. Van Valkenburg, M. E., *Analog Filter Design*, Holt, Rinehart, and Winston, New York, 1982.
6. Porat, B., *A Course In Digital Signal Processing*, Wiley, New York, 1997.
7. Chen, W. K., *Passive and Active Filters*, Wiley, New York, 1986.

Problems

2.1-1 Find the response of an LTIC system with transfer function $H(s) = \frac{3s}{s^2+2s+2}$ to the following everlasting sinusoidal inputs:

- (a) $3\sin(t + \pi/5)$
- (b) $-7\cos(2t - \pi/3)$
- (c) $10\sin(3t - 80^\circ)$

2.1-2 Consider the all-pass filter specified by the transfer function $H(s) = \frac{(s-2)}{s+2}$. Determine the system response to the following everlasting sinusoids:

- | | |
|----------------------|-------------------------------|
| (a) 1 | (b) $\cos(t + \pi/3)$ |
| (c) $\sin(5t)$ | (d) $\sin(25t)$ |
| (e) $e^{-j\omega t}$ | (f) $\sin(\omega t + \theta)$ |

Comment on the filter response.

2.1-3 Find the response of an LTIC system with transfer function $H(s) = \frac{3s}{s^2+2s+2}$ to everlasting input $x(t) = e^{-t} \cos(2t)$. Is such an input physically practical? Why or why not.

2.1-4 Unlike $H(s)$, a pole-zero plot is an incomplete description of an LTIC system. Explain why.

2.1-5 For each of the following transfer functions, determine and plot the poles and zeros of $H(s)$, and use the pole and zero information to predict overall system behavior. Confirm your predictions by graphing the system's frequency response (magnitude and phase).

(a) $H(s) =$

$$\frac{s^3}{s^3+20s^2+200s+1000}$$

(b) $H(s) =$

$$\frac{0.1s^4+1.4s^2+0.9}{s^4+1.2s^3+6.8s^2+3.6s+9}$$

(c) $H(s) =$

$$\frac{0.03s^4+1.76s^2+15.22}{s^4+2.32s^3+9.79s^2+13.11s+17.08}$$

(d) $H(s) =$

$$\frac{0.94s^4+11.33s^2+33.99}{s^4+0.94s^3+12.66s^2+5.64s+36}$$

2.2-1 Consider the LTIC system with transfer function $H(s) = \frac{3s}{s^2+2s+2}$. Using Eq. (2.15), plot the delay response of this system. Is the delay response constant for signals within the system's passband?

2.2-2 Consider the all-pass filter specified by the transfer function $H(s) = \frac{(s-2)}{s+2}$. Verify the all-pass character of the filter by plotting the system frequency response (magnitude and phase). Can this filter be considered distortionless? Explain.

2.2-3 Consider a system with transfer function given by

$$H(\omega) = ae^{j[\phi_0 \operatorname{sgn}(\omega) - \omega t_g]}$$

(a) Sketch the magnitude and the phase response of this system.

(b) Show that if the input to this system is a lowpass signal $x(t)$, then the output is $y(t) = a\hat{x}(t - t_g)$, where $\hat{x}(t)$ is a distorted version of $x(t)$ with Fourier transform given by $\hat{X}(\omega) = X(\omega)e^{j\phi_0 \operatorname{sgn}(\omega)}$.

2.2-4 A general bandpass signal with spectrum centered at $\omega = \omega_c$ is represented as $x_{bp}(t) = x(t) \cos[\omega_c t + \theta(t)]$, where $x(t)$ is the envelope of the signal and $\theta(t)$ is a time-varying phase. When this signal is passed through the system in Eq. (2.17), show that the system output is given by $y_{bp}(t) = |a|x(t - t_g) \cos[\omega_c(t - t_g) + \phi_0 + \theta(t - t_g)]|$.

2.3-1 Show that a filter with frequency response

$$H(\omega) = \frac{2(10^5)}{\omega^2+10^{10}}e^{-j\omega t_0}$$

is unrealizable. Can this filter be made approximately realizable by choosing a sufficiently large t_0 ? Use your own (reasonable) criterion of approximate realizability to determine t_0 .

2.3-2 Determine if the filters with the following transfer functions are physically realizable. If they are not realizable, explain if and how they can be approximately realized. Be as specific as possible.

(a) $H(\omega) = 10^{-6} \operatorname{sinc}(10^{-6}\omega)$

(b) $H(\omega) = 10^{-4} \Lambda \left(\frac{\omega}{40000\pi} \right)$

(c) $H(\omega) = 2\pi\delta(\omega)$

- 2.3-3** Consider the simple RC circuit shown in Fig. P2.3-3. Let $R = 1 \text{ k}\Omega$ and $C = 1 \text{ nF}$.

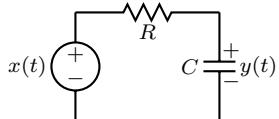


Figure P2.3-3

- (a) Find the system transfer function.
- (b) Plot the magnitude and phase responses.
- (c) Show that a lowpass signal $x(t)$ with bandwidth $W \ll 10^6$ will be transmitted practically without distortion. Determine the output.
- (d) Determine the approximate output if a bandpass signal $x(t) = g(t) \cos(\omega_c t)$ is passed through this filter. Assume that $\omega_c = 3 \times 10^6$ and that the envelope $g(t)$ has a very narrow band, on the order of 50 Hz.

- 2.4-1** What are the two impairments caused by windowing? What window characteristic is most related to each impairment? How is the effect of each impairment reduced? Can either window impairment be entirely eliminated? Explain your answers.

- 2.4-2** An ideal LPF impulse response $h(t) = \frac{B}{\pi} \text{sinc}(\frac{Bt}{\pi})$ is windowed using a rectangular window $w(t) = \Pi(t/T)$. Normally, when a rectangular window is used, the resulting stopband of the filter decays at a rate of -20 dB/decade . However, when T is chosen as $T = 6\pi/B$, the resulting stopband of the filter decays at twice the rate, or -40 dB/decade . Explain how this is possible. Do other choices of T produce similar results? Do any choices of T produce faster rates of decay? Explain.

- 2.4-3** For each of the following windows, determine a suitable width T so that when applied to an ideal LPF with impulse response $h(t) = \frac{10}{\pi} \text{sinc}(\frac{10t}{\pi})$, the resulting transition band is approximately 1 rad/s.

- (a) a rectangular window

(b) a triangular window

(c) a Hann window

(d) a Hamming window

(e) a Blackman window

- 2.5-1** Consider a microphone intended for use in a music recording studio. Determine a suitable frequency response (lowpass, highpass, bandpass, or bandstop) for the microphone, and provide suitable values (such as δ_p and ω_s) to specify the response.

- 2.5-2** Develop a “magnitude response” description for a typical coffee filter. Determine the type of response (lowpass, highpass, bandpass, or bandstop) that is appropriate, and provide suitable values (such as δ_p and ω_s) to specify the response.

- 2.6-1** Determine the filter $H(s)$ that results by applying the following transformations on a lowpass prototype filter given as $H_p(s) = \frac{1}{s^3 + 2s^2 + 2s + 1}$. Plot the corresponding magnitude response. In each case, designate ω_0 as the prototype’s 1-dB cutoff frequency.

(a) A lowpass-to-lowpass transformation
 $\omega_1 = 3 \text{ rad/s}$.

(b) A lowpass-to-highpass transformation
 $\omega_1 = 5 \text{ rad/s}$.

(c) A lowpass-to-bandpass transformation with $\omega_1 = 3 \text{ rad/s}$ and $\omega_2 = 4 \text{ rad/s}$.

(d) A lowpass-to-bandstop transformation with $\omega_1 = 4 \text{ rad/s}$ and $\omega_2 = 5 \text{ rad/s}$.

- 2.6-2** Determine the filter $H(s)$ that results by applying the following transformations on a highpass prototype filter given as $H_p(s) = \frac{s^3}{s^3 + 2s^2 + 2s + 1}$. Plot the corresponding magnitude response. In each case, designate ω_0 as the prototype’s 1-dB cutoff frequency.

(a) A lowpass-to-lowpass transformation with $\omega_1 = 3 \text{ rad/s}$.

(b) A lowpass-to-highpass transformation with $\omega_1 = 5 \text{ rad/s}$.

(c) A lowpass-to-bandpass transformation with $\omega_1 = 3 \text{ rad/s}$ and $\omega_2 = 4 \text{ rad/s}$.

- (d) A lowpass-to-bandstop transformation with $\omega_1 = 4$ rad/s and $\omega_2 = 5$ rad/s.

2.6-3 Consider a complex prototype filter with frequency response

$$H_p(j\omega) = \Pi(\omega - 0.5).$$

- (a) Sketch $|H_p(j\omega)|$.
- (b) Sketch the magnitude response $|H(j\omega)|$ that results from a lowpass-to-lowpass transformation with $\omega_0 = 1$ and $\omega_1 = 2$.
- (c) Sketch the magnitude response $|H(j\omega)|$ that results from a lowpass-to-highpass transformation with $\omega_0 = 1$ and $\omega_1 = 2$.
- (d) Sketch the magnitude response $|H(j\omega)|$ that results from a lowpass-to-bandpass transformation with $\omega_0 = 1$, $\omega_1 = 2$, and $\omega_2 = 4$.
- (e) Sketch the magnitude response $|H(j\omega)|$ that results from a lowpass-to-bandstop transformation with $\omega_0 = 1$, $\omega_1 = 2$, and $\omega_2 = 4$.

2.6-4 Repeat Prob. 2.6-3 for a complex prototype filter with frequency response

$$H_p(j\omega) = \Pi\left(\frac{\omega+1}{2}\right).$$

2.6-5 Consider a simple lowpass prototype $H_p(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$.

- (a) Apply the lowpass-to-lowpass transformation $s \rightarrow \frac{\omega_0}{\omega_1} s$. Determine the resulting filter and plot its magnitude response.
- (b) Apply a transformation $s \rightarrow -\frac{\omega_0}{\omega_1} s$ (notice the added negative sign). Determine the resulting filter and plot its magnitude response. Comment on the results, including why this transformation rule is not used in practice.
- (c) Apply a transformation $s \rightarrow \frac{\omega_0 s^2}{\omega_1}$ (notice the squaring of s). Determine the resulting filter and plot its magnitude response. Comment on the results.

2.6-6 For the following transformations, show that the dc point of a lowpass prototype maps to the geometric, not arithmetic, mean of ω_1 and ω_2 .

- (a) lowpass-to-bandpass transformation
- (b) lowpass-to-bandstop transformation

2.7-1 Determine the transfer function $H(s)$ and plot the magnitude response $H(j\omega)$ for a third-order lowpass Butterworth filter if the 3-dB cutoff frequency is $\omega_c = 100$.

2.7-2 Prove Eq. (2.38), the equation to determine Butterworth filter order.

2.7-3 Using the lowest order possible, find the transfer function $H(s)$ and plot the magnitude response $|H(j\omega)|$ for a lowpass Butterworth filter that satisfies $\alpha_p \leq 3$ dB, $\alpha_s \geq 14$ dB, $\omega_p \geq 100000$ rad/s, and $\omega_s \leq 150000$ rad/s. It is desirable to surpass, as much as possible, the requirement of α_s . Determine the actual specifications α_p , α_s , ω_p , and ω_s of your design.

2.7-4 Determine the lowest order K and the cut-off frequency ω_c of a lowpass Butterworth filter that satisfies the the following specifications. When possible, choose ω_c to provide sensible passband and stopband buffer zones.

- (a) $\alpha_p \leq 0.5$ dB, $\alpha_s \geq 20$ dB, $\omega_p \geq 100$ rad/s, and $\omega_s \leq 200$ rad/s.
- (b) $\delta_p \leq 0.0115$, $\delta_s \leq 10^{-3}$, $\omega_p \geq 1000$ rad/s, and $\omega_s \leq 2000$ rad/s.
- (c) The gain at $3\omega_c$ is required to be no greater than -50 dB.

2.7-5 Find the transfer function $H(s)$ and plot the magnitude response for a highpass Butterworth filter that satisfies $\alpha_p \leq 1$ dB, $\alpha_s \geq 20$ dB, $\omega_p \leq 20$ rad/s, and $\omega_s \geq 10$ rad/s.

2.7-6 Find the transfer function $H(s)$ and plot the magnitude response for a bandpass Butterworth filter that satisfies $\alpha_p \leq 3$ dB, $\alpha_s \geq 17$ dB, $\omega_{p1} \leq 100$ rad/s, $\omega_{p2} \geq 250$ rad/s, $\omega_{s1} \geq 40$ rad/s, and $\omega_{s2} \leq 500$ rad/s.

2.7-7 Find the transfer function $H(s)$ and plot the magnitude response for a bandstop Butterworth filter that satisfies $\alpha_p \leq 3$ dB, $\alpha_s \geq 24$ dB, $\omega_{p1} \geq 20$ rad/s, $\omega_{p2} \leq 60$ rad/s, $\omega_{s1} \leq 30$ rad/s, and $\omega_{s2} \geq 38$ rad/s.

2.7-8 Show that C_K satisfies the recursion $C_K(x) = 2xC_{K-1}(x) - C_{K-2}(x)$ for $K > 1$. To do so, expand Eq. (2.42) using Euler's equation, substitute the result into the right-hand side of the recursion relation, and then simplify.

2.7-9 Repeat Prob. 2.7-1 for a Chebyshev filter.

2.7-10 Design a lowpass Chebyshev filter to satisfy the specifications $\alpha_p \leq 1$ dB, $\alpha_s \geq 22$ dB, $\omega_p \geq 100$ rad/s, and $\omega_s \leq 200$ rad/s. Plot the magnitude response to verify specifications are met.

2.7-11 Design a lowpass Chebyshev filter with 1 dB of passband ripple whose gain drops to -50 dB at $3\omega_c$, where ω_c is the 3-dB cutoff frequency. Plot the magnitude response to verify specifications are met.

2.7-12 Find the transfer function $H(s)$ and plot the magnitude response for a highpass Chebyshev filter that satisfies $\alpha_p \leq 1$ dB, $\alpha_s \geq 22$ dB, $\omega_p \leq 20$ rad/s, and $\omega_s \geq 10$ rad/s.

2.7-13 Find the transfer function $H(s)$ and plot the magnitude response for a bandpass Chebyshev filter that satisfies $\alpha_p \leq 1$ dB, $\alpha_s \geq 17$ dB, $\omega_{p_1} \leq 100$ rad/s, $\omega_{p_2} \geq 250$ rad/s, $\omega_{s_1} \geq 40$ rad/s, and $\omega_{s_2} \leq 500$ rad/s.

2.7-14 Repeat Prob. 2.7-1 for an inverse Chebyshev filter.

2.7-15 Repeat Prob. 2.7-12 for an inverse Chebyshev filter.

2.7-16 Show that the zeros of an inverse Chebyshev lowpass filter are located at

$$z_k = j\omega_s \sec \left(\frac{\pi(2k-1)}{2K} \right)$$

2.7-17 Using functions from MATLAB's Signal Processing Toolbox, repeat Prob. 2.7-1 for an elliptic filter. Plot the magnitude response to verify specifications are met.

Chapter 3

Sampling: The Bridge from Continuous to Discrete

Although discrete-time systems offer clear advantages in performance and flexibility over continuous-time systems, our interests most frequently lie with continuous-time signals. Sampling theory provides a necessary bridge between the continuous-time and discrete-time worlds. Through sampling, a continuous-time signal is converted to discrete-time, ready for processing by any discrete-time system.

The sampling process necessarily discards much of the original signal. If the sampling rate is sufficiently high, however, the original signal is completely recoverable, either exactly or within some error tolerance, from its samples. The necessary quantitative framework for ideal sampling is provided by the sampling theorem, which is derived in the following section. Later sections discuss practical sampling, reconstruction, aliasing, sampling of bandpass signals, and spectral sampling. Practical sampling also requires quantization, and this chapter thus covers analog-to-digital conversion as well as digital-to-analog conversion.

3.1 Sampling and the Sampling Theorem

Generally, *sampling* is considered as any process that records a signal at discrete instances. In this chapter, we restrict our attention to *uniform sampling*. In uniform sampling, sample values are equally spaced from one another by a fixed sampling interval T . Although we focus on temporal sampling where the independent variable is time, our results apply equally well to other cases such as spatial sampling. The reciprocal of the sampling interval is called the *sampling frequency* (or *sampling rate*) $F_s = \frac{1}{T}$, which has units of hertz.

There are two common views of sampling: point sampling and impulse sampling. Figure 3.1 illustrates both. *Point sampling*, as shown in Fig. 3.1b, offers an intuitive and practical representation of the original signal shown in Fig. 3.1a. Point sampling produces a sequence of (usually) finite-valued numbers. Mathematically, we designate point sampling as

$$x[n] = x(nT). \quad (3.1)$$

Following common convention, we use square brackets to emphasize that point sampling results in a discrete-time sequence that is only defined for integer values of the argument. Chapter 4 fully covers discrete-time signals, which include point-sampled signals.

Impulse sampling, as shown in Fig. 3.1c, looks similar to point sampling but is constructed of scaled Dirac delta functions. Although the physically unrealizable and unbounded amplitudes of the impulses somewhat diminish the intuition of this view, impulse sampling is often easier to

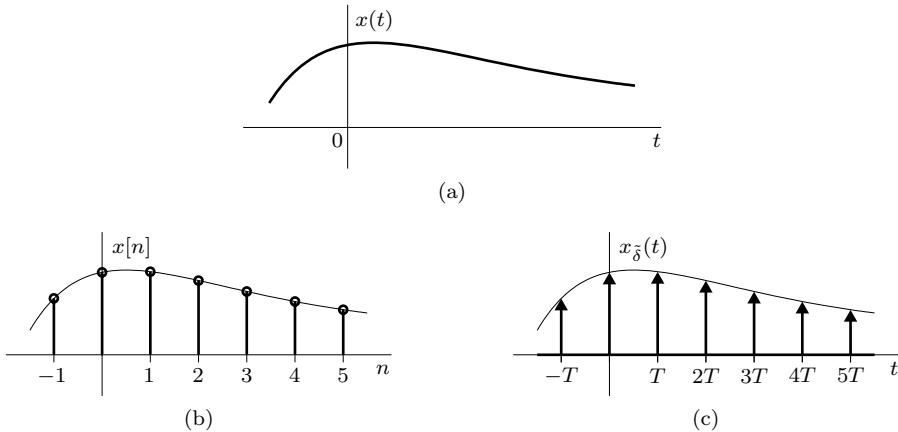


Figure 3.1: Views of sampling: (a) original signal, (b) point sampling, and (c) impulse sampling.

mathematically analyze. Further, impulse sampling offers advantages to better understand multirate systems as well as mixed systems with both continuous-time and discrete-time components. Unlike point sampling, which produces a truly discrete-time sequence, impulse sampling produces a continuous-time signal, although it is zero almost everywhere. Mathematically, we designate an impulse-sampled signal as $x_{\tilde{\delta}}(t)$. Here and elsewhere, the tilde designates a periodic replication; thus, $\tilde{\delta}(t)$ is a periodic replication of Dirac delta functions $\delta(t)$.

Despite the very different constructions of $x[n]$ and $x_{\tilde{\delta}}(t)$, point sampling and impulse sampling provide entirely equivalent information. Intuitively, we expect nothing less given the similarities in Figs. 3.1b and 3.1c. This equivalence is further confirmed in the frequency domain. As we shall see in Ch. 6, the Fourier spectra of point-sampled and impulse-sampled signals are identical. For the remainder of this chapter, we deal primarily with impulse sampling. This is particularly appropriate given the foundations laid in previous chapters for continuous-time signals and systems.

The *sampling theorem*, also called the *Nyquist sampling theorem* or the *Nyquist criterion*, states that a real signal whose spectrum is bandlimited to B Hz can be reconstructed exactly, without any error, from its samples taken uniformly at a rate $F_s > 2B$ samples per second. In other words, the minimum sampling frequency just exceeds twice the hertzian bandwidth. Stated mathematically for the lowpass case where $X(\omega) = 0$ for $|\omega| > 2\pi B$,

$$\text{lossless sampling requires } F_s > 2B. \quad (3.2)$$

Section 3.4 introduces the sampling of bandpass signals.[†]

The minimum sampling rate, $F_s = 2B$, required to recover $x(t)$ from its samples $x_{\tilde{\delta}}(t)$ is called the *Nyquist rate* for $x(t)$. The corresponding sampling interval, $T = 1/2B$, is called the *Nyquist interval* for $x(t)$. Samples of a signal taken at its Nyquist rate are the *Nyquist samples* of that signal. The statement that the Nyquist rate 2B Hz is the minimum sampling rate required to preserve the information of $x(t)$ seems to contradict Eq. (3.2), which asserts that the sampling rate F_s needs to be *greater than* 2B Hz in order to preserve the information of $x(t)$. Strictly speaking, Eq. (3.2) is the correct statement. However, if the spectrum $X(\omega)$ contains no impulse or its derivatives at the highest frequency B Hz, then the minimum sampling rate 2B Hz is adequate. It is rare to observe $X(\omega)$ with an impulse or its derivatives at the highest frequency. If such a situation does occur, we

[†]A bandpass signal whose spectrum exists over a frequency band $f_c - \frac{B}{2} < |f| < f_c + \frac{B}{2}$ has a bandwidth of B Hz and, as in the lowpass case, is uniquely determined by $2B$ samples per second. In general, however, the needed samples do not follow a single uniform rate; rather, two interlaced sampling trains, each at a rate B samples per second, are often required [1].

should use Eq. (3.2) and sample at a rate above Nyquist.[†] In practice, where filters are not ideal, sampling rates are usually chosen modestly above the Nyquist rate; a rate 20% greater is common.

To prove, and more importantly to understand, the sampling theorem, it is helpful to investigate the sampling operation in both the time and frequency domains. Consider a signal $x(t)$ (Fig. 3.2a) whose spectrum is bandlimited to B Hz (Fig. 3.2d). Although the spectra in Fig. 3.2 are shown as real, our arguments are valid for the complex case as well. For convenience, spectra are shown as functions of $\frac{\omega}{2\pi}$, or Hz.

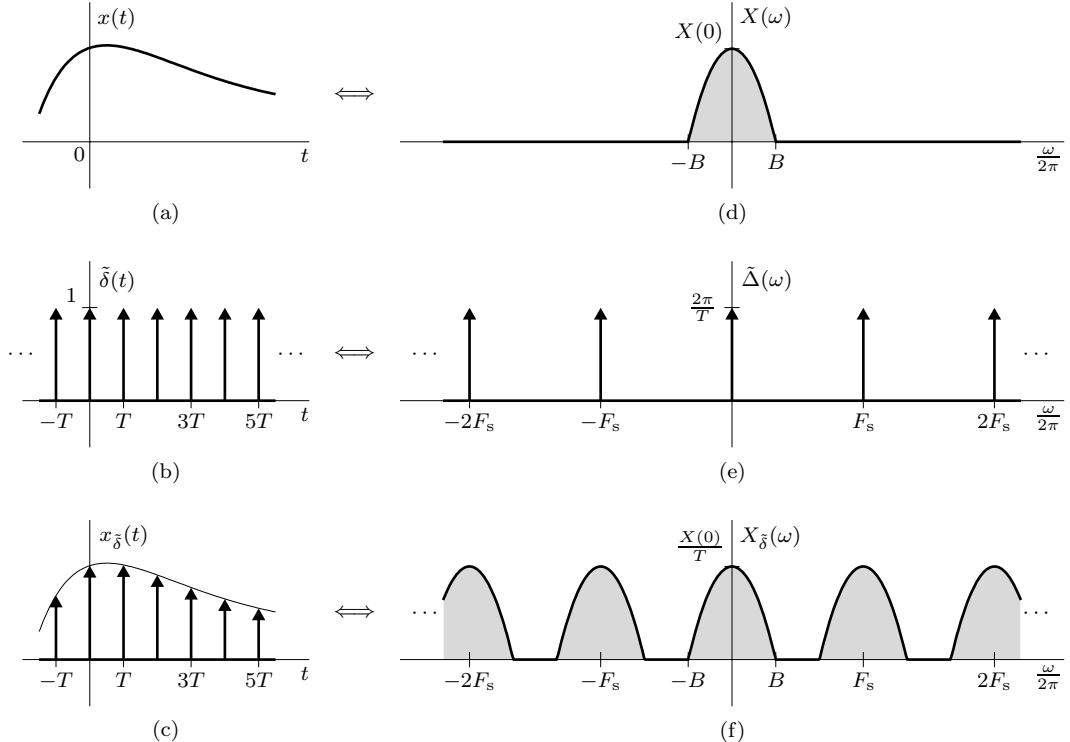


Figure 3.2: Impulse sampling: (a)–(c) time domain and (d)–(f) frequency domain.

Impulse sampling signal $x(t)$ at a rate of F_s Hz (F_s samples per second) is accomplished by multiplying $x(t)$ by the unit impulse train $\tilde{\delta}(t)$. As shown in Fig. 3.2b, $\tilde{\delta}(t)$ consists of unit impulses spaced periodically every $T = 1/F_s$ seconds. The resulting impulse-sampled signal $x_{\tilde{\delta}}(t)$, shown in Fig. 3.2c, consists of scaled impulses spaced every T seconds. The n th impulse, located at $t = nT$, has a strength $x(nT)$, the value of $x(t)$ at $t = nT$. Thus,

$$x_{\tilde{\delta}}(t) = x(t)\tilde{\delta}(t) = x(t) \sum_{n=-\infty}^{\infty} \delta(t - nT) = \sum_{n=-\infty}^{\infty} x(nT)\delta(t - nT). \quad (3.3)$$

As shown in Ex. 1.7, the T -periodic unit impulse train $\tilde{\delta}(t)$ is expressed by trigonometric Fourier series as

$$\tilde{\delta}(t) = \frac{1}{T} [1 + 2 \cos(\omega_s t) + 2 \cos(2\omega_s t) + 2 \cos(3\omega_s t) + \dots], \quad (3.4)$$

where $\omega_s = 2\pi F_s$ is the radian sampling frequency. Therefore,

$$x_{\tilde{\delta}}(t) = x(t)\tilde{\delta}(t) = \frac{1}{T} [x(t) + 2x(t)\cos(\omega_s t) + 2x(t)\cos(2\omega_s t) + 2x(t)\cos(3\omega_s t) + \dots], \quad (3.5)$$

[†]See Prob. 3.1-1 for an interesting exception.

To find $X_{\tilde{\delta}}(\omega)$, the Fourier transform of $x_{\tilde{\delta}}(t)$, we take the Fourier transform of the right-hand side of Eq. (3.5), term by term. The transform of the first term in the brackets is $X(\omega)$. The transform of the second term, $2x(t) \cos(\omega_s t)$, is $X(\omega - \omega_s) + X(\omega + \omega_s)$ (see Eq. (1.90)). This represents spectrum $X(\omega)$ shifted to ω_s and $-\omega_s$. Similarly, the transform of the third term, $2x(t) \cos(2\omega_s t)$, is $X(\omega - 2\omega_s) + X(\omega + 2\omega_s)$, which represents the spectrum $X(\omega)$ shifted to $2\omega_s$ and $-2\omega_s$, and so on to infinity. These results mean that the spectrum $X_{\tilde{\delta}}(\omega)$ consists of $X(\omega)$ repeating periodically with period $\omega_s = \frac{2\pi}{T}$ rad/s, or $F_s = \frac{1}{T}$ Hz, as depicted in Fig. 3.2f. Thus, including the scale factor $1/T$ found in Eq. (3.5),

$$X_{\tilde{\delta}}(\omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X(\omega - k\omega_s). \quad (3.6)$$

If we are to reconstruct $x(t)$ from $x_{\tilde{\delta}}(t)$, we should be able to recover $X(\omega)$ from $X_{\tilde{\delta}}(\omega)$. Error-free recovery is possible if there is no overlap between adjacent edges of the successive replicates in $X_{\tilde{\delta}}(\omega)$. Figure 3.2f indicates that this requires

$$F_s > 2B \quad \text{or} \quad T < \frac{1}{2B}.$$

This is the sampling theorem of Eq. (3.2).

As long as the sampling frequency F_s is greater than twice the signal bandwidth B (in hertz), $X_{\tilde{\delta}}(\omega)$ consists of nonoverlapping repetitions of $X(\omega)$. Figure 3.2f shows that the gap between the two adjacent spectral repetitions is $F_s - 2B$ Hz. In this case, $x(t)$ is recovered from its samples by passing $x_{\tilde{\delta}}(t)$ through any ideal lowpass filter with gain T and bandwidth between B and $F_s - B$ Hz. The block diagram representation of the sampling and recovery process is shown in Fig. 3.3. As long as $x(t)$ is sampled above the Nyquist rate, the reconstructor output $\hat{x}(t)$ is equal to $x(t)$.

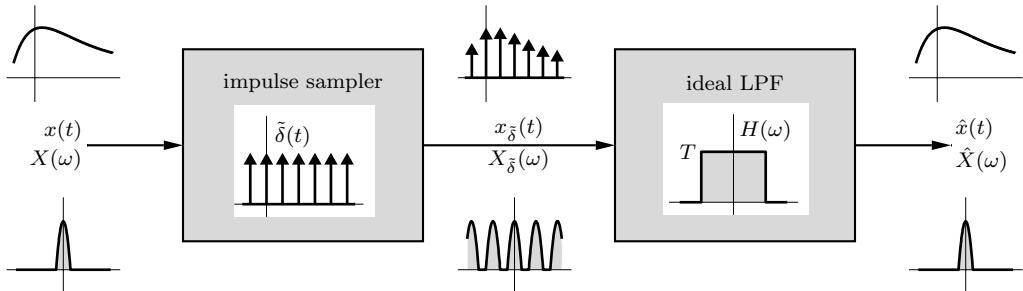


Figure 3.3: Block representation of impulse sampling and reconstruction.

An Elegant Alternative

To provide deeper understanding of sampling and the sampling theorem, we now present an elegant alternative to finding $X_{\tilde{\delta}}(\omega)$. Starting with Eq. (3.4) and taking its Fourier transform yield

$$\tilde{\Delta}(\omega) = \frac{1}{T} [2\pi\delta(\omega) + 2\pi [\delta(\omega - \omega_s) + \delta(\omega + \omega_s)] + 2\pi [\delta(\omega - 2\omega_s) + \delta(\omega + 2\omega_s)] + \dots].$$

More compactly, we see that

$$\tilde{\delta}(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \iff \tilde{\Delta}(\omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s). \quad (3.7)$$

In other words, the Fourier transform of the unit impulse train $\tilde{\delta}(t)$ is just another (scaled) impulse train, as shown in Figs. 3.2b and 3.2e.

The spectrum $X_{\tilde{\delta}}(\omega)$ of our impulse-sampled signal is now simple to obtain. Since $x_{\tilde{\delta}}(t)$ is a product of $x(t)$ and $\delta(t)$, $X_{\tilde{\delta}}(\omega)$ is found through convolution as

$$\begin{aligned} X_{\tilde{\delta}}(\omega) &= \frac{1}{2\pi} X(\omega) * \tilde{\Delta}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\lambda) \tilde{\Delta}(\omega - \lambda) d\lambda \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\lambda) \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s - \lambda) d\lambda \\ &= \frac{1}{T} \sum_{k=-\infty}^{\infty} \int_{-\infty}^{\infty} X(\lambda) \delta(\omega - k\omega_s - \lambda) d\lambda \\ X_{\tilde{\delta}}(\omega) &= \frac{1}{T} \sum_{k=-\infty}^{\infty} X(\omega - k\omega_s) \end{aligned} \quad (3.8)$$

Thus, the spectrum of an impulse-sampled signal is a scaled and periodic replication of the original spectrum, where each copy is spaced ω_s rad/s (or F_s Hz) apart. Equation (3.8) confirms, as stated in Table 1.4, that the transform of a discrete (sampled) signal is necessarily periodic.

The sampling theorem demonstrated here uses samples taken at uniform intervals. This condition is not necessary. Samples can be taken arbitrarily at any instants as long as the sampling times are recorded and there are, on average, $2B$ samples per second [2]. The essence of the sampling theorem has been known to mathematicians for a long time in the form of the *interpolation formula*, which we present later as Eq. (3.13). The origin of the sampling theorem was attributed by H. S. Black to Cauchy in 1841. The essential idea of the sampling theorem was rediscovered in the 1920s by Carson, Nyquist, and Hartley.

▷ Example 3.1 (Nyquist Sampling, Undersampling, and Oversampling)

Using signal $x(t) = \text{sinc}^2(5t)$ (Fig. 3.4a) with spectrum $X(\omega) = 0.2\Lambda(\frac{\omega}{20\pi})$ (Fig. 3.4b), plot the impulse-sampled signals and spectra when sampling is at the Nyquist rate, at half the Nyquist rate (undersampling), and at twice the Nyquist rate (oversampling).

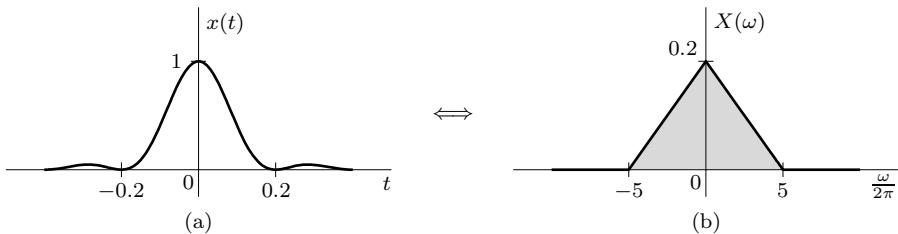


Figure 3.4: (a) $x(t) = \text{sinc}^2(5t)$ and (b) $X(\omega) = 0.2\Lambda(\frac{\omega}{20\pi})$

The bandwidth of $x(t)$ is 5 Hz (10π rad/s). Consequently, the Nyquist rate is 10 Hz, and the Nyquist interval is $T = \frac{1}{2B} = 0.1$ seconds. The spectrum $X_{\tilde{\delta}}(\omega)$ of the sampled signal $x_{\tilde{\delta}}(t)$ consists of components $\frac{1}{T}X(\omega) = \frac{0.2}{T}\Lambda(\frac{\omega}{20\pi})$ spaced periodically every F_s Hz. The table in Fig. 3.5 summarizes the $F_s = 10$ Hz (Nyquist), $F_s = 5$ Hz (undersampling at half Nyquist), and $F_s = 20$ Hz (oversampling at twice Nyquist) cases.

In the first case, we use the Nyquist sampling rate of 10 Hz (Fig. 3.5a). The spectrum $X_{\tilde{\delta}}(\omega)$, shown in Fig. 3.5b, consists of back to back, nonoverlapping repetitions of $\frac{1}{T}X(\omega)$ spaced every 10 Hz. Hence, $X(\omega)$ can be recovered from $X_{\tilde{\delta}}(\omega)$ using an ideal lowpass filter of bandwidth 5 Hz.

In the second case, the sampling rate is 5 Hz, which represents an undersampling condition. As shown in Fig. 3.5c, only one nonzero impulse exists at $t = 0$; all other impulses are eliminated

since they occur where $x(t)$ equals zero. The spectrum $X_{\delta}(\omega)$, shown in Fig. 3.5d, consists of overlapping repetitions of $\frac{1}{T}X(\omega)$ spaced every 5 Hz. Not surprisingly, since the time-domain signal is a single impulse function, these overlapping spectra add to a constant. Due to the overlapping spectral components, $X(\omega)$ is not recoverable from $X_{\delta}(\omega)$, and $x(t)$ cannot be reconstructed from the samples $x_{\delta}(t)$.

In the final case, the sampling rate is 20 Hz, which represents an oversampling condition. The impulses of $x_{\delta}(t)$ are spaced more closely together, as shown in Fig. 3.5e. Consequently, the repetitions of $\frac{1}{T}X(\omega)$ are spaced more widely apart, as shown in Fig. 3.5f, leaving large gaps between copies. Hence, $X(\omega)$ can be recovered from $X_{\delta}(\omega)$ using a filter with constant gain for frequencies below 5 Hz and zero gain beyond 10 Hz. These filter specifications, while technically unrealizable, can be closely satisfied by practical, realizable filters.

Sampling Frequency F_s	Sampling Interval T	$\frac{1}{T}X(\omega)$	Comments
10	0.1	$2\Lambda(\frac{\omega}{20\pi})$	Nyquist sampling
5	0.2	$\Lambda(\frac{\omega}{20\pi})$	Undersampling
20	0.05	$4\Lambda(\frac{\omega}{20\pi})$	Oversampling

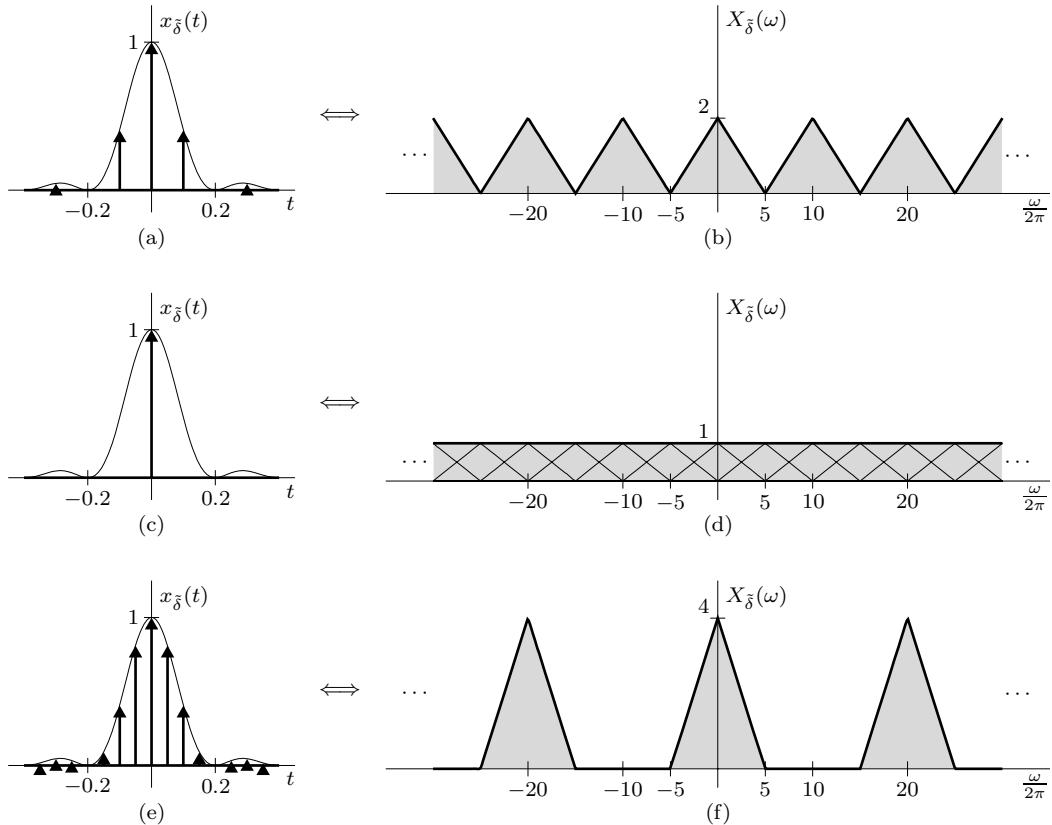


Figure 3.5: Investigating different sampling rates: (a)–(b) Nyquist sampling, (c)–(d) undersampling, and (e)–(f) oversampling.

For Skeptics Only

Rare is a reader who, at first encounter, is not skeptical of the sampling theorem. It seems impossible that Nyquist samples can define the one and only signal that passes through those sample values. We can easily picture an infinite number of signals passing through a given set of samples. However, among this infinite number of signals, only one has the minimum bandwidth $B \leq 1/2T$ Hz, where T is the sampling interval (see Prob. 3.2-1).

To summarize, for a given set of samples taken at a rate F_s Hz, there is only one signal of bandwidth $B \leq F_s/2$ that passes through those samples. All other signals that pass through those samples have bandwidth higher than $F_s/2$ Hz, and the samples are sub-Nyquist for those signals.

▷ Drill 3.1 (Determining the Nyquist Rate and the Nyquist Interval)

Determine the Nyquist rate and the Nyquist interval for the signals

$$\begin{array}{ll} \text{(a)} & x_a(t) = \text{sinc}(100t) \\ \text{(c)} & x_c(t) = \text{sinc}(50t) \text{sinc}(100t) \end{array} \quad \begin{array}{ll} \text{(b)} & x_b(t) = \text{sinc}(50t) + \text{sinc}(100t) \\ \text{(d)} & x_d(t) = \text{sinc}(50t) * \text{sinc}(100t) \end{array}$$

△

3.1.1 Practical Sampling

In proving the sampling theorem, we assumed ideal samples obtained by multiplying a signal $x(t)$ by an impulse train that is physically unrealizable. In practice, we multiply a signal $x(t)$ by a T -periodic train $\tilde{p}(t)$ (Fig. 3.6b) that is comprised of component pulses $p(t)$. Mathematically, $\tilde{p}(t)$ is a periodic replication of $p(t)$ constructed as

$$\tilde{p}(t) = \sum_{n=-\infty}^{\infty} p(t - nT). \quad (3.9)$$

Although the component pulses $p(t)$ typically possess finite duration that is smaller than T , this condition is not necessary.

The *pulse-sampled signal* $x_{\tilde{p}}(t)$, which equals the product $\tilde{p}(t)x(t)$, is illustrated in Fig. 3.6c. Is it possible, we might wonder, to recover or reconstruct $x(t)$ from $x_{\tilde{p}}(t)$? Perhaps surprisingly, the answer is affirmative, provided that the sampling rate is not below the Nyquist rate. The signal $x(t)$ can be recovered by lowpass filtering $x_{\tilde{p}}(t)$ as if it were sampled by an impulse train.

The plausibility of this result becomes apparent when we consider the fact that reconstruction of $x(t)$ requires knowledge of the impulse-sampled values. This information is available or built into the sampled signal $x_{\tilde{p}}(t)$ because the n th sampled pulse strength is $x(nT)$. To prove the result analytically, we observe that the pulse train $\tilde{p}(t)$ depicted in Fig. 3.6b, being a periodic signal, has a Fourier series given by

$$\tilde{p}(t) = \sum_{k=-\infty}^{\infty} \tilde{P}_k e^{jk\omega_s t},$$

where $\omega_s = 2\pi F_s$. As shown in Fig. 3.6e, the spectrum $\tilde{P}(\omega)$ of pulse train $\tilde{p}(t)$ is an impulse train weighted by the scaled Fourier series coefficients $2\pi\tilde{P}_k$. The pulse-sampled signal $x_{\tilde{p}}(t)$ is therefore represented as

$$x_{\tilde{p}}(t) = \tilde{p}(t)x(t) = \sum_{k=-\infty}^{\infty} \tilde{P}_k e^{jk\omega_s t} x(t),$$

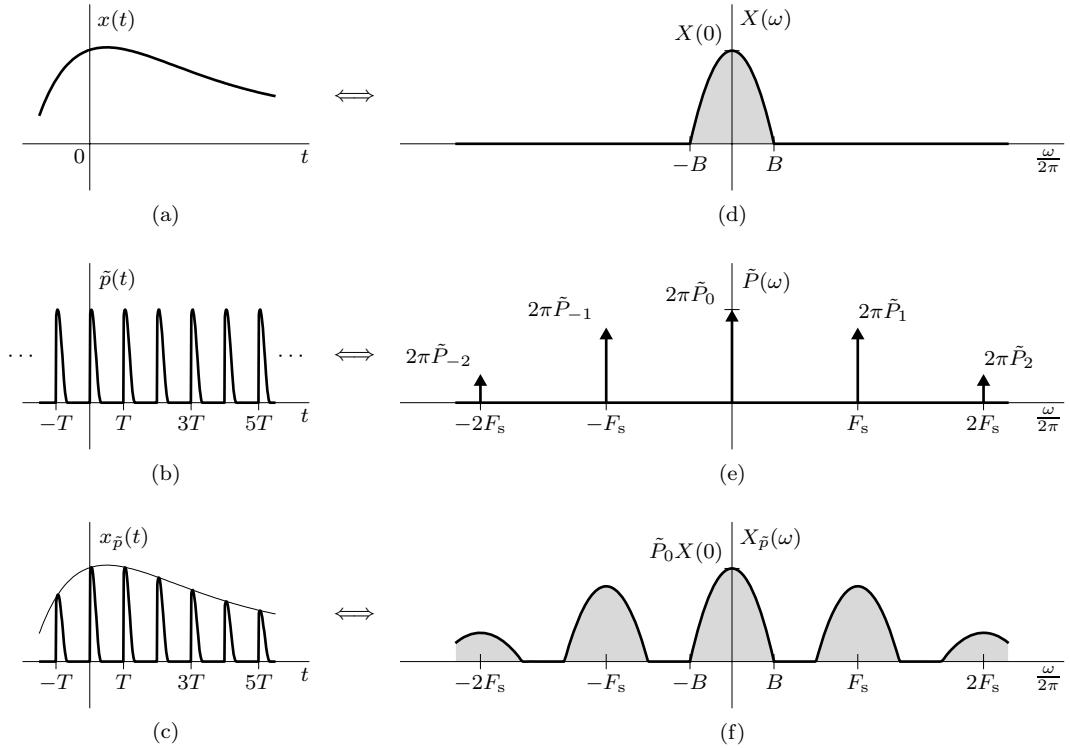


Figure 3.6: Practical pulse sampling: (a)–(c) time domain and (d)–(f) frequency domain.

and using the frequency-shifting property, its spectrum is

$$X_{\tilde{p}}(\omega) = \sum_{k=-\infty}^{\infty} \tilde{P}_k X(\omega - k\omega_s).$$

As shown in Fig. 3.6f, $X_{\tilde{p}}(\omega)$ is comprised of copies of $X(\omega)$, each separated by F_s and weighted by the appropriate Fourier series coefficient \tilde{P}_k . As long as $F_s > 2B$, a lowpass filter with gain $1/\tilde{P}_0$ is all that is necessary to recover $x(t)$ from $x_{\tilde{p}}(t)$. The block diagram representation of the pulse sampling and recovery process shown in Fig. 3.7 follows from Fig. 3.3 if $\tilde{\delta}(t)$ is replaced with $\tilde{p}(t)$.

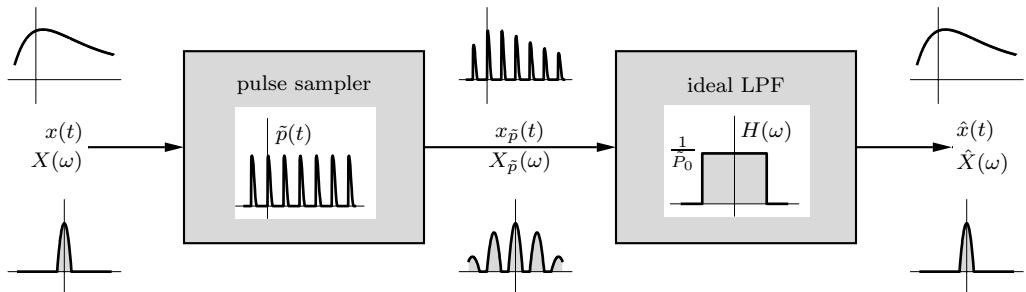


Figure 3.7: Block representation of pulse sampling and reconstruction.

▷ **Example 3.2 (Rectangular Pulse Sampling)**

Consider the signal $x(t) = \text{sinc}^2(5t)$ and the pulse train $\tilde{p}(t) = \sum_{n=-\infty}^{\infty} \Pi(\frac{t-0.1n}{0.025})$. Plot the pulse-sampled signal $x_{\tilde{p}}(t) = \tilde{p}(t)x(t)$ and its spectrum $X_{\tilde{p}}(\omega)$. Is $x_{\tilde{p}}(t)$ sampled above, below, or at the Nyquist rate? If $x_{\tilde{p}}(t)$ is applied to a unit-gain ideal lowpass filter with a bandwidth of 5 Hz, what is the output signal $y(t)$?

From Ex. 3.1, we know that $x(t)$, shown in Fig. 3.8a, has spectrum $X(\omega) = 0.2\Lambda(\frac{\omega}{20\pi})$, shown in Fig. 3.8d. The bandwidth of $x(t)$ is 5 Hz.

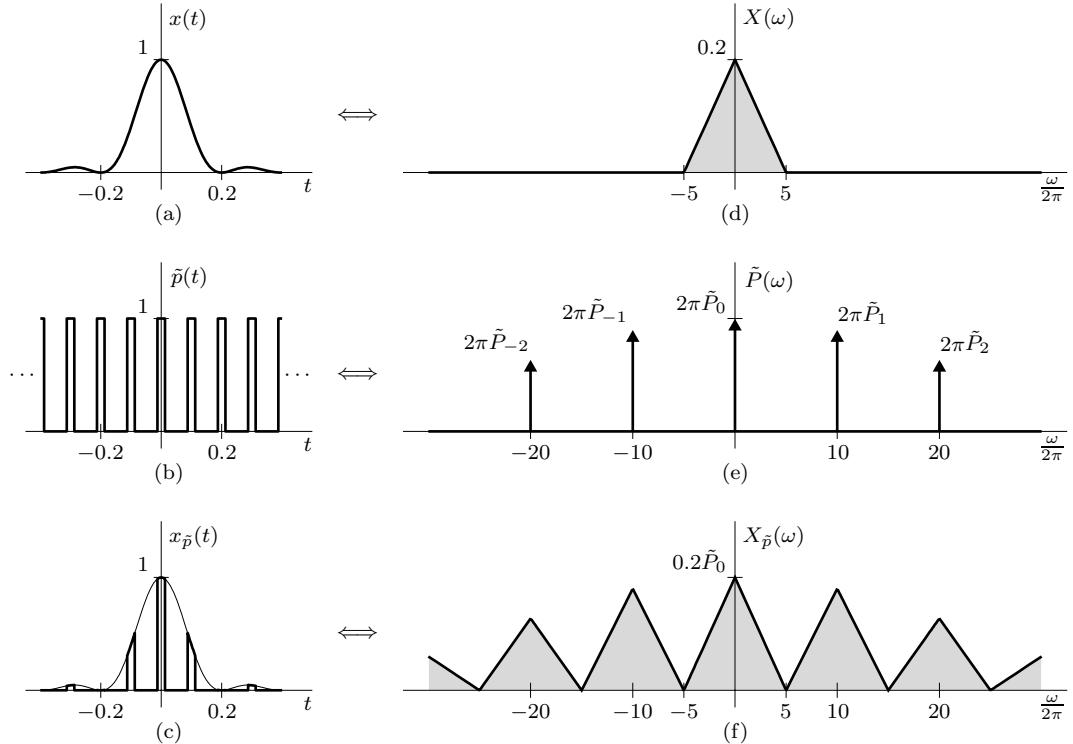


Figure 3.8: Rectangular pulse sampling: (a)–(c) time domain and (d)–(f) frequency domain.

As shown in Fig. 3.8b, $\tilde{p}(t)$ is a 25% duty-cycle rectangular pulse sequence with period $T = 0.1$ s. Using Eqs. (1.57) and (1.58), the corresponding Fourier series is

$$\tilde{p}(t) = \sum_{k=-\infty}^{\infty} \tilde{P}_k e^{j k \omega_s t} = \sum_{k=-\infty}^{\infty} \frac{1}{4} \text{sinc}(k/4) e^{j k \omega_s t},$$

where $\omega_s = \frac{2\pi}{T} = 20\pi$. Taking the Fourier transform yields

$$\tilde{P}(\omega) = \sum_{k=-\infty}^{\infty} 2\pi \tilde{P}_k \delta(\omega - k\omega_s) = \sum_{k=-\infty}^{\infty} \frac{2\pi}{4} \text{sinc}(k/4) \delta(\omega - k\omega_s),$$

which is shown in Fig. 3.8e. Since $F_s = \frac{1}{T} = 10$ Hz is twice the signal bandwidth of 5 Hz, sampling occurs exactly at the Nyquist rate.

As shown in Fig. 3.8c, the pulse-sampled signal $x_{\tilde{p}}(t)$ is the product of $x(t)$ and $\tilde{p}(t)$. Using the multiplication property of Eq. (1.96), the spectrum $X_{\tilde{p}}(\omega)$ is the convolution of $X(\omega)$ and $\tilde{P}(\omega)$ scaled by $\frac{1}{2\pi}$, as shown in Fig. 3.8f. When $x_{\tilde{p}}(t)$ is passed through a unit-gain ideal lowpass filter, the output $y(t)$ is just $x(t)$ scaled by $\tilde{P}_0 = \frac{1}{4}$. Thus,

$$y(t) = \frac{1}{4}x(t).$$

Example 3.2 ◀

▷ Drill 3.2 (Not All Pulses Are Appropriate for Pulse Sampling)

Show that the pulse $p(t)$ used to construct the sampling pulse train $\tilde{p}(t)$ in Eq. (3.9) cannot have zero area if we wish to reconstruct a lowpass signal $x(t)$ by lowpass filtering the sampled signal.

◀

3.2 Signal Reconstruction

The process of reconstructing a continuous-time signal $x(t)$ from its samples is also known as *interpolation*. In Sec. 3.1, we saw that a signal $x(t)$ bandlimited to B Hz can be reconstructed (interpolated) exactly from its samples $x_{\tilde{\delta}}(t)$ if the sampling frequency $F_s > 2B$ Hz or, equivalently, the sampling interval $T < 1/2B$. This reconstruction is accomplished by passing the sampled signal through an ideal lowpass filter with gain T and bandwidth between B and $F_s - B$ Hz, as shown in Fig. 3.3. From a practical viewpoint, a good bandwidth choice is the middle value $F_s/2$ Hz. This value allows for small deviations in the ideal filter characteristics on either side of the cutoff frequency. With this choice of cutoff frequency and gain T , the ideal lowpass filter required for signal reconstruction (or interpolation) is

$$H(\omega) = T \Pi\left(\frac{\omega}{2\pi F_s}\right) = T \Pi\left(\frac{\omega T}{2\pi}\right), \quad (3.10)$$

and signal reconstruction is described as

$$\hat{X}(\omega) = X_{\tilde{\delta}}(\omega)H(\omega). \quad (3.11)$$

The reconstructor output $\hat{X}(\omega)$ is equal to $X(\omega)$ if sampling is performed above the Nyquist rate. The interpolation process here is expressed in the frequency domain as a filtering operation. Now, we shall examine this process from a time-domain viewpoint.

Ideal Interpolation: A Time-Domain View

The ideal interpolation filter frequency response given in Eq. (3.10) is illustrated in Fig. 3.9a. The inverse Fourier transform of $H(\omega)$ yields the impulse response of this filter,

$$h(t) = \text{sinc}(t/T). \quad (3.12)$$

This impulse response, depicted in Fig. 3.9b, corresponds to the special but most-common case when the filter bandwidth is set to $F_s/2$. Observe the interesting fact that $h(t) = 0$ at all sample instants $t = nT$ except $t = 0$. When the sampled signal $x_{\tilde{\delta}}(t)$ is applied at the input of this ideal filter, the output is $\hat{x}(t)$, a reconstruction of the original signal $x(t)$.

To determine $\hat{x}(t)$, first notice that each sample in $x_{\tilde{\delta}}(t)$, being an impulse, generates a sinc pulse of height equal to the strength of the sample, as illustrated in Fig. 3.9c. Adding all the sinc pulses

generated by the samples results in $\hat{x}(t)$. More formally, the n th sample of the input $x_{\delta}(t)$ is the impulse $x(nT)\delta(t - nT)$, and the filter output of this impulse is $x(nT)h(t - nT)$. Summing over all impulses that comprise $x_{\delta}(t)$, the filter output is thus

$$\begin{aligned}\hat{x}(t) &= \sum_{n=-\infty}^{\infty} x(nT)h(t - nT) \\ &= \sum_{n=-\infty}^{\infty} x(nT)\text{sinc}\left(\frac{t - nT}{T}\right).\end{aligned}\quad (3.13)$$

In other words, we can view the reconstruction of $x(t)$ as a sum of weighted, shifted sinc functions, as illustrated in Fig. 3.9c. This construction is similar to others we have seen, where signals are constructed of weighted sums of shifted functions such as impulses or complex exponentials. Considering Eq. (3.13) another way, each sample in $x_{\delta}(t)$, being an impulse, generates a sinc pulse of height equal to the strength of the sample. Addition of the sinc pulses generated by all the samples results in $\hat{x}(t)$.

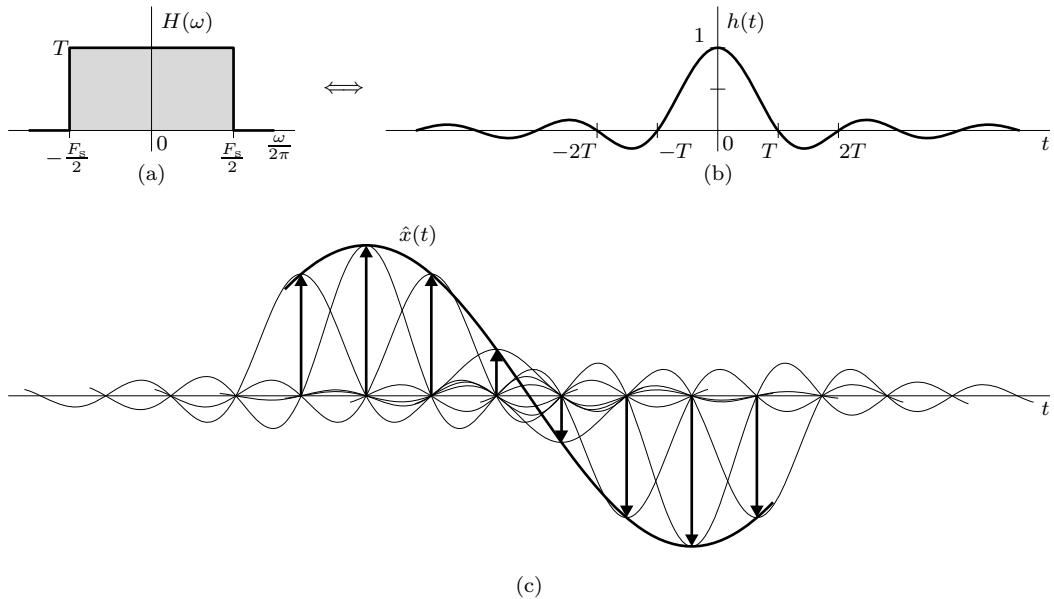


Figure 3.9: Ideal interpolation with an $F_s/2$ reconstruction filter: (a) frequency response, (b) impulse response, and (c) signal reconstruction.

Equation (3.13) is the *interpolation formula*. Since $h(t) = 0$ at all sample instants $t = nT$ except $t = 0$, this formula returns values $x(nT)$ without much work; only one term in the sum is nonzero. For all times that are not sample instants ($t \neq nT$), however, the formula requires *every* sample value from $-\infty$ to ∞ in order to produce $\hat{x}(t)$. You can never sample a signal long enough to perform ideal reconstruction. Consequently, more practical measures are required for real-world signal interpolation.

▷ Example 3.3 (A Simple Case of Ideal Interpolation)

Find a signal $x(t)$ that is bandlimited to $F_s/2$ Hz and whose samples are

$$x(0) = 1 \quad \text{and} \quad x(\pm T) = x(\pm 2T) = x(\pm 3T) = \dots = 0.$$

There is one and only one signal that satisfies the conditions of this problem, and it is determined using the interpolation formula of Eq. (3.13). Since all but one of the samples are zero, only one term (corresponding to $n = 0$) in the summation on the right-hand side of Eq. (3.13) survives. Further, since $x(t)$ is bandlimited to $F_s/2$ Hz, the reconstruction $\hat{x}(t)$ equals $x(t)$ exactly. Thus,

$$\hat{x}(t) = \text{sinc}(t/T) = x(t).$$

This signal is identical to that shown in Fig. 3.9b.

Example 3.3 ◀

▷ **Example 3.4 (Approximating Ideal Interpolation)**

Consider the 0.25-Hz sinusoid $\cos(\pi t/2)$ sampled at a rate $F_s = 1$ Hz. Reconstruct this signal from its samples over $-2 \leq t \leq 2$ (one period) using a suitable truncation of the ideal interpolation formula. Compute the resulting RMS error of the truncated approximation $\hat{x}_N(t)$.

Although the interpolation formula requires samples from all time to exactly reconstruct a signal, a truncated reconstruction $\hat{x}_N(t)$ that is nearly error-free over an interval is possible using a finite set of samples. Since samples weight sinc functions that asymptotically decay, the impact of each sample is primarily local in nature. Thus, we need only include a limited number of samples outside the desired interval of reconstruction.

In the current case, the reconstruction interval contains 5 samples. We extend this number by 10 in both directions to obtain our approximation, which is computed in MATLAB.

```
01 x = @(t) cos(pi*t/2); t = linspace(-2,2,501); xhatN = 0; T = 1;
02 for n = -12:12,
03     xhatN = xhatN + x(n*T)*sinc((t-n*T)/T);
04 end
05 plot(t,xhatN);
06 dt = t(2)-t(1); RMSerror = sqrt(sum((xhatN-x(t)).^2)*dt)
RMSerror = 0.0030
```

Line 01 defines the original signal, creates a time vector for the reconstruction, initializes the estimate to zero, and sets the sampling interval. The loop of lines 02–04 evaluates the sum of Eq. (3.13) truncated to $10 + 5 + 10 = 25$ terms. Notice that the samples $x(nT)$ only take on values of 1, 0, or -1 ; the CT nature of our reconstruction comes from the sinc functions. The result, plotted in line 05 and shown in Fig. 3.10, is nearly indistinguishable from the original sinusoid $\cos(\pi t/2)$. The RMS error of the interpolation, computed in line 06, is less than 1% of the peak signal value. By increasing the number of terms (line 02), the reconstruction is improved to any desired level.

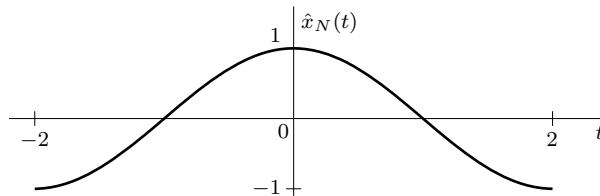


Figure 3.10: Approximating ideal interpolation using a truncated sum.

Example 3.4 ◀

▷ **Drill 3.3 (Interpolating a Discrete Rectangular Pulse)**

A discrete rectangular pulse is described by samples

$$x(nT) = \begin{cases} 1 & |n| \leq N \\ 0 & \text{otherwise} \end{cases}.$$

Use Eq. (3.13) to interpolate these samples for the $N = 5$ case. Plot the result. Is the interpolation $\hat{x}(t)$ a continuous-time rectangular pulse? Explain.

△

Practical Interpolation: The Zero-Order Hold Filter

To construct a more practical interpolation system, we assign our interpolation filter with a very simple impulse response $h(t) = \Pi(t/T)$, depicted in Fig. 3.11a. This is a gate pulse centered at the origin with unit height and width T (the sampling interval).[†] Taking the Fourier transform of this impulse response, the frequency response of the filter is

$$H(\omega) = T \operatorname{sinc}\left(\frac{\omega T}{2\pi}\right).$$

The magnitude response $|H(\omega)|$ for this filter, illustrated in Fig. 3.11b, is a coarse approximation of the ideal lowpass filter, shown shaded, required for exact interpolation, and it portends a somewhat crude and imprecise interpolation. This filter is also known as a *zero-order hold* (ZOH) filter.

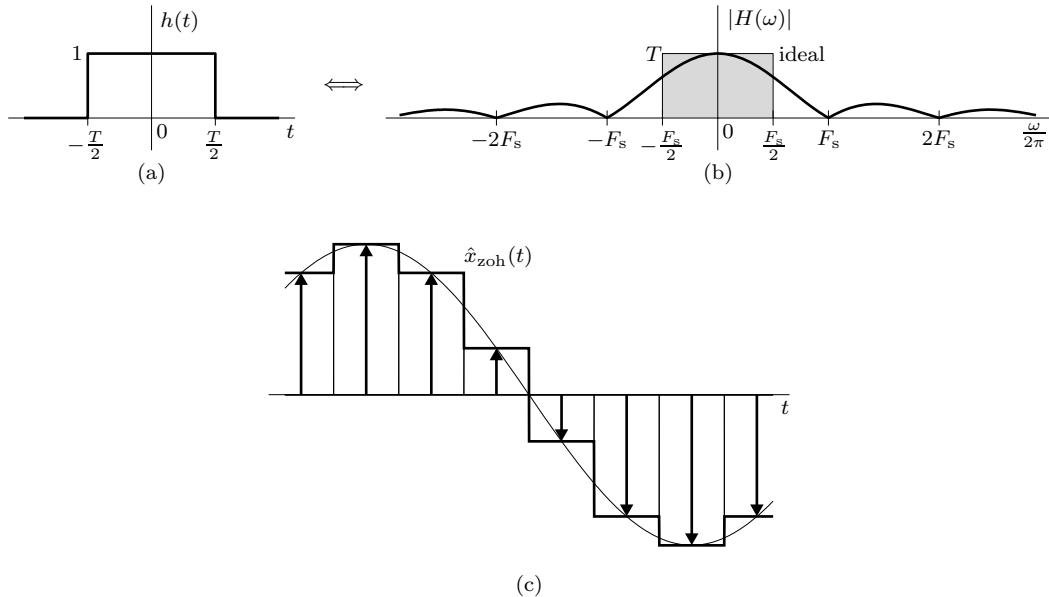


Figure 3.11: ZOH interpolation: (a) impulse response, (b) magnitude response, and (c) signal reconstruction.

The sampled signal $x_{\tilde{\delta}}(t)$ consists of an impulse train where the n th impulse at $t = nT$ has strength $x(nT)$. When passed through a ZOH, each sample in $x_{\tilde{\delta}}(t)$, being an impulse, produces at

[†]For convenience, Fig. 3.11a presents a noncausal, and therefore unrealizable, impulse response. A simple delay of $T/2$ renders $h(t)$ causal at a cost of a $T/2$ delay in the filter output.

the output a gate pulse of height equal to the strength of the sample. Summed together, the filter output $\hat{x}_{\text{zoh}}(t)$ is a staircase approximation of $x(t)$, as shown in Fig. 3.11c. More generally, when an impulse-sampled signal is passed through a filter, the output is a weighted sum of shifted impulse response functions. In the case of ideal interpolation, the output is a superposition of sinc functions. For ZOH interpolation, the output is a superposition of nonoverlapping but adjacent rectangular pulses.

We can improve on the ZOH filter by using a *first-order hold* filter, which results in a straight-line (linear), rather than staircase, approximation. The linear interpolation filter, whose impulse response is the triangle pulse $\Lambda(\frac{t}{2T})$, results in an interpolation in which successive sample tops are connected by straight-line segments (see Prob. 3.2-4).

3.3 Practical Difficulties in Sampling and Reconstruction

Consider the signal sampling and reconstruction procedure, illustrated in Fig. 3.3. If $x(t)$ is sampled at the Nyquist rate $F_s = 2B$ Hz, the spectrum $X_{\tilde{\delta}}(\omega)$ consists of repetitions of $X(\omega)$ without any gap between successive cycles, as depicted in Fig. 3.12a. To recover $x(t)$ from $x_{\tilde{\delta}}(t)$, we need to pass the sampled signal $x_{\tilde{\delta}}(t)$ through an ideal lowpass filter, shown shaded in Fig. 3.11b. As seen in Sec. 2.3, such a filter is unrealizable; it can be approximated with ever-decreasing error only with ever-increasing time delay in the response. In other words, we require infinite time delay to recover the signal $x(t)$ from its samples. A practical solution to this problem is to sample the signal at a rate higher than the Nyquist rate ($F_s > 2B$). The resulting spectrum $X_{\tilde{\delta}}(\omega)$ consists of repetitions of $X(\omega)$ that are spaced with a finite gap between successive cycles, as illustrated in Fig. 3.12b. Thanks to the spectral gaps, we can now recover $X(\omega)$ from $X_{\tilde{\delta}}(\omega)$ using a lowpass filter with a gradual transition between passband and stopband. But even in this case, the filter gain must be zero for frequencies above $F_s - B$ in order to suppress unwanted spectrum. According to the Paley-Wiener criterion (Eq. (2.19)), it is impossible to realize even this filter. The only advantage in this case is that the required filter can be closely approximated with a smaller time delay. All this means that it is impossible in practice to recover a bandlimited signal $x(t)$ exactly from its samples, even if the sampling rate is higher than the Nyquist rate. However, as the sampling rate increases, the recovered signal approaches the desired signal more closely.

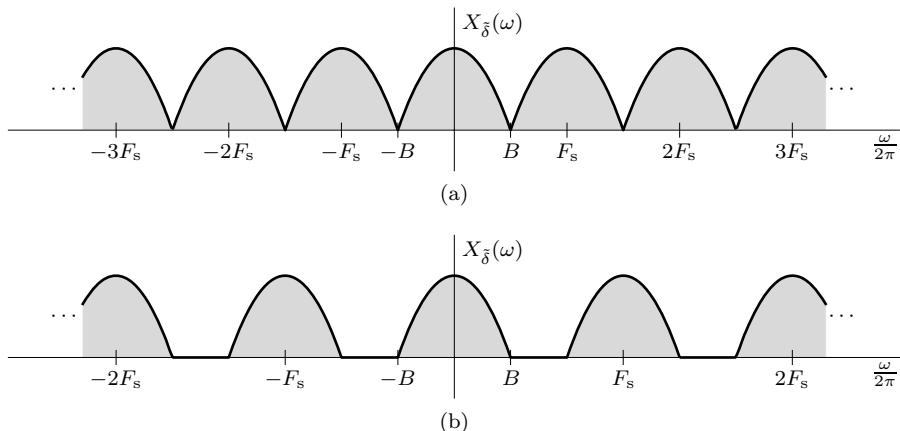


Figure 3.12: Sampled signal spectra: (a) Nyquist rate and (b) above Nyquist rate.

The Treachery of Aliasing

There is another fundamental, practical difficulty in reconstructing a signal from its samples. The sampling theorem assumes that the signal $x(t)$ is bandlimited. However, *all practical signals are timelimited*, meaning that they are of finite duration or width. We can demonstrate (see Prob. 3.2-17) that a signal cannot be timelimited and bandlimited simultaneously. If a signal is timelimited, it cannot be bandlimited, and vice versa, although it can be simultaneously non-timelimited and non-bandlimited. Clearly, all practical signals, being timelimited, possess infinite bandwidth, as shown in Fig. 3.13a. The sampled signal's spectrum $X_{\tilde{\delta}}(\omega)$ consists of overlapping cycles of $X(\omega)$ repeating every F_s

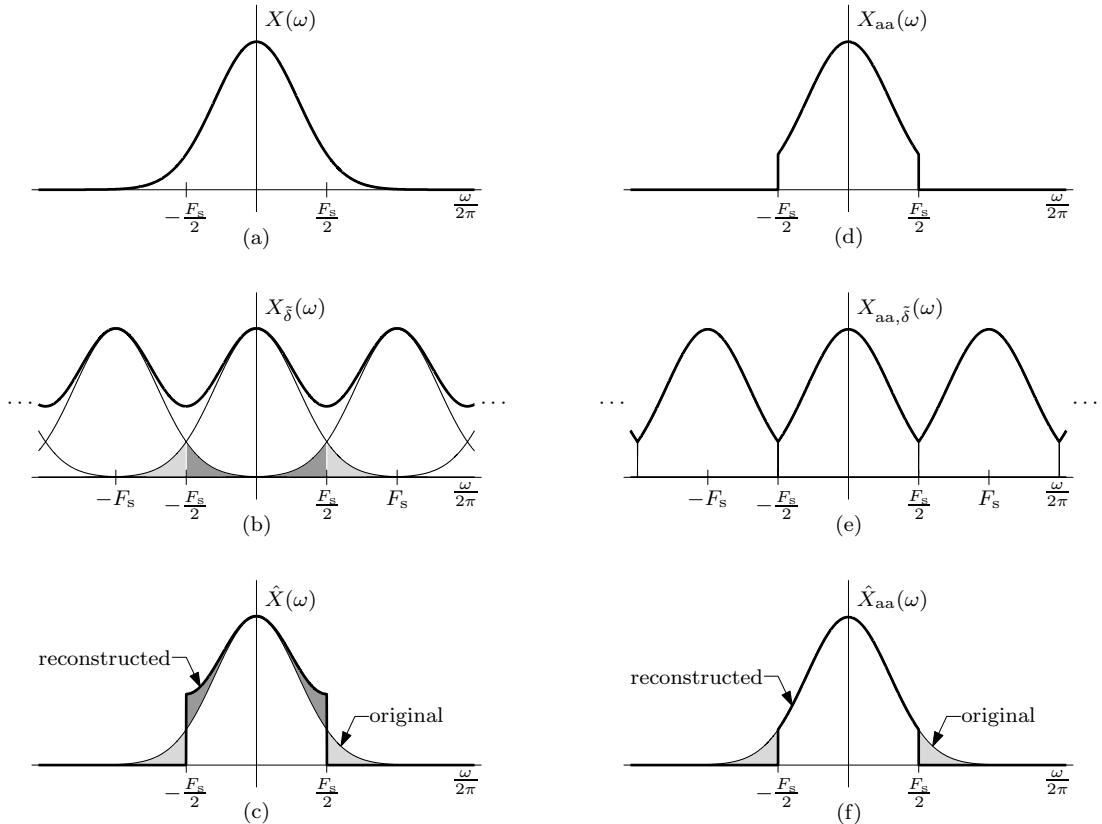


Figure 3.13: Frequency-domain demonstration of the aliasing effect: (a)–(c) original signal, sampled, and reconstructed, and (d)–(f) filtered signal, sampled, and reconstructed.

Because of the infinite bandwidth of $X(\omega)$, spectral overlap in $X_{\tilde{\delta}}(\omega)$ is unavoidable, regardless of the sampling rate. Sampling at a higher rate reduces but does not eliminate the overlap between repeating spectral cycles. Figure 3.13b shows a somewhat simplified picture where from the infinite number of repeating cycles, only the neighboring spectral cycles significantly overlap. All practical spectra must decay at higher frequencies, so such a situation is not uncommon. When broader overlap exists that produces interference from cycles other than immediate neighbors, the picture is a little more complicated. We treat this situation later with a discussion of multiple folding.

Because of the overlapping tails, $X_{\tilde{\delta}}(\omega)$ no longer has complete information about $X(\omega)$, and it is no longer possible, even theoretically, to recover $x(t)$ exactly from the sampled signal $x_{\tilde{\delta}}(t)$. If the sampled signal is passed through an ideal lowpass reconstruction filter with cutoff frequency $F_s/2$ Hz, the output $\hat{X}(\omega)$ does not equal $X(\omega)$, as shown in Fig. 3.13c. Instead, $\hat{X}(\omega)$ is distorted from

$X(\omega)$ in two distinct ways:

1. The tail of $X(\omega)$ beyond $|f| > F_s/2$ Hz, shown shaded light gray in Fig. 3.13, is lost. This results in the loss of high-frequency ($|f| > F_s/2$) signal content.
2. The lost tail reappears with lower frequency, shown shaded dark gray in Fig. 3.13. For the spectra of real signals, the tail appears to fold or invert about $F_s/2$. Thus, a component of frequency $F_s/2 + f_1$ shows up as or impersonates a component of lower frequency $F_s/2 - f_1$ in the reconstructed signal. This causes distortion of low-frequency ($|f| < F_s/2$) signal content.

The frequency $F_s/2$ is called the *folding frequency*, and the tail inversion is known as *spectral folding* or *aliasing*. When aliasing occurs, we not only lose components above the folding frequency, but these very components also reappear (alias) at lower frequencies.

The aliasing problem is analogous to that of an army with a disgruntled platoon that has defected to the enemy side. The army is in double jeopardy. First, the army loses this platoon as a fighting force. In addition, during actual fighting, the army must contend with the sabotage of the defectors; by using a loyal platoon to neutralize the defectors, the army has in fact lost two platoons in nonproductive activity.

Defectors Eliminated: The Anti-Aliasing Filter

To the commander of the betrayed army, the solution to this problem is obvious. As soon as the commander gets wind of the defection, he incapacitates, by whatever means, the defecting platoon *before the fighting begins*. This way he loses only one (the defecting) platoon. This is a partial solution to the double jeopardy of betrayal and sabotage, a solution that partly rectifies the problem and cuts the losses in half.

We follow exactly the same procedure. The potential defectors are all the frequency components beyond the folding frequency $F_s/2$ Hz. We must eliminate (suppress) these components from $x(t)$ *before sampling* $x(t)$. An ideal unit-gain lowpass *anti-aliasing filter* with a cutoff of $F_s/2$ Hz is just what is needed. As shown in the block representation of Fig. 3.14, we emphasize that the anti-aliasing operation must be performed *before the signal is sampled*. Within a scale factor, the ideal anti-aliasing filter is identical to the ideal reconstruction filter of Fig. 3.9. As shown in Fig. 3.13d, the anti-aliasing filter essentially bandlimits its input, excising potential defectors from the ranks. In this way, we only lose the components beyond the folding frequency $F_s/2$ Hz. These suppressed components cannot reappear to corrupt components below the folding frequency.

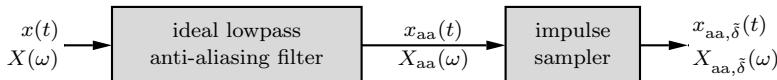


Figure 3.14: Anti-aliasing filters precede sampling.

Figures 3.13e and 3.13f show the sampled signal spectrum $X_{aa,\tilde{\delta}}(\omega)$ and the reconstructed signal spectrum $\hat{X}_{aa}(\omega)$ when an anti-aliasing scheme is used. The troublesome tails are handily eliminated, and the reconstructed signal spectrum $\hat{X}_{aa}(\omega)$ equals the original signal spectrum $X(\omega)$ for frequencies $|f| < F_s/2$. Thus, although we lose the spectrum beyond $F_s/2$ Hz (shown shaded in Fig. 3.13f), the spectrum for all the frequencies below $F_s/2$ remains intact. The anti-aliasing filter cuts aliasing-related distortion in half.

Anti-aliasing filters also help to reduce noise. Noise, generally, has a wideband spectrum. Without an anti-aliasing filter, high-frequency noise components alias into the signal band, causing unwanted and unnecessary distortions. An anti-aliasing filter suppresses the noise spectrum beyond frequency $F_s/2$, thereby improving the signal-to-noise ratio.

Ideal anti-aliasing filters are unrealizable. In practice, we use filters with steep cutoffs that sharply attenuate the spectrum beyond the folding frequency. An increasingly popular alternative

is to first use a low-order anti-aliasing filter with relatively poor performance, next oversample the signal, and then apply a highly selective digital filter prior to further processing; these ideas are developed in later chapters.

▷ Drill 3.4 (Graphical Verification of Aliasing in Sinusoids)

By plotting the appropriate spectra, verify that samples of $x(t) = \cos[(\omega_s/2 + \omega_1)t]$, where $0 \leq \omega_1 < \omega_s/2$, are equivalent to samples of $y(t) = \cos[(\omega_s/2 - \omega_1)t]$. Is the result also true for $\omega_1 > \omega_s/2$? Explain.

△

Sampling Causes Non-Bandlimited Signals to Appear Bandlimited

Figure 3.13b shows the spectrum of samples of signal $x(t)$ consists of overlapping copies of $X(\omega)$. This means $x_{\tilde{\delta}}(t)$ are sub-Nyquist samples of $x(t)$. However, we may also view the spectrum in Fig. 3.13b as the spectrum $\hat{X}(\omega)$ (Fig. 3.13c) repeating periodically every F_s Hz without overlap. The spectrum $\hat{X}(\omega)$ is bandlimited to $F_s/2$ Hz. Hence, the samples $x_{\tilde{\delta}}(t)$, which are sub-Nyquist samples of $x(t)$, are Nyquist samples for the signal $\hat{x}(t)$. In other words, sampling a non-bandlimited signal $x(t)$ at a rate F_s Hz makes the samples appear as if they are the Nyquist samples of a bandlimited signal $\hat{x}(t)$ with bandwidth $F_s/2$ Hz. A similar conclusion also applies if $x(t)$ is bandlimited but sampled at a sub-Nyquist rate. The frequency band from $-F_s/2$ to $F_s/2$ is called the *fundamental band*.

Endless Curses of Aliasing: Multiple Overlap, Multiple Folding

The picture in Fig. 3.13b does not fully illustrate the woes caused by aliasing. We deliberately simplified the picture to avoid confusion. Figure 3.15 provides a more accurate picture. As Eq. (3.6) requires, the sampled signal's spectrum $X_{\tilde{\delta}}(\omega)$ (light solid line) is a sum of the scaled original signal's spectrum $\frac{1}{T}X(\omega)$ replicated every F_s Hz (light dashed lines). Because the original signal is non-bandlimited, each replication of $X(\omega)$ overlaps with every other. In this case, however, the sampling rate is significantly sub-Nyquist, which results in significant and extended overlap between the replicated spectra. Unlike Fig. 3.13b, at any given frequency, more than two overlapping pieces are significant to $X_{\tilde{\delta}}(\omega)$. To appreciate the true dimensions of the aliasing problem, we need to take account of multiple overlap. The curse of aliasing is a kind of mini *eternal damnation* for the *original sin* of undersampling or not using an anti-aliasing filter. Had we used an anti-aliasing filter, we would have been saved.

When sampling time-domain signals that are real, there is another way to picture how the spectrum $X_{\tilde{\delta}}(\omega)$ is constructed. In such cases, first note that we only need to know $X_{\tilde{\delta}}(\omega)$ over the positive fundamental band, shown shaded in Fig. 3.15. Everything else is known due to required symmetry and periodicity. Over this interval, we obtain $X_{\tilde{\delta}}(\omega)$ from a single copy of $\frac{1}{T}X(\omega)$, folded repeatedly at intervals of $F_s/2$. To visualize the process, consider $\frac{1}{T}X(\omega)$ printed for $\omega \geq 0$ on a transparent piece of paper. Next, we fold this paper accordion style along the frequency axis, making folds every $F_s/2$. The first aliased segment follows the first fold, the second aliased segment follows the second fold, and so forth. Because $\frac{1}{T}X(\omega)$, as well as all practical spectra, extends to infinity, the process continues ad infinitum, repeatedly folding every $F_s/2$. The end result is shown in the shaded region of Fig. 3.15 using a heavy solid line. To obtain $X_{\tilde{\delta}}(\omega)$, we add together all the segments.

The multiple folding process of Fig. 3.15 is clumsier than using Eq. (3.6) to compute $X_{\tilde{\delta}}(\omega)$. Primarily, it provides a useful way to conceptualize and visualize aliasing. The folding analogy shown in Fig. 3.15, it should be stressed, does not work properly on the spectra of complex time-domain signals, which lack the necessary frequency-domain symmetry; such cases are better treated using the standard superposition of overlapping replications. Further, Fig. 3.15 gives a simplified treatment of multiple folding on the assumption that, in addition to $x(t)$, the spectrum $X(\omega)$ is

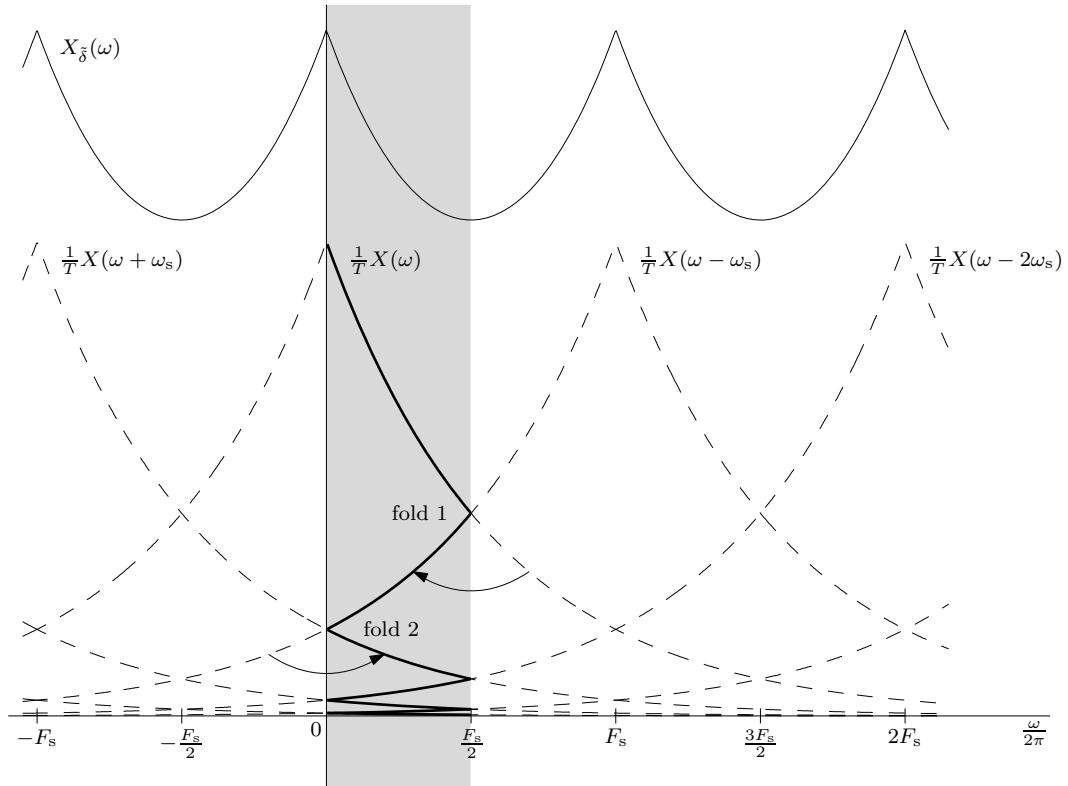


Figure 3.15: Viewing aliasing through multiple folding.

real. If $x(t)$ is real but $X(\omega)$ is not, as is most commonly the case, the segments following any odd-numbered fold of $\frac{1}{T}X(\omega)$ require an additional sign change in the phase spectrum.[†] To understand these phase reversals, notice in Fig. 3.15 that the segments following odd-numbered folds actually come from the *negative-frequency portions* of the replicates centered at F_s , $2F_s$, and so on. Thus, since $\angle X(-\omega) = -\angle X(\omega)$, we must accordingly update the phase of these segments, which are being folded from the *positive-frequency portion* of $\frac{1}{T}X(\omega)$.

Identity Fraud

What multiple overlap (or folding) really means is that every spectral component in $X(\omega)$ outside $F_s/2$ Hz gets aliased somewhere in the fundamental frequency band. It is as if when we sample a signal at F_s Hz, the spectral components beyond $F_s/2$ Hz are made outlaws and exiled from the homeland of $-F_s/2$ to $F_s/2$ Hz. Unfortunately, every one of these outlaws sneaks back into the country by assuming an identity of one of the law-abiding citizens. No resident is immune from this identity fraud, and every victim pays a heavy price: their true identity is forever lost.

There exist, however, limits to this fraud. The spectrum of any practical signal must decay with frequency. Hence, segments aliased from higher frequencies become negligible in effect. Much depends on how fast the spectrum $X(\omega)$ decays with frequency. If the spectrum decays quickly, only one aliased segment may be of significance. In contrast, for a slowly decaying spectrum, several aliased segments may need to be considered.

We should also note that noise accompanies most signals. The most common form of noise encountered in practice is so-called white noise, which has a very wide-band spectrum. Hence,

[†]Careful observers will notice that these segments correspond to the shaded bands of Fig. 3.17c.

aliasing can have serious consequences, even when the interfering noise is small. It is therefore crucial to use an anti-aliasing filter, which, by cutting off the tail beyond $F_s/2$, wipes out the entire gang of outlaws.

Although we cannot stop the imposters completely, we can, with proper choice of sampling rate and use of anti-aliasing filters, render the damage they do inconsequential. When proper precautions are taken, most practical cases involve little or no significant aliasing effects. As we shall see in Sec. 3.5, the problem of spectral aliasing has a time-domain dual, which results when spectral sampling is performed at an inadequate rate.

▷ Drill 3.5 (Anti-Aliasing Filter Type Is Important)

Ideally, an anti-aliasing filter has a sharp transition at $F_s/2$. For anti-aliasing applications, explain why, despite superior transition band characteristics, an elliptic filter, particularly one with even order, may be inferior to a Butterworth filter of the same order.

△

3.3.1 Aliasing in Sinusoids

Let us now focus our attention on aliasing in sinusoids. Figure 3.16 shows how, when sampled at the same rate, sinusoids of two different frequencies can generate identical sets of samples. Both sinusoids are sampled at a rate $F_s = 10$ Hz ($T = 0.1$ s). The frequencies of the two sinusoids, 1 Hz (period 1 s) and 11 Hz (period 1/11 s), differ by $F_s = 10$ Hz. In fact, a frequency difference equal to any integer multiple of F_s produces a similar result.

The root cause of aliasing, as illustrated in Fig. 3.16, is the sampling rate, which is adequate for the lower frequency sinusoid but is clearly inadequate for the higher frequency sinusoid. The figure visibly shows that between the successive samples of the higher frequency sinusoid, entire cycles are bypassed or ignored, which is an unmistakable indication that the sampling rate is sub-Nyquist (too low). Clearly, the samples do not properly represent the 11-Hz signal. The *apparent frequency* of the samples, which always lies in the fundamental band, is equal to 1 Hz in this example. If these samples are used to reconstruct a signal using a lowpass filter of bandwidth $F_s/2$, we shall obtain a sinusoid of frequency 1 Hz.

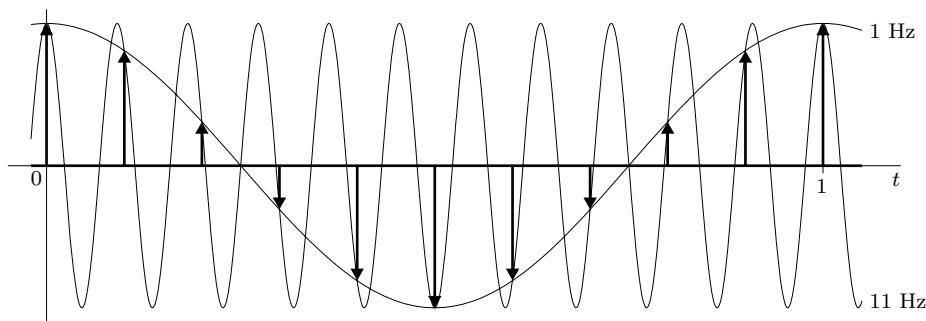


Figure 3.16: Demonstration of aliasing.

A Fascinating Picture

We have seen earlier that sampling any signal at frequency F_s Hz causes it to appear as a signal that is bandlimited to $F_s/2$ Hz. This means that if we sample a sinusoid of frequency f_0 Hz at a rate F_s Hz, then the resulting samples will appear as samples of a sinusoid with frequency no greater than $F_s/2$ Hz. What happens if we increase the frequency f_0 continuously without limit? Regardless how

high f_0 becomes, the resulting sampled sinusoid cannot have frequency greater than $F_s/2$ Hz. As we shall see, the sampling process produces a rather fascinating picture as we increase f_0 continuously without limit.

Keeping Up Appearances

The fundamental band is like an exclusive VIP club for signals, and sampled sinusoids are intent on keeping up appearances that they belong. Regardless of the underlying CT signal's true frequency, the apparent frequency of any sampled sinusoid is always within the fundamental band. Let us now demonstrate this fact. To begin, we show that samples of a sinusoid of frequency f_0 Hz are identical to those of a sinusoid of frequency $f_0 + mF_s$ Hz (integer m). The samples of $\cos[2\pi(f_0 + mF_s)t + \theta]$ are

$$\cos[2\pi(f_0 + mF_s)nT + \theta] = \cos(2\pi f_0 nT + \theta + 2\pi mn) = \cos(2\pi f_0 nT + \theta).$$

The result follows because mn is an integer and $F_s T = 1$. Thus, sinusoids of frequencies that differ by an integer multiple of F_s result in the same set of samples. Conversely, the samples of sinusoids in any frequency band of width F_s Hz are unique; that is, no two sinusoids in that band have the same samples (when sampled at a rate F_s Hz). For instance, frequencies in the fundamental band from $-F_s/2$ to $F_s/2$ have unique samples.[†]

From the preceding discussion, we conclude that if a continuous-time sinusoid of frequency f_0 Hz is sampled at a rate of F_s Hz, then aliasing causes the resulting samples to appear as samples of a continuous-time sinusoid with *apparent frequency* f_a that is in the fundamental band ($-F_s/2 \leq f_a \leq F_s/2$). While apparent frequency can be found by adding or subtracting integer multiples of F_s to f_0 until the fundamental band is reached, it is easier to compute f_a according to (see Prob. 3.2-10)

$$f_a = \langle f_0 + F_s/2 \rangle_{F_s} - F_s/2, \quad (3.14)$$

where $\langle a \rangle_b$ is the *modulo operation* a modulo b . Figure 3.17a plots apparent frequency f_a versus the actual frequency f_0 of the input sinusoid.

Especially for real signals, the difference between positive and negative frequencies is often unimportant. For example, a sinusoid of frequency f_0 oscillates at the same rate as a sinusoid of frequency $-f_0$; they both *appear* to have frequency f_0 . Thus, apparent frequency is most often reported as $|f_a|$, computed by taking the absolute value of Eq. (3.14). A plot of $|f_a|$ versus sinusoid frequency f_0 is shown in Fig. 3.17b.

In fact, as is commonly done, it is possible to express real sinusoids exclusively in terms of positive apparent frequency $|f_a|$ with a possible phase reversal. This is because $\cos(-\omega_0 t + \theta) = \cos(\omega_0 t - \theta)$. The sign of θ changes if f_a is negative.[‡] Consider, for example, the sinusoid $\cos(2\pi 8000t + \theta)$ sampled at a rate $F_s = 3000$ Hz. Using Eq. (3.14), we obtain $f_a = \langle 8000 + 1500 \rangle_{3000} - 1500 = -1000$. Hence, $|f_a| = 1000$, and the samples appear as if they come from $\cos(2\pi 1000t - \theta)$. Observe the sign change of the phase because f_a is negative. The frequency intervals where such phase changes occur are shown shaded in Fig. 3.17b.

In light of our discussions of apparent frequency, let us consider a sampled sinusoid of frequency $f_0 = F_s/2 + f_1$, where $0 \leq f_1 < F_s/2$. According to Eq. (3.14),

$$f_a = \langle F_s/2 + f_1 + F_s/2 \rangle_{F_s} - F_s/2 = f_1 - F_s/2.$$

Since $f_1 < F_s/2$, the absolute apparent frequency is $|f_a| = F_s/2 - f_1$, which confirms our earlier observations of frequency folding. Notice, however, the phase of the sinusoid will change sign because f_a is negative.

[†]There is one exception. Consider $\cos(\omega t + \theta)$ and $\cos(-\omega t + \theta)$, where $|\omega| < \pi$. Although the frequencies ω and $-\omega$ are different, the samples are identical if θ is zero. In the more general case that θ is nonzero, however, the samples are different, as expected of two distinct frequencies.

[‡]See Drill 3.7 for additional insights about this rule.

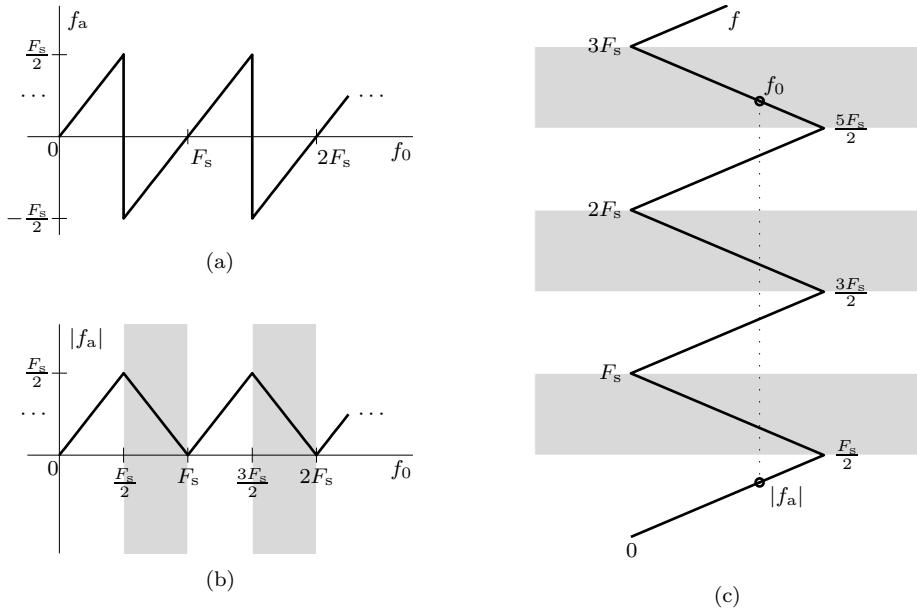


Figure 3.17: Apparent frequency of a sampled sinusoid of frequency f_0 : (a) f_a , (b) $|f_a|$, and (c) multiple folding to obtain $|f_a|$.

▷ Drill 3.6 (Investigating Equivalence of Aliased Sinusoids)

Show that samples of 90-Hz and 110-Hz sinusoids of the form $\cos(2\pi f_0 t)$ are identical when sampled at a rate of 200 Hz. Are the samples identical if the sinusoids are of the more general form $\cos(2\pi f_0 t + \theta)$? □

▷ Drill 3.7 (An Alternate Form for Apparent Sinusoids)

For $f_a < 0$, the apparent sinusoid of signal $\cos(2\pi f_0 t + \theta)$ is $\cos(2\pi|f_a|t - \theta)$, which includes a sign change of the phase term θ . If instead the form of our sinusoidal signal is $\sin(2\pi f_0 t + \theta)$, show that when $f_a < 0$, the apparent sinusoid is $-\sin(2\pi|f_a|t - \theta)$, which includes a sign change in amplitude as well as phase. □

Multiple Folding to Determine Apparent Frequency

Another instructive way of viewing the mapping of an undersampled frequency to an apparent frequency is through multiple folding. Mark a narrow strip of paper like a tape measure using units of frequency rather than units of length. Next, fold this tape accordion fashion, as shown in Fig. 3.17c, making folds every integer multiple of the folding frequency $F_s/2$. The frequency $|f_a|$ corresponding to frequency f_0 marked on the tape is found using a simple projection, as shown with the dotted line in Fig. 3.17c. If f_0 most recently follows an odd-numbered fold (shown shaded), then f_a is negative; this is the case shown in Fig. 3.17c, where f_0 follows the fifth fold. If f_0 follows an even-numbered fold, f_a is positive. It is no coincidence that Fig. 3.17c bears such a striking resemblance to Fig. 3.17b; they really are one and the same. Still, the multiple folding process (Fig. 3.17c) is clumsier than using Eq. (3.14) to compute $|f_a|$ (or f_a). Its primary virtue lies in the conceptual understanding of the frequency folding phenomenon, to be discussed soon.

▷ **Example 3.5 (Determining Apparent Frequency)**

A continuous-time sinusoid $\cos(2\pi f_0 t + \theta)$ is sampled at a rate $F_s = 1000$ Hz. Determine the apparent (aliased) sinusoid of the resulting samples if the input signal frequency f_0 is (a) 400 Hz, (b) 600 Hz, (c) 1000 Hz, (d) 2400 Hz, and (e) -3300 Hz

The folding frequency is $F_s/2 = 500$. If $|f_0| > 500$ Hz, then the sinusoid's frequency lies outside the fundamental band, and aliasing occurs during sampling. Otherwise, $|f_0| < 500$ Hz, and no aliasing occurs. Although we compute the results more or less analytically, graphical methods, such as demonstrated in Fig. 3.17, produce identical results.

(a) $f_0 = 400$ Hz

Here, $|f_0| < 500$ Hz, and there is no aliasing. The apparent sinusoid is $\cos(2\pi 400t + \theta)$ with $f_a = f_0 = 400$.

(b) $f_0 = 600$ Hz

Since $|f_0| > 500$ Hz, aliasing occurs, and $f_a = \langle 600 + 500 \rangle_{1000} - 500 = -400$ Hz. Hence, the apparent sinusoid is $\cos(-2\pi 400t + \theta)$ or, equivalently, $\cos(2\pi 400t - \theta)$. The latter result follows from basic trigonometry or the observation that since $f_a < 0$, the phase sign changes when expressing the apparent sinusoid in terms of $|f_a|$.

(c) $f_0 = 1000$ Hz

Aliasing again occurs, and $f_a = \langle 1000 + 500 \rangle_{1000} - 500 = 0$ Hz. Hence, the aliased frequency is 0 Hz (dc), and no phase sign change. The apparent sinusoid is $\cos(2\pi 0t + \theta) = \cos(\theta)$. In this case, all sample values are constant.

(d) $f_0 = 2400$ Hz

Aliasing occurs, and $f_a = \langle 2400 + 500 \rangle_{1000} - 500 = 400$ Hz. Since f_a is positive, there is no sign change for the phase and the apparent sinusoid is $\cos(2\pi 400t + \theta)$.

(e) $f_0 = -3300$ Hz

Since $|f_0| > 500$ Hz, aliasing occurs, and $f_a = \langle -3300 + 500 \rangle_{1000} - 500 = -300$ Hz. Hence, the apparent sinusoid is $\cos(-2\pi 300t + \theta)$ or, equivalently, $\cos(2\pi 300t - \theta)$.

Example 3.5 ◁

▷ **Drill 3.8 (Computing Apparent Frequency)**

A sinusoid of frequency f_0 Hz is sampled at a rate of $F_s = 100$ Hz. Determine the apparent frequency f_a as well as $|f_a|$ of the samples if f_0 is (a) 40 Hz, (b) 60 Hz, (c) 140 Hz, and (d) 160 Hz.

◀

3.4 Sampling of Bandpass Signals

A typical bandpass signal spectrum $X(\omega)$ is shown in Fig. 3.18a, where the negative-frequency segment is shown shaded. For convenience, we plot the spectra as functions of $\frac{\omega}{2\pi}$, which has units of Hz. Let the passband have lower and upper frequency limits of f_1 and f_2 , respectively, so that the signal bandwidth is $B = f_2 - f_1$ Hz. The highest frequency in the spectrum is f_2 , and according to the earlier development, we should need a minimum sampling rate $2f_2$ Hz. Intuitively, however,

we feel we are overpaying the price. The actual signal bandwidth is only B Hz. Hence, should we not expect the minimum sampling rate to be $2B$ rather than $2f_2$ Hz?

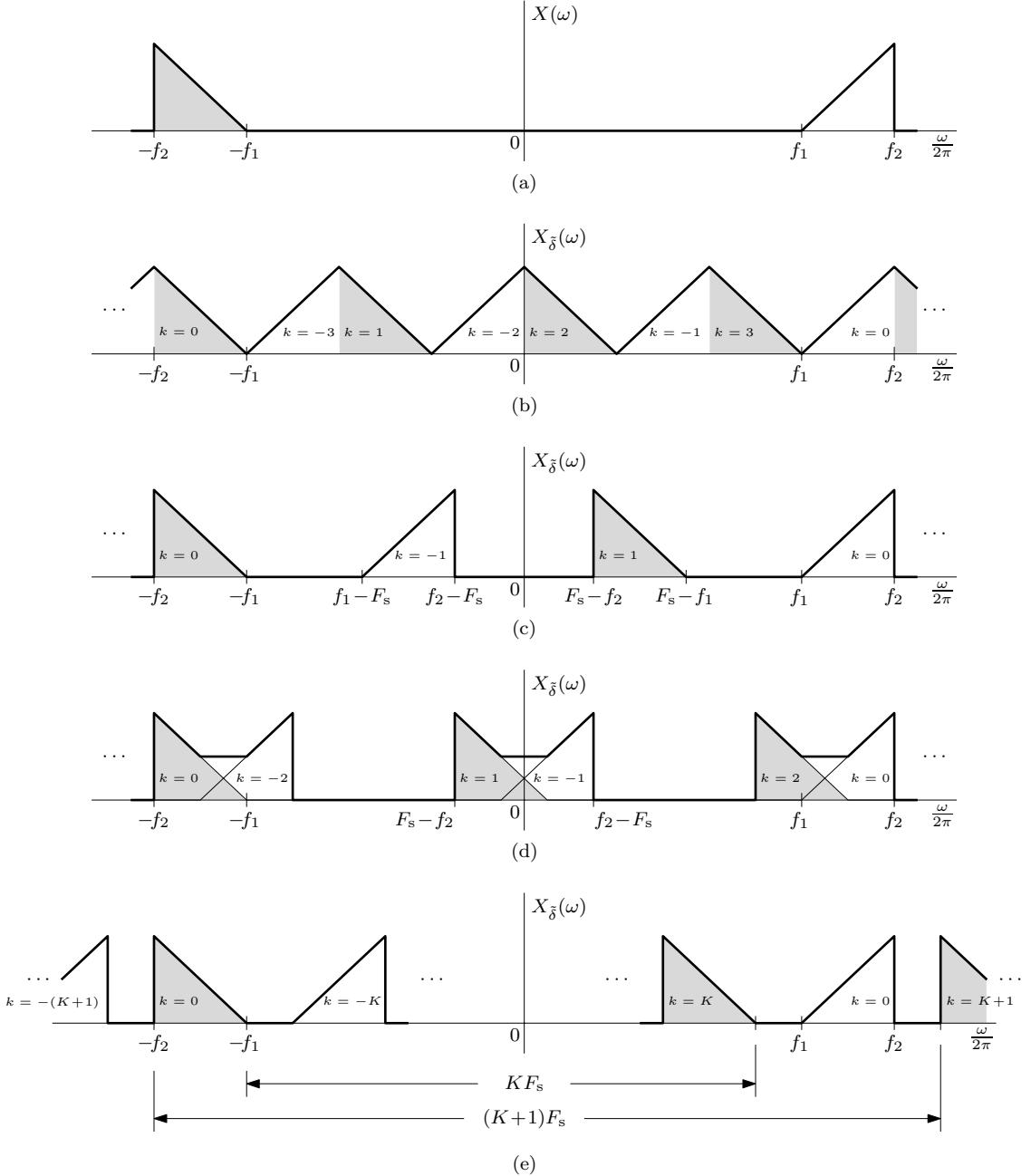


Figure 3.18: Sampling a bandwidth B bandpass signal: (a) original spectrum, (b) $f_2 = 4B$ and $F_s = 2B$, (c) $f_2 = 4B$ and $F_s = 4.75B$, (d) $f_2 = 4B$ and $F_s = 3.25B$, and (e) general sampling case.

Our hunch is correct. We can show that it is possible to reconstruct a bandpass signal $x(t)$ from its samples taken at a rate $2B$ Hz [1,2]. However, such a scheme involves more complex *second-order sampling* that uses two *interlaced* sampling trains, each at a rate of B samples per second. The sampling is taken in pairs, and consequently, the samples are *nonuniformly* spaced. However, even

using our usual uniform sampling, termed *first-order sampling*, we shall show that it is possible to get by with a rate equal to or slightly higher than $2B$ Hz. In fact, if f_2 is an integral multiple of B , then we need a sampling rate of $F_s = 2B$ Hz exactly.

When uniform samples are taken at a rate F_s Hz, the sampled spectrum consists of the original spectrum (scaled by $1/T$) replicated every F_s Hz, as given in Eq. (3.6). We can treat this replication process separately for the positive- and negative-frequency portions of $X(\omega)$. We replicate the positive-frequency portion (unshaded) first, spacing each copy at intervals F_s from the original. In a similar way, we then replicate the negative-frequency portion (shaded). Consistent with Eq. (3.6), we use k to identify the replicates: $k = 0$ designates the original-position spectrum, $k = \pm 1$ corresponds to copies closest to the original ($\pm F_s$ away), and so forth.

In order to gain intuitive understanding of the process, let us first consider a special (and simpler) case where f_2 is an integer multiple of the bandwidth B . Let $f_2 = 4B$. If we choose the sampling rate $F_s = 2B$, then within the frequency interval $-f_2$ to f_2 , the sampled spectrum $X_{\delta}(\omega)$ shows eight alternating back to back segments without any overlap and without any idle bands, as shown in Fig. 3.18b. Just as night must follow day, Eq. (3.6) ensures that negative-frequency copies (shaded) alternate with positive-frequency copies (unshaded). With $F_s = 2B$ there is no overlap between repetitions, and the original bandpass signal $x(t)$ can be reconstructed by passing the sampled signal through a B -Hz passband centered between f_1 and f_2 .

We can fit fewer than eight segments by choosing a higher repetition (sampling) frequency F_s . In such cases, there will be idle bands, and the repetitions will not be back to back. The case $f_2 = 4B$ and $F_s = 4.75B$, shown in Fig. 3.18c, results in four nonoverlapping pieces. Since there is no overlap, the original bandpass signal $x(t)$ can be recovered, more easily, in fact, than the case of Fig. 3.18b, where there are no idle bands to allow practical filter characteristics. The case $f_2 = 4B$ and $F_s = 3.25B$, shown in Fig. 3.18d, causes overlap, and the original bandpass signal $x(t)$ cannot be recovered.

As Fig. 3.18 illustrates, different sub-Nyquist sampling rates can produce quite different results. We see for this $f_2 = 4B$ case, for example, that increasing the sampling rate from $F_s = 2B$ (Fig. 3.18b) to $F_s = 3.25B$ (Fig. 3.18d) actually *reduces* performance and makes recovery of $x(t)$ impossible. By further increasing the sampling rate to $F_s = 4.75B$, we can once again recover $x(t)$. We would like to better understand this behavior.

Figure 3.18e shows a portion of the spectrum of $X_{\delta}(\omega)$ for the general case where f_2 is not required to be an integer multiple of B . In this general case, we cannot typically fit an integer number of repetitions that are back to back. There are necessarily idle bands, and replicates may or may not overlap. If there is no overlap between segments, as shown in Fig. 3.18e, then we can reconstruct the signal $x(t)$ from its samples. Counting each shaded and unshaded pair as one replicate of $X(\omega)$, K represents the number of replicates found within the band from $-f_1$ to f_1 . Including the original copy ($k = 0$), there are $K + 1$ copies within the band from $-f_2$ to f_2 . From Fig. 3.18e, it follows that to be able to reconstruct $x(t)$, that is, to avoid spectral overlap, we must have[†]

$$KF_s < 2f_1 \quad \text{and} \quad (K + 1)F_s > 2f_2.$$

Substituting $f_1 = f_2 - B$ and combining the preceding equations yield

$$\frac{2}{K + 1} \left(\frac{f_2}{B} \right) < \frac{F_s}{B} < \frac{2}{K} \left(\frac{f_2}{B} - 1 \right). \quad (3.15)$$

Equation (3.15) specifies the upper and lower bounds of the sampling frequency F_s for a given value of the integer K . Clearly, the maximum value of $K + 1$ is the largest integer less than f_2/B . Thus,

$$K_{\max} = \left\lfloor \frac{f_2}{B} \right\rfloor - 1. \quad (3.16)$$

[†]If there are no impulses at f_1 and f_2 , we can use \leq rather than $<$.

Using Eq. (3.15), Fig. 3.19 plots the upper and lower bounds of F_s/B versus f_2/B for various K . The shaded area is a forbidden region that, if entered, causes spectral overlap and makes recovery of $x(t)$ impossible. Conversely, the unshaded regions satisfy both inequalities in Eq. (3.15). If the sampling rate is chosen to lie in an unshaded region, then overlap is avoided, and $x(t)$ is recoverable from its samples. The $K = 0$ region, which has no replicates in the band 0 to f_2 other than the original spectrum, corresponds to sampling frequencies that are greater than the Nyquist rate of $2f_2$. All other unshaded regions, such as the $K = 1$ and $K = 2$ regions, correspond to sub-Nyquist sampling rates where recovery of $x(t)$ is still possible. To realize the lowest possible sampling rate, we must select K as large as possible. Notice that the sampling rate can never fall below the minimum rate of $F_s = 2B$, which is represented by the horizontal line $F_s/B = 2$. This minimum rate is achievable only when f_2 is an integer multiple of B . In all other cases, the sampling rate is higher than this minimum rate $2B$ Hz. As f_2/B becomes large, however, our minimum achievable sampling rate tends toward the minimum of $2B$.

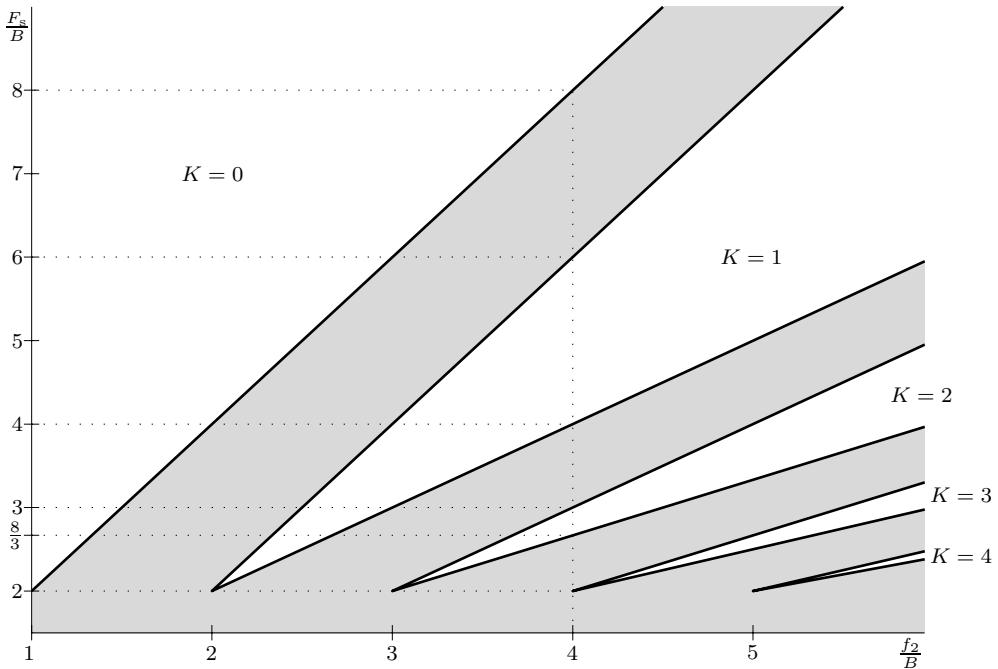


Figure 3.19: Permissible sampling rates (unshaded) for bandpass signals.

Let us reconsider the Fig. 3.18 case of sampling a bandpass signal with $f_2 = 4B$. Nyquist suggests that we sample this signal at a rate $F_s > 2f_2 = 8B$, which is part of the $K = 0$ region of Fig. 3.19. Looking above $f_2/B = 4$ in Fig. 3.19 (dotted lines), however, we see that several sub-Nyquist sampling rates are also permissible. Figure 3.18b, for example, uses the minimum possible sampling rate, which corresponds to the extreme corner of the $K = 3$ region. Similarly, Fig. 3.18c, with $F_s/B = 4.75$, is comfortably within the acceptable $K = 1$ region of $4 < F_s/B < 6$. Figure 3.18d, on the other hand, attempts $K = 2$ by using $F_s/B = 3.25$, which falls within the shaded forbidden region. Consequently, the replicates overlap, and $x(t)$ cannot be recovered from samples taken at this rate. To successfully achieve the $K = 2$ condition requires $8/3 < F_s/B < 3$.

It is generally not good practice to set the sampling rate at the lowest permissible value because slight deviations, which occur normally in practical equipment, may push the sampling rate into a forbidden zone and cause overlap. A better choice is to select a value halfway between the forbidden zone boundaries for a given K . For instance, when $f_2 = 2.7B$, $K_{\max} = 1$, and the lower and upper bounds on the sampling rate, as determined from Eq. (3.15) or Fig. 3.19, are $2.7B$ and $3.4B$. A good

choice would be $F_s = 3.05B$, which is about 13% above the lowest permissible rate of $2.7B$ Hz and about 44% below the Nyquist rate of $5.4B$ Hz. For the lowest rate of $2.7B$ Hz, any small downward drift in the sampling rate would cause spectral overlap, making signal recovery impossible.

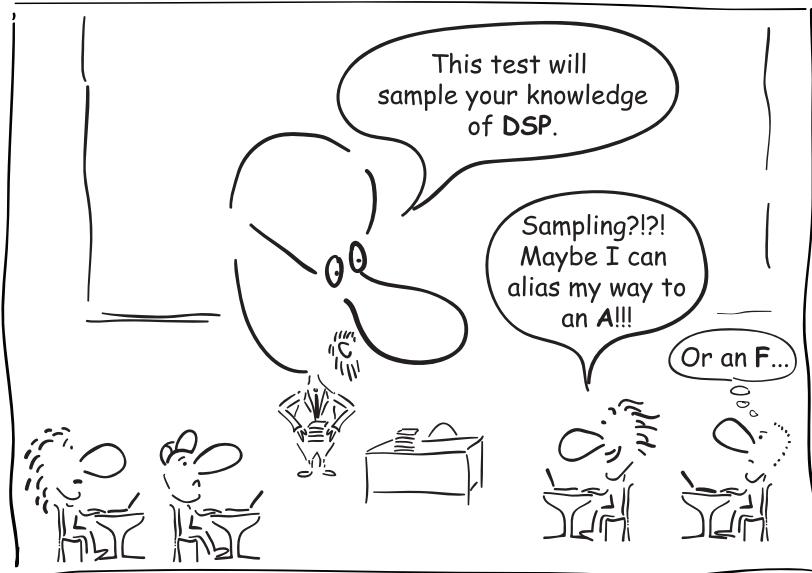
▷ Example 3.6 (Permissible Sampling Rates of Bandpass Signals)

Determine the permissible range of the uniform sampling rate for a bandpass signal with $f_1 = 15$ kHz and $f_2 = 25$ kHz.

In this case, $B = 25 - 15 = 10$ kHz, and $f_2/B = 25/10 = 2.5$. Using Eq. (3.16), $K_{\max} = \lfloor 2.5 \rfloor - 1 = 1$. Thus, there are two permissible ranges, corresponding to $K = 1$ and $K = 0$.

Using $K = 1$, the permitted range of the normalized sampling rate F_s/B , as found from Eq. (3.15) or Fig. 3.19, is $2.5 < F_s/B < 3$ or, equivalently, $25 \leq F_s \leq 30$ kHz. It is also possible, although wasteful, to occupy the $K = 0$ region and set $F_s > 2f_2 = 50$ kHz. Thus, we can sample at any rate between 25 and 30 kHz and beyond 50 kHz. We cannot, however, sample at a rate between 30 and 50 kHz. The minimum sampling rate is 25 kHz. It appears rather paradoxical that sampling at a rate 25 kHz is permitted, but 49 kHz is not. Below Nyquist, we enter a marshland of aliasing where some ground is solid and some is not. As K_{\max} increases, additional care is required to sidestep the ever-increasing pockets of unstable (forbidden) ground.

Example 3.6 ◀



Aliasing during sampling: occasional hero or treacherous imposter?

Aliasing: An Occasional Hero?

In the signal processing opera, aliasing generally plays the role of a villain. However, it is capable of occasionally performing the part of a hero. It plays precisely such a role in the uniform sampling of bandpass signals. Recall Sec. 3.1, where we showed that in order to be able to reconstruct a bandlimited signal from its uniform samples, the sampling rate must not be less than twice the highest frequency in its spectrum. Since f_2 designates the highest frequency of the bandpass signal $x(t)$, the minimum sampling frequency would seem to be $2f_2$ Hz. However, we showed that it is possible to recover the signal using a sampling rate as low as $2B$ Hz, where B is usually much smaller than f_2 . How is this possible? Simply, it is because the lower frequency band 0 to f_1 is unused

(empty) for bandpass signals. Aliasing deftly uses this empty band to allow sub-Nyquist sampling rates.

To further understand how aliasing accomplishes this role and satisfies, in some sense, the Nyquist criterion, again consider the spectrum $X_{\tilde{\delta}}(\omega)$ in Fig. 3.18b, where f_2 happens to be an integer multiple of B ($f_2 = 4B$), and the sampling rate is sub-Nyquist at $F_s = 2B < 2f_2$. Let us consider the spectrum within the band $-B$ to B . This is a lowpass spectrum of bandwidth B , which we denote as $X_{lp}(\omega)$. The entire spectrum $X_{\tilde{\delta}}(\omega)$ can be interpreted as $X_{lp}(\omega)$ repeating periodically at intervals of $F_s = 2B$ Hz. Thus, $X_{\tilde{\delta}}(\omega)$ also represents the signal $x_{lp}(t)$ sampled uniformly at a rate of $2B$ Hz. The samples of the bandpass signal $x(t)$ are identical to the samples of the lowpass signal $x_{lp}(t)$, both sets of samples being taken at a uniform rate of $F_s = 2B$ Hz. Moreover, the samples of $x_{lp}(t)$ contain the complete information of $x(t)$.

Observe that the sampling rate of $2B$ Hz is the Nyquist rate for $x_{lp}(t)$. Thus, by its nature of making high frequencies look like low frequencies, aliasing succeeds in making the bandpass signal $x(t)$ appear as a lowpass signal $x_{lp}(t)$. We are able to sample the bandpass signal at a lower rate, that is, one that satisfies the Nyquist criterion for $x_{lp}(t)$, and still be able to reconstruct $x(t)$ from these samples. Although we explain this behavior of aliasing for a special case where f_2 is an integer multiple of B , a similar argument applies to the general case.

▷ Drill 3.9 (Sampling a Bandpass Signal)

For a bandpass signal with $f_1 = 5$ kHz, $f_2 = 7$ kHz, show that the minimum sampling rate is bound by $14/3$ kHz. Furthermore, to recover the signal from its samples, show that the sampling rate may take any value in the ranges $14/3 < F_s < 5$, $7 < F_s < 10$, or $F_s > 14$ kHz.

△

▷ Drill 3.10 (Sampling a Complex Bandpass Signal)

Consider a complex bandpass signal $x(t)$ with spectrum $X(\omega) = \frac{\omega-1}{2}\Pi\left(\frac{\omega-2}{2}\right)$. Sketch $X(\omega)$, calculate its spectral width, and determine the sampling rates F_s that allow $x(t)$ to be recovered from its samples. What is the Nyquist rate for this signal?

△

3.5 Time-Sampling Dual: The Spectral Sampling Theorem

As in other cases, the sampling theorem has a dual. In Sec. 3.1, we discussed the time-sampling theorem, where we showed that a signal bandlimited to B Hz can be reconstructed from its samples taken at a rate $F_s > 2B$ samples/s. Note that the signal spectrum exists over the frequency range $-B$ to B Hz. Therefore, $2B$ is the spectral width (not the bandwidth, which is B) of the signal. This fact means that a bandlimited signal $x(t)$ can be reconstructed from samples taken at a rate F_s that exceeds $2B$, the spectral width of $X(\omega)$.

The *spectral sampling theorem* states that the spectrum $X(\omega)$ of a signal $x(t)$ timelimited to a duration of τ seconds can be reconstructed from the samples of $X(\omega)$ taken at a rate R samples/Hz, where $R > \tau$ (the signal width or duration) is in seconds. To understand and develop this theorem, we first relate spectral sampling to the periodic replication of a signal in time.

Periodic Replication in Time Produces Spectral Sampling

To begin, we define the T_0 -periodic replication of an energy signal $x(t)$ as $\tilde{x}(t) = \sum_{n=-\infty}^{\infty} x(t - nT_0)$. Conceptually, we obtain $\tilde{x}(t)$ (Fig. 3.20c) from the convolution of $x(t)$ (Fig. 3.20a) and a T_0 -periodic

delta train $\tilde{\delta}(t)$ (Fig. 3.20b). That is,

$$\tilde{x}(t) = x(t) * \tilde{\delta}(t) = \sum_{n=-\infty}^{\infty} x(t - nT_0). \quad (3.17)$$

Although we omit the details here, Eq. (3.17) is developed in the same straightforward manner as Eq. (3.6).

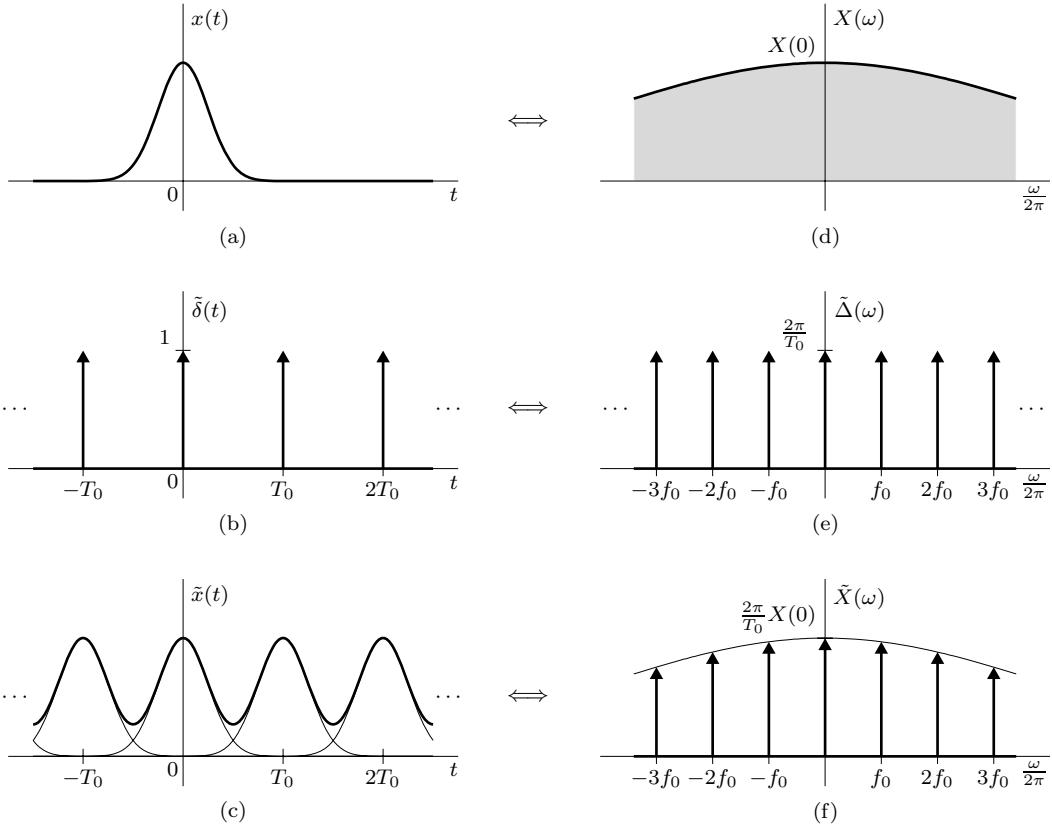


Figure 3.20: Periodic replication in time produces spectral sampling: (a)–(c) time domain and (d)–(f) frequency domain.

With the help of the convolution property, the Fourier transform of Eq. (3.17) yields[†]

$$\tilde{X}(\omega) = X(\omega)\tilde{\Delta}(\omega) = \sum_{k=-\infty}^{\infty} \frac{2\pi}{T_0} X(k\omega_0)\delta(\omega - k\omega_0), \quad (3.18)$$

where $\omega_0 = 2\pi f_0$ and $f_0 = 1/T_0$. Graphically, we see that $\tilde{X}(\omega)$ (Fig. 3.20f) is just the product of $X(\omega)$ (Fig. 3.20d) and the impulse train $\tilde{\Delta}(\omega)$ (Fig. 3.20e). Further, notice that Fig. 3.20 is just the dual of Fig. 3.2. Equation (3.18) tells us that the spectrum $\tilde{X}(\omega)$ is just a scaled, impulse-sampled version of the original spectrum $X(\omega)$. In other words, periodic replication in the time-domain produces sampling in the frequency domain. Much like the temporal sampling rate F_s is the

[†]Expressing the spectrum of $\tilde{x}(t)$ using a Fourier series produces a somewhat simpler-looking result that is equivalent to Eq. (3.18). See Prob. 3.4-4.

reciprocal of the time spacing T , the *spectral sampling rate* R , which has units of samples/Hz, is the reciprocal of the frequency spacing f_0 ,

$$R = \frac{1}{f_0} = T_0. \quad (3.19)$$

With these results, we are well equipped to understand the spectral sampling theorem.

Understanding the Spectral Sampling Theorem

Just as the sampling theorem of Sec. 3.1 specifies the conditions necessary to recover a time-domain signal from its samples, the spectral sampling theorem tells us the conditions necessary to recover a spectrum from its samples. And just as we cannot recover a time-domain signal that is non-bandlimited from its samples, we also cannot use spectral samples to recover the spectrum of a non-timelimited signal.

Figure 3.20 helps illustrate the problem with non-timelimited signals. A non-timelimited signal (Fig. 3.20a), when periodically replicated, has overlap that distorts the underlying time-domain signal (Fig. 3.20c). There is no way to undo the distortion of $\tilde{x}(t)$ to recover $x(t)$, and thus, there is no way to recover $X(\omega)$ from the spectral samples $\tilde{X}(\omega)$. This is precisely analogous to trying to recover a non-bandlimited signal from its samples: the spectral overlap that results from sampling makes recovery of the original spectrum through filtering impossible (see Fig. 3.13).

Even if a signal $x(t)$ is timelimited, there is no guarantee that samples of its spectrum are sufficient to recover the original spectrum. Like the Nyquist criterion of Eq. (3.2), which says that the sampling rate F_s must exceed the spectral width $2B$ to recover a bandlimited signal from its samples, a Nyquist criterion governs the recovery of $X(\omega)$ from $\tilde{X}(\omega)$ and requires that the repetition rate T_0 exceed the signal duration τ . This implies that, for a time-domain signal with finite duration τ ,

$$\text{lossless spectral sampling requires } R > \tau. \quad (3.20)$$

This is the *spectral sampling theorem*, and it basically states that if the successive cycles of $x(t)$ appearing in $\tilde{x}(t)$ do not overlap, then $x(t)$ can be recovered from $\tilde{x}(t)$.

Figure 3.21 illustrates the recovery of a timelimited signal from its periodic extension. The original timelimited signal (Fig. 3.21c), with width $\tau < T_0$ and midpoint T_c , is obtained by multiplying its periodic extension $\tilde{x}(t)$ (Fig. 3.21a) with a time-domain window $\Pi\left(\frac{t-T_c}{T_0}\right)$ (Fig. 3.21b). Mathematically,

$$x(t) = \tilde{x}(t)\Pi\left(\frac{t-T_c}{T_0}\right). \quad (3.21)$$

As we shall see next, this recovery implies indirectly that $X(\omega)$ can, through spectral interpolation, be reconstructed from its samples, provided that Eq. (3.20) is satisfied.

Spectral Interpolation

Again consider a timelimited signal $x(t)$ with width $\tau < T_0$ and midpoint T_c , an example of which is shown in Fig. 3.21c. Similar to the interpolation formula of Eq. (3.13), the *spectral interpolation formula* allows us to reconstruct the spectrum $X(\omega)$ of signal $x(t)$ from the samples $X(k\omega_0)$. Using $\Pi\left(\frac{t-T_c}{T_0}\right) \iff T_0 \text{sinc}\left(\frac{\omega T_0}{2\pi}\right) e^{-j\omega T_c}$ and the frequency-domain convolution property, the Fourier transform of Eq. (3.21) yields

$$X(\omega) = \frac{1}{2\pi} \tilde{X}(\omega) * T_0 \text{sinc}\left(\frac{\omega T_0}{2\pi}\right) e^{-j\omega T_c}.$$

Using Eq. (3.18), we obtain

$$X(\omega) = \sum_{k=-\infty}^{\infty} X(k\omega_0) \delta(\omega - k\omega_0) * \text{sinc}\left(\frac{\omega T_0}{2\pi}\right) e^{-j\omega T_c}.$$

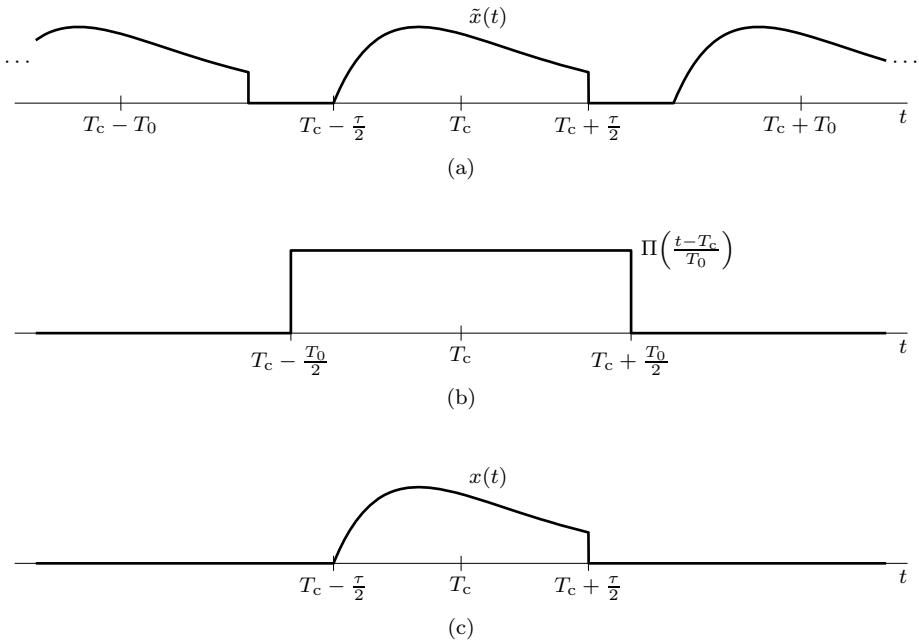


Figure 3.21: Recovering a finite-duration signal from its periodic extension.

Simplifying, the spectral interpolation formula is thus

$$X(\omega) = \sum_{k=-\infty}^{\infty} X(k\omega_0) \operatorname{sinc}\left(\frac{\omega T_0}{2\pi} - k\right) e^{-j(\omega - k\omega_0)T_c}. \quad (3.22)$$

If the pulse $x(t)$ is centered at the origin, then $T_c = 0$, and the exponential term in Eq. (3.22) vanishes. In such a case, Eq. (3.22) is the exact dual of Eq. (3.13). Notice that both interpolation formulas base reconstruction on weighted sinc functions.

▷ Example 3.7 (Spectral Interpolation)

Find the unit duration signal $x(t)$, centered at the origin, whose spectrum $X(\omega)$, when sampled at intervals of $f_0 = 1$ Hz (the Nyquist rate), is

$$X(k2\pi f_0) = \begin{cases} 1 & k = 0 \\ 0 & \text{otherwise} \end{cases}.$$

We use Eq. (3.22) with $T_c = 0$ to construct $X(\omega)$ from its samples. Since all but one of the Nyquist samples are zero, only the $k = 0$ term in the summation survives. Thus, with $X(0) = 1$ and $T_0 = \tau = 1$, we obtain

$$X(\omega) = \operatorname{sinc}\left(\frac{\omega}{2\pi}\right) \quad \text{and} \quad x(t) = \Pi(t).$$

For a signal of unit duration, this is the only spectrum with the sample values $X(0) = 1$ and $X(2\pi k) = 0$ ($k \neq 0$). No other spectrum satisfies these conditions.

Example 3.7 ◀

3.6 Analog-to-Digital Conversion

An *analog* signal is characterized by the fact that its amplitude can take on any value over a continuous range. Hence, an analog signal amplitude can take on an infinite number of values. In contrast, a *digital* signal amplitude can take on only a finite number of values. Although digital signals are either continuous- or discrete-time, practical digital signals are almost exclusively the latter. Thus, unless stated otherwise, when we refer to digital signals, we mean signals that are both digital and discrete-time.

Sampling a continuous-time analog signal alone does not yield a digital signal. As discussed earlier, sampling records a signal at discrete instants (continuous-time to discrete-time conversion). The recorded values still take on any value in a continuous range. To complete the conversion to a digital signal, samples must undergo a quantization process. *Quantization* uses one of various rules to round a sample's amplitude to a permissible value, or *quantization level*. Quantization is accomplished through *analog-to-digital (A/D) conversion*. As we shall soon see, there are many different types of analog-to-digital converters. Since the sampling and quantization processes almost always occur together, analog-to-digital converters often perform both operations.

Figure 3.22 illustrates one possible quantization scheme. The amplitudes of the original analog signal $x(t)$ lie in the range $(-V_{\text{ref}}, V_{\text{ref}})$. This range is partitioned into $L = 8$ subintervals, each of width $\Delta = 2V_{\text{ref}}/L$. Next, each sample amplitude $x(nT)$ is approximated by the midpoint of the subinterval in which the sample falls. It is clear that each quantized sample $x_q[n]$ is approximated by one of L possible values or states. This is an L -ary digital signal (see Sec. 1.1.2).

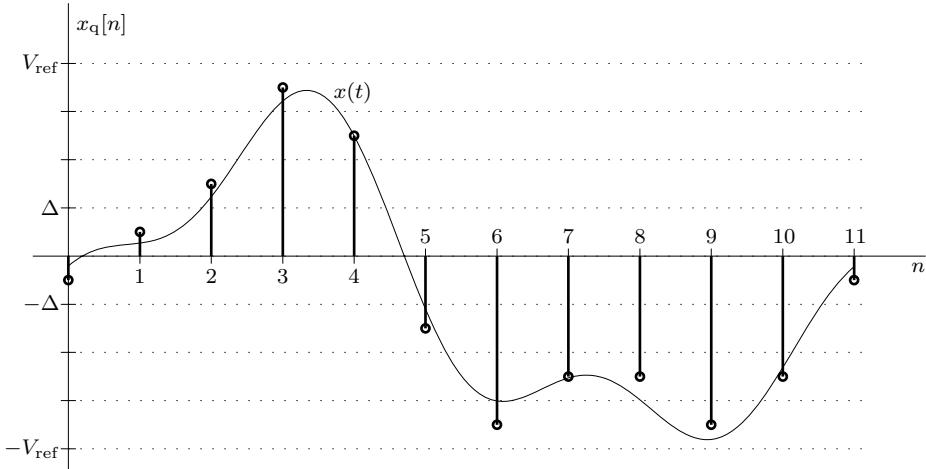


Figure 3.22: Analog-to-digital conversion produces signal quantization.

There are many ways to encode or represent the L possible states of each sample. Typically, each sample is represented by one or more symbols. A *symbol* is the smallest unit of data transmitted at any given time. In Fig. 3.22, each sample $x_q[n]$ is shown as a single pulse, or symbol, taking one of L values between $-V_{\text{ref}}$ and V_{ref} . In this case, one of $L = 8$ unique symbols encodes any particular sample. Particularly for large L , the single symbol per sample strategy requires an unwieldy number of distinct symbols. An attractive alternative is to represent each sample as a sequence of symbols taken from a smaller set, the smallest useable set being two.

A digital signal composed of a sequence of binary symbols, each of which has only two possible states, is known as a *binary signal*. Such a signal is very desirable because of its simplicity, economy, and ease of engineering. If L is a power of 2, as is most common, we can efficiently convert L -ary signal samples into binary numbers and thus a binary signal. A binary number is comprised of *binary digits*, or *bits*. To represent L states (quantization levels), we need a minimum of $B = \log_2 L$

binary digits. For example, $L = 8$ requires $B = \log_2 8 = 3$ bits to represent each of the eight levels. Alternately, we can say that B binary symbols can form $2^B = L$ distinct combinations, each one representing one level.

To transmit or digitally process this binary data, we need to assign a distinct electrical pulse to each of the two binary states. One possible way is to assign a negative pulse to a binary 0 and a positive pulse to a binary 1 so that each sample is now represented by a group of B binary pulses (pulse code). In communications jargon, this is known as *pulse-code modulation* (PCM).

Figure 3.23 illustrates these ideas for the $L = 8$ case. Each of the eight levels, numbered from 0 to 7, is assigned one binary code word of three digits. Although in this case we use the *natural binary code* (NBC), which is formed by the standard binary representation of the numbers 0 to $L - 1$, other binary codes, such as Gray codes (described later), are also possible. Figure 3.23 shows a pulse-code waveform for each code as well. Notice that if the strategy of Fig. 3.23 is used to encode the data of Fig. 3.22, the number of symbols transmitted per second, or *baud rate* (Bd), must exceed the sampling rate $F_s = 1/T$ by a factor of B , the number of digits (or symbols) per sample. Thus, in this case, data transmission occurs at a rate of $3F_s$ Bd.[†]

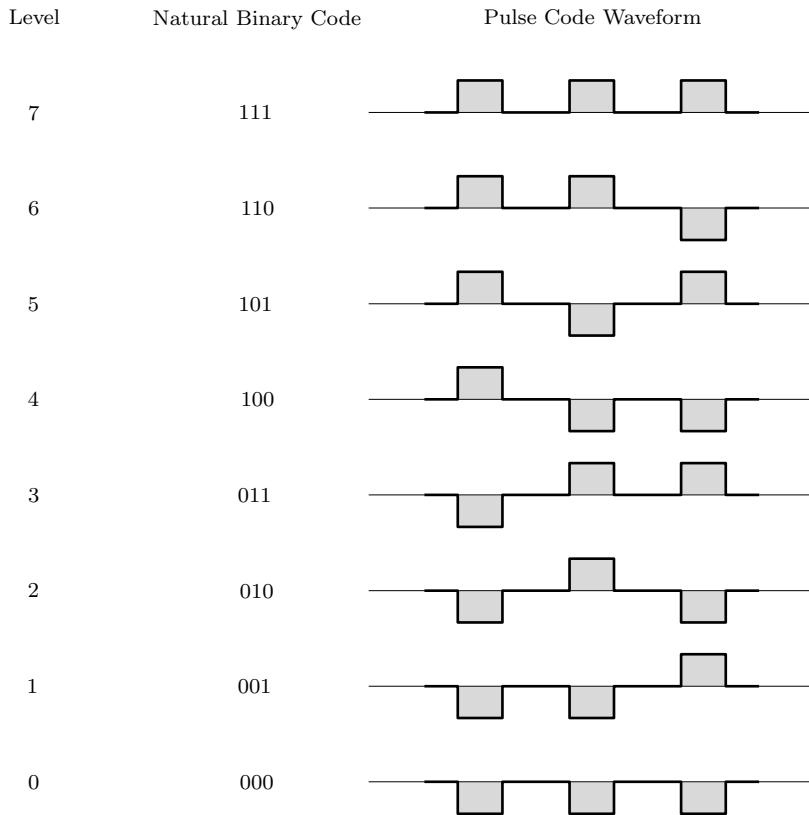


Figure 3.23: Binary representation and pulse coding.

Small symbol sets, such as the binary system, are crucial to practical signal representation. Intuitively, it is much easier to engineer a binary system, which needs to distinguish only between two states, than an L -ary system, which needs to distinguish between L states. The pulse-code waveforms of Fig. 3.23, for example, use positive and negative pulses to form the two required symbols. These pulses are separated by a wide margin, which makes it relatively easy to correctly

[†]Baud rate is commonly, and incorrectly, referred to as bit rate. Baud rate is equivalent to bit rate only in the special case when the symbols are binary.

identify symbols even when pulses are distorted by system effects or contaminated by noise. An analogous L -ary system would have smaller margins between its L -level pulses, making the system more susceptible to distortion and noise effects.

English, Chinese, and Beyond

To further highlight the advantages of small symbol sets, let us consider a simple analogy. English is comprised of 26 letters (symbols). Assuming an average of five letters per word, it takes 250000 symbols to represent a typical 50000-word book. Chinese, on the other hand, represents each word with a single Hanzi character (symbol).[†] In Chinese, the same 50000-word book now requires just 50000 symbols – a fivefold reduction compared with English. To read a book in Chinese, however, one must be familiar with hundreds or thousands of Hanzi symbols rather than just 26 English letters. Clearly, there is a tradeoff between the number of symbols in a language and the length needed to represent information. This is a basic principle of *information theory*.

In an extreme case, we can imagine some super language with enough unique symbols to represent every possible book in the world. Any book, including this textbook, could be written as a single symbol. There is a certain appeal to such an approach: you could read any book in one sitting without ever turning a single page! The practical problems of such an approach, however, are severe. To distinguish every book, each symbol becomes impossibly intricate. Further, one would have to prepare a dictionary that gives the meaning of each symbol. Anyone literate in this language would have to master this massive dictionary! Just to recognize a symbol requires you to know every symbol available, something akin to having pre-read every book. Further, consider what happens if there is even a single error in the printing of the book. In English, you might recognize this as a misspelled word. In Chinese, an entire word might change. With our super language, however, an error means that you read an entirely different book altogether. Imagine your disappointment when, having wanted to read a book on digital signal processing, you find instead that you actually read a book on grammar and punctuation.

Digitizing Voice and Audio

The human ear is sensitive to about 20 kHz. Subjective tests, however, show that voice intelligibility is not affected if components above 3400 Hz and below 300 Hz are suppressed [3]. Since the objective in voice communications is intelligibility rather than high fidelity, the components above 3400 Hz are typically eliminated by a lowpass filter. Consequently, sampling rates of 8 kHz are completely adequate for voice communication applications, such as digital telephony. Rates are intentionally kept higher than the Nyquist sampling rate of 6.8 kHz to ensure practical filters, which are required for signal reconstruction. In traditional phone systems, each sample is quantized using 7 or 8 bits per sample, which results in 128 or 256 levels to represent each sample ($2^7 = 128$ and $2^8 = 256$). Such a digitized telephone signal consists of $7 \times 8000 = 56000$ or $8 \times 8000 = 64000$ bits/s (56 or 64 kbps) for data transmission.[‡]

The compact disc (CD) is another relatively recent application of A/D conversion. Intended for high-fidelity audio applications, CDs require the full 20-kHz audio bandwidth. Although the Nyquist sampling rate is only 40 kHz, an actual sampling rate of 44.1 kHz allows realizable filters, as discussed earlier.* To reduce quantization noise and improve dynamic range, CD signals are quantized using 16 bits, which provides a rather large number of levels ($L = 65536$). For a stereo signal, then, CDs produce over 1.4 Mbps (1.4 million bits per second) of data.

[†]For simplicity, we ignore that some words require two or (rarely) more Hanzi characters.

[‡]Modern sophisticated voice coders, such as those used with CDMA and GSM digital wireless phone systems, compress voice signals to remove redundant data and produce excellent voice quality with bit rates as low as 4 kbps. Such approaches, however, come at a substantial increase in processing complexity.

*The seemingly strange specific rate choice of 44.1 kHz originates from an early desire to fit digital audio onto video tape designed for 3 samples (colors) of 490/2 lines at a 60-Hz refresh rate ($3 \times 245 \times 60 = 44100$).

A Historical Note

The ancient Indian writer Pingala developed advanced mathematical concepts for describing prosody and in doing so presented the first known description of a binary numeral system, possibly as early as the 8th century BCE.[†] Gottfried Wilhelm Leibnitz (1646–1716) was the first mathematician in the West to work out systematically the binary representation (using 1s and 0s) for any number. He felt a spiritual significance in this discovery, believing that 1, representing unity, was clearly a symbol for God, while 0 represented nothingness. He reasoned that if all numbers can be represented merely by the use of 1 and 0, this surely proves that God created the universe out of nothing!

▷ Example 3.8 (Computing Sampling Rate, Number of Bits, and Baud Rate)

A signal $x(t)$ bandlimited to 3 kHz is sampled at a rate $33\frac{1}{3}\%$ higher than the Nyquist rate. The maximum acceptable error in the sample amplitude (the maximum error due to quantization) is 0.5% of the peak amplitude V_{ref} . The quantized samples are binary coded. Find the required sampling rate, the number of bits required to encode each sample, and the bit rate of the resulting PCM signal.

The Nyquist sampling rate is $F_{\text{Nyq}} = 2 \times 3000 = 6000$ Hz. The actual sampling rate is

$$F_s = 6000 \times \frac{4}{3} = 8000 \text{ Hz.}$$

The quantization step is Δ , and using a quantization strategy such as shown in Fig. 3.22, the maximum quantization error is $\pm\Delta/2$, where $\Delta = 2V_{\text{ref}}/L$. The maximum error due to quantization, $\Delta/2$, should be no greater than 0.5% of the peak amplitude V_{ref} . Therefore

$$\frac{\Delta}{2} = \frac{V_{\text{ref}}}{L} = \frac{0.5}{100} V_{\text{ref}} \implies L = 200.$$

For binary coding, L must be a power of 2. Hence, the next higher value of L that is a power of 2 is $L = 256$. Because $\log_2 256 = 8$, we need 8 bits to encode each sample. Therefore, the bit rate of the PCM signal is

$$8 \times 8000 = 64000 \text{ bits/s.}$$

Example 3.8 □

▷ Drill 3.11 (Computing the Number of Bits and Baud Rate)

The American Standard Code for Information Interchange (ASCII) has 128 characters that are binary coded. Suppose that a certain computer generates 100000 such characters per second. Determine the number of bits required to encode each character as well as the baud rate necessary to transmit the computer output.

□

▷ Drill 3.12 (Quantization with M -ary Symbols)

Consider a case of M -ary symbols (symbols that can take M distinct values). Show that to represent L levels, the required number of M -ary digits is at least $\log_M(L)$. Find the minimum number of M -ary digits needed to encode $L = 64$ levels when $M = 2, 3, 4, 5, 6, 7$, and 8 . Remember that the number of digits in a code must always be an integer.

□

[†]See *The Heritage of Thales*, by W. S. Anglin and J. Lambek.

3.6.1 Analog-to-Digital Converter Transfer Characteristics

For a binary scheme using a B -bit code, an ADC divides the input signal range into $L = 2^B$ equal steps (quantization levels), and every signal amplitude is approximated by one of 2^B amplitudes. Upper and lower references define the converter's input operating range. *Unipolar* ADCs typically set one reference to zero, while *bipolar* converters commonly set the lower reference equal to the negative of the upper reference. Although we focus our attention on bipolar converters, unipolar ADCs are treated in a nearly identical manner.

Using 3-bit quantization, Fig. 3.24 shows the input-output or transfer characteristics of four common types of bipolar ADCs. In each case, the ADC input references are set to $\pm V_{\text{ref}}$, which is typically equal to or slightly larger than the peak amplitude of the input signal x . The output x_q is quantized to $2^3 = 8$ levels, uniformly separated in steps of Δ . We stress that the quantization process is both nonlinear and nonreversible. Once a signal x is quantized, it can never be recovered from x_q , although the error between the two can be made arbitrarily small by increasing the number of bits B . The difference between x and x_q for $|x| \leq V_{\text{ref}}$ is termed *quantization error* (light gray), while the error that occurs for $|x| > V_{\text{ref}}$ is called *saturation error* (dark gray). With proper signal conditioning and reference voltage selection, saturation error is easily avoided. Quantization error, on the other hand, is unavoidable.

Asymmetric converters, such as shown in Figs. 3.24a and 3.24b, include zero as a quantization level and consequently have more quantization levels either below or above zero. Both Figs. 3.24a and 3.24b, for example, have four levels below zero and three levels above. Asymmetry is inevitable in any scheme with an even number of steps if we maintain zero as a quantization level. Alternately, symmetric converters, such as those shown in Figs. 3.24c and 3.24d, do not include zero as a quantization level and consequently have the same number of quantization levels above and below zero. Except possibly near peak input amplitudes, the quantization error of rounding converters (Figs. 3.24a and 3.24c) is between $-\Delta/2$ and $\Delta/2$, while that of truncating converters (Figs. 3.24b and 3.24d) is skewed between $-\Delta$ and 0. Mathematically, the quantized output is given as

$$\begin{aligned} x_q &= \frac{V_{\text{ref}}}{2^{B-1}} \left\lfloor \frac{x}{V_{\text{ref}}} 2^{B-1} + \frac{1}{2} \right\rfloor \quad (\text{rounding, asymmetric}), \\ x_q &= \frac{V_{\text{ref}}}{2^{B-1}} \left\lfloor \frac{x}{V_{\text{ref}}} 2^{B-1} \right\rfloor \quad (\text{truncating, asymmetric}), \\ x_q &= \frac{V_{\text{ref}}}{2^{B-1}} \left(\left\lfloor \frac{x}{V_{\text{ref}}} 2^{B-1} \right\rfloor + \frac{1}{2} \right) \quad (\text{rounding, symmetric}), \\ \text{or} \quad x_q &= \frac{V_{\text{ref}}}{2^{B-1}} \left(\left\lfloor \frac{x}{V_{\text{ref}}} 2^{B-1} - \frac{1}{2} \right\rfloor + \frac{1}{2} \right) \quad (\text{truncating, symmetric}). \end{aligned} \quad (3.23)$$

When $|x|$ nears or exceeds V_{ref} , Eq. (3.23) may return values outside the L allowable levels. In such cases, the values should be clamped to the nearest permitted level.

Once a signal is quantized, each sample is assigned a code, typically binary. The transfer characteristic in Fig. 3.24a, for example, has eight quantization levels and requires a three-bit code word. Code word assignments can be done in many different ways, and each assignment has its advantages and drawbacks. Figure 3.24a shows four common coding methods: *two's complement*, *offset binary*, and two of many possible *Gray codes*. Although these code assignments are shown next to the asymmetric rounding converter, they apply equally well to Figs. 3.24b, 3.24c, and 3.24d.

To interpret the value of a binary code requires that the transfer characteristic be known. For asymmetric converters, such as shown in Figs. 3.24a and 3.24b, the value x_q corresponding to a two's complement code word $c_{B-1}c_{B-2}\dots c_1c_0$ is computed as

$$x_q = V_{\text{ref}} \left(-c_{B-1}2^0 + c_{B-2}2^{-1} + \dots + c_12^{-(B-2)} + c_02^{-(B-1)} \right). \quad (3.24)$$

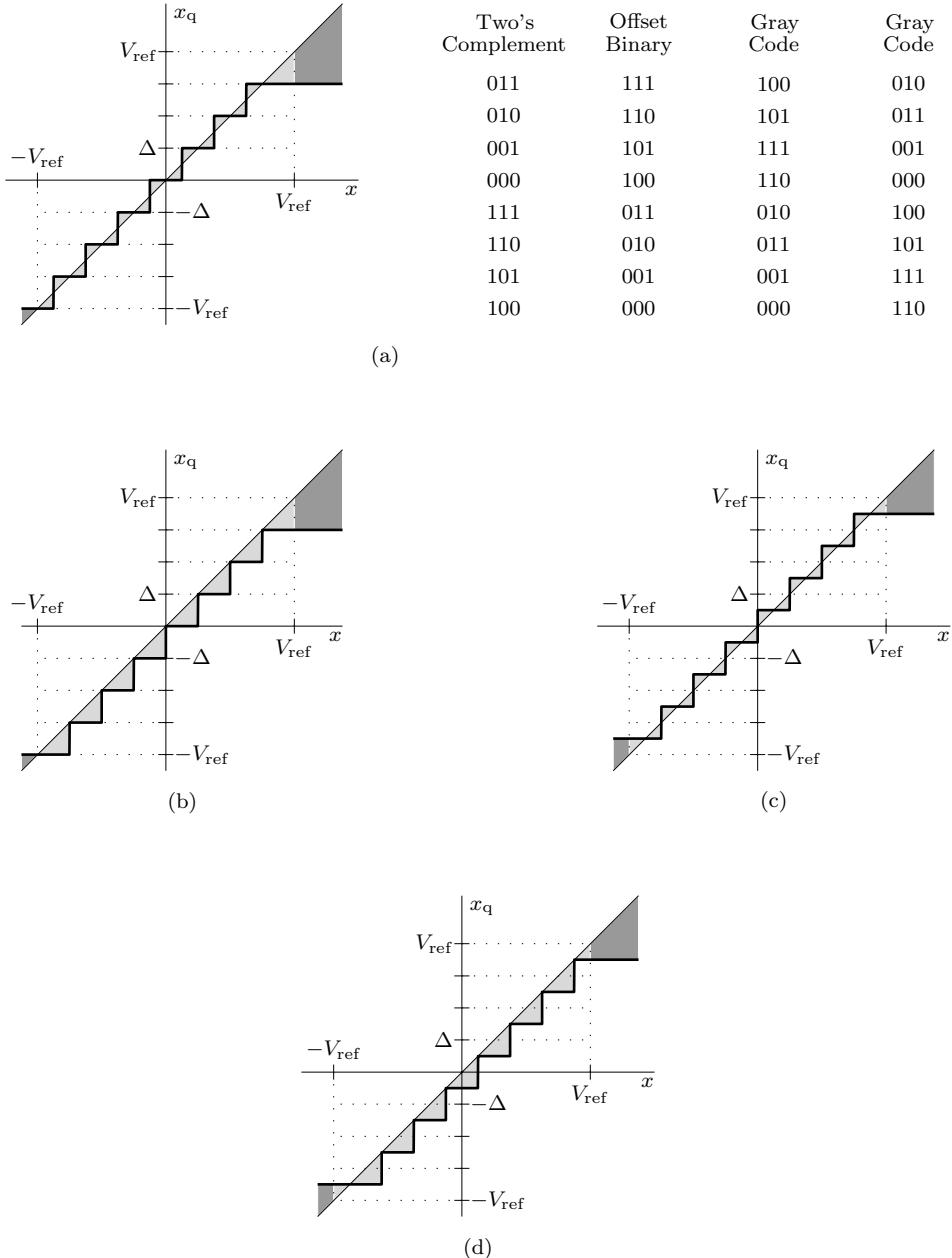


Figure 3.24: Bipolar ADC transfer characteristics: (a) rounding asymmetric, (b) truncating asymmetric, (c) rounding symmetric, and (d) truncating symmetric.

Offset binary requires a minor adjustment and is, for asymmetric converters, given as

$$x_Q = V_{\text{ref}} \left(-1 + c_{B-1}2^0 + c_{B-2}2^{-1} + \cdots + c_12^{-(B-2)} + c_02^{-(B-1)} \right). \quad (3.25)$$

Equations (3.24) and (3.25) are adjusted for symmetric converters by simply adding $\Delta/2$. It is more difficult to compute the value x_Q corresponding to a Gray code. Typically, a Gray code is converted

to another type, such as two's complement, which allows simple interpretation and processing.[†] Unipolar converters are treated in a similar manner. For example, the code word of a unipolar converter using natural binary code is

$$x_q = V_{\text{ref}} \left(c_{B-1} 2^{-1} + c_{B-2} 2^{-2} + \cdots + c_1 2^{-(B-1)} + c_0 2^{-B} \right). \quad (3.26)$$

In two's complement, the most significant bit c_{B-1} can be interpreted as a sign digit. Due to the existence of simple and efficient arithmetic rules, two's complement is generally preferred over offset binary and Gray code in digital processing. Every commercial microprocessor and digital signal processor, for example, has instructions to add or subtract two's complement numbers. Few processors offer native support for offset binary or Gray code. Gray codes have the advantage, however, that the codes assigned to adjacent quantization levels differ by one digit only. Hence, if a digit is read wrong due to noise or other interfering signals, there is a minimum error in reading its value. In communications applications, where signals are transmitted over long distances using imperfect or distortion-causing transmission media, there is a finite probability of error in reading received digits. The use of Gray codes results in the least damage to the received signal.

▷ Example 3.9 (Interpreting Two's Complement and Offset Binary)

Assuming a 3-bit asymmetric converter is used, determine the value x_q that corresponds to the two's complement code word 110. What is the value if the code word is offset binary? How do the values change if the converter is instead a symmetric converter?

Using Eq. (3.24), the two's complement code 110 corresponds to

$$x_q = V_{\text{ref}} \left(-1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} \right) = -V_{\text{ref}}/2.$$

If the number is offset binary, Eq. (3.25) yields a value

$$x_q = V_{\text{ref}} \left(-1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} \right) = V_{\text{ref}}/2.$$

If the converter is symmetric rather than asymmetric, the values require an offset of $\Delta/2 = V_{\text{ref}}/8$. Thus, the two's complement 110 corresponds to $-V_{\text{ref}}/2 + V_{\text{ref}}/8 = -3V_{\text{ref}}/8$, and the offset binary 110 corresponds to $V_{\text{ref}}/2 + V_{\text{ref}}/8 = 5V_{\text{ref}}/8$. Each of these values is easily confirmed with Fig. 3.24.

Example 3.9 ◀

▷ Drill 3.13 (Identifying Quantization Method)

Identify the quantization method used in Fig. 3.22 as symmetric or asymmetric, rounding or truncating, and linear or nonlinear.

◀

[†]A Gray code $g_{B-1} \cdots g_1 g_0$ can be constructed from a two's complement of offset binary code $c_{B-1} \cdots c_1 c_0$ using the rule

$$g_{B-1} = c_{B-1} \quad \text{and} \quad g_b = c_b \oplus c_{b+1},$$

where \oplus designates the exclusive-or operation. Using this rule, the offset binary in Fig. 3.24a converts to the first column of Gray code, and the two's complement converts to the second column of Gray code. We can recover $c_{B-1} \cdots c_1 c_0$ from $g_{B-1} \cdots g_1 g_0$ according to

$$c_{B-1} = g_{B-1} \quad \text{and} \quad c_b = g_b \oplus g_{b+1}.$$

▷ **Example 3.10 (Investigating Different Quantization Methods)**

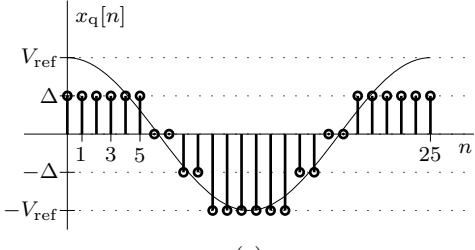
Consider the signal $x(t) = \cos(2\pi t/25)$. Using sampling interval $T = 1$ s, reference $V_{\text{ref}} = 1$, and $B = 2$ bits, plot the quantized signal $x_q[n]$ over one period of $x(t)$ using each of the four quantization methods of Eq. (3.23).

MATLAB is well equipped to solve this problem. First, basic parameters are established.

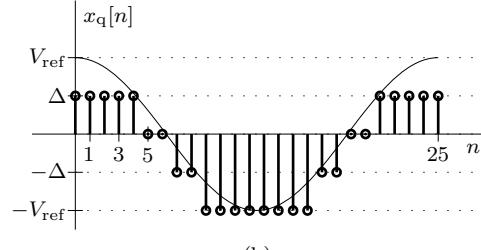
```
01 T = 1; Vref = 1; B = 2; Delta = 2*Vref/(2^B); x = @(t) cos(2*pi*t/25); n = (0:25);
```

Next, we compute and plot the quantized output for a rounding, asymmetric converter using Eq. (3.23). Line 03 clamps any values computed outside the permitted L levels.

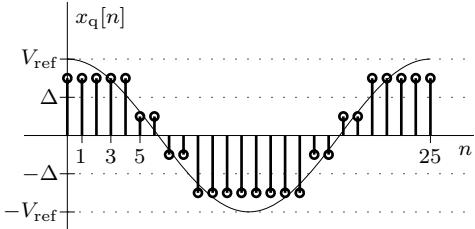
```
02 xqra = (Vref/2^(B-1))*floor(x(n*T)/Vref*2^(B-1)+1/2);
03 xqra(xqra>Vref-Delta) = Vref-Delta; xqra(xqra<-Vref) = -Vref;
04 subplot(221); stem(n,xqra);
```



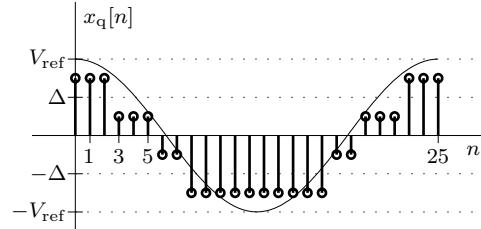
(a)



(b)



(c)



(d)

Figure 3.25: Quantization of $x(t) = \cos(2\pi t/25)$: (a) rounding asymmetric, (b) truncating asymmetric, (c) rounding symmetric, and (d) truncating symmetric.

Following a similar procedure, lines 02–04 are adapted for the other three methods. As shown in Fig. 3.25, the four results, while similar, are not identical. Still, each quantized signal conveys the basic character of $x(t)$. Notice that some of the waveforms appear bottom-heavy with a seeming negative offset. Although undesirable, such characteristics become negligible with large B .

Example 3.10 ◇

▷ **Drill 3.14 (Quantization Can Amplify Noise)**

Consider the common situation where a signal $x(t)$ becomes 0 for an extended period of time. Explain why an asymmetric, rounding converter is better than an asymmetric, truncating converter in the case where low-level noise corrupts $x(t)$. Which symmetric converter is best, rounding or truncating? Explain.

◇

Nonlinear Quantization

So far we have discussed *linear quantization*, which uses a constant quantization step Δ . In linear quantization, the quantization boundaries are equally spaced from one another, whether the input value is large or small. Such a scheme offers an unfair deal to smaller signal amplitudes, however, since linear quantization causes more relative damage to lower amplitude levels than to higher amplitude levels. A rounding symmetric converter, for example, has a maximum quantization error of $\Delta/2$. If such a converter has $\Delta = 0.002$, the maximum quantization error is 0.001. For a high input amplitude near 1, the maximum quantization error is only 0.1% of the signal amplitude. At a low amplitude such as 0.002, however, the maximum quantization error is 50%.

The relative damage can be equalized if the quantization step Δ is not fixed but becomes progressively larger for higher amplitudes, as shown in Fig. 3.26a. Such a strategy is termed *nonlinear quantization*.[†] Nonlinear quantization is widely used in communications applications where it allows us to compress useful signal information and reduce data rates. For example, telephone-grade audio using linear quantization requires 12-bit codes to match the perceived quality of nonlinear quantization using only 8-bit codes. In other words, linear quantization requires 50% more bits over nonlinear quantization for comparable quality.

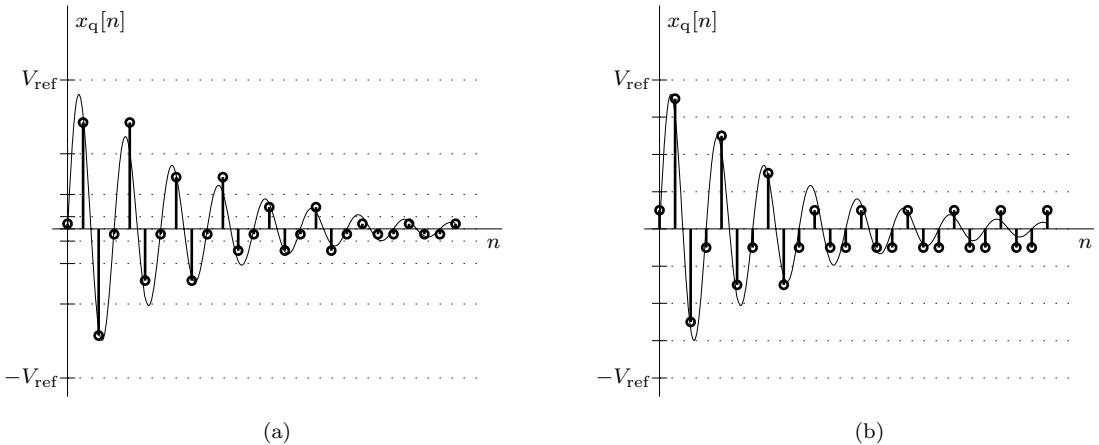


Figure 3.26: Signal quantization: (a) nonlinear and (b) linear.

In practice, logarithmic input-output characteristics are used to achieve nonlinear quantization. Figure 3.26a, for instance, uses *μ -law compression*, a North American standard. First, the input x is compressed according to

$$x_\mu = \frac{V_{\text{ref}} \text{sgn}(x)}{\ln(1 + \mu)} \ln(1 + \mu|x|/V_{\text{ref}}), \quad (3.27)$$

where the compression parameter μ determines the level of compression. Large μ increases resolution for small signal amplitudes, all at the price of more coarseness for large amplitudes. The logarithmically compressed x_μ is then quantized using a standard linear quantizer, such as the rounding symmetric converter used for Fig. 3.26. The transfer characteristics of this step are shown in Fig. 3.27a.

While compression enhances small-amplitude signals, the nonlinear nature of Eq. (3.27) also distorts the waveform. Thus, the compression must be undone (expanded) at the final destination to yield the quantized signal x_q . This is an important step as compressed data cannot generally be used directly for standard signal processing. Figure 3.27b shows the input-output transfer characteristics

[†]It is important to note that when describing quantization as linear or nonlinear, we are referring to the manner in which the quantization boundaries are established and *not the system property of linearity*. In fact, linear quantizers and nonlinear quantizers are both *nonlinear systems* (superposition is not guaranteed to hold).

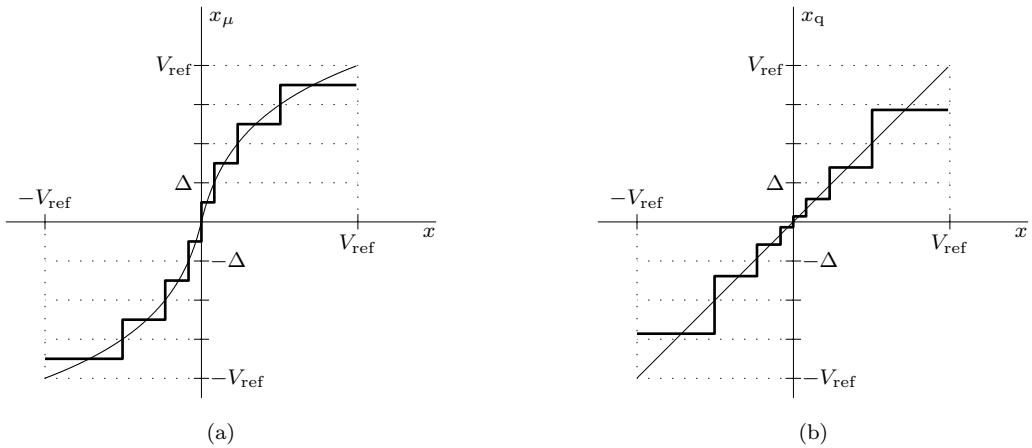


Figure 3.27: Nonlinear quantization transfer characteristics: (a) μ -law compression with rounding, symmetric quantization and (b) expansion to final quantized signal.

following the expansion step. Expansion clearly restores the underlying linear relationship between input and quantized output but necessarily forces a nonlinear spacing of the quantization levels.

Data compression and expansion increase processor load and hardware costs, but the reduced data rates are often well worth the price. The decaying sinusoid in Fig. 3.26a highlights how nonlinear quantization effectively tracks even small signals. The linear quantization of Fig. 3.26b, on the other hand, struggles to capture the detail of small signals, although it is less coarse for large amplitudes than the nonlinear case.

3.6.2 Analog-to-Digital Converter Errors

We know that analog-to-digital conversion produces quantization and saturation errors. These errors, while undesirable, are fundamental and indeed inevitable to the quantization process. Unfortunately, other errors can also occur that further degrade x_q . These errors, which reflect hardware imperfections and limitations, are typically classified as either static or dynamic.[†] In broad terms, *static errors* relate to dc performance, while dynamic errors involve time dependencies such as input frequency or sampling clock rates. Unlike quantization and saturation errors, however, it is possible to correct or compensate some, although not all, types of static and dynamic errors.

We begin with a 3-bit unipolar rounding converter. Figure 3.28 illustrates how the four basic types of static errors affect this converter. These plots take the form of the transfer characteristic plots of Fig. 3.24, but they also assume that the input x is a dc signal.

An *offset error* shifts the ADC transfer characteristic either right or left. The right shift in Fig. 3.28a, for example, shifts the waveform x_q downward. Except for those parts shifted below 0 (or above V_{ref} for a left shift), which are forever lost by the clamping action of the converter, an offset error is easily corrected by subtracting the offset.

A *gain error*, such as shown in Fig. 3.28b, is also relatively easy to characterize and correct. A gain error changes the slope of the transfer characteristic, causing x_q to appear larger or smaller than x . With a slope greater than 1, such as in Fig. 3.28b, x_q appears larger, or magnified, compared with x . Values pushed beyond the reference are clamped, causing irrecoverable signal loss and distortion. When the slope is less than 1, x_q appears smaller than x . In this case, the basic waveform shape

[†]Fully characterizing an analog-to-digital converter is quite complex, so we do not attempt a comprehensive treatment here. IEEE Standard 1057, for example, deals with this topic and is nearly 100 pages long. Instead, we overview the most important static errors and comment on dynamic errors to provide a sense of what can and does go wrong during analog-to-digital conversion.

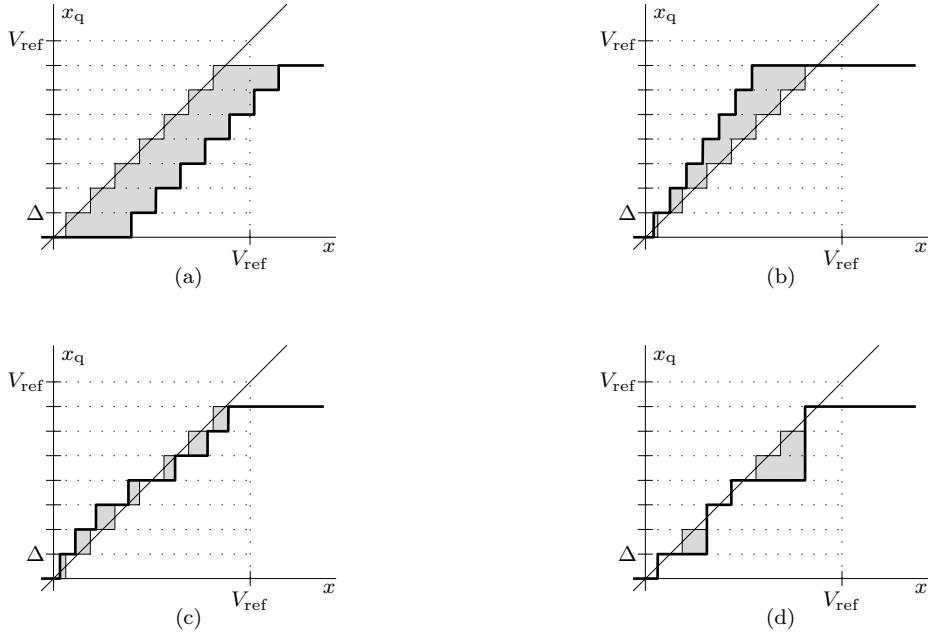


Figure 3.28: Static ADC errors: (a) offset, (b) gain, (c) nonlinearity, and (d) missing codes.

remains intact, although the relative quantization error increases (Δ seems bigger when we shrink our signal). By applying a reciprocal gain factor to x_q , gain errors are largely compensated.

The effects of *nonlinearity errors*, depicted in Fig. 3.28c, are not as systematic as those of gain or offset errors. In some sense, we can think of nonlinearity errors as being the static error beyond a converter's gain and offset errors. It is usually not simple to either characterize or correct nonlinearity errors. *Missing codes* occur when a converter is unable to produce, regardless of input, certain quantization levels. Figure 3.28d illustrates this last static error, where three of the eight codes are missing. It is not possible to correct missing code errors. Using the same exact transfer characteristics as Fig. 3.28, Fig. 3.29 illustrates the various effects of static errors while sampling a sinusoid.

Dynamic errors are more involved than static errors. Dynamic errors might cause two input sinusoids, which only differ in frequency, to convert with entirely different gains. Variations in the sampling clock, also called *clock jitter* or *timing jitter*, cause signal conversions to occur at entirely different times than anticipated. Frequency-dependent interactions further complicate the picture. Typical transfer characteristic curves, such as those in Fig. 3.28, are woefully inadequate to describe dynamic errors. We do not pursue the details of dynamic errors here. Suffice it to say that dynamic errors further erode converted signal quality.

A Pack of Wolves

No more than a shepherd desires to see a wolf among his flock, we do not care to see any particular type of error affect our A/D conversions. Unfortunately, a lone wolf is not the least of our worries. Just as wolves hunt in packs, different converter errors attack our signal together, and they can inflict significant combined damage. Still, we need not cry wolf just because an A/D converter has errors. It is wise to first determine the nature of the pack that hunts us. The *effective number of bits* (ENOB) is a number, expressed in bits, that helps quantify the combined impact of various errors on analog-to-digital converter performance. Often found in a converter's data sheets, ENOB provides an indication of ADC accuracy under real-world conditions. Any practical converter's ENOB is less

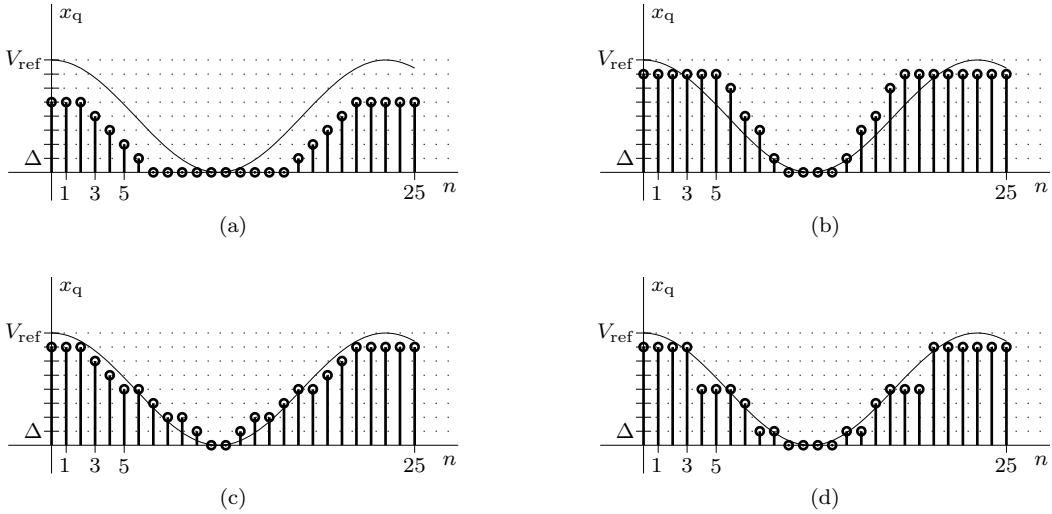


Figure 3.29: Sampling a sinusoid with static ADC errors: (a) offset, (b) gain, (c) nonlinearity, and (d) missing codes.

than its actual number of bits B . High-quality 16-bit converters, for example, typically have ENOBs of 12 to 13.5, which means that 3.5 to 4 bits of accuracy are being lost to converter errors.

▷ **Drill 3.15 (ADC Errors Reduce Dynamic Range)**

Determine the reduction in dynamic range, in dB, of a 16-bit analog-to-digital converter that has an ENOB equal to 12.25 when compared with an ideal (ENOB = 16).

◀

3.6.3 Analog-to-Digital Converter Implementations

Analog-to-digital conversion involves reading sample values and assigning the proper binary code words. Figure 3.30a shows a simplified block representation of a practical A/D converter.

In practice, the process of coding involves a sequence of comparisons and hence is not instantaneous. To give the coder time to perform its job, we sample the signal and hold this value until the next sample. Conceptually, this so-called sample-and-hold (S/H) operation first impulse samples the signal $x(t)$ to yield the impulse sequence $x_{\delta}(t)$. Next, the hold operation produces a rectangular pulse of width T for each impulse. The impulse response $h(t)$ of the hold operation, shown in Fig. 3.30b, is identical to the impulse response of the ideal zero-order hold (Fig. 3.11a) except for a delay of $T/2$ seconds, which is needed to ensure that the converter is causal and realizable. The sample-and-hold output $x_{sh}(t)$, shown in Fig. 3.30c, has an apparent, and unavoidable, delay of $T/2$ when compared with the original input $x(t)$. During each hold period T , the coder assigns a code word to the sample according to some desired transfer characteristic rule, such as one of those shown in Fig. 3.24. Timing for the entire process is typically driven by an external clock.

Irrespective of transfer characteristic, there are many different strategies to implement the coder of an ADC. The coder details, in fact, largely determine the nature of the A/D converter itself. Thus, coder type becomes synonymous with ADC type. Each type of coder (converter) has its own advantages and disadvantages, making some appropriate for certain applications and not others.

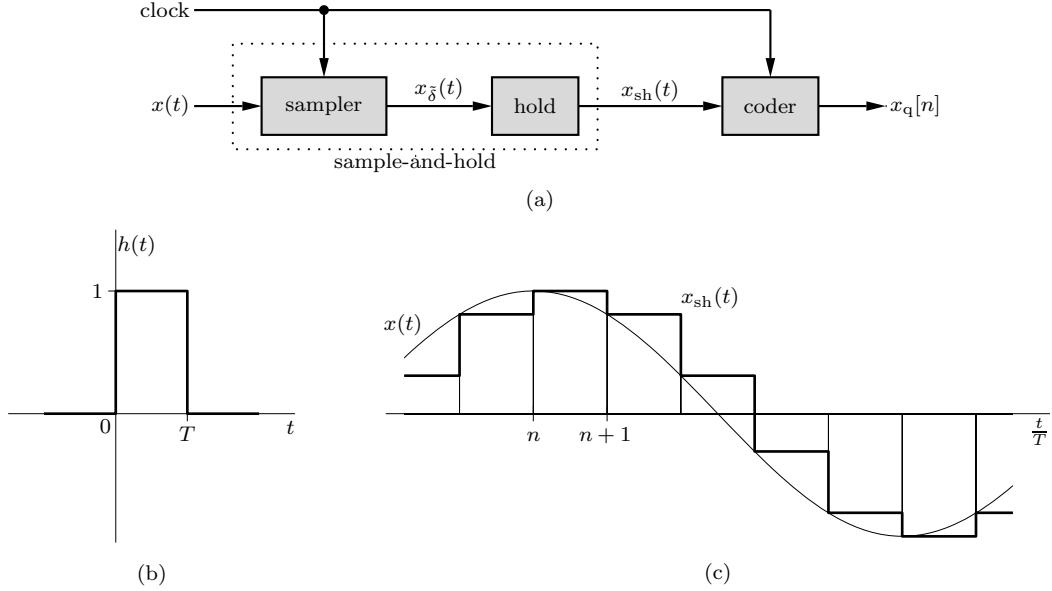


Figure 3.30: A practical A/D converter: (a) block representation, (b) hold operation impulse response, and (c) output of the sample-and-hold circuit.

Counting Converters

Counting converters are among the simplest to understand. As shown in the block diagram of Fig. 3.31, a counting converter centers on a simple binary counter. Starting from zero, the counter steps until its output, which is converted using a digital-to-analog converter (DAC), reaches the S/H value $x_{sh}(t)$. This comparison triggers the counter to stop, at which time the counter value is the code of sample $x_q[n]$.

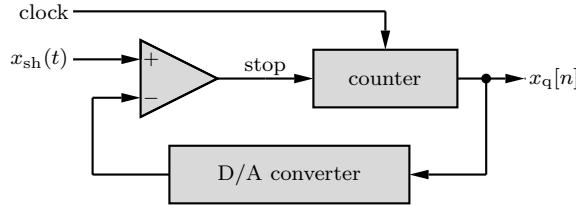


Figure 3.31: Counting converter.

Counting converters possess an undesirable characteristic that the conversion time is proportional to input amplitude. Further, if the converter uses a large number of bits B , the counting process can take a relatively long period of time. For example, a 16-bit counting converter can require up to $2^{16} = 65536$ separate steps to obtain a result. Consequently, counting converters are slow and not particularly well suited to high-speed applications. However, counting converters are simple, affordable, and can be extremely accurate.

Successive Approximation Converters

As shown in Fig. 3.32, a successive approximation converter is similar in appearance to a counting converter but operates by making B sequential comparisons to generate a B -bit code word. Rather than a counter, a successive approximation converter utilizes a successive approximation register (SAR). One can imagine that the SAR answers B successive questions, beginning with whether or

not the sample is in the upper or lower half of the allowed range. The most significant digit of the output is set to 1 or 0 accordingly. The next most significant digit is set according to whether the sample is in the upper or lower half of the previous subinterval. This process continues a total of B times until the last digit in the code is generated.

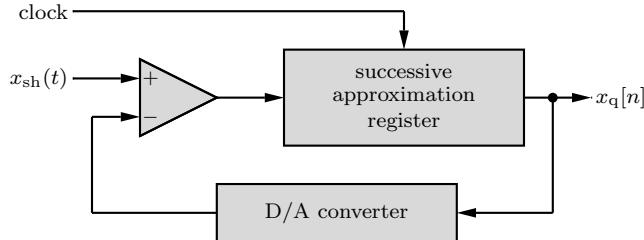


Figure 3.32: Successive approximation converter.

Successive approximation converters are much faster than counting converters. A fixed number B steps are required, as compared with 2^B steps possible for counting converters. Successive approximation converters offer good resolutions, moderate complexity, and wide operating ranges.

Flash Converters

As much as counting converters are simple and slow, *flash* converters are complex and fast. Also known as *parallel* or *direct* converters, flash architectures code samples nearly instantaneously using a massive parallel circuit. As shown in Fig. 3.33, a flash converter uses L resistors, connected in series, to establish the required $L - 1$ quantization boundaries across the device operating range. If all the resistors are equal, the voltage drop across any resistor is a constant Δ . The input $x_{sh}(t)$ is simultaneously compared to each of these boundaries using $L - 1$ comparators, the outputs of which drive coding logic to produce the desired code or sample $x_q[n]$.

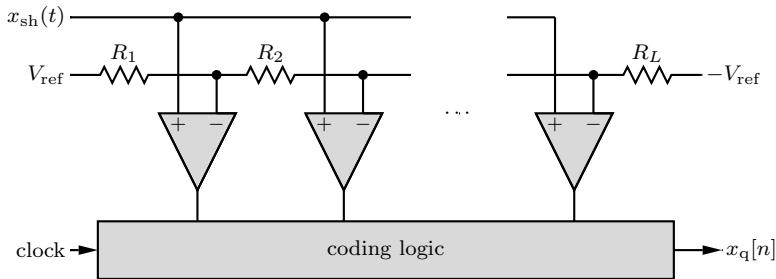


Figure 3.33: Flash converter.

Flash converters are among the fastest types. They code each sample in a single step, regardless of bit resolution. Some flash converters are so near to instantaneous in sampling that the sample-and-hold operation is unnecessary. This speed, however, comes at a significant price. A 16-bit flash converter, for example, requires 65535 comparators, making the device expensive, bulky, and power hungry.

Other Converters

There are many other A/D converter types than counting, successive approximation, and flash. Half-flash converters, for example, combine the parallelism and speed of a flash conversion with a two-step successive conversion. The result is a nice balance between complexity and speed. A 16-bit half-flash converter, for example, requires only $2^{B/2} - 1 = 2^8 - 1 = 255$ comparators and completes

its conversion in just two steps. Oversampling converters, such as sigma-delta ($\Sigma-\Delta$) converters, are also quite popular. These converters initially sample much faster than F_s but at a resolution much less than B bits (commonly at just 1-bit resolution). Following some signal processing, they output a B -bit signal at the desired rate F_s . Certain specialized applications utilize rather exotic converters, such as multistage noise shaping (MASH) converters, bandpass $\Sigma-\Delta$ modulators, clockless architectures, and others.

▷ Drill 3.16 (Flash Converters)

If an asymmetric bipolar flash converter, such as shown in Fig. 3.33, has $R_1 = R_2 = \dots = R_L$, is the converter rounding or truncating? Leaving all other resistors as R , what happens if R_1 is changed to $\frac{3}{2}R$ and R_L is changed to $\frac{1}{2}R$? □

3.7 Digital-to-Analog Conversion

Digital-to-analog (D/A) conversion essentially follows the steps in A/D conversion in reverse order, producing a continuous-time analog signal from a stream of digital samples. A block representation of the D/A conversion process is shown in Fig. 3.34. To begin, we decode the quantized samples $x_q[n]$, each of which is most commonly represented by a B -bit code word. Conceptually, this step produces an impulse train $\hat{x}_\delta(t)$, which is an estimate of the original nonquantized impulse-sampled signal $x_\delta(t)$. As long as the number of bits B is large (quantization error is small), $x_\delta(t)$ and $\hat{x}_\delta(t)$ will be nearly equivalent (as shown in Fig. 3.34). The impulses $\hat{x}_\delta(t)$ are next applied to a ZOH circuit that holds the sample value until the next sample arrives. The impulse response of a practical ZOH circuit is shown in Fig. 3.30b. The output of the hold circuit $\hat{x}_{zoh}(t)$ is a staircase approximation of the analog signal $x(t)$. This output is identical to that in Fig. 3.30c, except that in this case the samples are quantized, while those in Fig. 3.30c are the actual sample values. In addition to any error caused by quantization, the sharp edges of the staircase approximation represent distortion as well. Fortunately, much of this latter distortion can be corrected by a *reconstruction or anti-imaging filter*. With a reasonably large number of bits B and a good reconstruction filter, the final output $\hat{x}(t)$ closely approximates (within a delay $T/2$) the original signal $x(t)$. In such cases, it is common to write the output of a D/A converter as $x(t)$ rather than $\hat{x}(t)$.

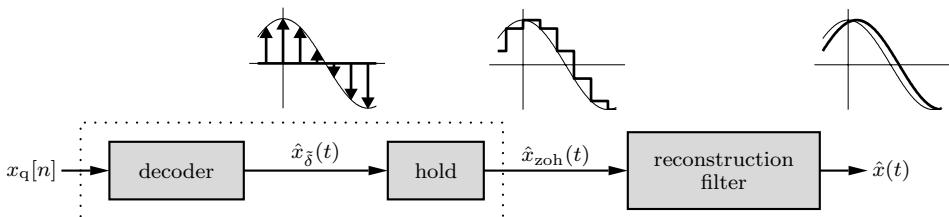


Figure 3.34: Block representation of a practical D/A converter.

Often, the decoder and ZOH operations are combined in a single circuit. Figure 3.35 shows a typical case for a unipolar D/A converter. The bits $c_{B-1}, c_{B-2}, \dots, c_0$ that comprise the code word for $x_q[n]$ each control a switch. For a low logic (0), the switch connects to ground. For a high logic (1), the switch connects to the inverting terminal of the op-amp, which is at virtual ground. Either way, the switch appears connected to ground. Thus, using the rules for connections of resistors, the load seen by V_{ref} is $2R$. The total current i is $V_{ref}/2R$. This current splits evenly at the first branch, yielding $i_{B-1} = i/2$. Similarly, we find $i_{B-2} = i/4, \dots, i_1 = i/2^{B-1}$, and $i_0 = i/2^B$. The currents from all branches switched to the inverting terminal (logic-1 bits) sum together and produce, by

means of flowing through the $2R$ feedback resistor, the output voltage $-\hat{x}_{\text{zoh}}(t)$. Mathematically,

$$-\hat{x}_{\text{zoh}}(t) = -i_{\text{sum}}2R = -\sum_{b=0}^{B-1} c_b \frac{i}{2^{B-b}} 2R = -V_{\text{ref}} \sum_{b=0}^{B-1} c_b 2^{b-B}. \quad (3.28)$$

Within a sign factor, this output is identical to Eq. (3.26).

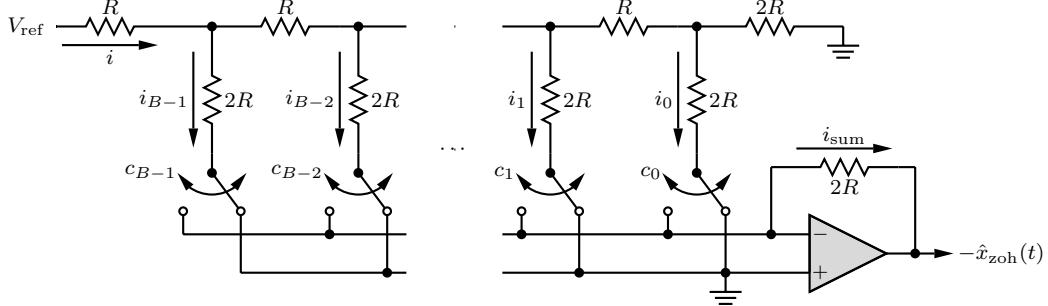


Figure 3.35: Unipolar D/A converter circuit.

3.7.1 Sources of Distortion in Signal Reconstruction

As shown in Sec. 3.2, to reconstruct a lowpass signal $x(t)$ of bandwidth $B < F_s/2$ Hz from its samples, we need to pass the samples through an ideal lowpass filter with bandwidth between B and $F_s/2$ Hz. As we have seen, however, practical D/A decoders use zero-order hold circuits rather than ideal lowpass filters, which are not realizable. Consequently, two types of distortion impact the reconstruction of signal $x(t)$ from its quantized samples. The first error is due to the use of quantized samples instead of the actual samples. This quantization error can be reduced as much as desired by increasing the number of quantization levels. The second source of error or distortion stems from the use of a hold circuit rather than an ideal lowpass filter. We note here that a hold circuit is also a lowpass filter, albeit not the ideal lowpass filter that is required for perfect signal reconstruction. The distortion due to a hold circuit, known as *aperture effect*, can be corrected by using a proper reconstruction filter with a frequency response that is the reciprocal of the frequency response of the hold circuit. We shall analyze each of these errors.

The Aperture Effect: Distortion from the Zero-Order Hold

As explained earlier, the Fig. 3.11 construction discussed in Sec. 3.2 is a noncausal version of a ZOH. By delaying this impulse response by $T/2$ ($T = 1/F_s$), we obtain the causal (and realizable) response used in A/D and D/A operations (Figs. 3.30a and 3.34). The delay adds a factor $e^{-j\omega T/2}$ to the frequency response in Eq. (3.10). Hence, the realizable ZOH transfer function $H_{\text{zoh}}(\omega)$ is given by

$$H_{\text{zoh}}(\omega) = T \operatorname{sinc}\left(\frac{\omega T}{2\pi}\right) e^{-j\omega T/2}. \quad (3.29)$$

We now place a reconstruction filter $H(\omega)$ in cascade with the ZOH circuit in an attempt to make the total response $H_{\text{zoh}}(\omega)H(\omega)$ equivalent to an ideal lowpass filter $H_{\text{lpf}}(\omega) = T \Pi\left(\frac{\omega}{2\pi F_s}\right)$. Substituting Eq. (3.29) and solving for $H_{\text{lpf}}(\omega)/H_{\text{zoh}}(\omega)$, we obtain

$$\frac{H_{\text{lpf}}(\omega)}{H_{\text{zoh}}(\omega)} = \frac{\Pi\left(\frac{\omega T}{2\pi}\right) e^{j\omega T/2}}{\operatorname{sinc}\left(\frac{\omega T}{2\pi}\right)}.$$

The factor $e^{j\omega T/2}$ in the numerator represents a time advance by $T/2$, which is unrealizable. We ignore this factor and accept a time delay of $T/2$ in the final output. The resulting reconstruction filter transfer function $H(\omega)$ is given by

$$H(\omega) = \frac{\Pi\left(\frac{\omega T}{2\pi}\right)}{\text{sinc}\left(\frac{\omega T}{2\pi}\right)} = \begin{cases} \frac{\omega T/2}{\sin(\omega T/2)} & |\omega/2\pi| < F_s/2 \\ 0 & |\omega/2\pi| > F_s/2 \end{cases}. \quad (3.30)$$

As shown in Fig. 3.36, $H(\omega)$ is the inverse of $\text{sinc}(\omega T/2\pi)$ over the band 0 to $1/2T = F_s/2$ Hz, and is 0 outside this band.

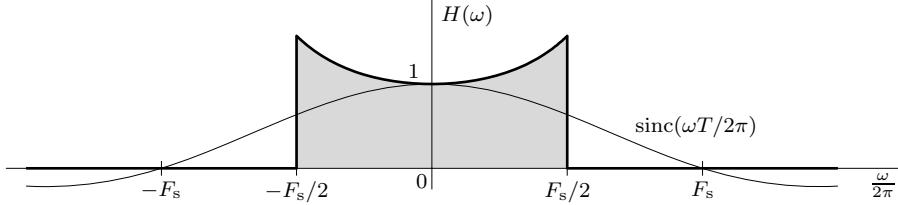


Figure 3.36: D/A reconstruction filter $H(\omega) = \Pi(\omega T/2\pi)/\text{sinc}(\omega T/2\pi)$.

Since the hold circuit gain is of the form $\sin(x)/x$, the reconstruction filter compensates this distortion with gain of the form $x/\sin(x)$. The reconstruction filter has zero gain outside the passband, as required of the ideal filter. Thus, the reconstruction filter has two functions: correcting the $\sin(x)/x$ distortion and lowpass filtering. The infinitely sharp cutoff of the filter in Fig. 3.36 is, of course, itself unrealizable. In practice, however, the sampling rate typically exceeds the Nyquist rate, which creates a gap that allows the filter to have gradual transition band characteristics.

The two functions of the correction filter can also be tackled using digital filters and techniques. For example, a digital filter with gain of the form $x/\sin(x)$ can be placed just before D/A conversion. The spectral cutoff beyond B Hz can also be facilitated by digitally increasing the sampling rate, also prior to D/A conversion. As seen earlier, higher sampling rates make it easier to closely realize the function of an ideal filter using a gradual cutoff filter (see Fig. 3.12). Indeed, if the sampling rate is high enough, the $\sin(x)/x$ distortion is negligible, and even the spectral cutoff beyond B Hz is unnecessary because the hold operation itself, being a lowpass filter, provides this function. Digital processing can increase the sampling rate internally through an interpolation operation, discussed later in Chs. 4 and 6.

Distortion from Linear Quantization

To better understand the general distortion caused by linear quantization, where each quantization level is separated by Δ , we consider the particular case where quantization errors occupy the range $(-\Delta/2, \Delta/2)$. For a large number of levels L , the quantization error e_q is equally likely to take any value in the range $(-\Delta/2, \Delta/2)$. In this case, the mean square quantization error E_q is given by[†]

$$E_q = \frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} e_q^2 de_q = \frac{\Delta^2}{12}.$$

[†]Those who are familiar with the theory of probability can derive this result directly by noting that because the area under a probability density function must be unity, the probability density of the quantization error e_q is $f(e_q) = 1/\Delta$ over the range $|e_q| \leq \Delta/2$ and is zero elsewhere. Hence,

$$E_q = \int_{-\Delta/2}^{\Delta/2} e_q^2 f(e_q) de_q = \int_{-\Delta/2}^{\Delta/2} \frac{e_q^2}{\Delta} de_q = \frac{\Delta^2}{12}.$$

Given a sufficiently large number of samples, E_q represents the energy (power) of the quantization noise. Noting that $\Delta = 2V_{\text{ref}}/L$ and $L = 2^B$, the mean square quantization noise is thus

$$E_q = \frac{V_{\text{ref}}^2}{3L^2} = \frac{V_{\text{ref}}^2/3}{4^B}. \quad (3.31)$$

Clearly, the quantization noise can be reduced as much as needed by suitably increasing the number of bits B , which also increases the number of quantization levels L .

▷ **Example 3.11 (Quantization Noise Calculations)**

A periodic sinusoidal signal $x(t)$ of amplitude 12 volts is quantized and binary coded. Assuming the signal-to-quantization noise ratio must exceed 20000, determine the smallest number of bits B needed to encode each sample.

Using Eq. (3.31), the quantization noise power is

$$E_q = \frac{V_{\text{ref}}^2/3}{4^B}.$$

To minimize quantization errors, the converter reference should be set equal to the maximum signal amplitude, or $V_{\text{ref}} = 12$. Thus, the sinusoidal signal $x(t)$ has power $P_x = \frac{V_{\text{ref}}^2}{2}$, and the signal-to-quantization noise ratio requirement is

$$P_x/E_q = \frac{V_{\text{ref}}^2/2}{V_{\text{ref}}^2 4^{-B}/3} = \frac{3}{2} 4^B \geq 20000.$$

Solving for B yields

$$B \geq \log(40000/3)/\log(4) = 6.8514.$$

Because B must be an integer, we chose $B = 7$ as the smallest number of bits required to encode each sample.

Example 3.11 ◇

3.8 Summary

This chapter details the process of converting a continuous-time signal to a discrete-time signal (sampling) and the reverse (reconstruction). Ideal sampling records a signal at discrete instants and, in the frequency domain, causes a periodic replication of the original signal spectrum. A primary goal in proper sampling is to preserve signal information. To this end, the sampling theorem states that a signal bandlimited to B Hz can be reconstructed exactly from its samples if the sampling rate $F_s > 2B$ Hz (Nyquist criterion).

In the frequency domain, the sampling theorem requires that the spectral replicates be nonoverlapping. In such cases, signal reconstruction is accomplished by applying a filter to select, among all replicates, the original spectrum. Such a reconstruction, although possible theoretically, poses practical problems such as the need for ideal filters, which are unrealizable. In practice, therefore, there is always some error in reconstructing a signal from its samples.

Practical signals, being timelimited, are not bandlimited. When sampled, the spectral replicates overlap and display aliasing, where high-frequency components appear lower in frequency. Samples of a real sinusoid of frequency $F_s/2 + f_1$ Hz appear as samples of a sinusoid of lower frequency $F_s/2 - f_1$ Hz. Aliasing errors are largely eliminated by bandlimiting a signal to half the sampling frequency, $F_s/2$ Hz. Such bandlimiting, done prior to sampling, is accomplished with an anti-aliasing filter. On occasion, such as the sampling of bandpass signals, aliasing is used to the advantage of the system.

The sampling theorem is important in signal analysis, processing, and transmission because it allows us to replace a continuous-time signal with a discrete sequence of numbers. It becomes possible, and often advantageous, to process continuous-time signals using discrete-time systems, including digital filters. In the communications field, the transmission of a continuous-time message reduces to the transmission of a sequence of numbers. This opens the door to modern techniques of communicating continuous-time signals by pulse trains.

The dual of the sampling theorem states that for a signal timelimited to τ seconds, its spectrum $X(\omega)$ can be reconstructed from the samples of $X(\omega)$ taken at uniform intervals no greater than $1/\tau$ Hz. In other words, the spectrum should be sampled at a rate not less than τ samples/Hz.

As the name implies, analog-to-digital converters not only sample but also quantize a signal. Quantization is a necessary step to allow practical digital signal processing, which would otherwise require an infinite number of bits just to represent a single sample. Although there are many quantization strategies, most converters perform linear quantization and encode each sample using B binary digits. Different A/D converter architectures are available, each with advantages and disadvantages, although all A/D converters are susceptible to errors, such as static and dynamic errors. Digital-to-analog conversion is the reverse process of A/D conversion. Distortions during D/A conversion, including quantization and aperture effects, are largely controllable through proper choice of bit resolution, sampling rate, and reconstruction filter.

With a solid understanding of sampling principles, we are now ready to undertake the general topic of discrete-time signals and systems, which is the subject of the next chapter.

References

1. Linden, D. A., "A Discussion of Sampling Theorems," *Proc. IRE*, Vol. 47, July 1959, pp. 1219–1226.
2. Siebert, W. M., *Circuits, Signals, and Systems*, MIT/McGraw-Hill, New York, 1986.
3. Bennett, W. R., *Introduction to Signal Transmission*, McGraw-Hill, New York, 1970.
4. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, NY, 2005.

Problems

3.1-1 Typically, Nyquist sampling, where $F_s = 2B$, is sufficient unless $X(\omega)$ contains impulses (or its derivatives) at the highest frequency B . Show that $x(t) = \cos(2\pi Bt)$, which contains impulses at frequency B , is an exception and can be recovered exactly using Nyquist sampling. Further, show that the signal $x(t) = \sin(2\pi Bt)$ is completely lost when it is Nyquist sampled. What happens if $x(t) = \cos(2\pi Bt + \phi)$?

3.1-2 Figure P3.1-2 shows Fourier spectra of real signals $x_1(t)$ and $x_2(t)$. Determine the Nyquist sampling rates for signals

- | | |
|--------------------|------------------|
| (a) $x_1(t)$ | (b) $x_2(t/2)$ |
| (c) $x_1^2(3t)$ | (d) $x_2^3(t)$ |
| (e) $x_1(t)x_2(t)$ | (f) $1 - x_1(t)$ |

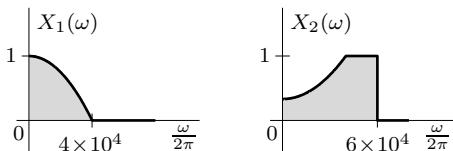


Figure P3.1-2

3.1-3 Determine the Nyquist sampling rate and the Nyquist sampling interval for

- (a) $x_a(t) = \text{sinc}^2(100t)$
- (b) $x_b(t) = 0.01 \text{sinc}^2(100t)$
- (c) $x_c(t) = \text{sinc}(100t) + 3\text{sinc}^2(60t)$
- (d) $x_d(t) = \text{sinc}(50t) \text{sinc}(100t)$

3.1-4 Consider the signal $x(t) = 3\cos(6\pi t) + \sin(18\pi t) + 2\cos[(28 - \epsilon)\pi t]$, where ϵ is a very small number $\rightarrow 0$.

- (a) Sketch $X(\omega)$, the spectrum of $x(t)$. Determine the minimum sampling rate required to be able to reconstruct $x(t)$ from these samples.
- (b) Does the minimum sampling rate of part (a) change if $\epsilon = 0$? Explain.
- (c) Sketch the spectrum of the sampled signal when the sampling rate is 25% above the Nyquist rate (show the

spectrum over the frequency range ± 50 Hz only). How would you reconstruct $x(t)$ from these samples?

3.1-5 Find the spectrum $X_{\tilde{\delta}}(\omega)$ of the impulse-sampled signal $x_{\tilde{\delta}}(t) = x(t)\tilde{\delta}(t)$, where the sampling train $\tilde{\delta}(t)$ consists of shifted unit impulses at instants $nT + \tau$ instead of at nT (for all positive and negative integer values of n).

3.1-6 A signal is bandlimited to 12 kHz. The band between 9 and 12 kHz has been corrupted by excessive noise to the point that the information within this band is nonrecoverable. Determine the minimum sampling rate for this signal so that the uncorrupted portion of the band can be recovered. If we filter out the corrupted spectrum prior to sampling, determine the minimum sampling rate.

3.1-7 A continuous-time signal $x(t) = \Lambda\left(\frac{t-1}{2}\right)$ is sampled at rates of (a) 10 Hz, (b) 2 Hz, and (c) 1 Hz. Sketch the resulting sampled signals. Because $x(t)$ is timelimited, its bandwidth is infinite. However, most of its energy is concentrated in a small band. Determine a reasonable minimum sampling rate that allows reconstruction of this signal with a small error. Carefully justify what you consider negligible or small error.

3.1-8 Consider the signal $x(t) = \cos(20\pi t) + 5\text{sinc}^2(5t)$.

- (a) Determine and sketch the spectrum $X(\omega)$ of signal $x(t)$ when sampled at a rate of 10 Hz. Can $x(t)$ be reconstructed by lowpass filtering the sampled signal? Explain.
- (b) Repeat part (a) for the sampling frequency $F_s = 20$ Hz.
- (c) Repeat part (a) for the sampling frequency $F_s = 21$ Hz.

3.1-9 Refer to Fig. P3.1-9 for plots of the bandpass spectra $X(\omega)$ and $Y(\omega)$.

- (a) The highest frequency in $X(\omega)$ is 30 Hz. According to the Nyquist criterion (Sec. 3.1), the minimum sampling frequency needed to sample $x(t)$ is 60 Hz. Letting the sampling rate $F_s = 60$

Hz, sketch the spectrum $X_{\tilde{\delta}}(\omega)$ of the sampled signal $x_{\tilde{\delta}}(t)$. Can you reconstruct $x(t)$ from these samples? How?

- (b) A certain student looks at $X(\omega)$ and concludes, since its bandwidth is only 10 Hz, that a sampling rate of 20 Hz is adequate for sampling $x(t)$. Using the sampling rate $F_s = 20$ Hz, sketch the spectrum $X_{\tilde{\delta}}(\omega)$ of the sampled signal $x_{\tilde{\delta}}(t)$. Can she reconstruct $x(t)$ from these samples? Explain.
- (c) The same student, using the same reasoning, looks at spectrum $Y(\omega)$ and again concludes that she can use a sampling rate of 20 Hz. Setting $F_s = 20$ Hz, sketch the spectrum $Y_{\tilde{\delta}}(\omega)$ of the sampled signal $y_{\tilde{\delta}}(t)$. Can she reconstruct $y(t)$ from these samples? Explain.

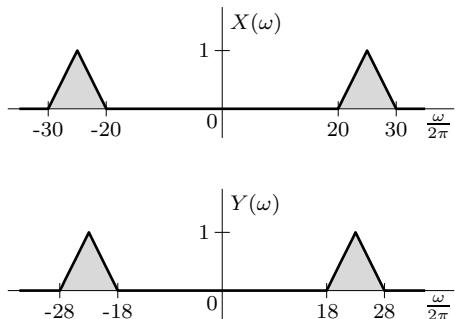


Figure P3.1-9

- 3.1-10** A signal $x(t)$ with spectrum $X(\omega)$ is shown in Fig. P3.1-10. By inspection of $X(\omega)$, determine all the sample values $x(nT)$ when $F_s = 1/T = f_1 + f_2$ Hz.

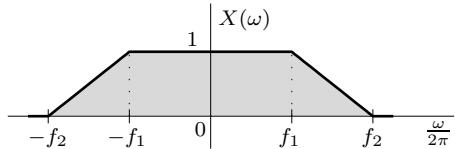


Figure P3.1-10

- 3.1-11** In digital communications, it is important to know the upper theoretical limit on the rate of digital pulses that can be transmitted over a channel of bandwidth B Hz. In digital transmission, the relative shape

of the pulse is not important. We are interested in knowing only the amplitude represented by the pulse. In binary communications where pulse amplitudes are taken as 1 or -1 , each pulse represents one piece of information. Consider one independent amplitude value (not necessarily binary) as one piece of information. Show that $2B$ independent pieces of information per second can be transmitted correctly (assuming no noise) over a channel of bandwidth B Hz. Prove also the converse that a channel of bandwidth B Hz can transmit at most $2B$ independent pieces of information per second correctly (assuming no noise). This important principle in communication theory states that 1 Hz of bandwidth can transmit two independent pieces of information per second.

- 3.1-12** A pulse $p(t) = \Pi(1250t)$ is periodically replicated to form the pulse train shown in Fig. P3.1-12. Mathematically,

$$\tilde{p}(t) = \sum_{n=-\infty}^{\infty} p(t - 125n).$$

Using $x(t) = \text{sinc}(200t)$, find and sketch the spectrum of the sampled signal $x_{\tilde{p}}(t) = \tilde{p}(t)x(t)$. Explain whether it is possible to reconstruct $x(t)$ from these samples. If the sampled signal is passed through an ideal lowpass filter of bandwidth 100 Hz and unit gain, find the filter output. What is the filter output if its bandwidth B Hz is between 100 and 150 Hz? What happens if the filter bandwidth exceeds 150 Hz?

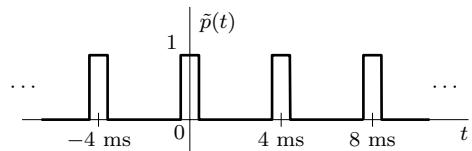


Figure P3.1-12

- 3.2-1** (a) Show that the signal $x(t)$ reconstructed from its samples $x(nT)$ using Eq. (3.13) has a bandwidth $B \leq 1/2T$ Hz.
(b) Show that $x(t)$ is the smallest bandwidth signal that passes through sam-

ples $x(nT)$. Hint: Use *reductio ad absurdum* (proof by contradiction).

- 3.2-2** Derive the reconstruction formula of Eq. (3.13). To do so, substitute Eqs. (3.3) and (3.12) into the inverse Fourier transform of Eq. (3.11) and simplify the resulting expression.

- 3.2-3** Show that the block diagram in Fig. P3.2-3 represents a realization of a causal ZOH operation. You can do this by showing that the unit impulse response $h(t)$ of this system is $\Pi\left(\frac{t-T/2}{T}\right)$.

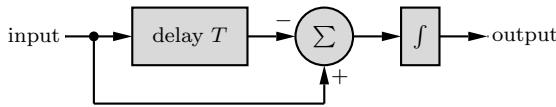


Figure P3.2-3

- 3.2-4** A *first-order hold* (FOH) circuit can also be used to reconstruct a signal $x(t)$ from its samples. The impulse response of this circuit is

$$h(t) = \Lambda\left(\frac{t}{2T}\right),$$

where T is the sampling interval.

- (a) Using an impulse-sampled signal $x_{\delta}(t)$, show that the FOH circuit performs *linear interpolation*. In other words, the filter output consists of samples connected by straight-line segments.
- (b) Determine the frequency and magnitude responses of the FOH filter. Compare these responses with both the ideal and ZOH reconstruction filters.
- (c) This filter, being noncausal, is unrealizable. By delaying its impulse response, the filter can be made realizable. What is the minimum delay required to make it realizable? How does this delay affect the reconstructed signal and the filter frequency response?
- (d) Show that the causal FOH circuit in part (c) can be realized by a cascade of two ZOH circuits, where each ZOH is constructed as shown in Fig. P3.2-3.

- 3.2-5** A *predictive first-order hold* circuit is also sometimes used to reconstruct a signal $x(t)$ from its samples. The impulse response of this circuit is

$$h(t) = \begin{cases} 1 + t/T & 0 \leq t < T \\ 1 - t/T & T \leq t < 2T \\ 0 & \text{otherwise} \end{cases},$$

where T is the sampling interval.

- (a) Sketch $h(t)$. Is this system causal?
- (b) Pick an example sampled signal $x_{\delta}(t)$ and show the corresponding output of the predictive FOH operation. Does the name “predictive first-order hold” make sense? Explain.
- (c) Plot the magnitude responses of the predictive FOH filter. How does the character of this circuit’s output compare with the original signal $x(t)$.

- 3.2-6** In Sec. 3.1.1, we used timelimited pulses, each with width less than the sampling interval T , to achieve practical sampling. Show that it is not necessary to restrict the sampling pulse width. We can use sampling pulses of arbitrarily large duration and still be able to reconstruct the signal $x(t)$ as long as the pulse rate is no less than the Nyquist rate for $x(t)$.

Next, consider signal $x(t)$, which is bandlimited to B Hz. Define a sampling pulse as $p(t) = e^{-at}u(t)$. Find the spectrum $X_{\tilde{p}}(\omega)$ of the sampled signal $x_{\tilde{p}}(t)$, which is formed as the product of $x(t)$ and $\tilde{p}(t)$, the T -periodic replication of $p(t)$. Show that $x(t)$ can be reconstructed from $x_{\tilde{p}}(t)$ provided that the sampling rate is no less than $2B$ Hz. Explain how you would reconstruct $x(t)$ from the sampled signal.

- 3.2-7** In Ex. 3.2, the sampling of a signal $x(t)$ was accomplished by multiplying the signal by a pulse train $\tilde{p}(t)$, resulting in the sampled signal depicted in Fig. 3.8c. This procedure is known as *natural sampling*. Figure P3.2-7 shows the so-called *flat-top sampling* of the same signal $x(t) = \text{sinc}^2(5t)$.

- (a) Show that the signal $x(t)$ can be recovered from flat-top samples if the sampling rate is no less than the Nyquist rate.

- (b) Explain how you would recover $x(t)$ from its flat-top samples.
- (c) Find and sketch the sampled signal spectrum $X_{\tilde{p}}(\omega)$.

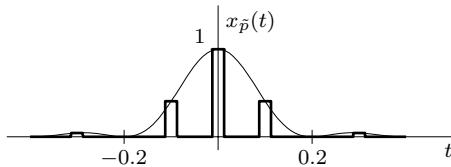


Figure P3.2-7

3.2-8 A sinusoid of frequency f_0 Hz is sampled at a rate $f_s = 20$ Hz. Find the apparent frequency of the sampled signal if f_0 is

- | | |
|-----------|-----------|
| (a) 8 Hz | (b) 12 Hz |
| (c) 20 Hz | (d) 21 Hz |
| (e) 22 Hz | (f) 32 Hz |

3.2-9 A sinusoid of unknown frequency f_0 is sampled at a rate 60 Hz. The apparent frequency of the sampled signal is 20 Hz. Determine f_0 if it is known that f_0 lies in the range

- | | |
|-----------------|------------------|
| (a) 0 to 30 Hz | (b) 30 to 60 Hz |
| (c) 60 to 90 Hz | (d) 90 to 120 Hz |

3.2-10 Using a logical or graphical argument, prove Eq. (3.14). Why can't apparent frequency be computed as $f_a = \langle f_0 \rangle_{F_s}$?

3.2-11 A signal $x(t) = 3 \cos(6\pi t) + \cos(16\pi t) + 2 \cos(20\pi t)$ is sampled at a rate 25% above the Nyquist rate. Sketch the spectrum of the sampled signal. How would you reconstruct $x(t)$ from these samples? If the sampling frequency is 25% below the Nyquist rate, what are the frequencies of the sinusoids present in the output of the filter with cutoff frequency equal to the folding frequency? Do not write the actual output; give just the frequencies of the sinusoids present in the output.

3.2-12 This example is one of those interesting situations, leading to a curious result in the category of defying gravity. The sinc function can be recovered from its samples

taken at extremely low frequencies in apparent defiance of the sampling theorem.

Consider a sinc pulse $x(t) = \text{sinc}(4t)$ for which $X(\omega) = \frac{1}{4}\Pi(\omega/8\pi)$. The bandwidth of $x(t)$ is $B = 2$ Hz, and its Nyquist rate is 4 Hz.

- (a) Sample $x(t)$ at a rate 4 Hz, and sketch the spectrum of the sampled signal.
- (b) To recover $x(t)$ from its samples, we pass the sampled signal through an ideal lowpass filter of bandwidth $B = 2$ Hz and gain $G = T = 1/4$. Sketch this system, and show that for this system $H(\omega) = \frac{1}{4}\Pi(\omega/8\pi)$. Show also that when the input is the sampled $x(t)$ at a rate 4 Hz, the output of this system is indeed $x(t)$, as expected.
- (c) Now sample $x(t)$ at 2 Hz, which is half the Nyquist rate. Apply this sampled signal at the input of the lowpass filter used in part (b). Find the output.
- (d) Repeat part (c) for the sampling rate 1 Hz.
- (e) Show that the response to the sampled $x(t)$ of the lowpass filter in part (b) is $x(t)$ if the sampling rate is $2/N$, where N is any positive integer. This means that we can recover $x(t)$ from its samples taken at arbitrarily small rate by letting $N \rightarrow \infty$.
- (f) The mystery may be clarified a bit if you examine the problem in the time domain. Find the samples of $x(t)$ when the sampling rate is $2/N$ (N integer).

3.2-13 In digital communication systems, transmission of digital data is encoded using bandlimited pulses in order to utilize the channel bandwidth efficiently. Unfortunately, bandlimited pulses are non-timelimited; they have infinite duration, which causes pulses representing successive digits to interfere and possibly cause errors. This difficulty can be resolved by shaping a pulse $p(t)$ in such a way that it is bandlimited yet causes zero interference at the sampling instants. To transmit R pulses/second, we require a minimum bandwidth $R/2$ Hz (see Prob. 3.1-11).

The bandwidth of $p(t)$ should be $R/2$ Hz, and its samples, in order to cause no interference at all other sampling instants, must satisfy the condition

$$p(nT) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases},$$

where $T = \frac{1}{R}$. Because the pulse rate is R pulses per second, the sampling instants are located at intervals of $1/R$ s. Hence, the preceding condition ensures that a pulse at any instant will not interfere with the amplitude of any other pulse at its center. Find $p(t)$. Is $p(t)$ unique in the sense that no other pulse satisfies the given requirements?

3.2-14 The problem of pulse interference in digital data transmission is outlined in Prob. 3.2-13, which shapes the pulse $p(t)$ to eliminate interference. Unfortunately, the pulse found is not only noncausal (and unrealizable) but also has a serious drawback that because of its slow decay (as $1/t$), it is prone to severe interference. To make the pulse decay more rapidly, Nyquist proposed relaxing the bandwidth requirement from $R/2$ to $kR/2$ Hz and $1 \leq k \leq 2$. The new pulse still requires the property of noninterference with other pulses, described mathematically as

$$p(nT) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases},$$

where $T = \frac{1}{R}$. Show that this condition is satisfied only if the pulse spectrum $P(\omega)$ has an odd-like symmetry over $0 \leq \omega \leq R$ and about the set of dotted axes shown in Fig. P3.2-14. The bandwidth of $P(\omega)$ is $kR/2$ Hz ($1 \leq k \leq 2$).

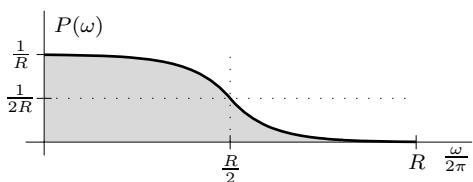


Figure P3.2-14

3.2-15 The Nyquist samples ($T = 1/2B$) of a signal $x(t)$ bandlimited to B Hz are

$$x(nT) = \begin{cases} 1 & n = 0, 1 \\ 0 & \text{all } n \neq 0, 1 \end{cases}.$$

Show that

$$x(t) = \frac{\text{sinc}(2Bt)}{1 - 2\pi Bt}.$$

This pulse, known as the *duobinary pulse*, is used in digital transmission applications.

3.2-16 A signal bandlimited to B Hz is sampled at a rate $F_s > 2B$ Hz. Show that

$$\int_{-\infty}^{\infty} x(t) dt = T \sum_{n=-\infty}^{\infty} x(nT)$$

and

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = T \sum_{n=-\infty}^{\infty} |x(nT)|^2.$$

Hint: Use Eq. (3.13) and Fourier transform properties.

3.2-17 Prove that a signal cannot be simultaneously timelimited and bandlimited.

Hint: Assume that a signal is simultaneously timelimited and bandlimited so that $X(\omega) = 0$ for $|\omega| \geq 2\pi B$. In this case, $X(\omega) = X(\omega)\Pi(\frac{\omega}{4\pi B'})$ for $B' > B$. Inverse transform the right-hand expression to show that the corresponding time-domain waveform cannot be timelimited.

3.2-18 The signal $x(t) = \sin(0.7\pi t) \cos(0.5\pi t)$ is sampled using $F_s = 1$ Hz to yield a discrete-time signal $x[n]$. Next, $x[n]$ is filtered using an ideal high-pass digital filter that eliminates all frequencies below $\frac{3}{10}F_s$, the output of which is called $y[n]$. Finally, $y[n]$ is passed through a perfect reconstruction filter at the rate $F_s = \frac{1}{2}$ Hz. Find a simplified expression for $y(t)$, the output of the reconstructor. Can this system operate in “real time”?

3.2-19 Which, if either, is a better anti-aliasing filter: a Chebyshev filter or an inverse Chebyshev filter? Fully justify your answer.

3.3-1 Consider the following specifications for bandpass signals. For each case, find the minimum permissible sampling rate and the ranges of sampling frequency that will allow reconstruction of the bandpass signal from its uniform samples.

- (a) $f_1 = 50$ kHz and $f_2 = 70$ kHz
- (b) $f_1 = 17.5$ kHz and $f_2 = 22.5$ kHz
- (c) $f_2 = 25$ kHz and $B = 10$ kHz
- (d) $f_1 = 4$ kHz and $B = 6$ kHz
- (e) $f_1 = 50$ kHz and $f_2 = 100$ kHz

3.3-2 A bandpass signal has bandwidth of 5 kHz centered at $f_c = 20$ kHz. Determine the permissible ranges of sampling frequency so that the original signal can be recovered from its samples. Redo the problem if the frequencies outside the band from 18 to 22 kHz have been corrupted by excessive noise to the point that the information contained in those frequencies is nonrecoverable. Determine the minimum sampling frequency so that the uncorrupted portion of the band can be recovered. If we filter out the corrupted spectrum prior to sampling, determine the minimum sampling rate.

3.3-3 Let $x(t)$ be a signal whose Fourier transform $X(\omega)$ is nonzero only over $3 \leq |\omega| \leq 9$. Further, only frequencies $5 \leq |\omega| \leq 7$ contain useful information (the other frequencies can be considered to contain noise or different channel information).

- (a) What is the smallest sampling rate that will enable exact reconstruction of the useful signal if we do not perform any filtering on $x(t)$ before sampling?
- (b) How will the answer of part (a) change if it is permitted to pass $x(t)$ through a filter before sampling? Explain.

3.3-4 The spectrum $X(\omega)$ for a real bandpass signal $x(t)$ is shown in Fig. P3.3-4.

- (a) Determine the permissible ranges of the sampling frequency F_s that allow the original signal to be recovered from its samples.
- (b) Sketch $X_{\tilde{x}}(\omega)$, the spectrum of the sampled signal, for the sampling frequencies $F_s = 5, 6$, and 10 kHz. Can we reconstruct $x(t)$ using these sampling rates? How?
- (c) What happens if the sampling rate F_s is slightly below 5 kHz? What if F_s is just above 6 kHz? How about the case when F_s is just below 10 kHz?

- (d) Consider the sampling rates $F_s = 5.5$ and 11 kHz. Determine what happens for both sampling rates, and state, with reasons, which rate is preferable.

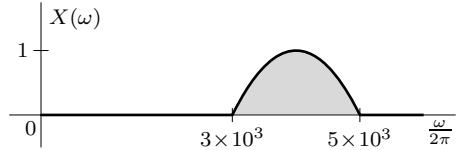


Figure P3.3-4

3.3-5 A sinusoid $\cos(\omega_0 t)$ is a bandpass signal with zero bandwidth. This implies that the sampling rate that will allow reconstruction of this signal from its samples can be arbitrarily small. Show that this is indeed the case. Describe a system that will permit reconstruction (within a constant factor) of this sinusoid from its samples taken uniformly at a sampling rate approaching zero.

3.4-1 The Fourier transform of a signal $x(t)$, bandlimited to B Hz, is $X(\omega)$. The signal $x(t)$ is repeated periodically at intervals T , where $T = 1.25/B$. The resulting signal $\tilde{x}(t)$ is

$$\tilde{x}(t) = \sum_{n=-\infty}^{\infty} x(t - nT).$$

Show that $\tilde{x}(t)$ can be expressed as

$$\tilde{x}(t) = C_0 + C_1 \cos(1.6\pi B t + \theta),$$

where

$$\begin{aligned} C_0 &= \frac{1}{T} X(0), \\ C_1 &= \frac{2}{T} |X(2\pi/T)|, \\ \text{and } \theta &= \angle X(2\pi/T). \end{aligned}$$

Recall that a bandlimited signal is not timelimited and hence has infinite duration. The periodic repetitions are all overlapping.

3.4-2 A signal $x(t)$ is timelimited to τ seconds. Its spectrum $X(\omega)$ is uniformly sampled at frequency intervals of $1/\tau$ Hz. The sample values are

$$X\left(\frac{2\pi k}{\tau}\right) = \begin{cases} 1 & k = 0, 1 \\ 0 & \text{all } k \neq 0, 1 \end{cases}.$$

Show that

$$X(\omega) = \frac{\text{sinc}\left(\frac{\omega\tau}{2\pi}\right)}{1 - \frac{\omega\tau}{2\pi}}.$$

- 3.4-3** A real signal $\tilde{x}(t)$ is constructed as a T_0 -periodic replication of signal $x(t)$. For $\omega > 0$, the spectrum of $\tilde{x}(t)$ is

$$\tilde{X}(\omega) = \sum_{k=0}^{100} (1 - \omega/10)\delta(\omega - 0.1k).$$

Determine the piece $x(t)$ and the spacing interval T_0 used to construct $\tilde{x}(t)$.

- 3.4-4** Let an energy signal $x(t)$ with Fourier transform $X(\omega)$ be used to construct a T_0 -periodic function $\tilde{x}(t)$ according to

$$\tilde{x}(t) = \sum_{n=-\infty}^{\infty} x(t - nT_0).$$

Show that the Fourier series representation of $\tilde{x}(t)$ is

$$\tilde{x}(t) = \frac{1}{T_0} \sum_{k=-\infty}^{\infty} X(k\omega_0)e^{jk\omega_0 t},$$

where $\omega_0 = 2\pi/T_0$. This formula, called the *Poisson sum formula*, tells us that the Fourier series of $\tilde{x}(t)$ is a scaled and sampled version of the Fourier transform of $x(t)$.

- 3.5-1** A compact disc (CD) records audio signals digitally using a binary code. Assume that the audio signal bandwidth is 15 kHz.

- (a) What is the Nyquist rate?
- (b) If the Nyquist samples are quantized into $L = 65536$ levels and then binary coded, what number of binary digits is required to encode a sample?
- (c) Determine the number of binary digits/s (bits/s) required to encode the audio signal.
- (d) For practical reasons discussed in the text, signals are sampled at a rate well above the Nyquist rate. Practical CDs use 44100 samples/s. If $L = 65536$, determine the number of pulses/s required to encode the signal.

- 3.5-2** A TV signal (video and audio) has a bandwidth of 4.5 MHz. This signal is sampled, quantized, and binary coded.

- (a) Determine the sampling rate if the signal is to be sampled at a rate 20% above the Nyquist rate.

- (b) If the samples are quantized into 1024 levels, determine the number of binary pulses required to encode each sample.
- (c) Determine the binary pulse rate (bits/s) of the binary coded signal.

- 3.5-3** (a) In a certain A/D scheme, there are 16 quantization levels. Give one possible binary code and one possible quaternary (4-ary) code. For the quaternary code, use 0, 1, 2, and 3 as the four symbols. Use the minimum number of digits in your code.

- (b) To represent a given number of quantization levels L , if we require a minimum of B_M digits for an M -ary code, show that the ratio of the number of digits in a binary code to the number of digits in a quaternary (4-ary) code is 2, that is, $B_2/B_4 = 2$.

- 3.5-4** Five telemetry signals, each of bandwidth 1 kHz, are quantized and binary coded. These signals are time division multiplexed (signal bits interleaved). Determine the number of quantization levels L so that the maximum error in sample amplitudes is 0.2% of the peak signal amplitude. Assuming the signals must be sampled at least 20% above the Nyquist rate, determine the data rate (bits per second) of the multiplexed signal.

- 3.5-5** An application requires a bipolar ADC to help detect a very short duration event.

- (a) Which type of ADC is likely the most appropriate for the job: counting, flash, half-flash, successive approximation, or other? Explain.
- (b) An ADC can be asymmetric or symmetric and rounding or truncating. Which type is most appropriate for this application? Explain.

- 3.5-6** A digital communication channel is capable of transmitting 56600 bits per second, about the rate of a high-speed dial-up modem at the turn of the century. We want to use this channel to transmit a single analog

signal $x(t)$ with lowpass content. The signal magnitude is limited to $|x(t)| \leq x_{\max}$. System specifications require that the error between the digitized signal and $x(t)$ must not exceed $\pm 10^{-3}x_{\max}$.

- (a) What is the required number of bits B of the ADC?
- (b) What is the maximum bandwidth of the analog input for which the channel can be reliably used without aliasing errors?
- (c) Suppose that our desired message $x(t)$ is streamed audio, which has a maximum frequency content of around 3500 Hz. Will the system work? If yes, justify how. If not, explain what you might do to best accommodate this particular message signal so that the channel can be used.

3.5-7 Assuming an input $|x(t)| \leq 1$, sketch the transfer characteristic for a 2-bit symmetric rounding converter. If, due to gain and offset errors, the input $x(t)$ appears as $\frac{2}{3}x(t) + 0.25$, sketch the resulting transfer characteristic.

3.5-8 Repeat Prob. 3.5-7 for a 2-bit asymmetric truncating converter.

3.5-9 Show that a hold circuit is not necessary for a bipolar ADC when sampling a signal whose frequency satisfies

$$f \leq F_s 2^{-(B+1)} / \pi.$$

Hint: Assume that the input is a sinusoid $x(t) = V_{\text{ref}} \sin(2\pi f t)$. For a hold to be unnecessary, the signal should not change by more than 1/2 a quantization level during any sample period.

3.6-1 A signal $x(t)$ is converted to a binary signal. If the signal-to-quantization-noise ratio (SQNR) is required to be at least 47 dB, determine the minimum number of quantization levels L required, assuming that $x(t)$ is a sinusoid. Determine the actual SQNR obtained with this minimum L .

3.6-2 Repeat Prob. 3.6-1 for the signal $x(t)$ shown in Fig. P3.6-2.

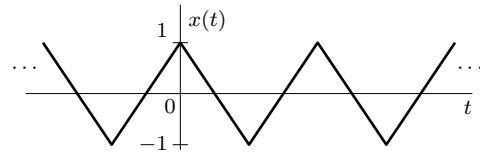


Figure P3.6-2

3.6-3 A signal $x(t)$ is converted to a 10-bit ($B = 10$) binary signal. The signal-to-quantization-noise ratio (SQNR) is found to be 30 dB. The desired SQNR is 42 dB. It is decided to increase the SQNR to the desired value by increasing the number of quantization levels L . Determine the new values of L and B .

Chapter 4

Discrete-Time Signals and Systems

This chapter examines the basics of discrete-time (DT) signals and systems. There exist many similarities as well as some important differences between discrete-time and continuous-time concepts. To encourage comparison between the two, this chapter follows a similar presentation order to Ch. 1.

Preliminaries and Connections

A *discrete-time signal* is basically a sequence of numbers. Such signals exist naturally in situations that are inherently discrete-time in nature, such as population studies, amortization problems, national income models, and radar tracking. As shown in Ch. 3, they also arise as a result of sampling continuous-time signals in sampled data systems and digital filtering. Such signals can be denoted by $x[n]$, $y[n]$, and so on, where the variable n takes integer values, and $x[n]$ denotes the n th number in the sequence labeled x . To emphasize its integer nature, this notation encloses the discrete-time variable n within square brackets instead of parenthesis, which we generally reserve for enclosing continuous-time variables such as t .

A discrete-time signal, when obtained by uniformly sampling a continuous-time signal $x(t)$, can also be expressed as $x(nT)$, where T is the sampling interval, and n again takes only integer values. Thus, $x(nT)$ denotes the value of the signal $x(t)$ at $t = nT$, and the representation $x(nT)$ is entirely equivalent to the customary discrete-time notation $x[n]$, meaning $x[n] = x(nT)$. Since it is typically more convenient, we favor the notation $x[n]$ over $x(nT)$ throughout most of this book. A typical discrete-time signal is depicted in Fig. 4.1, which shows both notations. In this figure, a continuous-time exponential $x(t) = e^{-t}$ is sampled every $T = 0.1$ seconds to produce the discrete-time signal $x(nT) = e^{-nT} = e^{-0.1n} = x[n]$. A discrete-time signal is also called a *discrete-time sequence*, and we shall use both terms interchangeably.

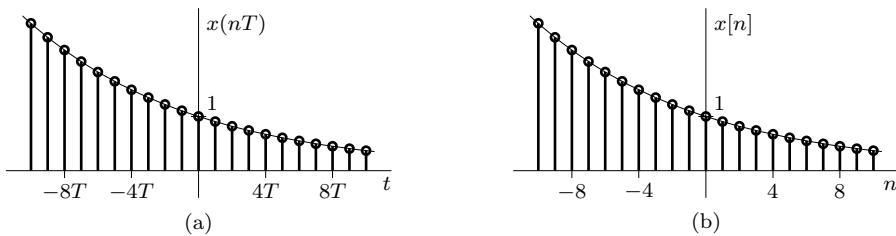


Figure 4.1: Representations of a discrete-time signal $e^{-0.1n}$: (a) $x(nT)$ and (b) $x[n]$.

Systems whose inputs and outputs are discrete-time signals are called *discrete-time systems*. A digital computer is a familiar example of this type of system. We shall concentrate on *single-input, single-output (SISO) systems*, where a DT system processes a single input sequence $x[n]$ to produce

a single output sequence $y[n]$. More complex cases, such as *multiple-input, single-output (MISO)*, *single-input, multiple-output (SIMO)*, and *multiple-input, multiple-output (MIMO) systems*, share the same foundations as SISO systems.

In many applications such as digital filtering, continuous-time signals are processed by discrete-time systems using appropriate interfaces at the input and the output, as illustrated in Fig. 4.2. A continuous-time signal $x(t)$ is first sampled to convert it into a discrete-time signal $x[n]$, which is then processed by a discrete-time system to yield the output $y[n]$. A continuous-time signal $y(t)$ is finally constructed from $y[n]$. We use the notations C/D and D/C to designate the conversions from continuous-time to discrete-time and from discrete-time to continuous-time, respectively. As we shall see later in our discussion, discrete-time systems have several advantages over continuous-time systems. For this reason, there is an accelerating trend toward processing continuous-time signals with discrete-time systems. Operating together, the three blocks in Fig. 4.2 often perform better than a purely continuous-time system.

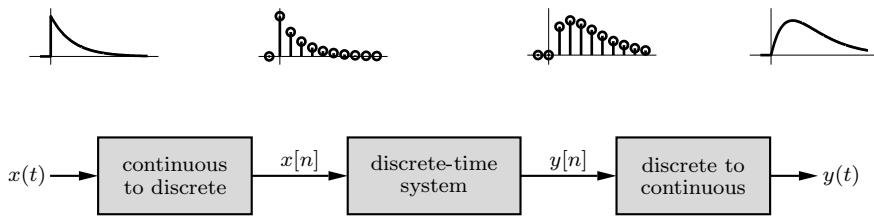


Figure 4.2: Processing a continuous-time signal by a discrete-time system.

A C/D converter samples an input $x(t)$ to yield a discrete-time signal $x[n]$ according to

$$x[n] = x(nT). \quad (4.1)$$

In contrast, a D/C converter ideally converts a digital output $y[n]$ into an analog output $y(t)$ using the interpolation formula

$$y(t) = \sum_{n=-\infty}^{\infty} y[n] \operatorname{sinc}\left(\frac{t - nT}{T}\right). \quad (4.2)$$

This expression is equivalent to the ideal interpolation formula of Eq. (3.13). In fact, the C/D and D/C operations of Fig. 4.2 are basically identical to the sampling and D/A operations discussed in Ch. 3 except that quantization effects are absent. As noted previously, it is mathematically difficult, not to mention inconvenient, to rigorously account for quantization. Further, quantization more often causes distractions than insights during the development of discrete-time signals and systems concepts. Therefore, unless stated otherwise, our treatment of discrete-time signals assumes that no quantization is present. Since most practical applications quantize signals with a relatively large number of quantization levels, this assumption is usually well justified.

Reducing Packaging Costs

An observant reader will note a strikingly close kinship of a discrete-time signal to an impulse-sampled continuous-time signal. The information inherent in the two is identical; only the packaging is different. A discrete-time signal is packaged in a very simple and spartan form: a sequence of numbers. In contrast, an impulse-sampled signal is wrapped in an exquisite package of impulses, the strengths of which follow the same sequence of numbers. Such an elegant packaging has its cost. The CT techniques used for impulse-sampled signals are often more complex than needed to handle the spartan form. For this reason, we develop a parallel set of techniques to handle discrete-time signals, techniques that are closely connected to continuous-time techniques. These discrete-time techniques are in form that is simpler to handle. The sampling theorem, aliasing, and other concepts discussed in Ch. 3 have direct relevance and application to discrete-time signals.

4.1 Operations on the Independent DT Variable

The *time-shifting*, *time-reversal*, and *time-scaling* operations discussed for continuous-time systems also apply to discrete-time systems with some modification. In general, discrete-time expansions, compressions, and shifts are restricted to integer factors. The DT time-scaling operation is particularly interesting, and we provide separate treatments for compression and expansion.

4.1.1 DT Time Shifting

Consider a signal $x[n]$ (Fig. 4.3a) and the same signal right shifted (delayed) by 5 units (Fig. 4.3b), which we shall denote by $y[n]$. Whatever occurs in $x[n]$ at some instant n also occurs in $y[n]$ five units later at the instant $n + 5$. Therefore,

$$y[n + 5] = x[n] \quad \text{and} \quad y[n] = x[n - 5].$$

In other words, the signal $x[n - 5]$ in Fig. 4.3b, being the signal in Fig. 4.3a delayed by 5 units, is the same as $x[n]$ with n replaced by $n - 5$. Now,

$$x[n] = (0.9)^n \quad \text{over} \quad 3 \leq n \leq 10.$$

Therefore,

$$y[n] = x[n - 5] = (0.9)^{n-5} \quad \text{over} \quad 3 \leq n - 5 \leq 10 \quad \text{or} \quad 8 \leq n \leq 15.$$

Figure 4.3c illustrates a shift that results in a time advance rather than a time delay.

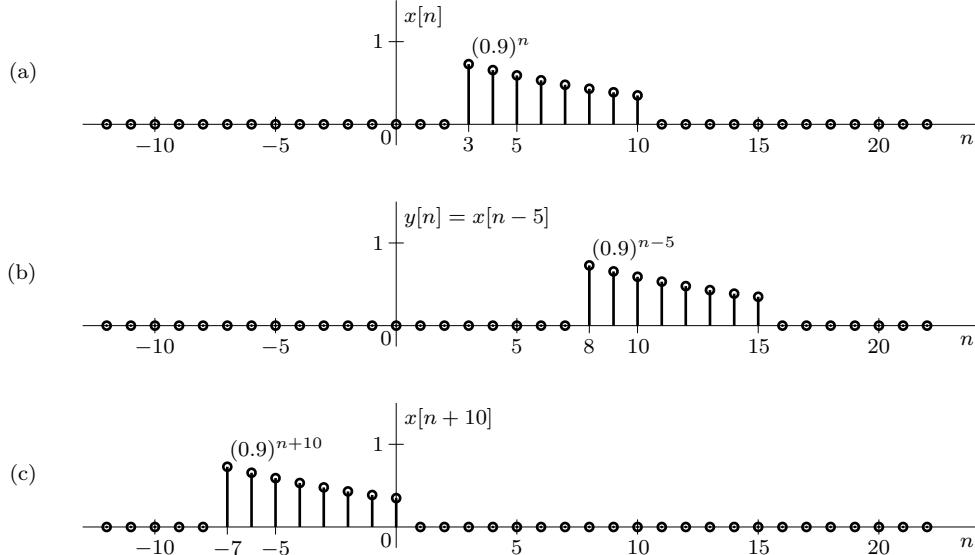


Figure 4.3: Time shifting a DT signal: (a) original signal, (b) delay by 5, and (c) advance by 10.

In general, then, to shift a sequence $x[n]$ by m units (m integer), we replace n with $n - m$ as

$$x[n] \rightarrow x[n - m]. \quad (4.3)$$

If m is positive, the shift is to the right and corresponds to a delay, such as the $m = 5$ case of Fig. 4.3b. If m is negative, such as the $m = -10$ case shown in Fig. 4.3c, the shift is to the left and

corresponds to an advance.[†] We emphasize that the DT time shift of Eq. (4.3) requires an integer shift value m .

▷ **Drill 4.1 (DT Time Shifting)**

Show that $x[n]$ in Fig. 4.3a left shifted by 3 units can be expressed as $y[n] = 0.729(0.9)^n$ for $0 \leq n \leq 7$ and is zero otherwise. Sketch the shifted signal $y[n]$.

△

4.1.2 DT Time Reversal

To invert $x[n]$ in Fig. 4.4a, imagine $x[n]$ as a rigid wire frame. Rotating this frame 180° about the vertical axis results in the desired inverted signal $y[n]$ shown in Fig. 4.4b. Whatever happens in Fig. 4.4a at some instant n also happens in Fig. 4.4b at the instant $-n$ so that

$$y[-n] = x[n] \quad \text{and} \quad y[n] = x[-n].$$

The expression for $y[n]$ is the same as that for $x[n]$ with n replaced by $-n$. For the case shown in Fig. 4.4b, $x[n] = (0.9)^n$ over $3 \leq n \leq 10$ and $y[n] = x[-n] = (0.9)^{-n}$ over $3 \leq -n \leq 10$ or, equivalently, over $-3 \geq n \geq -10$.

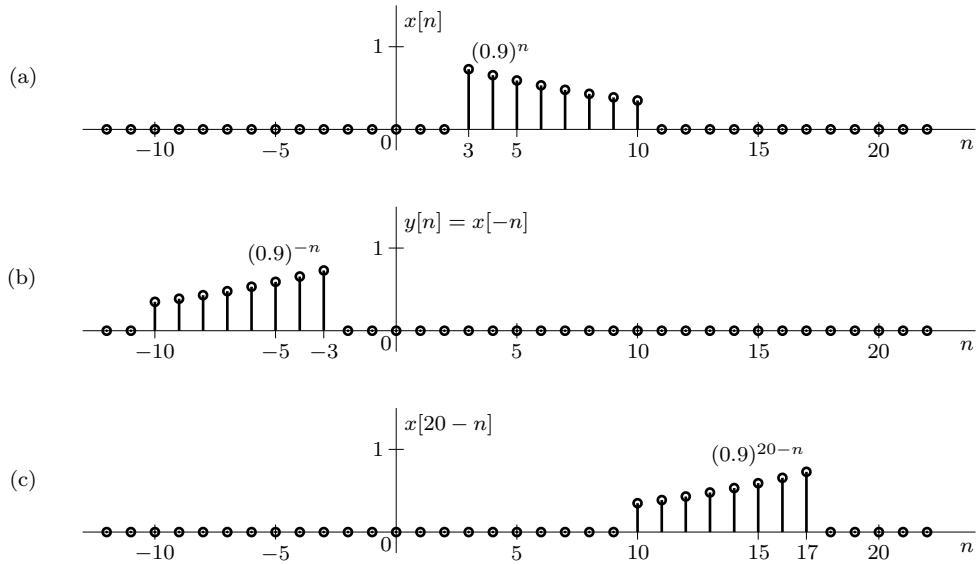


Figure 4.4: Time reversing a DT signal: (a) original signal, (b) reversal, and (c) reversal and shift by 20.

In general, then, to reverse a sequence $x[n]$, we replace n with $-n$ as

$$x[n] \rightarrow x[-n]. \quad (4.4)$$

The origin $n = 0$ is the anchor point, which remains unchanged during time reversal because at $n = 0$, $x[n] = x[-n] = x[0]$. Note that while the reversal of $x[n]$ about the vertical axis is $x[-n]$, the reversal of $x[n]$ about the horizontal axis is $-x[n]$.

[†]The terms “delay” and “advance” are meaningful when the independent variable is time. For other independent variables, such as frequency or distance, it is often more appropriate to use the terms “right shift” and “left shift.”

As in the continuous-time case, care must be taken while performing combined operations on the independent variable. Time reversal and time shifting, for example, are jointly encountered as $x[m - n]$ in discrete-time convolution, discussed later. Two steps yield $x[m - n]$ from $x[n]$. First, we time reverse the signal $x[n]$ to obtain $x[-n]$. Second, we shift $x[-n]$ by m . Recall that a time shift of m is accomplished by replacing n with $n - m$. Hence, shifting $x[-n]$ by m units is $x[-(n - m)] = x[m - n]$. To illustrate, consider using $x[n]$ in Fig. 4.4a to produce $x[20 - n]$. We first time reverse $x[n]$ to obtain $x[-n]$, as shown in Fig. 4.4b. Next, we right shift $x[-n]$ by $m = 20$ to obtain $x[m - n] = x[20 - n]$, as shown in Fig. 4.4c. It is also possible to produce the same result in a reversed order of operations. Repeating our example, we first left shift $x[n]$ to obtain $x[n + 20]$. Subsequent time reversal yields the desired result of $x[20 - n]$. The reader is encouraged to graphically verify that this alternate procedure produces the same Fig. 4.4c result.

▷ Drill 4.2 (DT Time Reversal)

Sketch the signal $x[n]$, which equals $e^{-0.5n}$ for $-3 \leq n \leq 2$ and is zero otherwise. Determine and sketch the time-reversed signal $y[n] = x[-n]$.

△

▷ Drill 4.3 (Combined DT Time Shifting and Reversal)

Show that $x[-n - m]$ can be obtained from $x[n]$ by first right shifting $x[n]$ by m units and then time reversing this shifted signal.

△

4.1.3 DT Time Scaling: Sampling Rate Conversion

As with the continuous-time case, the time-scaling operation either compresses or expands a discrete-time signal. A compressed DT signal has fewer points than the original, while an expanded DT signal increases the number of points from the original, usually by inserting zeros. If desired, an interpolation filter fills in the gaps of zeros caused by expansion. Using the original signal of Fig. 4.5a, Fig. 4.5b shows compression by 2, Fig. 4.5c shows expansion by 2, and Fig. 4.5d shows interpolation of Fig. 4.5c. When applied to a signal sampled at some rate F_s , a DT time scaling operation alters the sampling rate. Compression decreases the sampling rate, while expansion increases it. Systems that operate with different sampling rates, called *multirate systems*, are becoming increasingly important in modern digital signal processing applications.

Compression, Downsampling, and Decimation

Replacing n with Mn in $x[n]$ compresses the signal by factor M to produce

$$x_{\downarrow}[n] = x[Mn]. \quad (4.5)$$

Because of the restriction that discrete-time signals are defined only for integer values of the argument, M must be an integer. The value of $x[Mn]$ at $n = 0$ is $x[0]$, at $n = 1$ is $x[M]$, at $n = 2$ is $x[2M]$, and so on. This means that $x[Mn]$ selects every M th sample of $x[n]$ and deletes all the samples in between. Since compression reduces the number of samples by factor M , a signal $x[n]$ at rate F_s produces a signal $x_{\downarrow}[n]$ at reduced rate F_s/M . For this reason, the DT compression operation is also called *downsampling*. To help illustrate, Fig. 4.5a shows a signal $x[n]$, and Fig. 4.5b shows the downsampled signal $x_{\downarrow}[n] = x[2n]$, which is obtained by deleting odd-numbered samples of $x[n]$.

In the continuous-time case, time compression merely speeds up the signal without loss of any data. In contrast, compression of $x[n]$ causes a loss of samples, although, as we shall soon see, not

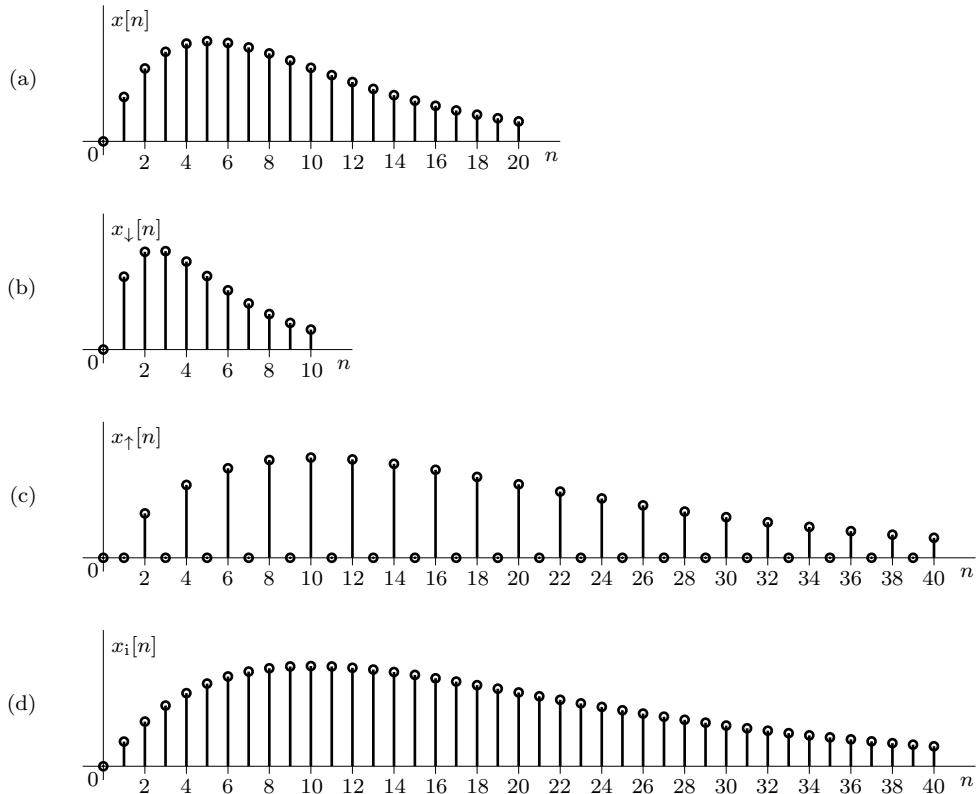


Figure 4.5: Time scaling a DT signal: (a) original signal, (b) compression by 2, (c) expansion by 2, and (d) expansion by 2 followed by interpolation.

necessarily a loss of information. If $x[n]$ is the result of oversampling some CT signal $x(t)$ by a factor M or greater, then clearly $x[Mn]$ still satisfies Nyquist and retains complete information about $x[n]$. Since reducing the sampling rate of a signal can cause aliasing, downsampling is sometimes preceded by a digital anti-aliasing filter, also called a *decimation filter*. We shall discuss in Ch. 6 the nature of the appropriate decimation filter for accomplishing the desired objective of conservation (or at least limiting the loss) of the original data. The combined operation of filtering followed by compression is referred to as *decimation*.

▷ Example 4.1 (Downsampling and Aliasing)

Consider the signal $x[n] = \cos(2\pi n/4)$, which are samples of the signal $\cos(2\pi t)$ taken at $F_s = \frac{1}{T} = 4$ (twice the Nyquist rate). Sketch $x[n]$, $x[2n]$, and $x[4n]$. Comment on the results.

MATLAB easily generates plots of $x[n]$, $x[2n]$, and $x[4n]$.

```
01 n = 0:20; x = @n cos(2*pi*n/4);
02 subplot(311); stem(n,x(n)); subplot(312); stem(n,x(2*n)); subplot(313); stem(n,x(4*n));
```

The original sinusoid signal $x[n]$, being sampled at twice the Nyquist rate, displays four samples per cycle (twice the minimum), as shown in Fig. 4.6a. When compressed by a factor of 2, half the samples are eliminated, and the sampling rate is effectively cut in half. In other words, the sampling rate goes from twice Nyquist to exactly Nyquist. Figure 4.6b confirms this result and shows that $x[2n]$ has two samples per cycle, the bare minimum needed to represent the sinusoid. Since $x[n]$ is

oversampled by a factor of 2, the compressed signal $x[2n]$ still retains complete information about the original signal, even though half the samples of $x[n]$ are thrown away.

Continuing the downsampling process, it is not surprising that when $x[n]$ is compressed by a factor of 4 to produce $x[4n]$, as shown in Fig. 4.6c, the result displays the classic symptoms of aliasing. The sampling rate is essentially cut to half the Nyquist rate, causing higher frequencies to impersonate lower frequencies. Such aliasing typically causes an irrecoverable loss of signal information and is thus undesirable.

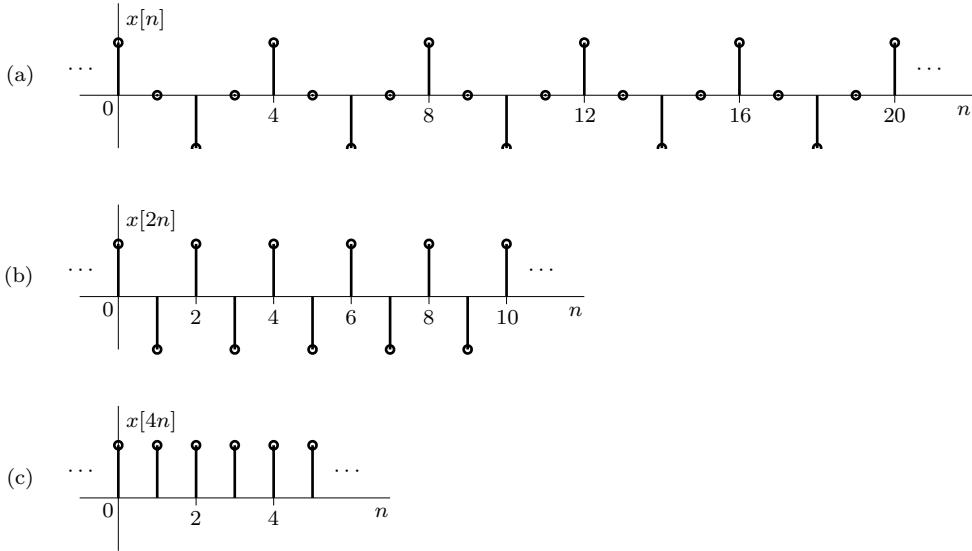


Figure 4.6: Downsampling a sinusoid: (a) $x[n] = \cos(2\pi n/4)$, (b) $x[2n]$, and (c) $x[4n]$.

Example 4.1 □

▷ Drill 4.4 (Preserving Odd-Numbered Samples during Compression)

As shown in Fig. 4.5, $x[2n]$ represents a compression by 2 that preserves the even-numbered samples of the original signal $x[n]$. Show that $x[2n+1]$ also compresses $x[n]$ by a factor of 2, but in doing so preserves the odd-numbered samples.

□

Expansion, Upsampling, and Interpolation

An *interpolated* signal is generated in two steps: an expansion followed by filtering. To begin, we expand $x[n]$ by an integer factor L to obtain the expanded signal $x_{\uparrow}[n]$ given as

$$x_{\uparrow}[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{otherwise} \end{cases}. \quad (4.6)$$

To understand this expression, consider a simple case of expanding $x[n]$ by a factor $L = 2$. The expanded signal is $x_{\uparrow}[n] = x[n/2]$ for even n . Hence, $x_{\uparrow}[0] = x[0]$, $x_{\uparrow}[2] = x[1]$, $x_{\uparrow}[4] = x[2]$, and so on. Since a discrete-time signal is not defined for non-integer arguments, Eq. (4.6) defines $x_{\uparrow}[n]$ as zero whenever n/L is a fractional value. Continuing our $L = 2$ example, the odd-numbered samples $x_{\uparrow}[1], x_{\uparrow}[3], x_{\uparrow}[5], \dots$ are thus all zero. Figure 4.5c illustrates this case.

In general, for $n \geq 0$, the expansion of $x[n]$ by factor L is given by the sequence

$$x[0], \underbrace{0, 0, \dots, 0, 0}_{L-1 \text{ zeros}}, x[1], \underbrace{0, 0, \dots, 0, 0}_{L-1 \text{ zeros}}, x[2], \underbrace{0, 0, \dots, 0, 0}_{L-1 \text{ zeros}}, \dots$$

This sequence contains all the data of $x[n]$, although in an expanded form. The sampling rate of $x_{\uparrow}[n]$ is L times that of $x[n]$. Hence, this operation is also called *upsampling*.

The expanded signal in Fig. 4.5c, with the jaggedness caused by having all odd-numbered samples zero, seems a poor approximation of the underlying signal, particularly if that signal is bandlimited. By passing the expanded signal through an *interpolation filter*, a more suitable construction is achieved. We shall show in Ch. 6 that the optimum interpolating filter is usually an ideal digital lowpass filter, which, as any ideal filter, is realizable only approximately. This process of filtering to interpolate the zero-valued samples is called *interpolation*. Since the interpolated data is computed from the existing data, interpolation does not result in gain of information. Additionally, since an interpolated signal possesses the same increased sampling rate as $x_{\uparrow}[n]$, interpolation is, like expansion, also referred to as *upsampling*. Figure 4.5d illustrates interpolation by appropriately filtering the expanded signal of Fig. 4.5c.

A Word of Caution

In Sec. 4.6 we shall learn how to combine expansion and compression to achieve a fractional alteration of the sampling rate. It is not enough, however, to express such a combination using $x[Mn/L]$, a notation that, among other failings, does not indicate the presence of interpolation or decimation filters. Even without such filters, using expanders and compressors in combination requires great care. In general, the two operations do not commute, which is to say that their placement order is important to the final result. If we expand a signal by L and then compress it by M , the result is generally not the same as if we first compress the signal by M and then expand it by L . Section 4.6 clarifies these ideas.

▷ Drill 4.5 (Expansion and Interpolation)

A signal $x[n] = \sin(2\pi n/4)$ is expanded by $L = 2$ to produce $x_{\uparrow}[n]$. Next, this signal is interpolated according to $x_i[n] = \frac{1}{2}x_{\uparrow}[n-1] + x_{\uparrow}[n] + \frac{1}{2}x_{\uparrow}[n+1]$. Sketch $x[n]$, $x_{\uparrow}[n]$, and $x_i[n]$, and comment on the overall quality of the signal interpolation.

△

4.2 DT Signal Models

We now discuss some important discrete-time signal models that are encountered frequently in the study of discrete-time signals and systems.

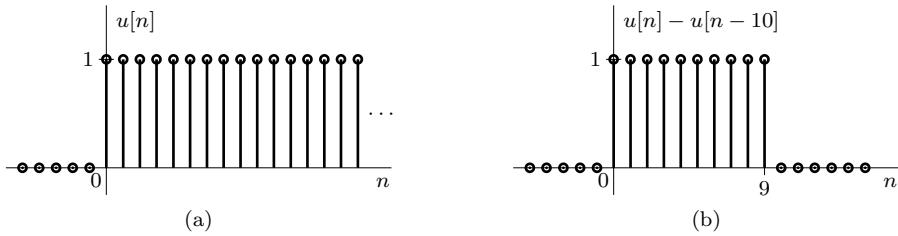
4.2.1 DT Unit Step Function $u[n]$

The discrete-time counterpart of the unit step function $u(t)$ is $u[n]$, which is shown in Fig. 4.7a and defined as

$$u[n] = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}. \quad (4.7)$$

If we want a signal to start at $n = 0$ (so that it has a zero value for all $n < 0$), we need only multiply the signal with $u[n]$.

Combinations of the unit step and its shift allow us to specify a particular region of time. The signal $u[n] - u[n-10]$, for example, is 1 for $0 \leq n \leq 9$ and 0 otherwise, as shown in Fig. 4.7b. Unlike

Figure 4.7: (a) The unit step $u[n]$ and (b) the combination $u[n] - u[n - 10]$.

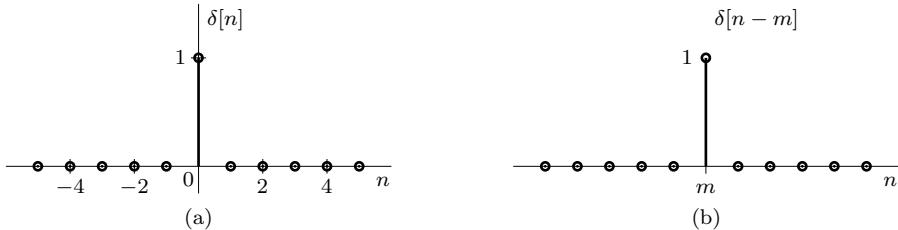
the continuous-time signal $u(t) - u(t - 10)$, which remains 1 until $t = 10$, notice that $u[n] - u[n - 10]$ lasts only until $n = 9$. This subtle but important point is easy to understand: the signal turns on (becomes 1) at $n = 0$ due to the $u[n]$ component and turns off (becomes 0) at $n = 10$ due to the $u[n - 10]$ component. Upcoming Ex. 4.2 illustrates the efficacy of the step function in describing piecewise signals, which have different descriptions over different ranges of n .

4.2.2 DT Unit Impulse Function $\delta[n]$

The discrete-time counterpart of the continuous-time impulse function $\delta(t)$ is the Kronecker delta function $\delta[n]$, defined as

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}. \quad (4.8)$$

This function, also called the *unit impulse sequence*, is shown in Fig. 4.8a. The shifted impulse sequence $\delta[n - m]$ is depicted in Fig. 4.8b. Unlike its continuous-time counterpart $\delta(t)$ (the Dirac delta function), the Kronecker delta function $\delta[n]$ is a very simple function that can physically exist and that requires no special esoteric knowledge of distribution theory.

Figure 4.8: Discrete-time impulse functions: (a) $\delta[n]$ and (b) $\delta[n - m]$.

Similar to the Dirac delta function, several useful properties accompany the DT unit impulse function.

- Multiplication by a DT Impulse:** If a DT function $x[n]$ is multiplied by a shifted Kronecker delta function, only the signal value that coincides with the impulse survives:

$$x[n]\delta[n - m] = x[m]\delta[n - m]. \quad (4.9)$$

This property follows directly from Eq. (4.8).

- The Sampling Property:** Equation (4.9) leads to the DT *sampling* or *sifting property*, which is given as

$$x[n] = \sum_{m=-\infty}^{\infty} x[m]\delta[n - m]. \quad (4.10)$$

Basically, Eq. (4.10) provides the rather obvious result that any discrete-time signal can be represented as a weighted sum of all its samples (shifted, scaled impulse functions).

3. Relationships between $\delta[n]$ and $u[n]$: Since the unit impulse is 1 at $n = 0$ and 0 elsewhere, it follows that

$$u[n] = \sum_{m=-\infty}^n \delta[m] \quad (4.11)$$

and

$$\delta[n] = u[n] - u[n - 1]. \quad (4.12)$$

These relationships between $u[n]$ and $\delta[n]$ are the discrete-time analogues to the integral and differential relationships between $u(t)$ and $\delta(t)$.

▷ **Example 4.2 (Signal Representation through Unit Impulse and Step Functions)**

Describe the signal $x[n]$ shown in Fig. 4.9 by a single expression, valid for all n , comprised of unit impulse and step functions.

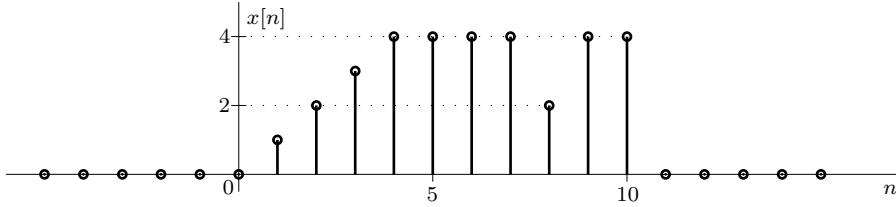


Figure 4.9: Signal representation through unit impulse and step functions.

There are many different ways of viewing $x[n]$. Although each view yields a different expression, they are all equivalent. We shall consider here just one possible expression.

The signal $x[n]$ can be broken into three components: (1) a ramp component $x_1[n]$ from $n = 0$ to 4, (2) a step component $x_2[n]$ from $n = 5$ to 10, and (3) an impulse component $x_3[n]$ represented by the negative spike at $n = 8$. Let us consider each one separately.

We express $x_1[n] = n(u[n] - u[n - 5])$ to account for the signal from $n = 0$ to 4. Assuming that the spike at $n = 8$ does not exist, we let $x_2[n] = 4(u[n - 5] - u[n - 11])$ account for the signal from $n = 5$ to 10. Once these two components are added, the only part that is unaccounted for is a spike of amplitude -2 at $n = 8$, which can be represented by $x_3[n] = -2\delta[n - 8]$. Hence

$$\begin{aligned} x[n] &= x_1[n] + x_2[n] + x_3[n] \\ &= n(u[n] - u[n - 5]) + 4(u[n - 5] - u[n - 11]) - 2\delta[n - 8]. \end{aligned}$$

We stress again that the expression is valid for all values of n . There are many other equivalent expressions for $x[n]$. For example, one may consider a step function from $n = 0$ to 10, subtract a ramp over the range $n = 0$ to 3, and subtract a spike at $n = 8$.

Example 4.2 ◀

▷ **Drill 4.6 (Using Kronecker Delta Functions to Represent Expansion)**

Show that the expansion of signal $x[n]$ by factor L , defined by Eq. (4.6), can be expressed using Kronecker delta functions as

$$x_{\uparrow}[n] = \sum_{m=-\infty}^{\infty} x[m]\delta[n - Lm]. \quad (4.13)$$

Using this representation, find and plot the $L = 4$ expansion of $x[n] = u[n + 4] - u[n - 5]$.

△

4.2.3 DT Exponential Function z^n

Chapter 1 details the versatile continuous-time exponential function e^{st} , which, depending on the value of the complex frequency s , can behave as a constant, a monotonic exponential, a sinusoid, an exponentially decaying sinusoid, and others. This generalized exponential function is intimately connected to the Laplace transform, which is useful in the analysis of continuous-time signals and systems. As might be expected, a generalized discrete-time exponential serves similar roles in the study of DT signals and systems. Connecting the DT exponential to the CT exponential provides much insight and is well worth the investment in time and effort.

Let us sample a CT exponential e^{st} in the usual way by letting $t = nT$. The resulting DT exponential is

$$e^{sTn} = z^n, \quad \text{where } z = e^{sT} \text{ and } s = \frac{1}{T} \ln(z). \quad (4.14)$$

The two forms e^{sTn} and z^n are just alternate expressions of the same DT exponential. The form e^{sTn} helps emphasize the connection to CT exponentials, while the form z^n is convenient to the study of DT signals and systems. It is straightforward to convert between the two forms. For example, $e^{-n} = (e^{-1})^n = (0.3679)^n$. Similarly, $2^n = e^{\ln(2)n} = e^{0.6931n}$. In the first case, $sT = -1$ and $z = e^{sT} = 0.3679$. In the second case, $z = 2$ and $sT = \ln(z) = 0.6931$. Notice that while we typically use z with DT signals, the variable z itself is complex and *continuous*.

Given its direct connection to the CT exponential, we expect that the DT exponential z^n can behave as a DT constant, monotonic exponential, sinusoid, exponentially decaying sinusoid, and others. This turns out to be true. Letting $r = |z|$ and $\Omega = \angle z$ so that $z = re^{j\Omega}$, the following functions are special cases of z^n :

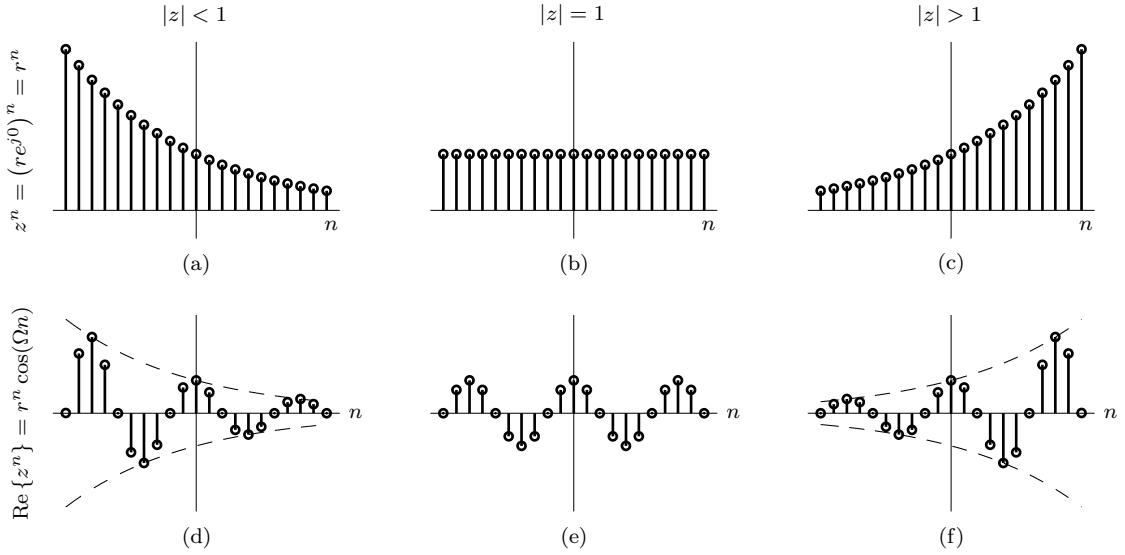
1. a constant $k = k1^n$ (where $z = 1e^{j0}$),
2. a monotonic exponential r^n (where $z = re^{j0}$),
3. a sinusoid $\cos(\Omega n) = \operatorname{Re}\{e^{j\Omega n}\}$ (where $z = 1e^{j\Omega}$), and
4. an exponentially varying sinusoid $r^n \cos(\Omega n) = \operatorname{Re}\{r^n e^{j\Omega n}\}$ (where $z = re^{j\Omega}$).

Figure 4.10 shows various examples of these DT exponential functions.

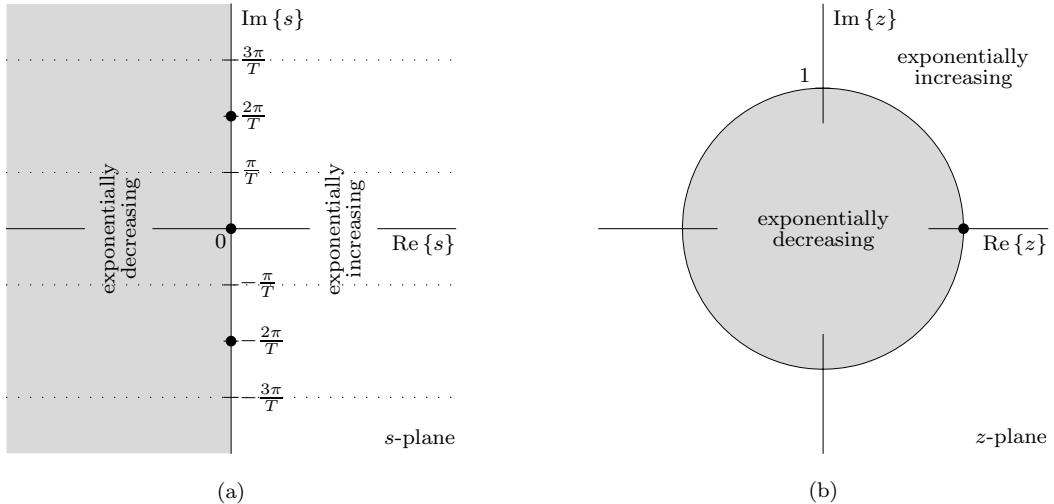
While these cases may seem essentially identical to those for the CT exponential, there are important distinctions between the continuous-time and discrete-time cases. Before detailing these differences, however, it is helpful to better connect the complex variable s (used with CT signals) to the complex variable z (used with DT signals).

Relating s and z

The signal e^{st} grows exponentially with s if $\operatorname{Re}\{s\} > 0$ (s in RHP) and decays exponentially if $\operatorname{Re}\{s\} < 0$ (s in LHP). It is constant or oscillates with constant amplitude if $\operatorname{Re}\{s\} = 0$ (s on the imaginary axis). As indicated in Fig. 4.11a, the location of s in the complex plane indicates whether the signal e^{st} grows exponentially (unshaded region), decays exponentially (shaded region), or oscillates with constant frequency (imaginary axis). By investigating the relationship between s

Figure 4.10: Various manifestations of z^n and $\text{Re}\{z^n\}$.

and z , we next establish similar criteria to determine the nature of z^n from the location of z in the complex plane.

Figure 4.11: Relating complex planes: (a) s -plane and (b) z -plane.

In simple terms, Eq. (4.14) tells us that s and z are related by $z = e^{sT}$ or, equivalently, $s = \frac{1}{T} \ln(z)$. Let us begin with the imaginary axis in Fig. 4.11a, which corresponds to non-decaying CT exponentials. Here, $s = 0 + j\omega$ and $z = e^{sT} = e^{j\omega T}$. In this case, $|z| = |e^{j\omega T}| = 1$, and we conclude that the s -plane imaginary axis becomes the z -plane *unit circle* $|z| = 1$. Similarly, the negative half-plane in Fig. 4.11a, which has $s = \sigma + j\omega$ and $\sigma < 0$, becomes $z = e^{sT} = e^{(\sigma+j\omega)T}$. In this case, $|z| = |e^{(\sigma+j\omega)T}| = e^{\sigma T}$. However, since $\sigma < 0$ (LHP), we see that $|z| < 1$. In other words, the left half of the s -plane, shown shaded in Fig. 4.11a, maps to the interior of the unit circle in the z -plane, shown shaded in Fig. 4.11b. Conversely, the right half-plane in Fig. 4.11a (unshaded) maps to the

exterior of the unit circle in Fig. 4.11b (also unshaded). Thus, $|z| < 1$ corresponds to decaying exponentials, $|z| = 1$ corresponds to constant-envelope exponentials, and $|z| > 1$ corresponds to growing exponentials, examples of which are shown in Fig. 4.10.

Let us study the relation $z = e^{sT}$ more carefully. The equation $z = e^{sT}$ is a rule, or *mapping*, that tells us how to go from one location to another. There are many types of mapping functions. The familiar equation $y = x$ is a linear mapping where every point x maps directly to its own corresponding point y (a one-to-one mapping). The equation $y = x^3$ is also a mapping, but one where multiple points x can map to a single point y . The expression $z = e^{sT}$ is a mapping that tells us how to get from s to z (and, to some extent, vice versa). It, like $y = x^3$, is a *many-to-one* mapping. In fact, an infinite number of points s map to any single point z (of which there are infinitely many). Realizing that sampling is used to travel from s to z , we recognize that the *many-to-one nature of this mapping is just a disguise of aliasing*.

To help further understand this many-to-one behavior, let us consider a simple case where $s = j\frac{2\pi}{T}k$ (k integer). These points, shown as heavy dots along the imaginary axis in Fig. 4.11a, map to the single value $z = e^{sT} = e^{j2\pi k} = 1$, shown with a similar heavy dot in Fig. 4.11b. Just as “all roads lead to Rome,” we see that all points $s = j\frac{2\pi}{T}k$ lead to $z = 1$. This case corresponds to an earlier observation that all sinusoids whose frequencies are integer multiples of the sampling rate $F_s = 1/T$ alias to DC. By extension, any segment of length $2\pi/T$ along the imaginary axis in Fig. 4.11a maps in a one-to-one fashion to the unit circle in Fig. 4.11b. In fact, by further extension, any s -plane strip of height $2\pi/T$ maps to cover the entire z -plane. Not surprisingly, the strip $-\pi/T < \text{Im}\{s\} < \pi/T$ corresponds to the fundamental band, discussed in Ch. 3 (pg. 171). Struck again by the curse of aliasing, any signal outside this region will look, following the mapping by $z = e^{sT}$, as if it came from the fundamental band.

▷ Example 4.3 (Aliasing in DT Exponentials)

Consider the DT exponential $\text{Re}\{z^n\}$ obtained from sampling the signal $\text{Re}\{e^{st}\}$ with $T = 1$. Using $z = e^{sT}$, plot $\text{Re}\{z^n\}$, as well as $\text{Re}\{e^{st}\}$ for reference, over $-5 \leq n \leq 5$ for the cases $s = -0.1$ and $s = -0.1 + j2\pi$. Comment on the results.

The desired signal plots, generated with MATLAB, are shown in Fig. 4.12.

```
01 t = linspace(-5.5,5.5,1001); n = (-5:5); T = 1;
02 s = -0.1+1j*0; z = exp(s*T);
03 subplot(121); stem(n,real(z.^n)); line(t/T,real(exp(s*t)));
04 s = -0.1+1j*2*pi; z = exp(s*T);
05 subplot(122); stem(n,real(z.^n)); line(t/T,real(exp(s*t)));
```

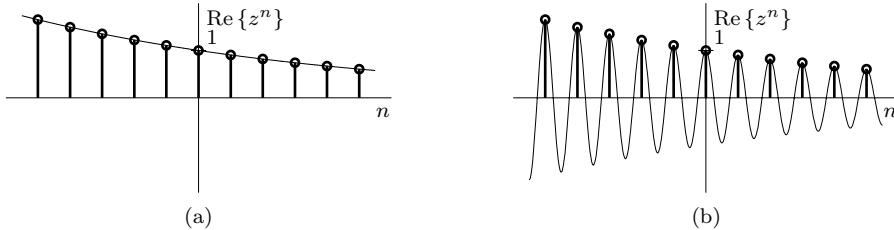


Figure 4.12: Plotting $\text{Re}\{z^n\} = \text{Re}\{e^{st}\}|_{t=nT}$ for (a) $s = -0.1$ (b) $s = -0.1 + j2\pi$.

The case $s = -0.1$ is in the fundamental band, so the DT exponential closely follows the CT exponential, as shown in Fig. 4.12a. The case $s = -0.1 + j2\pi$, however, is not in the fundamental band. In this case, aliasing occurs, and the DT exponential cannot closely follow the original CT exponential, as shown in Fig. 4.12b. Further, since $s = -0.1$ differs from $s = -0.1 + j2\pi$ by an integer factor of $j\frac{2\pi}{T}$, the DT exponentials in Figs. 4.12a and 4.12b are identical. As this example

demonstrates, aliasing affects all types of exponentials, not just simple (non-decaying) sinusoids, as considered earlier in Ch. 3.

Example 4.3 □

▷ Drill 4.7 (Relating CT and DT Exponentials)

Determine and sketch the DT exponentials z^n that result from sampling ($T = 1$) the following CT exponentials e^{st} :

$$\begin{array}{llll} \text{(a)} & e^{0t} = 1 & \text{(b)} & e^t \\ \text{(e)} & 2^t & \text{(f)} & 2^{-t} \\ \text{(c)} & e^{-0.6931t} & \text{(g)} & e^{-t/4} \\ \text{(d)} & (-e^{-0.6931})^t & \text{(h)} & e^{j\pi t} \end{array}$$

For each case, locate the value z in the complex plane and verify that z^n is exponentially decaying, exponentially growing, or non-decaying depending on whether z is inside, outside, or on the unit circle.

□

DT Complex Exponential and Sinusoid

A general DT complex exponential $e^{j\Omega n}$ is a complex-valued function of n . For convenience, we presently ignore the complex scale constant necessary to make $e^{j\Omega n}$ truly general with arbitrary magnitude and initial phase. Notice that Ωn is an angle in radians. Hence, the dimensions of the frequency Ω are *radians per sample*. At $n = 0, 1, 2, 3, \dots$, the complex exponential $e^{j\Omega n}$ takes on values $e^{j0}, e^{j\Omega}, e^{j2\Omega}, e^{j3\Omega}, \dots$, respectively. Rather than use the typical dual plots of magnitude and phase (or real part and imaginary part), Fig. 4.13 plots $e^{j\Omega n}$ directly in the complex plane for various values of n .

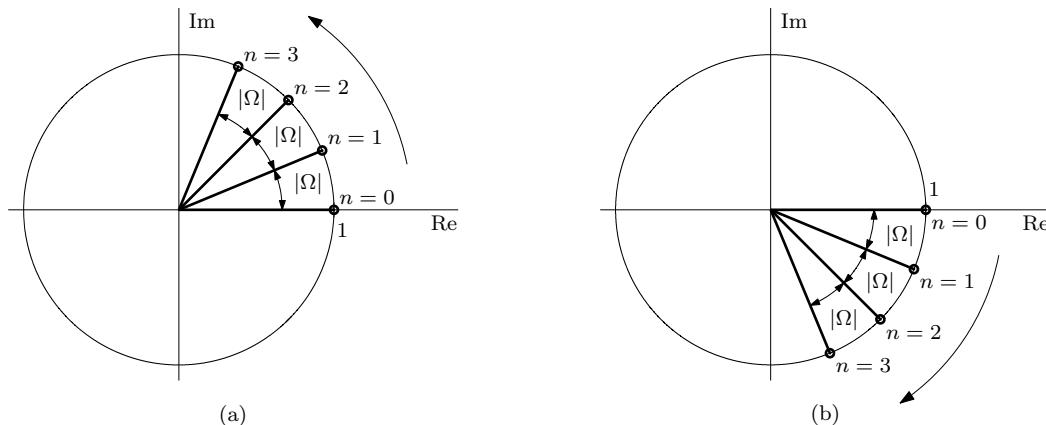


Figure 4.13: Loci of (a) $e^{j|\Omega|n}$ and (b) $e^{-j|\Omega|n}$.

Observe that $e^{j\Omega n}$, already in standard polar form, has a magnitude of 1 and an angle of Ωn . Therefore, the points $e^{j0}, e^{j\Omega}, e^{j2\Omega}, e^{j3\Omega}, \dots$, lie on a circle of unit radius (unit circle) at angles $0, \Omega, 2\Omega, 3\Omega, \dots$. For $\Omega > 0$, $e^{j\Omega n} = e^{j|\Omega|n}$ moves counterclockwise along the unit circle by an angle $|\Omega|$ for each unit increase in n , as shown in Fig. 4.13a. For $\Omega < 0$, $e^{j\Omega n} = e^{-j|\Omega|n}$ moves clockwise along the unit circle by an angle $|\Omega|$ for each unit increase in n , as shown in Fig. 4.13b. In either case, the locus of $e^{j\Omega n}$ may be viewed as a phasor rotating stepwise at a uniform rate of $|\Omega|$ radians per unit sample interval. The sign of Ω specifies the direction of rotation, while $|\Omega|$ establishes the rate of rotation, or frequency, of $e^{j\Omega n}$.

We know from Euler's formula that $e^{j\Omega n} = \cos(\Omega n) + j \sin(\Omega n)$, so DT sinusoids are directly connected to complex exponentials. For example, Fig. 4.14a plots $\cos(\pi n/5)$, which can be obtained from taking the real part of the complex exponential $e^{j\pi n/5}$. Bound by such a close relationship, sinusoids and complex exponentials share similar properties and peculiarities. We examine one peculiarity of DT sinusoids next.

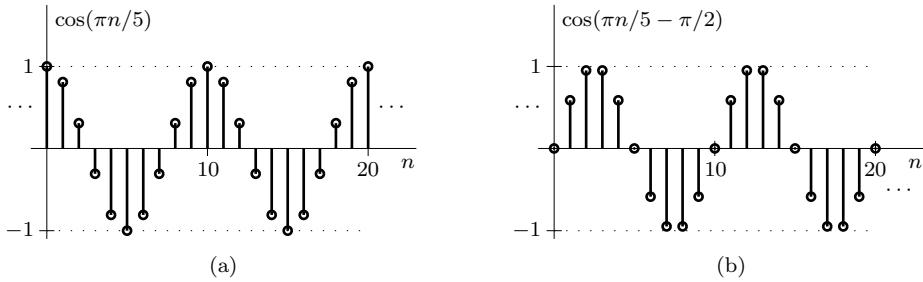


Figure 4.14: DT sinusoids: (a) $\cos(\pi n/5)$ and (b) $\cos(\pi n/5 - \pi/2)$.

Cousins in Striped Suits?

A DT sinusoid $\cos(\Omega n + \theta)$ can be viewed as a sampled relative of CT sinusoid $\cos(\omega n + \theta)$,

$$\cos(\omega t + \theta)|_{t=nT} = \cos(\omega nT + \theta) = \cos(\Omega n + \theta), \quad \text{where } \Omega = \omega T. \quad (4.15)$$

With a casual glance at Eq. (4.15), it may appear that DT sinusoids are cousins of CT sinusoids in striped suits. Things, however, are not quite that straightforward. In the CT case, the waveform $\cos(\omega t + \theta)$ is identical to $\cos(\omega t)$ except for a time shift of θ/ω . In the DT case, however, $\cos(\Omega n + \theta)$ is not generally a simple shift of $\cos(\Omega n)$. Figure 4.14 illustrates the point. In Fig. 4.14a we find the DT sinusoid $\cos(\pi n/5)$, which has frequency $\Omega = \pi/5$. As shown in the Fig. 4.14b plot of $\cos(\pi n/5 - \pi/2)$, however, a phase offset of $-\pi/2$ produces more than a simple shift in the waveform. Entirely different values are produced. For example, the peak value of $\cos(\Omega n)$ is 1, while the peak value of $\cos(\pi n/5 - \pi/2)$ is slightly less at 0.9511. Similarly, $\cos(\Omega n)$ is never 0 yet $\cos(\pi n/5 - \pi/2)$ is frequently 0.

The two CT sinusoids $\cos(\omega t)$ and $\cos(\omega t + \theta)$ are identical in shape and information; their only difference is a simple time shift. Now, an important question is whether the two corresponding DT sinusoids also contain identical information despite their different sample values? Yes, they do! How so? The signal $\cos(\Omega n + \theta)$ is just a sampled version of $\cos(\omega t + \theta)$. As long as the signal is not undersampled, $\cos(\omega t + \theta)$ can be recovered exactly from $\cos(\Omega n + \theta)$ regardless of θ .

Apparent Frequency and the Nonuniqueness of DT Exponentials

A continuous-time sinusoid $\cos(\omega t)$ has a unique waveform for every value of ω in the range 0 to ∞ . Increasing ω results in a sinusoid of ever-increasing frequency. Such is not the case for DT sinusoids or, more generally, DT exponentials. The culprit is the same phenomenon of frequency folding (aliasing) presented in Ch. 3, which itself is connected to the many-to-one nature of the mapping $z = e^{sT}$ discussed earlier. Moreover, inasmuch as DT sinusoids can be obtained by sampling CT sinusoids, we expect DT sinusoids to follow the fascinating behavior of sampled CT sinusoids observed in Sec. 3.3.1. We saw that when a CT sinusoid of frequency f_0 is sampled at a rate F_s Hz, the frequency of the resulting sampled sinusoid never exceeds $F_s/2$ Hz. In fact, the results of Sec. 3.3.1 can be directly applied to DT sinusoids. The implication is that a DT sinusoid frequency can never exceed a certain frequency, which, we shall see, is π .

For integer k , observe that

$$\begin{aligned} e^{j(\Omega+2\pi k)n} &= e^{j\Omega n}, \\ r^n e^{j(\Omega+2\pi k)n} &= r^n e^{j\Omega n}, \\ \text{and } \cos[(\Omega + 2\pi k)n] &= \cos(\Omega n). \end{aligned} \quad (4.16)$$

Equation (4.16) shows that a DT exponential (sinusoid) of frequency Ω is indistinguishable from an exponential (sinusoid) of frequency Ω plus or minus an integral multiple of 2π . Any DT exponential has a unique waveform only for the values of Ω in the *fundamental band* of frequencies from $-\pi$ to π . Every frequency Ω outside the fundamental band, no matter how large, is identical to an *apparent frequency* Ω_a located within the fundamental band. Similar to Eq. (3.14), the apparent frequency Ω_a of frequency Ω is given as

$$\Omega_a = \langle \Omega + \pi \rangle_{2\pi} - \pi. \quad (4.17)$$

Figure 4.15a shows the plot of apparent frequency Ω_a versus the actual frequency Ω of a DT exponential or sinusoid.

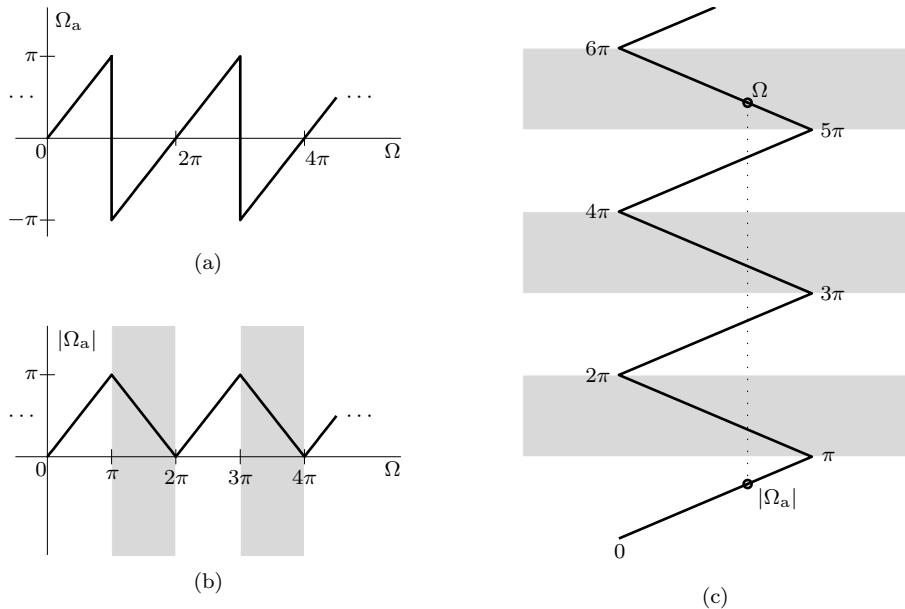


Figure 4.15: Apparent frequency of a DT exponential with frequency Ω : (a) Ω_a , (b) $|\Omega_a|$, and (c) multiple folding to obtain $|\Omega_a|$.

As in the CT case, the distinction between positive and negative frequencies is often unimportant. For example, since $\cos(\Omega n + \theta) = \cos(-\Omega n - \theta)$, we see that frequencies $\pm\Omega$ both produce the same rate of oscillation. Thus, apparent frequency is often expressed as $|\Omega_a|$, as shown in Fig. 4.15b. Although less efficient than using Eq. (4.17), we can also determine the apparent frequency $|\Omega_a|$ using multiple folding, as shown in Fig. 4.15c. To do so, we mark a narrow strip of paper tape-measure style using radians per sample, the units of Ω . Folding the tape accordion fashion every integer multiple of π , the frequency $|\Omega_a|$ is just the projection of Ω onto the $(0, \pi)$ segment, as shown using a dotted line in Fig. 4.15c.

Figure 4.15 also clarifies the frequency reversal effect, where increasing frequency Ω can actually decrease the apparent frequency Ω_a and reverse its direction (sign). This reversal effect is behind the amusing scenes in some western movies where wheels appear to rotate backward and slow down, even though the wagon is moving forward and speeding up.

As in the continuous case, it is possible to express real DT sinusoids exclusively in terms of positive apparent frequency since $\cos(-\Omega n + \theta) = \cos(\Omega n - \theta)$. When the sign of Ω_a is changed in such cases, the sign of the phase θ is also changed. The shaded bands in Figs. 4.15b and 4.15c designate frequencies that have negative apparent frequency. Using Fig. 4.15c, we see these shaded regions follow odd-numbered folds.

Notice that the lowest DT frequency is $\Omega = 0$, which corresponds to a constant. Similarly, the highest oscillation rate in a DT sinusoid (or exponential) occurs when $|\Omega| = \pi$. For example, the DT sinusoid $\cos(\pi n)$ produces the alternating sequence $1, -1, 1, -1, \dots$. Clearly, no other DT sequence can oscillate at a faster rate. Accounting for aliasing effects, high frequencies are those near $\Omega = (2k+1)\pi$, and low frequencies are those near $\Omega = 2\pi k$, where k is an integer. Figure 4.16 illustrates several DT sinusoids of various frequencies, including the highest oscillation rate $\Omega = \pi$.

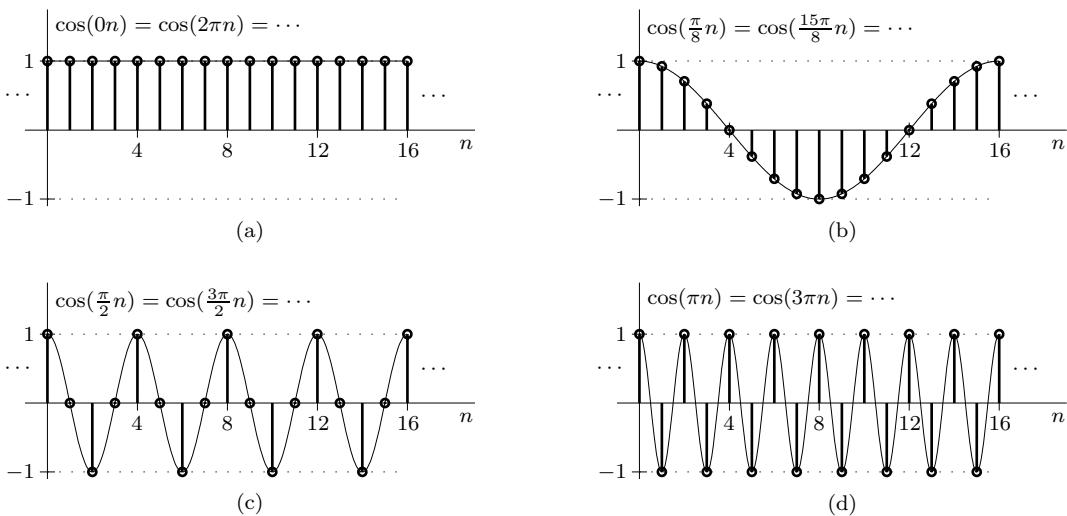


Figure 4.16: DT sinusoids of various frequencies: (a) lowest rate $\Omega = 0$ to (d) highest rate $\Omega = \pi$.

Figure 4.15 shows that discrete-time frequencies are bandlimited to $|\Omega| = \pi$. All frequencies outside this range alias into the fundamental band, which ranges from $-\pi$ to π radians/sample.[†] Any discrete-time sinusoid of frequency beyond the fundamental band, when plotted, appears and behaves, in every way, as some sinusoid of frequency in the fundamental band. It is impossible to distinguish between the two signals. Thus, in a basic sense, discrete-time frequencies beyond $|\Omega| = \pi$ do not exist. Yet, in a mathematical sense, we must admit the existence of sinusoids of frequencies beyond $|\Omega| = \pi$. What does this mean?

A Man Named Robert

To give an analogy, consider a fictitious person Mr. Robert Thompson. His mother calls him “Robby,” his acquaintances call him “Bob,” his close friends call him by his nickname, “Shorty.” Yet, Robert, Robby, Bob, and Shorty are one and the same person. However, we cannot say that only Mr. Robert Thompson exists, or only Robby exists, or only Shorty exists, or only Bob exists. All these four persons exist, although they are one and the same person. In the same way, we cannot say that the frequency $\pi/2$ exists and that the frequency $5\pi/2$ does not exist; they are both the same entity, called by different names.

It is in this sense that we have to admit the existence of frequencies beyond the fundamental band. Indeed, as we shall see later, mathematical expressions in the frequency domain automatically

[†]When advantageous, we sometimes use other contiguous ranges of width 2π in place of the range $-\pi$ to π . The range 0 to 2π , for instance, is used in many applications.

cater to this need by their built-in periodic nature. For this reason, discrete-time signal spectra are 2π -periodic.

Admitting the existence of frequencies with $|\Omega| > \pi$ also serves mathematical and computational convenience in digital signal processing applications. Values of frequencies beyond π also originate naturally in the process of sampling continuous-time sinusoids. Because there is no upper limit on the value of ω , there is no upper limit on the value of the resulting discrete-time frequency $\Omega = \omega T$, although aliasing reduces the apparent frequency to $|\Omega_a| < \pi$ if $|\Omega| > \pi$.

The Apparent Laziness of DT Exponentials

To provide additional insight regarding apparent frequency, we next use a discrete-time exponential rather than a sinusoid. As shown in Fig. 4.13, a discrete-time complex exponential $e^{j\Omega n}$ can be viewed as a phasor rotating at a uniform angular speed of $|\Omega|$ rad/sample, where the direction of rotation depends on whether Ω is positive (counterclockwise rotation) or negative (clockwise rotation). For $|\Omega| < \pi$, angular speed increases with $|\Omega|$, as is natural. When $|\Omega|$ increases beyond π , however, something interesting happens. Let $\Omega = \pi + x$ and $x < \pi$. Figures 4.17a, 4.17b, and 4.17c show $e^{j\Omega n}$ as n progresses from $0 \rightarrow 1$, $1 \rightarrow 2$, and $2 \rightarrow 3$, respectively. Since $\Omega = \pi + x > 0$, we can rightly view this progression as angular steps of size $\pi + x$ rad/sample in the counterclockwise direction (solid arcs). However, we may also interpret the phasor motion as being clockwise at the lower speed of $\pi - x$ rad/sample (dotted arcs). Either of these interpretations describes the phasor motion correctly. If this motion is seen by a human eye, which is a lowpass filter, it will automatically interpret the speed as $\pi - x$, the lower of the two speeds. This interpretation is precisely $|\Omega_a|$, which equals $\pi - x$ in this case. This is the genesis of why, as Ω increases beyond π , the frequency displays a counterintuitive behavior and appears to slow down. Apparent frequency views all DT exponentials as incredibly lazy, only traveling the smallest possible angle to move from one sample to the next.

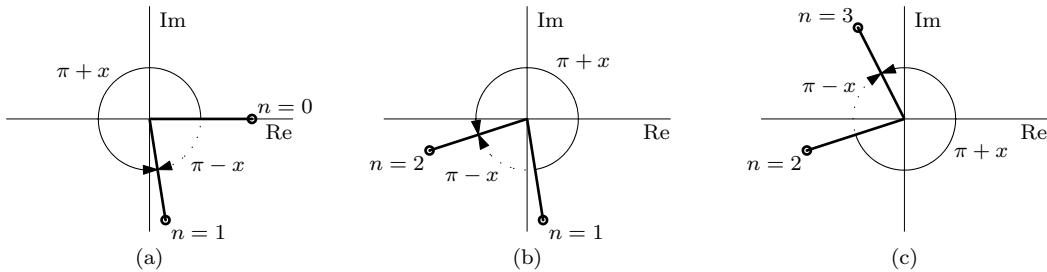


Figure 4.17: Progression of $e^{j(\pi+x)n}$ for n from (a) $0 \rightarrow 1$, (b) $1 \rightarrow 2$, and (c) $2 \rightarrow 3$.

▷ Example 4.4 (Expressing DT Sinusoids in Terms of Apparent Frequency)

Express the following signals in terms of (positive) apparent frequency $|\Omega_a|$:

- | | | |
|-------------------------------|-------------------------------|--------------------------------|
| (a) $\cos(0.5\pi n + \theta)$ | (b) $\cos(1.6\pi n + \theta)$ | (c) $\sin(1.6\pi n + \theta)$ |
| (d) $\cos(2.3\pi n + \theta)$ | (e) $\cos(34.6991n + \theta)$ | (f) $\sin(-2.3\pi n + \theta)$ |

(a)

As confirmed by Fig. 4.15, $\Omega = 0.5\pi$ is in the fundamental range already. There is no phase reversal, and the apparent sinusoid is

$$\cos(0.5\pi n + \theta).$$

(b)

Here, $\Omega_a = \langle 1.6\pi + \pi \rangle_{2\pi} - \pi = -0.4\pi$. Since Ω_a is negative, a sign change in θ is required to express

the sinusoid in terms of $|\Omega_a|$. Thus, the apparent sinusoid is

$$\cos(0.4\pi n - \theta).$$

The same result also follows from Fig. 4.15b.

(c)

We first convert the sine into cosine form as $\sin(1.6\pi n + \theta) = \cos(1.6\pi n - \frac{\pi}{2} + \theta)$. In part (b) we found $\Omega_a = -0.4\pi$. Hence, the apparent sinusoid is

$$\cos(0.4\pi n + \frac{\pi}{2} - \theta) = -\sin(0.4\pi n - \theta).$$

In this case, both the phase and the amplitude change signs.

(d)

In this case, $\Omega_a = \langle 2.3\pi + \pi \rangle_{2\pi} - \pi = 0.3\pi$. Hence, the apparent sinusoid is

$$\cos(0.3\pi n + \theta).$$

(e)

Using Eq. (4.17) and MATLAB, we compute the apparent frequency as

```
01 mod(34.6991+pi,2*pi)-pi
ans = -3.0000
```

Consequently, $|\Omega_a| = 3$ radians/sample, and because Ω_a is negative, there is a sign change of the phase θ . Hence, the apparent sinusoid is

$$\cos(3n - \theta).$$

(f)

In this case, $\Omega_a = \langle -2.3\pi + \pi \rangle_{2\pi} - \pi = -0.3\pi$. As in part (c), since Ω_a is negative and the signal is in the form of sine, both phase and amplitude change signs. The apparent sinusoid is thus

$$-\sin(0.3\pi n - \theta).$$

Example 4.4 □

▷ Drill 4.8 (Apparent Frequency of DT Sinusoids)

Show that a real sinusoid of frequency Ω equal to 2π , 3π , 5π , 3.2π , 22.1327 , and $\pi + 2$ can be expressed as a sinusoid of frequency 0 , π , π , 0.8π , 3 , and $\pi - 2$, respectively. In which cases does the phase change sign?

□

▷ Drill 4.9 (Exponentially-Varying DT Sinusoids)

Accurately sketch the exponentially varying sinusoids $x_a[n] = (0.9)^n \cos(\frac{\pi}{6}n - \frac{\pi}{3})$ and $x_b[n] = (1.1)^n \cos(\frac{49\pi}{6}n - \frac{\pi}{3})$. For both $x_a[n]$ and $x_b[n]$, determine their apparent frequencies, and locate the values z of their corresponding DT exponentials z^n in the complex plane.

□

4.3 DT Signal Classifications

Discrete-time signals are classified in much the same way as continuous-time signals. Of particular interest to our study, we consider the following DT signal classifications:

1. causal, noncausal, and anti-causal signals,
2. real and imaginary signals,
3. even and odd signals,
4. periodic and aperiodic signals, and
5. energy and power signals.

The first three DT signal classifications are nearly identical to the corresponding CT cases, so we treat them only briefly. Deeper discussion on these classifications is found in Sec. 1.4. The DT classifications of periodic or aperiodic and energy or power, which are substantively different than the CT cases, are treated in more depth.

4.3.1 Causal, Noncausal, and Anti-Causal DT Signals

Exactly as in continuous-time, a *causal* DT signal $x[n]$ extends to the right, beginning no earlier than $n = 0$. Mathematically, $x[n]$ is causal if

$$x[n] = 0 \quad \text{for } n < 0. \quad (4.18)$$

A signal that is not causal is called *noncausal*.

An *anti-causal* signal $x[n]$ extends to the left of $n = 0$. Mathematically, $x[n]$ is anti-causal if

$$x[n] = 0 \quad \text{for } n \geq 0. \quad (4.19)$$

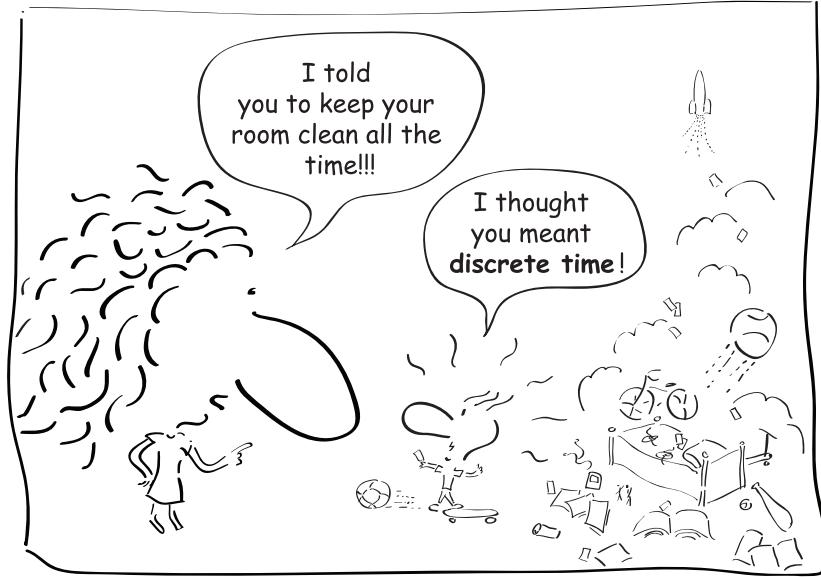
Any signal $x[n]$ can be decomposed into a causal component plus an anti-causal component.

A *right-sided* signal is nonzero only to the right of some finite time $n = N$, while a *left-sided* signal extends to the left of some finite point. Causal signals are right-sided, and anti-causal signals are left-sided. The converse, however, is not necessarily true. Discrete-time signals that stretch in either direction indefinitely are called *two-sided* or *everlasting* signals.

A Cause to Pause

Let us use our knowledge of causality to consider a subtlety of notation. Either explicitly or implicitly, all causal signals involve the unit step function. Recall that Eq. (1.1) defines the CT unit step function $u(t)$ to have a value $1/2$ at $t = 0$. This assignment of $u(0) = 1/2$ is preferable from the context of Fourier analysis, among others. In the DT case, however, we define $u[0]$ as 1 rather than $1/2$, a choice that is preferable from many, although not all, perspectives. One disadvantage of this choice is that sampling a CT unit step does not exactly produce a DT unit step. That is, according to our notation, $u(nT) \neq u[n]$. The two functions differ at time 0 by a value of $1/2$. It is impossible to sample *exactly* at $t = 0$ (just as it is impossible to generate a CT unit step), so the difference between the two models is mostly irrelevant in practice. Thus, we are well justified to consider $u(0)$ as whatever value is convenient and appropriate to the situation.[†]

[†]Mathematically, a CT function is undefined at points of discontinuity, and we can, without serious consequence, call such points whatever value suits our purpose. In the case of the unit step, the choices $u(0) = 1/2$, $u(0) = 1$, and $u(0) = 0$ all find common practical application.



There is a commanding difference between continuous-time and discrete-time.

4.3.2 Real and Imaginary DT Signals

As in continuous-time, a DT signal $x[n]$ is *real* if, for all time, it equals its own complex conjugate,

$$x[n] = x^*[n]. \quad (4.20)$$

A signal $x[n]$ is *imaginary* if, for all time, it equals the negative of its own complex conjugate,

$$x[n] = -x^*[n]. \quad (4.21)$$

Any signal $x[n]$ can be represented in rectangular form using its real and imaginary portions as

$$x[n] = \underbrace{\frac{x[n] + x^*[n]}{2}}_{\text{Re}\{x[n]\}} + j \underbrace{\left(\frac{x[n] - x^*[n]}{2j} \right)}_{\text{Im}\{x[n]\}}. \quad (4.22)$$

Recall that the imaginary portion of a signal, DT or otherwise, is always real.

4.3.3 Even and Odd DT Signals

The even/odd classification is also unchanged in going from continuous-time to discrete-time. A signal $x[n]$ is *even* if, for all time, it equals its own reflection,

$$x[n] = x[-n]. \quad (4.23)$$

A signal $x(t)$ is *odd* if, for all time, it equals the negative of its own reflection,

$$x[n] = -x[-n]. \quad (4.24)$$

Any signal $x[n]$ can be decomposed into an even portion plus an odd portion,

$$x[n] = \underbrace{\frac{x[n] + x[-n]}{2}}_{x_e[n]} + \underbrace{\frac{x[n] - x[-n]}{2}}_{x_o[n]}. \quad (4.25)$$

Conjugate Symmetries

Complex signals are commonly decomposed using conjugate symmetries rather than even and odd decompositions. A signal $x[n]$ is *conjugate symmetric*, or *Hermitian*, if

$$x[n] = x^*[-n]. \quad (4.26)$$

A signal $x(t)$ is *conjugate antisymmetric*, or *skew Hermitian*, if

$$x[n] = -x^*[-n]. \quad (4.27)$$

Any DT signal $x[n]$ can be decomposed into a conjugate-symmetric portion plus a conjugate-antisymmetric portion,

$$x[n] = \underbrace{\frac{x[n] + x^*[-n]}{2}}_{x_{\text{es}}[n]} + \underbrace{\frac{x[n] - x^*[-n]}{2}}_{x_{\text{ca}}[n]}. \quad (4.28)$$

4.3.4 Periodic and Aperiodic DT Signals

A discrete-time signal $x[n]$ is said to be N -periodic if, for some positive integer N ,

$$x[n] = x[n - N] \quad \text{for all } n. \quad (4.29)$$

The *smallest* value of N that satisfies the periodicity condition of Eq. (4.29) is the *fundamental period* N_0 . Figure 4.18 shows an example of a periodic signal with $N_0 = 6$. By definition, a periodic signal must be an everlasting signal that stretches from $n = -\infty$ to ∞ . A signal that does not satisfy Eq. (4.29) is *aperiodic*.

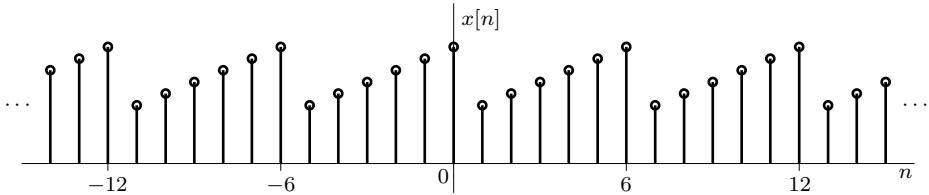


Figure 4.18: Periodic signal $x[n]$ with $N_0 = 6$.

The *fundamental frequency* of a periodic DT signal is related to the reciprocal of the fundamental period as

$$\mathcal{F}_0 = \frac{1}{N_0} \quad \text{or} \quad \Omega_0 = \frac{2\pi}{N_0}. \quad (4.30)$$

In Eq. (4.30), the units of \mathcal{F}_0 are *cycles per sample*, and the units of Ω_0 are *radians per sample*. As we shall see next, the fundamental frequency of a DT signal $x[n]$ need not equal the frequency of any components that comprise $x[n]$, *even in the case that $x[n]$ is a simple sinusoid*.

Not All DT Sinusoids Are Periodic

A continuous-time sinusoid $\cos(2\pi ft + \theta)$ is always periodic regardless of the value of its frequency f . Further, the fundamental period of a single CT sinusoid is always the reciprocal of its frequency, or $T_0 = 1/f = 2\pi/\omega$. Put another way, the fundamental frequency f_0 of a simple CT sinusoid is always equal to the sinusoid's frequency f .

A discrete-time sinusoid $\cos(2\pi\mathcal{F}n + \theta)$, on the other hand, is periodic only if its frequency $\mathcal{F} = \Omega/2\pi$ is a rational number (or Ω is a rational multiple of 2π). This property is the direct result

of the limitation that the period N_0 must be an integer. Further, as we shall see, the frequency \mathcal{F} of a periodic DT sinusoid need not equal the fundamental frequency \mathcal{F}_0 of that very same sinusoid.[†]

Ignoring phase θ for convenience, consider a DT sinusoid $\cos(2\pi\mathcal{F}n)$. Using Eq. (4.29), this signal is N_0 -periodic if

$$\begin{aligned}\cos(2\pi\mathcal{F}n) &= \cos[2\pi\mathcal{F}(n + N_0)] \\ &= \cos(2\pi\mathcal{F}n + 2\pi\mathcal{F}N_0).\end{aligned}$$

This result is possible only if $\mathcal{F}N_0$ is an integer. This integer, which we call m , should be as small as possible since N_0 is the fundamental period. Thus, a DT sinusoid is periodic if

$$\mathcal{F} = \frac{m}{N_0} \quad \text{or} \quad \Omega = 2\pi \frac{m}{N_0}. \quad (4.31)$$

Equation (4.31) tells us that a DT sinusoid is periodic only if \mathcal{F} is a rational number (or Ω is a rational multiple of 2π). Further, Eq. (4.31) confirms that unless $m = 1$, a DT sinusoid's frequency $\mathcal{F} = m/N_0$ is not equal to the signal's fundamental frequency $\mathcal{F}_0 = 1/N_0$. The fundamental period of a DT sinusoid is easily determined by writing \mathcal{F} in the reduced form of Eq. (4.31), where the numerator and denominator are *coprime*.[‡] The next example clarifies these ideas.

▷ Example 4.5 (Establishing the Periodicity of DT Sinusoids)

Sketch the DT sinusoid $\cos(2\pi\mathcal{F}n)$ for the three cases (a) $\mathcal{F} = \frac{3}{15}$, (b) $\mathcal{F} = \frac{1}{1.7\pi}$, and (c) $\mathcal{F} = \frac{1}{5.5}$. In each case, determine whether or not the signal is periodic. If it is, determine the fundamental period N_0 , and state whether the fundamental frequency equals the sinusoid's frequency \mathcal{F} .

Plots of the three DT sinusoids, generated using MATLAB, are shown in Fig. 4.19.

```
01 n = -12:12; subplot(311); stem(n,cos(2*pi*3/15*n));
02 subplot(312); stem(n,cos(2*pi*1/(1.7*pi)*n));
03 subplot(313); stem(n,cos(2*pi*1/5.5*n));
```

For reference, Fig. 4.19 also shows the CT sinusoids $\cos(2\pi\mathcal{F}t)$ that, when sampled using $T = 1$, result in the DT sinusoids $\cos(2\pi\mathcal{F}n)$ of interest.

(a)

In the first case, the frequency $\mathcal{F} = \frac{3}{15}$ is clearly rational, so the DT sinusoid is periodic. However, $\frac{3}{15}$ is not in reduced form since the greatest common divisor of the numerator and denominator is 3 and not 1. Dividing numerator and denominator by this factor yields $\mathcal{F} = \frac{1}{5} = \frac{m}{N_0}$. The fundamental period is thus $N_0 = 5$, which is easily verified using Fig. 4.19a. Since the fundamental frequency $\mathcal{F}_0 = \frac{1}{N_0} = \frac{1}{5}$ equals the sinusoid's frequency $\mathcal{F} = \frac{1}{5}$ (i.e., $m = 1$), we see that N_0 samples of the DT sinusoid contain exactly one cycle of the underlying CT sinusoid $\cos(2\pi\mathcal{F}t)$.

(b)

Due to the factor π in the denominator, the frequency $\mathcal{F} = \frac{1}{1.7\pi}$ is not rational. Thus, the DT sinusoid $\cos(2\pi\frac{1}{1.7\pi}n)$ is not periodic. Although the sequence $\cos(2\pi\frac{1}{1.7\pi}n)$ contains complete information of the underlying CT sinusoid, the DT sequence never repeats. Notice that $\cos(2\pi\frac{1}{1.7\pi}n)$ averages 1.7 π samples (an irrational number) per cycle, and no period N_0 can thus contain an integer number of cycles of the CT sinusoid $\cos(2\pi\mathcal{F}t)$.

As shown in Fig. 4.19b, we see that the sample at $n = 0$ equals 1, the sinusoid's peak. No other sample value, from $-\infty$ to ∞ , ever reaches this peak value. In fact, all samples are unique in this respect. Each sample value is seen once and only once.

[†]DT complex exponentials $e^{j\Omega n}$ have identical periodicity conditions to DT sinusoids. See Prob. 4.3-4.

[‡]Two integers are coprime if their greatest common divisor is 1.

(c)

In the third case, the frequency \mathcal{F} equals $\frac{1}{5.5}$. Multiplying both numerator and denominator by 2 yields $\mathcal{F} = \frac{2}{11}$, which is clearly rational. Thus, the DT sinusoid is periodic. Since 2 and 11 are coprime, the numerator and denominator establish $m = 2$ and $N_0 = 11$, respectively. Since $m \neq 1$, the fundamental frequency $\mathcal{F}_0 = \frac{1}{11}$ does not equal the sinusoid's frequency $\mathcal{F} = \frac{1}{5.5}$. This is simply a reflection that the DT sinusoid only repeats after $m = 2$ oscillations of the underlying CT sinusoid $\cos(2\pi\mathcal{F}t)$, as clarified in Fig. 4.19c. As this case suggests, the frequency of a periodic DT sinusoid is always an integer multiple of the signal's fundamental frequency, $\mathcal{F} = m\mathcal{F}_0$.

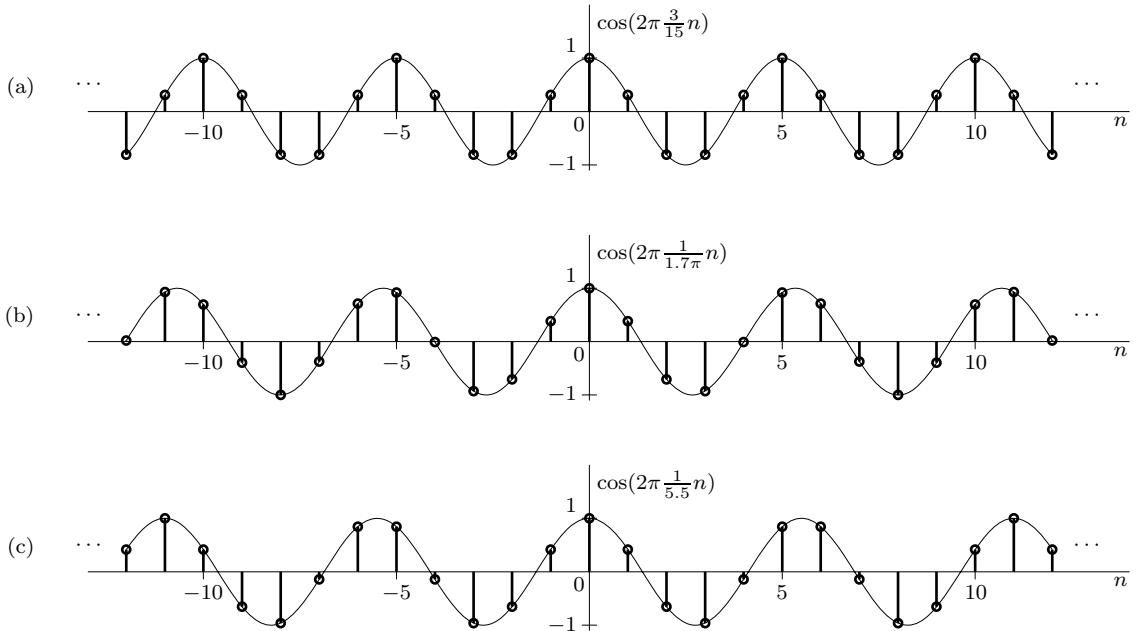


Figure 4.19: DT sinusoids: (a) $\cos(2\pi \frac{3}{15}n)$, (b) $\cos(2\pi \frac{1}{1.7\pi}n)$, and (c) $\cos(2\pi \frac{1}{5.5}n)$.

Example 4.5 ◀

Intuition Makes the Profound Look Trivial: Connections to the Sampling Theorem

A sampled CT sinusoid of frequency f results in a DT sinusoid of frequency $\mathcal{F} = fT$. To avoid aliasing, Eq. (3.2) requires that $|f| < F_s/2 = 1/2T$ or, equivalently, $|\mathcal{F}| < 0.5$. As shown in Fig. 4.15, this is equivalent to our earlier observation that all DT sinusoids are inherently bandlimited.[†] Frequencies \mathcal{F} separated by integer numbers are, as a consequence of aliasing, identical. For instance, discrete-time sinusoids with frequencies $\mathcal{F} = 0.3, 1.3, 2.3, \dots$ cycles/sample are all identical. Clearly, there is a strong connection between the sampling theorem, aliasing, and the bandlimiting phenomenon.

When we see the restriction $|\mathcal{F}| < 0.5$ on discrete-time frequencies, some unsettling questions come to mind. Why are discrete-time frequencies restricted when their continuous-time sisters have the freedom to take on any value? Does this restriction make discrete-time systems second-class citizens, who are forced to service only low-frequency signals?

To answer the first question, first notice that CT signals can have a period as large or as small as we can imagine. For DT signals, however, the period cannot be less than one sample ($N_0 \geq 1$). But a periodic signal with period of one sample results in a constant ($\mathcal{F} = 0$). Hence, the next best

[†]In terms of ω and Ω , the expressions are $\Omega = \omega T$, $|\omega| < \pi/T$, and $|\Omega| < \pi$.

choice is $N_0 = 2$. This is the minimum period (maximum frequency) that is possible. Thus, the highest possible discrete-time frequency has a period of 2 and a maximum frequency $\mathcal{F}_{\max} = 0.5$ cycles/sample. This profound result ($\mathcal{F} < \mathcal{F}_{\max} = 0.5$), which basically restates the venerable *sampling theorem*, is just an intuitively trivial fact. Intuition often makes profound results look trivial.

Coming back to the second question, the answer is an emphatic *No!* The limitation $\mathcal{F} < 0.5$ has no detrimental effect on the ability of discrete-time systems to process high-frequency continuous-time signals. Since $\mathcal{F} = fT$, we can decrease T enough to ensure that $\mathcal{F} < 0.5$ no matter how high is f , the frequency to be processed. In summary, a discrete-time system has a limit as to how high a frequency f it can process for a given value of the sampling interval T . However, we can process a continuous-time signal of any frequency, no matter how high, by decreasing T sufficiently.

▷ Drill 4.10 (Establishing the Periodicity of DT Sinusoids)

State with reasons whether the following sinusoids are periodic. If periodic, find the fundamental period N_0 , and determine whether the fundamental frequency is equal to the sinusoid's frequency.

- (a) $\cos(\frac{3\pi}{7}n)$ (b) $\cos(\frac{10}{7}n)$ (c) $\cos(\sqrt{\pi}n)$ (d) $\sin(2.35\pi n + 1)$

△

4.3.5 DT Energy and Power Signals

Arguing along the lines similar to those used for continuous-time signals, the size of a discrete-time signal $x[n]$ is measured by its energy E_x or its power P_x . The *energy* of DT signal $x[n]$ is defined as

$$E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2. \quad (4.32)$$

This definition is valid for real or complex $x[n]$. For this measure to be meaningful, the energy of a signal must be finite. A necessary condition for the energy to be finite is that the signal amplitude must $\rightarrow 0$ as $|n| \rightarrow \infty$. Otherwise the sum in Eq. (4.32) will not converge. If E_x is finite, the signal is called an *energy signal*.

In some cases, such as when the amplitude of $x[n]$ does not $\rightarrow 0$ as $|n| \rightarrow \infty$, signal energy is infinite. In such cases, a more meaningful measure of size is signal power P_x (if it exists), which is defined as the time average of energy and is given by

$$P_x = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2. \quad (4.33)$$

In this equation, the sum is divided by $2N+1$ because there are $2N+1$ samples within the interval from $-N$ to N . For an N_0 -periodic signal, the time averaging need be performed only over one period, such as[†]

$$P_x = \frac{1}{N_0} \sum_{n=0}^{N_0-1} |x[n]|^2. \quad (4.34)$$

If P_x is finite and nonzero, the signal is called a *power signal*. As in the continuous-time case, a discrete-time signal can be either an energy signal or a power signal but cannot be both at the same time. Some signals are neither energy nor power signals.

[†]The sum limits in Eq. (4.34) can use any contiguous set of N_0 points, not just the ones from 0 to $N_0 - 1$.

▷ **Example 4.6 (Energy and Power of DT Signals)**

Figure 4.20 shows a signal $x[n]$ as well as two periodic signals created by periodic extension of $x[n]$,

$$\tilde{x}_1[n] = \sum_{k=-\infty}^{\infty} x[n+6k] \quad \text{and} \quad \tilde{x}_2[n] = \sum_{k=-\infty}^{\infty} x[n+7k].$$

Determine the energy and power for each of these three DT signals.

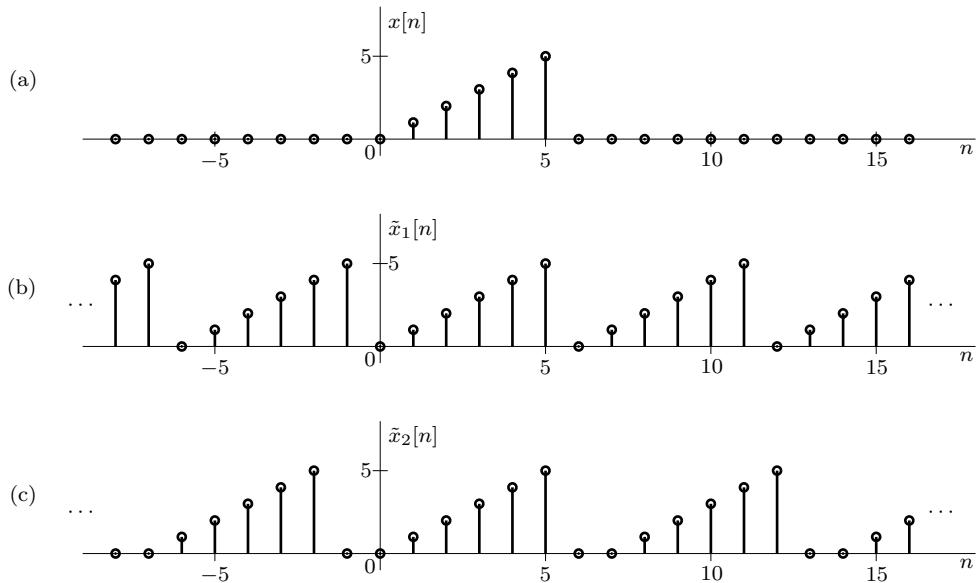


Figure 4.20: Computing power and energy for various DT signals.

Since $x[n] = n$ over $0 \leq n \leq 5$ and is 0 elsewhere, its energy is found using Eq. (4.32) as

$$E_x = \sum_{n=0}^5 n^2 = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 55.$$

Since E_x is finite, $x[n]$ is an energy signal and $P_x = 0$.

The signal $\tilde{x}_1[n]$ is a periodic replication of $x[n]$ with a period of 6. All periodic signals have infinite energy, so $E_{\tilde{x}_1} = \infty$. The power of $\tilde{x}_1[n]$ is found using Eq. (4.34) as

$$P_{\tilde{x}_1} = \frac{1}{6} \sum_{n=0}^5 n^2 = \frac{1}{6} E_x = \frac{55}{6} \approx 9.1667.$$

The signal $\tilde{x}_2[n]$ is also a periodic replication of $x[n]$ but has period 7. As with $\tilde{x}_1[n]$, the energy of $\tilde{x}_2[n]$ is infinite, $E_{\tilde{x}_2} = \infty$. The power of $\tilde{x}_2[n]$ is found using Eq. (4.34) as

$$P_{\tilde{x}_2} = \frac{1}{7} \sum_{n=0}^5 n^2 = \frac{1}{7} E_x = \frac{55}{7} \approx 7.8571.$$

Example 4.6 ◀

▷ **Drill 4.11 (Energy and Power of a Causal DT Exponential)**

Show that the signal $x[n] = z^n u[n]$ is an energy signal of energy $E_x = \frac{1}{1-|z|^2}$ if $|z| < 1$, that it is a power signal of power $P_x = 0.5$ if $|z| = 1$, and that it is neither an energy signal nor a power signal if $|z| > 1$.

△

4.4 DT Systems and Examples

Systems whose inputs and outputs are discrete-time signals are called *discrete-time systems*. In other words, discrete-time systems process discrete-time signals (inputs) to yield another set of discrete-time signals (outputs). By using the *operator* H to represent the system function, the system output $y[n]$ is compactly represented in terms of the input $x[n]$ as

$$y[n] = H\{x[n]\}.$$

Figure 4.21 illustrates this general relationship between input, output, and system operator.

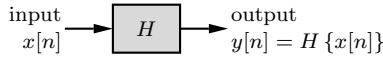


Figure 4.21: Block diagram for discrete-time system H .

Before we start with the usual and necessary mathematical analysis of discrete-time systems, we shall try to get some feel for discrete-time systems through several practical examples. In the first examples, the signals are inherently discrete-time. In the final examples, continuous-time signals are sampled, processed by discrete-time systems, and then reconstructed (see Fig. 4.2). In each case, we also present the DT system H in block diagram form, which provides a road map to hardware or software realization of the system. Typical DT systems diagrams require three basic blocks: *adder* (Fig. 4.22a), *scalar multiplier* (Fig. 4.22b), and *delay* (Fig. 4.22c). While labeling the DT time-shifting (delay) block as z^{-m} may seem somewhat mysterious at present, we shall learn in Ch. 7 that such a notation is well justified. In addition, we also have a *pick-off* or *branch* node (Fig. 4.22d), which is used to provide multiple copies of a signal.

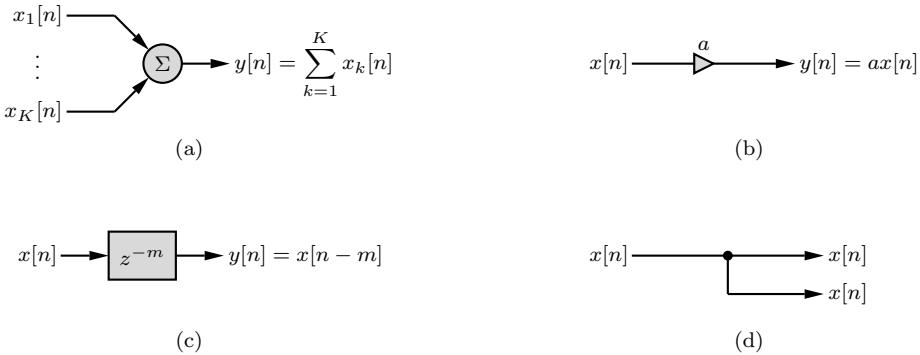


Figure 4.22: Elementary DT blocks: (a) adder, (b) scalar multiplier, (c) delay, and (d) branch.

▷ **Example 4.7 (Cash Register)**

Determine the nonrecursive and recursive input-output equations of a cash register, where the input $x[n]$ is the price of the n th item and the output $y[n]$ is the total value of all n items entered in the register. Provide a block diagram representation of the system.

A cash register's input and output are both DT signals. We can express the output $y[n]$ as

$$y[n] = \sum_{k=1}^n x[k] \quad n = 1, 2, 3, \dots \quad (4.35)$$

This is an input-output equation of a cash register, which is an example of an *accumulator* system. However, no practical cash register works this way. It is too wasteful to keep on adding the prices of all previous items every time a new item is added. In practice, we keep a running total of the prices of the previous items and update this total every time a new item is added. This is equivalent to expressing Eq. (4.35) in an alternate form as

$$\begin{aligned} y[n] &= \underbrace{\sum_{k=1}^{n-1} x[k]}_{y[n-1]} + x[n] \\ &= y[n-1] + x[n] \quad n = 1, 2, 3, \dots \end{aligned}$$

We can obtain this equation directly by observing that the output $y[n]$ is a sum of $y[n-1]$ (total price of previous $n-1$ items) and $x[n]$, the price of the n th item. Grouping input and output terms on separate sides of the equation yields

$$y[n] - y[n-1] = x[n] \quad n = 1, 2, 3, \dots \quad (4.36)$$

Equation (4.36) is an example of a *difference equation* that, because the largest delay is 1, is first-order. To be complete, Eq. (4.36) also requires knowledge of the *initial condition* $y[0] = 0$. We treat system order and initial conditions more thoroughly later on.

Recursive and Nonrecursive Forms

The cash register representation of Eq. (4.36) is a *recursive* form, and Eq. (4.35) is a *nonrecursive* form. Since both expressions properly model the system, what is the difference between the two? Which form is preferable? To answer these questions, let us examine how each computes its output.

In Eq. (4.35), the output $y[n]$ at any instant n is computed by adding the current and *all* past input values. As n increases, so do the number of terms being summed. In contrast, Eq. (4.36), written as $y[n] = y[n-1] + x[n]$, computes the output $y[n]$ as the addition of only two values: the previous output value $y[n-1]$ and the present input value $x[n]$. The computations are done recursively using the previous output values. For example, if the input starts at $n = 1$, we first compute $y[1]$. Once $y[1]$ is computed, we compute $y[2]$ using the value $y[1]$. Knowing $y[2]$, we compute $y[3]$, and so on.

The computational burden of the nonrecursive form increases with n . For the recursive form, however, the computational burden remains fixed for all time. Thus, although both forms describe a cash register's output in terms of its input, the recursive form is more efficient than the nonrecursive form.

Figure 4.23 shows the block diagram representation (implementation) of Eq. (4.36), the preferred recursive form. Clearly, the output $y[n]$ is just the sum of the input $x[n]$ and the previous output value $y[n-1]$.

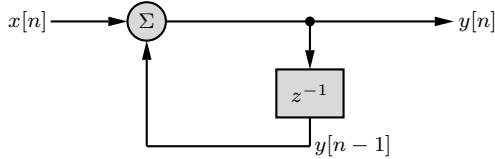


Figure 4.23: Cash register block representation.

Delay and Advance Forms

We may also write Eq. (4.36) in an alternate form by observing that the choice of index n is completely arbitrary. Thus, substituting $n + 1$ for n yields an alternate description of

$$y[n + 1] - y[n] = x[n + 1] \quad n = 0, 1, 2, \dots \quad (4.37)$$

Since it uses advance operations, the difference equation in Eq. (4.37) is said to be in *advance form*. Similarly, Eq. (4.36), which uses delay operations, is said to be in *delay form*. Delay form is more natural because delay operations are causal and hence realizable. In contrast, advance form, being noncausal, is unrealizable. Still, advance form is useful since it results in discrete-time equations that are identical in form to those for CT systems.

The Accumulator

The cash register represented by Eqs. (4.35) and (4.36) is a special case of an accumulator system with a causal input. The general equation of an accumulator is

$$y[n] = \sum_{k=-\infty}^n x[k].$$

This equation is equivalent to the recursive equation

$$y[n] - y[n - 1] = x[n]. \quad (4.38)$$

In discrete-time systems, the function of an accumulator is roughly analogous to that of an integrator in continuous-time systems.

Example 4.7 ◀

▷ Example 4.8 (Savings Account)

A person makes a deposit (the input) in a bank regularly at an interval of $T = 1$ month. The bank pays a certain interest on the account balance during the period T and mails out a periodic statement of the account balance (the output) to the depositor. Find the equation relating the output $y[n]$ (the balance) to the input $x[n]$ (the deposit). Assuming that a person invests \$100 monthly starting at $n = 1$ and earns a 0.5% monthly interest rate, how much money is earned at $n = 100$? Provide a block diagram representation of the system.

In this case, the signals are inherently discrete-time. Let

$x[n]$ = the n th deposit

$y[n]$ = the month- n account balance, including the n th deposit

r = interest rate per period T

The balance $y[n]$ is the sum of the previous balance $y[n - 1]$, the interest on $y[n - 1]$ during the period T , and the deposit $x[n]$, or

$$\begin{aligned}y[n] &= y[n - 1] + ry[n - 1] + x[n] \\&= (1 + r)y[n - 1] + x[n]\end{aligned}$$

Letting $a_1 = -(1 + r)$, the savings account is represented in standard delay form as

$$y[n] + a_1 y[n - 1] = x[n]. \quad (4.39)$$

Simple recursion, computed in MATLAB, offers one way to determine the money earned by investing \$100 monthly at 0.5% interest per month for 100 months.

```
01 r = 0.005; a1 = -(1+r); y(1) = 100;
02 for n = 2:100, y(n) = -a1*y(n-1)+100; end
03 y(100)-100*100
ans = 2933.37
```

Thus, on an investment of $100 \times \$100 = \$10,000$, \$2,933.37 is earned in interest. The annual percent yield (APY) of this account is computed as $(1.005)^{12} - 1 = 0.0617$, or 6.17%. As a note of caution, although in this case the month n complies with MATLAB's index requirements (positive integers starting at 1), this is not always the case; more often than not, the DT time variable n cannot serve (directly) as a MATLAB vector index.

Figure 4.24 illustrates the block representation of Eq. (4.39). The origin of the negative sign attached to the scale factor a_1 is easily seen by rewriting Eq. (4.39) as

$$y[n] = -a_1 y[n - 1] + x[n].$$

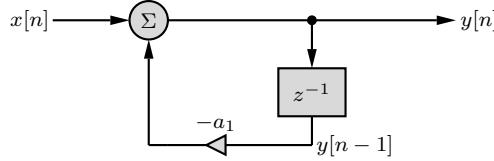


Figure 4.24: Savings account block representation.

As it turns out, the savings account model is rather flexible. A withdrawal from the account is a negative deposit. Therefore, this formulation can handle deposits as well as withdrawals. It also applies to a loan payment problem with the initial value $y[0] = -M$, where M is the amount of the loan. Since we can also treat a loan as an initial deposit with a negative value, we may treat a loan of M dollars taken at $n = 0$ as an input of $-M$ at $n = 0$.

Example 4.8 ◀

▷ Example 4.9 (Digital Differentiator)

Design a discrete-time system like the one in Fig. 4.2 to differentiate continuous-time signals. Construct a block diagram representation of the system, and investigate its response to a ramp input $x(t) = tu(t)$. Suggest an appropriate sampling interval if this is used in an audio system where the input signal bandwidth is below 20 kHz.

In this case, the output $y(t)$ is required to be the derivative of the input $x(t)$. The discrete-time system processes the samples of $x(t)$ to produce the discrete-time output $y[n]$, which is used to

construct $y(t)$. Let $x[n]$ and $y[n]$ represent the uniform samples of signals $x(t)$ and $y(t)$, respectively. If T is the sampling interval, then

$$x[n] = x(nT) \quad \text{and} \quad y[n] = y(nT).$$

The signals $x[n]$ and $y[n]$ are the input and output for the DT system. Now, we require that

$$y(t) = \frac{d}{dt}x(t).$$

Using the definition of a derivative and setting $t = nT$, we obtain

$$y(nT) = \left. \frac{d}{dt}x(t) \right|_{t=nT} = \lim_{T \rightarrow 0} \frac{x(nT) - x[(n-1)T]}{T}.$$

Using the notation in Eq. (3.1), this equation becomes

$$y[n] = \lim_{T \rightarrow 0} \frac{x[n] - x[n-1]}{T}.$$

This is the input-output relationship needed to achieve our objective. In practice, the sampling interval T cannot be zero. Assuming that T is sufficiently small, however, the system is approximated as

$$y[n] = \frac{1}{T}x[n] - \frac{1}{T}x[n-1]. \quad (4.40)$$

This approximation improves as T approaches 0. Figure 4.25 represents the system in block form.

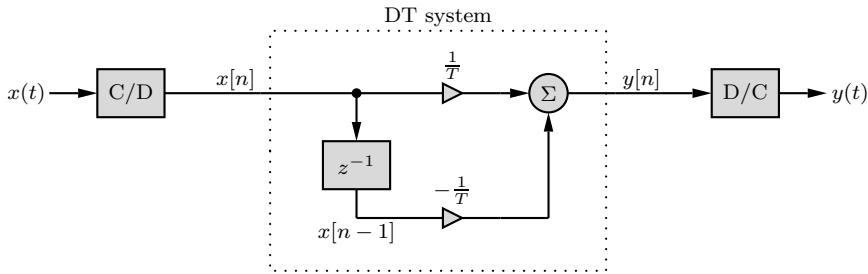


Figure 4.25: Digital differentiator block representation.

To see how well our digital differentiator works, let us consider the implementation in Fig. 4.25 with a ramp input $x(t) = tu(t)$, depicted in Fig. 4.26a. The samples of the input, taken every T seconds, are given as

$$x[n] = x(t)|_{t=nT} = tu(t)|_{t=nT} = nTu[n].$$

Figure 4.26b shows the sampled signal $x[n]$, and Fig. 4.26c shows the output $y[n]$, which is the $\frac{1}{T}$ -scaled difference between successive input samples. The CT output $y(t)$, constructed from $y[n]$ using a linear-interpolation rule, is very close to the ideal output of a unit step function $u(t)$. As T approaches zero, the output $y(t)$ approaches the desired output $u(t)$.

Clearly, the choice of sampling interval T is important to system performance. Chosen too small, the system operates at excessively high speed, increasing cost. Chosen too large, the system fails to properly operate. According to the sampling theorem of Ch. 3, an input with frequencies below $f_{\max} = 20$ kHz requires a sampling interval

$$T \leq \frac{1}{2f_{\max}} = \frac{1}{40000} = 25 \mu\text{s}.$$

However, the lower limit $T = 25 \mu\text{s}$, which is the Nyquist rate, does not deliver good differentiator performance except at relatively low frequencies. Later, once we have covered the frequency-domain analysis of DT signals and systems, we shall be able to improve our choice of sampling interval.

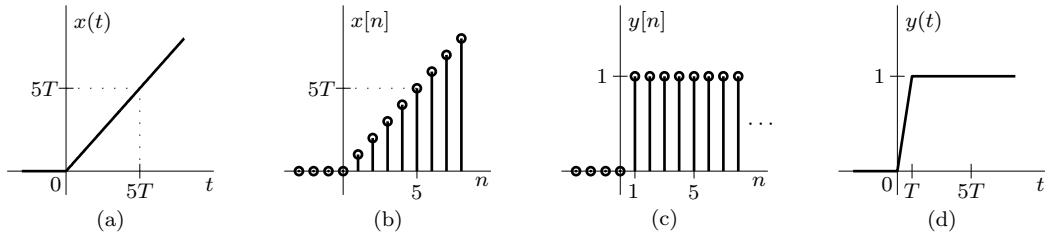


Figure 4.26: Digital differentiator operation: (a) CT input $x(t) = tu(t)$, (b) DT input $x[n]$, (c) DT output $y[n]$, and (d) CT output $y(t)$.

Backward and Forward Difference Systems

The digital differentiator in Fig. 4.25 is an example of what is known as a *backward difference system*. The reason for this naming is obvious from Eq. (4.40). To compute the derivative of $y(t)$, we use the difference between the present sample value and the previous (backward) sample value. The *forward difference* form of the digital differentiator, found using the difference between the next (forward) sample $x[n + 1]$ and the present sample $x[n]$, is given as

$$y[n] = \frac{1}{T}x[n + 1] - \frac{1}{T}x[n].$$

In the literature, the *first-order backward difference* is specified by the equation

$$y[n] = x[n] - x[n - 1]. \quad (4.41)$$

Clearly, the digital differentiator described by Eq. (4.40) is just the first-order backward difference scaled by $1/T$. Similarly, the *first-order forward difference* is specified by the equation

$$y[n] = x[n + 1] - x[n]. \quad (4.42)$$

The forward and backward difference systems each perform a function that is analogous to that of a DT differentiator. The backward difference system is causal and therefore physically realizable, while the forward difference system is not.

Example 4.9 ◀

▷ Example 4.10 (Digital Integrator)

Design discrete-time systems like the one in Fig. 4.2 to integrate continuous-time signals using (a) a backward-staircase approximation, (b) a forward-staircase approximation, and (c) a trapezoidal approximation. Construct a block diagram representation for each system.

A CT integrator operates as $y(t) = \int_{-\infty}^t x(\tau) d\tau$. Much like the digital differentiator of Ex. 4.9, it is possible to approximate a CT integrator with simple DT systems. Figure 4.27 illustrates the idea behind three different approximations. The first two methods use staircase approximations, with each “stair” extending either backward (Fig. 4.27a) or forward (Fig. 4.27b) from the sample points $x[n]$. The third method uses a trapezoidal approximation, where each trapezoid is formed using adjacent pairs of input samples.

(a) Backward Staircase Approximation

To begin, we approximate the input $x(t)$ by a staircase, where each stair extends backward of the respective input samples $x(nT)$. To approximate $y(t)$, the area of $x(t)$ from $-\infty$ to t , we simply

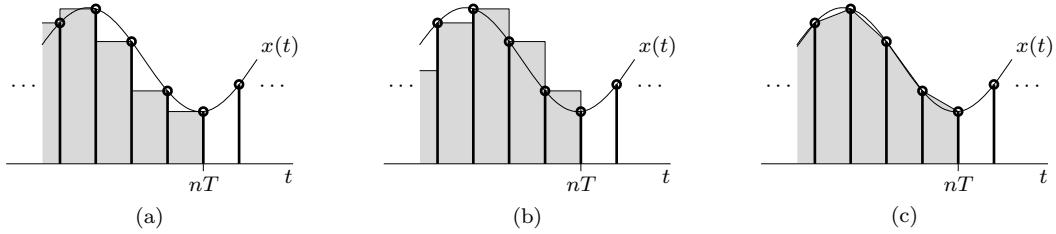


Figure 4.27: Digital integrator methods: (a) backward staircase, (b) forward staircase, and (c) trapezoidal.

add the areas of the rectangular strips that lie to the left of $x(nT)$ (shown shaded in Fig. 4.27a). As $T \rightarrow 0$, the area under the staircase approaches the exact area under $x(t)$. Thus,

$$y(nT) = \lim_{T \rightarrow 0} \sum_{k=-\infty}^n T x(kT).$$

Assuming that T is small enough to justify the assumption $T \rightarrow 0$, this equation can be expressed in terms of our DT samples as

$$y[n] = T \sum_{k=-\infty}^n x[k]. \quad (4.43)$$

Clearly, this model of the digital integrator is basically an accumulator. Equation (4.43) represents the nonrecursive form of the digital integrator. We can also express it in a recursive form as

$$y[n] - y[n-1] = T x[n]. \quad (4.44)$$

Equations (4.43) and (4.44) are equivalent; one can be derived from the other.

(b) Forward Staircase Approximation

In this case, we also approximate the input $x(t)$ by a staircase, but each stair extends forward rather than backward. Adding the rectangular strips to the left of $x(nT)$ (shown shaded in Fig. 4.27b) yields the model

$$y(nT) = \lim_{T \rightarrow 0} \sum_{k=-\infty}^n T x([k-1]T)$$

Paralleling the development in part (a), the recursive form of this system is

$$y[n] - y[n-1] = T x[n-1]. \quad (4.45)$$

Notice that Eq. (4.45) is identical to Eq. (4.44) except that the input is delayed by one unit.

(c) Trapezoidal Approximation

These two staircase methods (Eqs. (4.44) and (4.45)) approximate the desired integral and improve as $T \rightarrow 0$. Depending on whether the slope of $x(t)$ is positive or negative, however, one approximation tends to overstate the value, while the other tends to underestimate it. For a better result, would it not be preferable to take the average value of these two approximations? The trapezoidal approximation does precisely that.

To begin, we approximate the input $x(t)$ by trapezoids, as shown in Fig. 4.27c. As $T \rightarrow 0$, the area under the trapezoids approaches the exact area under $x(t)$. Using Fig. 4.27c, it follows that

$$y(nT) = \lim_{T \rightarrow 0} \sum_{k=-\infty}^n \frac{T}{2} [x(kT) + x([k-1]T)].$$

Clearly, the trapezoidal approximation is the average of the previous two staircase approximations.

Again assuming that T is small enough to justify $T \rightarrow 0$, this equation leads to

$$y[n] - y[n-1] = \frac{T}{2} (x[n] + x[n-1]). \quad (4.46)$$

This recursive form is well suited for system realization.

As Fig. 4.28 demonstrates, a generalized first-order DT system can realize each of our three digital integrators. The three cases differ only in the values of the scaling coefficients. In fact, the structure in Fig. 4.28 is also appropriate for the accumulator and cash register models of Ex. 4.7, the savings account model of Ex. 4.8, the digital differentiator models of Ex. 4.9, and many others. An amazing variety of tasks can be accomplished using very simple, low-order DT models.

Approximation Method	a_1	b_0	b_1
Backward staircase	-1	T	0
Forward staircase	-1	0	T
Trapezoidal	-1	$\frac{T}{2}$	$\frac{T}{2}$

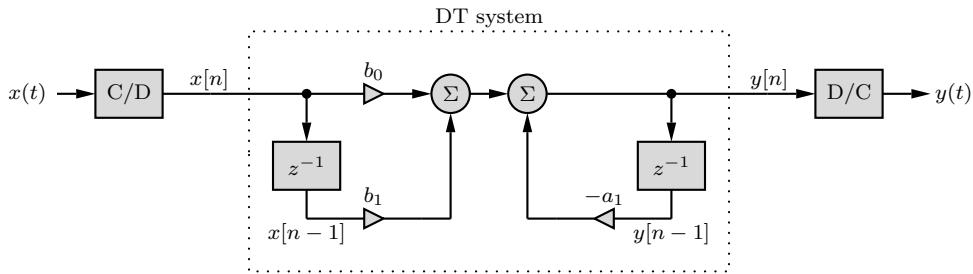


Figure 4.28: Digital integrator realizations using generalized first-order DT system.

Example 4.10 ◀

4.4.1 The Order and General Form of Difference Equations

Equations (4.38), (4.39), (4.40), and (4.46) are all examples of simple, first-order difference equations. Just as the highest-order derivative (or integral) of the input or output represents the order of a differential (or integral) equation, the highest-order difference of the input or output represents the *order* of a difference equation. Order is reflective of system complexity. For CT systems, order indicates the number of energy storage elements, such as capacitors and inductors, required for implementation. For DT systems, order indicates the required amount of memory.

In advance form, a general-order difference equation is expressed as

$$\begin{aligned} a_0 y[n+K] + a_1 y[n+K-1] + \cdots + a_{K-1} y[n+1] + a_K y[n] = \\ b_0 x[n+L] + b_1 x[n+L-1] + \cdots + b_{L-1} x[n+1] + b_L x[n] \end{aligned}$$

or, more compactly, as

$$\sum_{k=0}^K a_k y[n+K-k] = \sum_{l=0}^L b_l x[n+L-l]. \quad (4.47)$$

Here, the system order is clearly either K or L , whichever is largest.

4.4.2 Kinship of Difference Equations to Differential Equations

As the two previous examples demonstrate, difference equations ably serve as differentiators and integrators. We now show that a digitized version of a differential equation results in a difference equation. Let us consider a simple first-order differential equation

$$\frac{d}{dt}y(t) + cy(t) = x(t). \quad (4.48)$$

Consider uniform samples of $x(t)$ at intervals of T seconds. As usual, we use the notation $x[n]$ to denote $x(nT)$, the n th sample of $x(t)$. Similarly, $y[n]$ denotes $y(nT)$, the n th sample of $y(t)$. From the basic definition of a derivative, we can express Eq. (4.48) at $t = nT$ as

$$\lim_{T \rightarrow 0} \frac{y[n] - y[n-1]}{T} + cy[n] = x[n].$$

Clearing the fractions and rearranging the terms yield (assuming nonzero but very small T)

$$y[n] + \alpha y[n-1] = \beta x[n], \quad (4.49)$$

where

$$\alpha = \frac{-1}{1+cT} \quad \text{and} \quad \beta = \frac{T}{1+cT}.$$

We can also express Eq. (4.49) in advance form as

$$y[n+1] + \alpha y[n] = \beta x[n+1]. \quad (4.50)$$

Clearly, the differential equation of Eq. (4.48) can be approximated by a difference equation of the same order.

By extension, we can approximate a differential equation of K th order by a difference equation of K th order. For example, we can approximate a second derivative by observing that

$$\begin{aligned} \left. \frac{d^2}{dt^2} y(t) \right|_{t=nT} &= \lim_{T \rightarrow 0} \frac{1}{T} \left(\left. \frac{d}{dt} y(t) \right|_{t=nT} - \left. \frac{d}{dt} y(t) \right|_{t=(n-1)T} \right) \\ &\approx \frac{1}{T} \left(\frac{y[n] - y[n-1]}{T} - \frac{y[n-1] - y[n-2]}{T} \right) \\ &= \frac{1}{T^2} (y[n] - 2y[n-1] + y[n-2]). \end{aligned} \quad (4.51)$$

Similarly, we can show that a K th-order derivative can be approximated by

$$\left. \frac{d^K}{dt^K} y(t) \right|_{t=nT} \approx \frac{1}{T^K} \sum_{k=0}^K (-1)^k \binom{K}{k} y[n-k]. \quad (4.52)$$

Software packages often model and solve differential equations using such approximations, which only require the simple operations of addition, multiplication, and delay. Results can be made as close to the exact answer as required by choosing a sufficiently small value for T . To perform well across all input frequencies, approximations such as Eq. (4.52) normally require T to be much smaller than Nyquist demands. Later chapters treat the selection of T in greater depth and also provide simpler procedures, such as the impulse invariance and bilinear transform methods, to find a discrete-time system to realize a K th-order LTIC system.

► Example 4.11 (Modeling Differential Equations with Difference Equations)

Assuming a sampling interval $T = 0.1$, determine a difference equation model for the differential equation

$$\ddot{y}(t) + 4\dot{y}(t) + 3y(t) = 100x(t)$$

with initial conditions $y(0) = 0$ and $\dot{y}(0) = 10$.

Using Eq. (4.52), the differential equation is approximated as

$$\frac{1}{T^2} (y[n] - 2y[n-1] + y[n-2]) + \frac{4}{T} (y[n] - y[n-1]) + 3y[n] = 100x[n].$$

Combining terms and substituting $T = 0.1$ yield

$$143y[n] - 240y[n-1] + 100y[n-2] = 100x[n].$$

To compute the equivalent initial conditions, we note that $y[0] = y(0) = 0$. Further,

$$10 = \dot{y}(0) = \left. \frac{d}{dt} y(t) \right|_{t=0} \approx \frac{y(0) - y(0-T)}{T} = \frac{y[0] - y[-1]}{0.1} = -10y[-1].$$

Assuming that T is sufficiently small to justify Eq. (4.52), this leads to $y[-1] = -1$. Following normalization of the coefficient for $y[n]$, the differential equation is therefore modeled as

$$y[n] - \frac{240}{143}y[n-1] + \frac{100}{143}y[n-2] = \frac{100}{143}x[n]$$

with initial conditions $y[0] = 0$ and $y[-1] = -1$.

Example 4.11 ◀

▷ Drill 4.12 (Approximating Derivatives)

Following the same approach used to derive Eq. (4.51), find an approximation for $\frac{d^2}{dt^2}y(t)$ using *forward differences*. How does this result compare with Eq. (4.51)?

◀

▷ Drill 4.13 (Modeling Differential Equations with Difference Equations)

Find a causal difference equation and initial conditions that approximates the behavior of the second-order differential equation

$$\frac{d^2}{dt^2}y(t) + 3\frac{d}{dt}y(t) + 2y(t) = 1000x(t)$$

with initial conditions $y(0) = 0$ and $\dot{y}(0) = 3$. Use the sampling interval $T = 0.05$.

◀

Comments on Analog, Digital, Continuous-Time, and Discrete-Time

In terms of describing signals and systems, Secs. 1.1.1 and 1.1.2 fully explain the differences between continuous-time and discrete-time and between analog and digital. Briefly, the terms “continuous-time” and “discrete-time” qualify the nature of a signal along the time axis (horizontal axis). The terms “analog” and “digital”, in contrast, qualify the nature of the signal amplitude (vertical axis). Historically, discrete-time systems have been realized using digital computers, whereas continuous-time signals are processed through digitized samples rather than unquantized samples. Because of this fact, the terms “digital filters”, “digital systems”, and “discrete-time systems” are used synonymously in the literature. For this reason, we follow the loose convention in this book where the term “digital filter” implies *discrete-time system*, and “analog filter” means *continuous-time system*. Moreover, the terms “C/D” (continuous-to-discrete-time) and “D/C” will occasionally be used interchangeably with terms “A/D” (analog-to-digital) and “D/A”, respectively.

4.4.3 Advantages of Digital Signal Processing

There are many advantages of digital signal processing over its continuous-time, analog counterpart. We consider here some of the most important advantages.

1. Digital systems can tolerate considerable variation in signal values and, hence, are less sensitive to changes in the component parameter values caused by temperature variations, aging, and other factors. This results in a greater degree of precision and stability. Digital systems are easily duplicated in volume without having to worry about precise component values. Unlike analog systems, units do not require individual factory calibration or adjustment.
2. Digital systems are extremely flexible and easy to implement. Digital filter function is easily altered by simply changing the program. Additionally, more sophisticated algorithms are available for digital signal processing than for analog signal processing. It is quite difficult, for example, to realize and tune analog filters with double-digit orders, but triple-digit orders are readily achieved with digital filters.
3. Even in the presence of noise, reproduction with digital messages is extremely reliable, often without any deterioration whatsoever. Analog messages such as photocopies and films, on the other hand, lose quality at each successive stage of reproduction. Further, digital signals can be coded to yield extremely low error rates, high fidelity, error correction capabilities, and privacy.
4. Digital signals are easily and inexpensively stored electronically, without any loss of signal quality. Further, electronic storage permits efficient search, selection, and delivery of data.
5. Digital filters can be easily time shared and therefore can serve a number of inputs simultaneously. Compared with analog signals, it is easier and more efficient to multiplex several digital signals on a single channel.
6. Digital implementation permits the use of a wide variety of hardware options, including computers and microprocessors, DSPs (digital signal processors) and FPGAs (field programmable gate arrays), digital switching, and others. Digital systems can be fully integrated, and even highly complex systems can be placed on a single chip using VLSI (very large scale integrated) circuits. Being binary circuits, accuracy can be increased by increasing word length, subject to cost and complexity limitations.

Of course, one must weigh these advantages against the disadvantages of digital signal processing, such as increased system complexity due to A/D and D/A interfaces, limited range of frequencies available in practice (hundreds of megahertz to gigahertz), and greater power use than passive analog circuits (digital systems use power-consuming active devices).

4.5 DT System Properties

The properties that describe continuous-time systems are, with minor modifications, useful to describe discrete-time systems. Once again, we shall find that linearity and time invariance are the cornerstone properties of DT systems. Together, linearity and time invariance provide a direct path to characterize the response of discrete-time systems. Other important system properties include causality, stability, memory, and invertibility. Similar to our CT conventions, we shall sometimes use the notation $x[n] \implies y[n]$ to indicate that $y[n]$ is the response of a given system to the input $x[n]$, that is, $y[n] = H\{x[n]\}$.

4.5.1 Time Invariance

A system is time invariant or shift invariant (also *constant parameter*) if

$$x[n] \implies y[n] \quad \text{implies that} \quad x[n-m] \implies y[n-m] \quad (4.53)$$

for every input $x[n]$ and every shift m , assuming that the initial conditions are zero. This condition is satisfied if the system parameters do not change with the independent variable n . If the system parameters change with n , then Eq. (4.53) is not satisfied, and the system is time variant.

As Fig. 4.29 helps to illustrate, the time-invariance property is satisfied if a system acting on a delayed input (Fig. 4.29a) produces the same result as delaying the output of the system in response to the original input (Fig. 4.29b). As in the CT case, *time invariance implies that a system operator H commutes with the time-shifting operation z^{-m} .*

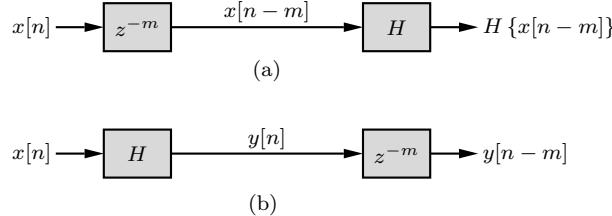


Figure 4.29: Visualizing time invariance: (a) a DT system acting on a delayed input and (b) the delayed output of a DT system.

To provide a brief example, consider a system described by

$$y[n] = e^{-n}x[n].$$

Delaying this output by m yields (Fig. 4.29b)

$$y[n-m] = e^{-(n-m)}x[n-m].$$

Following Fig. 4.29a, the output of a delayed input $x[n-m]$ is

$$H\{x[n-m]\} = e^{-n}x[n-m].$$

Since $y[n-m] = e^{-(n-m)}x[n-m] \neq e^{-n}x[n-m] = H\{x[n-m]\}$, the system is time variant.

▷ Example 4.12 (A Practical Time-Variant System)

In several probability applications and combinatorial analysis, we need to compute $\binom{N}{n}$, which represents the number of combinations of N items taken n at a time. This term, called the *binomial coefficient*, is given by

$$\binom{N}{n} = \frac{N!}{n!(N-n)!} \quad \text{for } n = 0, 1, 2, \dots, N.$$

Even for moderate values of N , direct computation of $\binom{N}{n}$ for $n = 0, 1, 2, \dots, N$ requires a large number of multiplications. To help address this difficulty, derive a recursive difference equation to solve for the binomial coefficients. Show that the system is time variant.

Although most systems considered in this book are time invariant, this example demonstrates a useful system that is time variant. To begin, consider a system whose output $y[n]$ is the binomial coefficient. In this case,

$$y[n] = \frac{N!}{n!(N-n)!} \quad \text{for } n = 0, 1, 2, \dots, N$$

$$y[n-1] = \frac{N!}{(n-1)!(N-n+1)!} \quad \text{for } n = 1, 2, 3, \dots, N+1$$

From these two equations, we have

$$y[n] - ay[n-1] = 0 \quad \text{for } n = 1, 2, 3, \dots, N,$$

where

$$a = \frac{N-n+1}{n}.$$

This is a first-order difference equation with time-varying parameter a . Because a system parameter varies with n , the system fails to satisfy the time-invariance condition of Eq. (4.53).

We can compute binomial coefficients $\binom{N}{n}$ by finding the output of this system iteratively using zero input and the initial condition $y[0] = \binom{N}{0} = 1$. This proves to be much more economical in terms of the number of multiplications and divisions required. To compute the entire set of coefficients for $n = 0, 1, 2, \dots, N$, we need only $N-1$ divisions and $N-1$ multiplications. Contrast this with the total of $2N^2-9N+10$ multiplications and $N-2$ divisions needed by the direct method [2].

Example 4.12 

4.5.2 Linearity

For discrete-time systems, the definition of *linearity* is identical to that for continuous-time systems, as given in Eq. (1.35). Consequently, there is little need to duplicate all the detailed discussions of Sec. 1.5.1. Rather, we highlight the most important aspects of linearity from a DT system perspective.

Linearity is defined by *superposition*, which includes the *additivity* and *homogeneity* properties discussed in Ch. 1. Briefly, the homogeneity (or scaling) property states that if $x[n] \Rightarrow y[n]$, then $c x[n] \Rightarrow c y[n]$ for all real or imaginary constant c . The additivity property states that if $x_1[n] \Rightarrow y_1[n]$ and $x_2[n] \Rightarrow y_2[n]$, then $x_1[n] + x_2[n] \Rightarrow y_1[n] + y_2[n]$, assuming that $y_1[n]$ and $y_2[n]$ are zero-state responses. Taken together, linearity states that for all possible constants c_1 and c_2 and all possible inputs $x_1[n]$ and $x_2[n]$,

$$c_1 x_1[n] + c_2 x_2[n] \Rightarrow c_1 y_1[n] + c_2 y_2[n]$$

or, equivalently, that

$$H\{c_1 x_1[n] + c_2 x_2[n]\} = c_1 \underbrace{H\{x_1[n]\}}_{y_1[n]} + c_2 \underbrace{H\{x_2[n]\}}_{y_2[n]}. \quad (4.54)$$

As in the CT case, *linearity implies that a system operator H commutes with the operations of summing and scaling*. When a system is linear, any sum of scaled inputs applied to the system (Fig. 4.30a) produces the same result as summing and scaling the individual outputs of each input (Fig. 4.30b). A system that does not satisfy Eq. (4.54) is a *nonlinear* system.

The Total Response of a Linear DT System

Just like the CT case explained in Ch. 1, a DT system's output for $n \geq 0$ is the result of two independent causes: the initial conditions of the system (system state) at $n = 0$ and the input $x[n]$ for $n \geq 0$. The *zero-input response* (ZIR) is the output due to initial conditions with the input set to zero. The *zero-state response* (ZSR) is the response due to the input $x[n]$ for $n \geq 0$ with the initial conditions set to zero. When all the appropriate initial conditions are zero, the system is said to be in *zero state*.[†] The system output is zero when the input is zero only if the system is in zero state.

[†]Initial conditions are expressed differently for CT and DT systems. The initial conditions of a CT system at $t = 0$ normally involve the output $y(t)$ and its derivatives at $t = 0^-$. Since derivatives are meaningless for DT signals, however, the appropriate initial conditions of a DT system at $n = 0$ involve the output $y[n]$ at times $n < 0$.

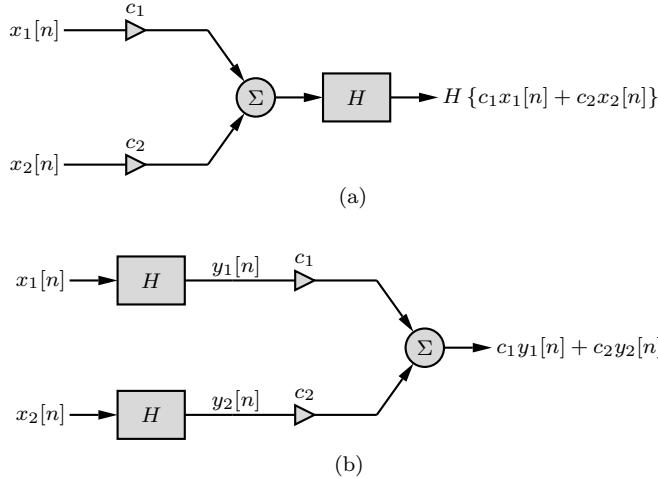


Figure 4.30: Visualizing linearity: (a) summing and scaling precede the DT system and (b) the DT system precedes summing and scaling.

If a system is linear, the total output must be the sum of the ZSR and the ZIR, that is,

$$\text{total response} = \text{zero-state response} + \text{zero-input response} = \text{ZSR} + \text{ZIR}. \quad (4.55)$$

In addition to this property, known as the *decomposition property*, linearity implies that both the zero-input and zero-state components must obey the principle of superposition with respect to each of their respective causes. For example, if we increase the initial condition k -fold, the zero-input component must also increase k -fold. Similarly, if we increase the input k -fold, the zero-state component must also increase k -fold.

Comments on Linear and Nonlinear Systems

The superposition (linearity) property greatly simplifies system analysis. Because of the decomposition property, we can evaluate separately the zero-input and zero-state responses. Moreover, if we express an input $x[n]$ as a sum of simpler functions,

$$x[n] = x_1[n] + x_2[n] + \cdots + x_m[n],$$

then, by virtue of linearity, the (zero-state) response $y[n]$ is given by the sum of the simpler individual responses,

$$y[n] = y_1[n] + y_2[n] + \cdots + y_m[n].$$

As we shall soon see, this apparently trivial observation makes finding the response of an LTID system easy.

As in the case of CT systems, all practical DT systems can exhibit nonlinear behavior. Finite word lengths and quantization effects can both cause an otherwise linear system to become nonlinear. Fortunately, with a little effort, such factors can be reduced or eliminated, thereby leaving system linearity intact.

▷ Example 4.13 (Linearity of Difference Equations)

Consider a general first-order difference equation given by

$$y[n+1] + a_1y[n] = b_0x[n+1] + b_1x[n].$$

Show that this system is linear. Are general-order difference equations also linear?

If the zero-state responses of this system to inputs $x_1[n]$ and $x_2[n]$ are $y_1[n]$ and $y_2[n]$, respectively, then

$$y_1[n+1] + a_1 y_1[n] = b_0 x_1[n+1] + b_1 x_1[n]$$

and

$$y_2[n+1] + a_1 y_2[n] = b_0 x_2[n+1] + b_1 x_2[n].$$

Multiplying these two equations by c_1 and c_2 , respectively, and then adding yields

$$(c_1 y_1[n+1] + c_2 y_2[n+1]) + a_1 (c_1 y_1[n] + c_2 y_2[n]) = \\ b_0 (c_1 x_1[n+1] + c_2 x_2[n+1]) + b_1 (c_1 x_1[n] + c_2 x_2[n]).$$

This equation shows that the response to input $c_1 x_1[n] + c_2 x_2[n]$ is $c_1 y_1[n] + c_2 y_2[n]$ regardless of whether the parameters a_1 , b_0 , and b_1 are constants or functions of n . Hence, the system is linear.

Following the same procedure, we can readily extend our results and show that the general-order difference equation of Eq. (4.47) is also linear. If the coefficients a_k and b_l in that equation are constants, then the DT system is linear and time invariant (LTID). If the coefficients a_k and b_l are functions of time, then the system is linear and time variant.

Example 4.13 ◇

▷ Drill 4.14 (Linear and Time-Invariant Systems)

Show that the accumulator of Eq. (4.38), the first-order backward difference of Eq. (4.41), and the first-order forward difference of Eq. (4.42) are linear and time invariant.

◇

▷ Drill 4.15 (Linear or Time-Invariant Systems)

To illustrate that discrete-time systems that are linear need not be time invariant and vice versa, show that (a) the discrete-time system specified by equation $y[n] = nx[n]$ is linear but not time invariant and (b) the system specified by $y[n] = |x[n]|^2$ is time invariant but not linear.

◇

4.5.3 The Zero-State Response of an LTID System

The zero-state response of a LTID system is analogous to the LTIC case, discussed in Sec. 1.5.3. The result is so important, however, that we repeat the development in its entirety.

Using the DT sampling property of Eq. (4.10), we first represent our input using a sum of scaled and shifted Kronecker delta functions,

$$\underbrace{x[n]}_{\text{input}} = \sum_{m=-\infty}^{\infty} \underbrace{x[m]}_{\substack{\text{scaled} \\ \text{sum}}} \underbrace{\delta[n-m]}_{\substack{\text{shifted} \\ \text{impulses}}}. \quad (4.56)$$

The output $y[n]$ of an LTID system H is thus

$$y[n] = H \{x[n]\} = H \left\{ \sum_{m=-\infty}^{\infty} x[m] \delta[n-m] \right\}. \quad (4.57)$$

Using the property of linearity, we know that summing and scaling commute with the system operation. Thus, Eq. (4.57) becomes

$$y[n] = \sum_{m=-\infty}^{\infty} x[m]H\{\delta[n-m]\}. \quad (4.58)$$

Designating $h[n]$ as the *impulse response* of an LTID system, $h[n] = H\{\delta[n]\}$, the time-invariance property allows us to write Eq. (4.58) as

$$\underbrace{y[n]}_{\text{output}} = \underbrace{\sum_{m=-\infty}^{\infty} x[m]}_{\substack{\text{sum} \\ \text{scaled}}} \underbrace{h[n-m]}_{\substack{\text{shifted} \\ \text{impulse} \\ \text{responses}}}. \quad (4.59)$$

Equation (4.59) is known as the *convolution sum*, expressed symbolically as $x[n] * h[n]$. The next chapter fully treats the time-domain solution of difference equations, including evaluation of the convolution sum.

▷ Example 4.14 (The Response of an LTID System)

Consider an LTID system that has an impulse response $h[n] = 2^{-n}u[n]$. Using the input $x[n] = u[n] - u[n-10]$, determine and sketch the zero-state response of this system over $-10 \leq n \leq 20$.

The signal $x[n] = u[n] - u[n-10]$ is 1 for $0 \leq n \leq 9$ and is 0 elsewhere. Using Eq. (4.56), we can therefore represent the input as

$$x[n] = \sum_{m=0}^9 \delta[n-m].$$

Following Eq. (4.59), the corresponding zero-state response is thus

$$y[n] = \sum_{m=0}^9 h[n-m] = \sum_{m=0}^9 2^{m-n}u[n-m].$$

MATLAB easily computes and plots the output $y[n]$, as shown in Fig. 4.31.

```
01 h = @(n) 2.^(-n).* (n>=0); y = @(n) 0;
02 for m = 0:9,
03     y = @(n) y(n)+h(n-m);
04 end
05 n = -10:20; stem(n,y(n)); xlabel('n'); ylabel('y[n]');
```

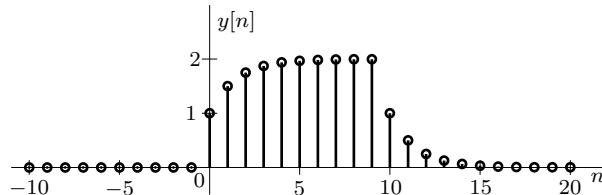


Figure 4.31: LTID system response $y[n] = \sum_{m=0}^9 2^{m-n}u[n-m]$.

Although the input has finite duration, the output, like the impulse response functions $h[n-m]$ that comprise it, has infinite duration.

Example 4.14 ▶

Digital DNA

The *impulse response* $h[n] = H\{\delta[n]\}$ acts as a sort of digital DNA for LTID systems: it contains all the information needed to describe an LTID system's zero-state responses *to the entirety of all possible inputs*. Since it is an external description, the impulse response may not, however, be sufficient to determine a system's zero-input response. Put another way, systems with identical $h[n]$ can differ internally and produce unique zero-input responses.

4.5.4 Causality

A *causal* (also known as a *physical* or *non-anticipative*) *system* is one for which the output at any instant $n = n_0$ depends only on the value of the input $x[n]$ for $n \leq n_0$. In other words, the value of the output at the present instant depends only on the past and present values of the input $x[n]$, not on its future values. In contrast, the response of a noncausal system begins before the input is applied. Clearly, noncausal systems are physically unrealizable.[†]

As we learned in the preceding section, the impulse response $h[n]$ completely characterizes the zero-state behavior of a linear and time-invariant system. As such, we can determine the causality of such systems based purely on $h[n]$. Because $\delta[n]$ starts at $n = 0$, the unit impulse response $h[n]$ of a *causal system* must be zero for $n < 0$, that is,

$$h[n] = 0 \quad \text{for } n < 0. \quad (4.60)$$

Alternately, an LTID system is noncausal if its impulse response $h[n]$ starts before $n = 0$.

Turning our attention to Eq. (4.47), let us determine the conditions that ensure the causality of difference equations. The left-hand side of Eq. (4.47) consists of the output at instants $n + K$, $n + K - 1$, $n + K - 2$, and so on. The right-hand side of Eq. (4.47) consists of the input at instants $n + L$, $n + L - 1$, $n + L - 2$, and so on. For a causal system, the output cannot depend on future input values. Thus, the system described by the difference equation of Eq. (4.47) is causal if and only if $L \leq K$.

A Preferred Form for Difference Equations

Practical systems that are functions of time are causal and require $L \leq K$. Setting $L = K$, which may require setting some coefficients equal to zero, we can express Eq. (4.47) in the more convenient form of

$$\begin{aligned} a_0y[n+K] + a_1y[n+K-1] + \cdots + a_{K-1}y[n+1] + a_Ky[n] = \\ b_0x[n+K] + b_1x[n+K-1] + \cdots + b_{K-1}x[n+1] + b_Kx[n]. \end{aligned}$$

This expression is valid for all values of n . Therefore, the equation is still valid if we replace n by $n - K$ throughout the equation. Such replacement yields the delay form of

$$\begin{aligned} a_0y[n] + a_1y[n-1] + \cdots + a_{K-1}y[n-(K-1)] + a_Ky[n-K] = \\ b_0x[n] + b_1x[n-1] + \cdots + b_{K-1}x[n-(K-1)] + b_Kx[n-K]. \end{aligned}$$

Written in compact form, the general causal difference equation is thus

$$\sum_{k=0}^K a_k y[n-k] = \sum_{l=0}^K b_l x[n-l]. \quad (4.61)$$

[†]This statement is true only when the independent variable is time. See the footnote on page 29.

Without loss of generality, the coefficient a_0 can be assumed to be unity. If $a_0 \neq 1$, we can divide the equation through by a_0 to render the coefficient a_0 unity.[†] Thus, we can also express Eq. (4.61) in an alternate form as

$$y[n] = \sum_{k=1}^K -a_k y[n-k] + \sum_{l=0}^K b_l x[n-l]. \quad (4.62)$$

Observe the recursive nature of this equation. This form of difference equation, favored in the literature and by signal processing practitioners the world over, represents a general K th-order causal LTI discrete-time (causal LTID) system.

If the coefficients $a_k = 0$ ($k = 1, \dots, K$), Eq. (4.62) reduces to

$$y[n] = \sum_{l=0}^K b_l x[n-l].$$

In this case, the output is computed exclusively in terms of the input terms, and the system is nonrecursive. To reiterate, a system represented by Eq. (4.62) is nonrecursive if $a_k = 0$ for $k = 1, 2, 3, \dots, K$. However, even if one of the a_k coefficients is nonzero, the system is recursive.

Figure 4.32 shows a block representation of a general difference equation system based on Eq. (4.62). This particular representation, called *direct form I* (DFI), is just one of many ways to realize the system. Alternate realizations are treated later in Ch. 7.

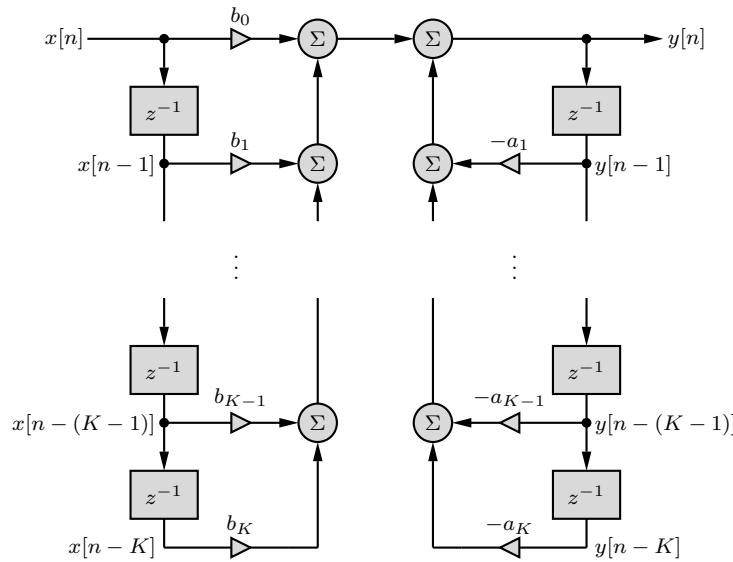


Figure 4.32: Direct form I realization of a general K th-order difference equation.

▷ **Drill 4.16 (Causality of Backward and Forward Difference Systems)**

Show that the backward difference of Eq. (4.41) is causal, whereas the forward difference of Eq. (4.42) is noncausal. Both systems are linear and time invariant, as shown in Drill 4.14.

△

[†]Coefficient normalization is not possible for the case $a_0 = 0$. This case, however, is of little practical interest since it does not occur for causal systems where the output is generated in response to current and past values of input and output.

▷ **Drill 4.17 (Properties of a Time-Reversal System)**

Show that a time-reversal system, specified by equation $y[n] = x[-n]$, is linear but noncausal.

△

4.5.5 Stability

We shall consider here two of several different definitions of stability. *Input-output stability*, also known as *external stability*, requires that, for all n , *bounded inputs* $x[n]$ yield *bounded outputs* $y[n]$. This stability is also known as *bounded-input, bounded-output (BIBO)* stability. Mathematically, BIBO stability requires

$$|y[n]| \leq K_y < \infty \quad \text{if} \quad |x[n]| \leq K_x < \infty, \quad (4.63)$$

where K_x and K_y are some finite constants. For a system to be BIBO stable, the stability condition of Eq. (4.63) must hold for every possible input. If the system violates this condition just for one input, it is unstable.

Internal stability, also known as *asymptotic stability*, is defined in a similar way but in terms of the initial conditions rather than the input. If the response to bounded initial conditions (with or without input) is unbounded, then the system is unstable. If the response decays with time to zero, then the system is asymptotically stable. If the response is neither unbounded nor decays to zero, then the system is *marginally stable*.

Asymptotically stable systems are BIBO stable. However, BIBO stability does not necessarily make a system asymptotically stable. Fortunately, in practice, external and internal stabilities are, with few exceptions, equivalent. Although systematic procedures to test stability are not presented until the next chapter, common sense and reasoning are often sufficient to establish stability. Consider, for example, the cash register system of Ex. 4.7. If the input is $x[n] = u[n]$, then from Eq. (4.35) it follows that the output $y[n] \rightarrow \infty$ as $n \rightarrow \infty$. Clearly, the system is BIBO unstable.

▷ **Drill 4.18 (Digital Differentiator Stability)**

Show that the digital differentiator of Ex. 4.9 is BIBO stable. Is a CT differentiator BIBO stable? Consider how both systems respond to unit step inputs.

△

4.5.6 Memory

If the output of a discrete-time system depends on input values other than the present one, then the system has *memory*. Systems with memory are also called *dynamic systems*. A system without memory is called a *memoryless, instantaneous, or static system*. The response of a memoryless system at any instant n depends at most on the input at the same instant n . For example, $y[n] = \sin(n)$ is an example of an instantaneous system, and $y[n] - y[n-1] = x[n]$ is an example of a system with memory (dynamic). To realize systems with memory, we need to use delay elements. All the examples discussed in Sec. 4.4 are cases of dynamic systems because the outputs at any instant depend in some way on the previous values of the inputs.

▷ **Drill 4.19 (Static and Dynamic Systems)**

Determine whether the following systems are static or dynamic. Which of the systems are causal?

- | | |
|-----------------------|------------------------|
| (a) $y[n+1] = x[n]$ | (b) $y[n] = (n+1)x[n]$ |
| (c) $y[n+1] = x[n+1]$ | (d) $y[n-1] = x[n]$ |

△

4.5.7 Invertibility

A discrete-time system H is invertible if an inverse system H_i exists such that the cascade of H and H_i results in an identity system. An *identity system* is defined as one whose output is identical to the input. In other words, the input of an invertible system can be uniquely determined from the corresponding output. There is no loss of information when a signal is processed by an invertible system, and every input produces a unique output.

The compression $y[n] = x[Mn]$ is not invertible because this operation discards $M - 1$ of every M input samples; these lost input values cannot be recovered by any means or system. Similarly, operations such as $y[n] = \cos(x[n])$ and $y[n] = |x[n]|^2$ are not invertible. In contrast, a cascade of a unit delay with a unit advance results in an identity system because its output is identical to the input. Clearly, the inverse of an ideal unit delay is an ideal unit advance, although this inverse is a noncausal system. Such an example highlights one difficulty of invertibility: practical DT systems that employ delay operations can have inverses that require unrealizable advance operations. Consequently, the definition of invertibility is often relaxed to allow an overall delay. That is, if the cascade of H and H_i produces an output $x[n-m]$ for some $m \geq 0$, then H_i is still considered a suitable and useful (although not ideal) inverse.

4.6 Digital Resampling

Section 4.1 introduces the ideas of expansion and compression for DT signals. Let us now investigate these ideas further. Combined with appropriate filtering, expansion and compression are the backbone operations for multirate systems. Properly used, these operations provide an effective means to *digitally* alter the sampling rate of a DT signal, even by non-integer factors. As we shall see, however, care must be taken to ensure proper results. For now, we investigate resampling from a time-domain perspective. In Ch. 6, we shall revisit these ideas using a frequency-domain perspective.

Block Representations of Operations on the DT Independent Variable

Let us consider the block representations of systems that perform basic time-scaling operations. The downsampling operation, represented as $\downarrow M$, is implemented using a *compressor*, as shown in Fig. 4.33a. To avoid aliasing, such as found in Ex. 4.1, a compressor is often preceded with a decimation filter H_d , as shown in Fig. 4.33b. We shall show later that a decimation filter is typically a digital lowpass anti-aliasing filter with a cutoff frequency of $F_s/2M$, where F_s is the sampling rate of the original signal $x[n]$. The combination of decimation filter and compressor is called a *decimator*.

As shown in Fig. 4.33c, the expansion (upsampling) operation, represented as $\uparrow L$, is implemented using an *expander*. An expander followed with an interpolation filter H_i , as shown in Fig. 4.33d, functions as an *interpolator*. We shall show later that an interpolation filter is preferably an ideal digital lowpass filter that operates at a rate LF_s and has cutoff frequency $F_s/2$.

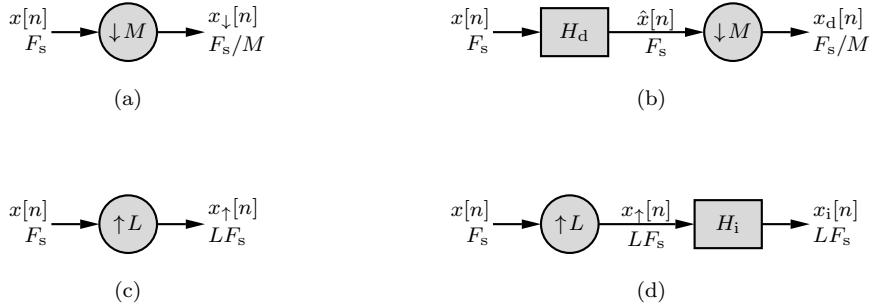


Figure 4.33: Block representations of DT time scaling including sampling rates: (a) compressor, (b) decimator, (c) expander, and (d) interpolator.

Compression and Expansion: Does Cascade Ordering Matter?

Expansion allows us to increase the sampling rate by an integer factor L , and compression allows us to decrease the sampling rate by an integer factor M . On occasion, however, we desire to change the sampling rate by a fractional amount L/M . Intuitively, we suspect that it is possible to achieve fractional sampling rate changes through a cascade combination of expansion and compression. As shown in Fig. 4.34, there are two ways to order this combination.



Figure 4.34: Expander and compressor ordering: (a) $\uparrow L$ precedes $\downarrow M$ and (b) $\downarrow M$ precedes $\uparrow L$.

Unfortunately, the operations of compression and expansion do not, in general, commute. In other words, expansion followed by compression (Fig. 4.34a) does not necessarily lead to the same result as compression followed by expansion (Fig. 4.34b). Let us proceed with an example to investigate these differences.

▷ Example 4.15 (Expansion and Compression Ordering)

Consider the signal $x[n] = \cos(\pi n)$, whose frequency is the highest possible without aliasing. Determine the output when this signal is applied to the following systems:

- (a) An upsample by $L = 4$ operation followed by a downsample by $M = 2$ operation.
- (b) A downsample by $M = 2$ operation followed by an upsample by $L = 4$ operation.
- (c) An upsample by $L = 3$ operation followed by a downsample by $M = 2$ operation.
- (d) A downsample by $M = 2$ operation followed by an upsample by $L = 3$ operation.

As shown in Fig. 4.35, notice that $x[n]$ alternates between the values 1 and -1 as $[\dots, 1, -1, 1, -1, \dots]$.

(a) $\uparrow 4, \downarrow 2$

This case corresponds to the ordering shown in Fig. 4.34a. Upsampling $x[n]$ by $L = 4$ requires that we insert $L - 1 = 3$ zeros between each sample,

$$[\dots, 1, 0, 0, 0, -1, 0, 0, 0, 1, 0, 0, 0, -1, \dots].$$

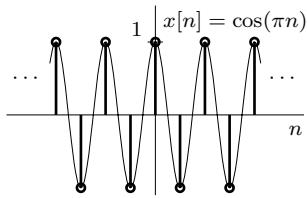


Figure 4.35: A DT signal $x[n] = \cos(\pi n)$.

Next, downsampling by $M = 2$ produces

$$[\dots, 1, 0, -1, 0, 1, 0, -1, 0, 1, \dots].$$

These steps are shown in Fig. 4.36. Intuitively, this result seems sensible as $M/L = 2/4 = 1/2$ suggests a net expansion by factor 2, as produced.

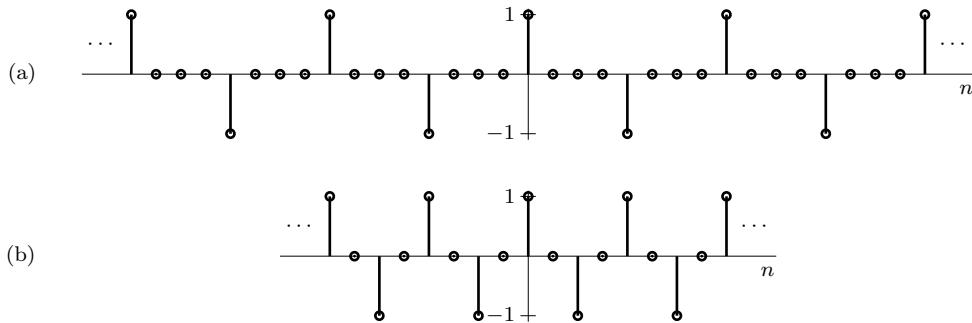


Figure 4.36: Signal $x[n] = \cos(\pi n)$ (a) expanded by $L = 4$ and then (b) compressed by $M = 2$.

(b) $\downarrow 2, \uparrow 4$

This case corresponds to the ordering shown in Fig. 4.34b. Downsampling $x[n]$ by $M = 2$ produces

$$[\dots, 1, 1, 1, \dots].$$

Next, upsampling by $L = 4$ produces

[..., 1, 0, 0, 0, 1, 0, 0, 0, 1, ...].

Even though the expansion and compression factors match those used in part (a), the result is different (Fig. 4.37). In this case, the order of compression and expansion is clearly important to the final result.

(c) $\uparrow 3, \downarrow 2$

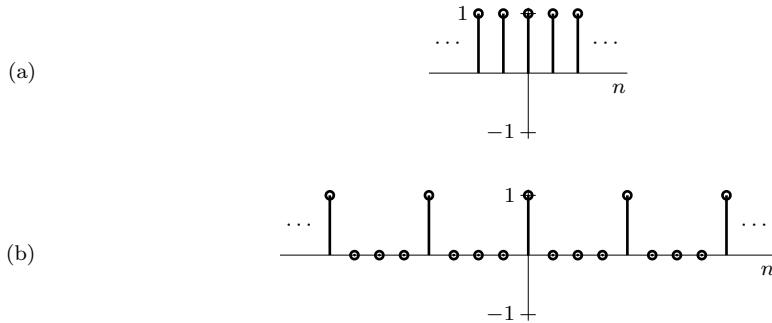
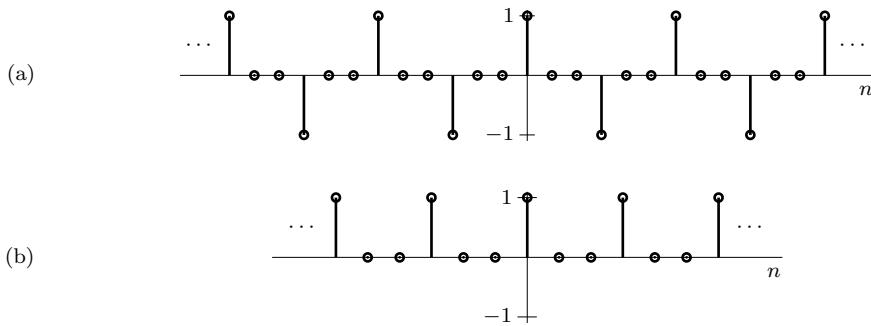
Upsampling $x[n]$ by $L = 3$ requires that we insert $L - 1 = 2$ zeros between each sample,

```
[..., 1, 0, 0, -1, 0, 0, 1, 0, 0, -1, ...]:
```

Next, downsampling by $M = 2$ produces

[..., 1, 0, 0, 1, 0, 0, 1, ...].

These steps are shown in Fig. 4.38. As in part (a), the final result is sensible. A factor $M/L = 2/3$ suggests that values originally located at integer multiples of $M = 2$ (spacing of 2) are subsequently

Figure 4.37: Signal $x[n] = \cos(\pi n)$ (a) compressed by $M = 2$ and then (b) expanded by $L = 4$.Figure 4.38: Signal $x[n] = \cos(\pi n)$ (a) expanded by $L = 3$ and then (b) compressed by $M = 2$.

located at integer multiples of $L = 3$ (spacing of 3). The signal $x[n]$ has values 1 spaced every 2 time units, and the final result has those same values spaced every 3 time units.

(d) $\downarrow 2, \uparrow 3$

Downsampling $x[n]$ by $M = 2$ produces

$$[\dots, 1, 1, 1, \dots].$$

Next, upsampling by $L = 3$ produces

$$[\dots, 1, 0, 0, 1, 0, 0, 1, \dots].$$

This result matches the case of part (c) and suggests, for this case at least, that the order of expansion and compression does not matter.

Discussion

When comparing parts (a) and (b), the order of expansion and compression clearly matters. In other words, the operations of expansion and compression do not, in general, commute. There are cases, as illustrated by parts (c) and (d), where expansion and compression do commute. Fortunately, there is a simple test to determine when the order of operations matters: if L and M are coprime, which is to say they have no common factors other than 1, then the operations of expansion and compression commute [3]. In other words, if L and M are coprime, then Fig. 4.34a and Fig. 4.34b are equivalent. In resampling, we are most often interested in the ratio M/L , so choosing M and L to be coprime is relatively simple.

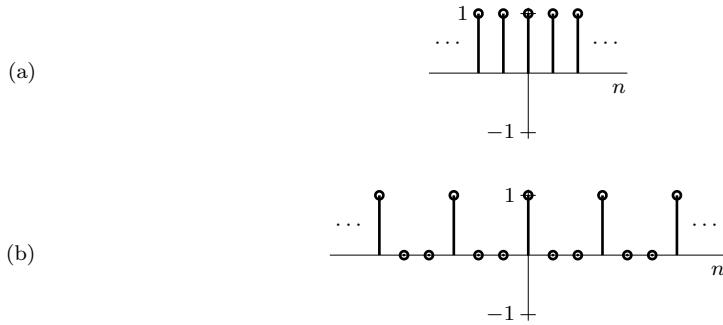


Figure 4.39: Signal $x[n] = \cos(\pi n)$ (a) compressed by $M = 2$ and then (b) expanded by $L = 3$.

Even if operation order does not matter mathematically, it may matter for other reasons. Placing an expander before a compressor, for example, requires that both components operate at faster rates than if placed in the reverse order, a notable disadvantage in many applications (see Prob. 4.6-2). Placing expansion prior to compression, however, can reduce aliasing errors. In the case of interpolation and decimation, as shown in Figs. 4.33b and 4.33d, placing the expander before the compressor allows the interpolation and decimation filters to be combined into a single filter. Just as a miserly person never throws anything away until the last possible moment, expansion before compression ensures that samples are not discarded until the end.

Example 4.15 \triangleleft

▷ **Drill 4.20 (Fractional Sampling Rate Conversion)**

Consider the signal $x[n] = \cos(\Omega n)$, where $\Omega = 4\pi/30$. Determine an expansion factor L and compression factor M that reduce the input's sample rating F_s by 40% to $0.6F_s$. Plot the resulting signal at each stage of an implementation that follows Fig. 4.34a.

\triangleleft

4.7 Summary

Discrete-time signals are sequences of numbers specified only at discrete instants. Some signals are inherently discrete-time in nature, and others become discrete-time following a sampling process. Commonly, a CT signal $x(t)$ is uniformly sampled at instants $t = nT$ to produce the DT signal $x[n] = x(nT)$. Varieties of operations, models, and classifications assist in the characterization and manipulation of DT signals.

Time shifting, reversal, and scaling are three useful operations on the independent DT variable. A signal $x[n]$ delayed (right shifted) by m time units is given by $x[n - m]$. On the other hand, $x[n]$ advanced (left shifted) by m time units is given by $x[n + m]$. The time reversal of signal $x[n]$ is given by $x[-n]$. These operations are the same as those for the continuous-time case. The time-scaling operation, however, is somewhat different because of the discrete nature of the variable n . Unlike the continuous-time case, where time compression results in the same data at a higher speed, time compression in the discrete-time case eliminates part of the data. Consequently, this operation is called downsampling. An anti-aliasing filter followed by downsampling is called decimation. Time expansion, or upsampling, increases the data by inserting zeros between samples. A filter often follows the upsampling process to perform interpolation, a process that replaces the inserted zeros with more appropriate values. Compression ($\downarrow M$) and expansion ($\uparrow L$) are constrained to integer

factors M and L .

The DT unit step, unit impulse, and exponential functions are similar to their CT counterparts, with a few noteworthy differences. The DT unit step $u[n]$ is basically identical to the CT unit step $u(t)$, except that $u[0] = 1$ and $u(0) = 1/2$. Unlike the Dirac delta function $\delta(t)$, which is not physically realizable, the DT unit impulse $\delta[n]$ is easily generated in practice. Continuous-time exponentials are usually represented using the natural base as e^{st} , where s is a complex constant. Discrete-time exponentials, by contrast, are more appropriately represented in the form z^n , where z is also a complex constant. The function z^n grows exponentially with n if $|z| > 1$ (z outside the unit circle), decays exponentially if $|z| < 1$ (z within the unit circle), and has constant envelope if $|z| = 1$ (z lies on the unit circle). When a CT exponential e^{st} is sampled uniformly every T seconds, the resulting DT exponential z^n requires $z = e^{sT}$; the many-to-one nature of this mapping is reflective of the aliasing phenomenon.

Sampling a continuous-time sinusoid $\cos(\omega t + \theta)$ every T seconds results in a discrete-time sinusoid $\cos(\Omega n + \theta)$, where $\Omega = \omega T$. Discrete-time sinusoids whose frequencies Ω differ by an integer multiple of 2π are identical. Consequently, a DT sinusoid with $|\Omega| > \pi$ has a lower apparent frequency Ω_a in the fundamental band of frequencies from $-\pi$ to π . The highest rate of oscillation in a discrete-time sinusoid occurs when its apparent frequency is π . The nonuniqueness of DT sinusoids, where different frequencies Ω produce the same waveforms, is again a consequence of aliasing and has profound implications in the study of discrete-time signals and systems.

A number of signal classifications, such as those involving causality, realness, and symmetry, are the same for discrete-time signals as for continuous-time signals. Other signal classifications are more substantially different. Discrete-time periodicity requires an integer-valued period, for example, and is more restrictive than the continuous-time case. To illustrate further, the CT sinusoid $\cos(\omega t + \theta)$ is always periodic, yet the DT sinusoid $\cos(\Omega n + \theta)$ is periodic only if Ω is a rational multiple of π . Discrete-time energy and power, while conceptually similar to the CT cases, utilize sums rather than integrals.

A system whose inputs and outputs are discrete-time signals is a discrete-time system. Many DT systems are characterized by difference equations, so the operations of summation, scalar multiplication, and delay are fundamental to most DT system realizations. More basically, an input $x[n]$ applied to a system H to produce output $y[n]$ is represented as $y[n] = H\{x[n]\}$. Discrete-time systems may be used to process discrete-time signals, or if using appropriate interfaces at the input and output, discrete-time systems may be used to process continuous-time signals. In the latter case, a continuous-time input signal is first converted into a discrete-time signal through sampling. The resulting discrete-time signal is then processed by a discrete-time system to yield a discrete-time output, which is then converted into a continuous-time output. Assuming a sufficiently high sampling rate, any CT system modeled by a general constant-coefficient linear differential equation can be implemented as a DT system using a constant-coefficient linear difference equation. Given the many advantages of digital signal processing, many CT systems are best implemented digitally.

As with CT systems, time invariance requires that the parameters of a DT system do not change with time. To be linear, the principle of superposition must hold for a DT system. Taken together, the properties of linearity and time invariance allow the response of a DT system to be conveniently represented using the convolution sum, which only requires knowledge of the system's impulse response $h[n]$ and input $x[n]$. Other DT system properties, such as causality, stability, memory, and invertibility, follow the same spirit as their CT counterparts.

The sampling rate of a DT signal can be increased through expansion and can be reduced through compression. Expansion by factor L inserts $L - 1$ zeros between each sample, and compression by factor M removes $M - 1$ out of every M samples. A decimator is just a compressor that is preceded by a digital lowpass anti-aliasing, or decimation, filter. An interpolator is just an expander that is followed by a digital lowpass smoothing, or interpolation, filter. Fractional sampling rate alteration is accomplished by cascading an expander with a compressor. The same result is also obtained by a compressor followed by an expander, as long as the expansion factor L and the compression factor M are coprime. If factors M and L are not coprime, $\downarrow M$ and $\uparrow L$ do not commute.

References

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
2. Cadzow, J. A., *Discrete-Time Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1973.
3. Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

Problems

4.1-1 For the signal shown in Fig. P4.1-1, sketch the signals

- | | |
|------------------------|----------------------|
| (a) $x[-n]$ | (b) $x[n+6]$ |
| (c) $x[n-6]$ | (d) $x[3-n]$ |
| (e) $x[3n]$ | (f) $x[\frac{n}{3}]$ |
| (g) $x[\frac{n+1}{4}]$ | |

Parts (f) and (g) require simple expansion (inserted zeros, no interpolation).

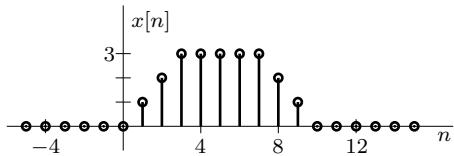


Figure P4.1-1

4.1-2 Repeat Prob. 4.1-1 using the signal in Fig. P4.1-2.

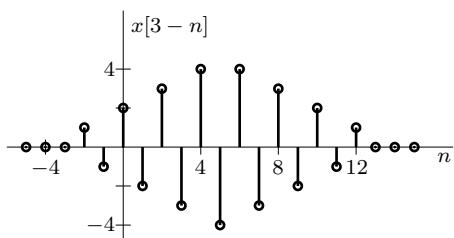


Figure P4.1-2

4.1-3 What values θ cause the DT sinusoid $\cos(\Omega n + \theta)$ to be a simple shifted version of $\cos(\Omega n)$?

4.2-1 Example 4.2 describes the signal in Fig. 4.9 using ramp, step, and impulse functions. Give two more expressions to describe this signal using ramp, step, and impulse functions. Use no more than one impulse.

4.2-2 Sketch the signals

- | | |
|------------------------------|--|
| (a) $u[n-2] - u[n-6]$ | |
| (b) $n(u[n] - u[n-7])$ | |
| (c) $(n-2)(u[n-2] - u[n-6])$ | |
| (d) $(8-n)(u[n-6] - u[n-9])$ | |

$$(e) (n-2)(u[n-2] - u[n-6]) + (8-n)(u[n-6] - u[n-9])$$

4.2-3 Show that

- | | |
|---|--|
| (a) $\delta[n] + \delta[n-1] = u[n] - u[n-2]$ | |
| (b) $2^{n-1} \sin\left(\frac{\pi n}{3}\right) u[n] =$ | $-0.5(2)^n \sin\left(\frac{5\pi n}{3}\right) (u[n] - \delta[n])$ |
| (c) $2^{n-1} \sin\left(\frac{\pi n}{3}\right) u[n] =$ | $-0.5(-2)^n \sin\left(\frac{2\pi n}{3}\right) u[n-1]$ |
| (d) $n(n-1)\gamma^n u[n] = n(n-1)\gamma^n u[n-2]$ | |
| (e) $(u[n] + (-1)^n u[n]) \sin\left(\frac{\pi n}{2}\right) = 0$ | |
| (f) $(u[n] + (-1)^{n+1} u[n]) \cos\left(\frac{\pi n}{2}\right) = 0$ | |

4.2-4 Describe each of the signals in Fig. P4.2-4 by a single expression valid for all n . Using only ramp and step functions, give at least two different expressions to describe each signal.

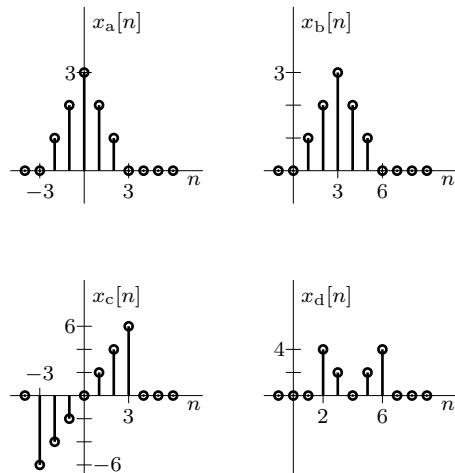


Figure P4.2-4

4.2-5 Using the signal $x_c[n]$ from Fig. 4.2-4, define $y[3-n] = x_c[n]$ and sketch

- | | |
|-----------------------|-----------------------|
| (a) $y[-n]$ | (b) $y[2-n]$ |
| (c) $y[n]\delta[n-2]$ | (d) $y[n+4]\delta[n]$ |
| (e) $y[n]u[n-2]$ | (f) $y[n]u[4-n]$ |

4.2-6 The following signals are in the form $e^{\lambda n}$. Express them in the form γ^n .

- | | |
|-------------------|------------------|
| (a) $e^{-0.5n}$ | (b) $e^{0.5n}$ |
| (c) $e^{-j\pi n}$ | (d) $e^{j\pi n}$ |

In each case, show the locations of λ and γ in the complex plane. Verify that the exponential is growing if γ lies outside the unit circle (λ in the RHP), is decaying if γ lies within the unit circle (λ in the LHP), and has a constant envelope if γ lies on the unit circle (λ on the imaginary axis).

- 4.2-7** Repeat Prob. 4.2-6 for the exponentials

$$\begin{array}{ll} \text{(a)} & e^{-(1+j\pi)n} \\ \text{(b)} & e^{-(1-j\pi)n} \\ \text{(c)} & e^{(1+j\pi)n} \\ \text{(d)} & e^{(1-j\pi)n} \\ \text{(e)} & e^{-(1+j\frac{\pi}{3})n} \\ \text{(f)} & e^{(1-j\frac{\pi}{3})n} \end{array}$$

- 4.2-8** Show that $\cos(0.6\pi n + \frac{\pi}{6}) + \sqrt{3} \cos(1.4\pi n + \frac{\pi}{3}) = 2 \cos(0.6\pi n - \frac{\pi}{6})$.

- 4.2-9** Express the following signals in terms of apparent frequencies:

$$\begin{array}{l} \text{(a)} \cos(0.8\pi n + \theta) \\ \text{(b)} \sin(1.2\pi n + \theta) \\ \text{(c)} \cos(6.9n + \theta) \\ \text{(d)} \cos(2.8\pi n + \theta) + 2 \sin(3.7\pi n + \phi) \\ \text{(e)} \text{sinc}(\pi n/2) \\ \text{(f)} \text{sinc}(3\pi n/2) \\ \text{(g)} \text{sinc}(2\pi n) \end{array}$$

- 4.2-10** Consider a sinusoid $\cos(k\pi n + \theta)$, where k is an odd integer. In this case, the frequency $\Omega = k\pi$ is on the border between the shaded and unshaded regions of Fig. 4.15b, and the apparent frequency can be assigned from either region.

- (a) Show that Eq. (4.17) computes the apparent frequency of this sinusoid as $\Omega_a = -\pi$, which corresponds to the shaded region of Fig. 4.15b. Express the sinusoid in terms of this apparent frequency.
- (b) Assuming that $\Omega = k\pi$ is assigned to the unshaded region, determine the apparent frequency Ω_a , and express the sinusoid in terms of this apparent frequency.
- (c) Letting $\Omega = 11\pi$ and $\theta = \pi/3$, evaluate the two apparent sinusoids from parts (a) and (b). Comment on the results.

- 4.2-11** A continuous-time sinusoid $\cos(\omega_0 t)$ is sampled at a rate $F_s = 100$ Hz. The sampled signal is found to be $\cos(0.6\pi n)$. If there is more than one possible value for ω_0 , find the general expression for ω_0 , and determine the three smallest values of $|\omega_0|$.

- 4.2-12** Samples of a continuous-time sinusoid $\cos(100\pi t)$ are found to be $\cos(\pi n)$. Find the sampling frequency F_s . Explain whether there is only one possible value for F_s . If there is more than one possible value, find the general expression for the sampling frequency, and determine the three largest possible values.

- 4.2-13** Express the following exponentials in the form $e^{j(\Omega n + \theta)}$, where $-\pi \leq \Omega < \pi$:

$$\begin{array}{ll} \text{(a)} & e^{j(8.2\pi n + \theta)} \\ \text{(b)} & e^{j4\pi n} \\ \text{(c)} & e^{-j1.95n} \\ \text{(d)} & e^{-j10.7\pi n} \end{array}$$

Repeat the problem if Ω is required to be in the range $0 \leq \Omega < 2\pi$.

- 4.2-14** A discrete-time processor uses a sampling interval $T = 0.5 \mu\text{s}$. What is the highest frequency of a signal that can be processed with this processor without aliasing? If a signal of frequency 2 MHz is sampled by this processor, what is the (aliased) frequency of the resulting sampled signal?

- 4.2-15** Continuous-time sinusoids $10 \cos(11\pi t + \frac{\pi}{6})$ and $5 \cos(29\pi t - \frac{\pi}{6})$ are sampled using a sampling interval of $T = 0.1$ s. Express the resulting discrete-time sinusoids in terms of their apparent frequencies.

- 4.2-16** Consider a signal $x(t) = 10 \cos(2000\pi t) + \sqrt{2} \sin(3000\pi t) + 2 \cos(5000\pi t + \frac{\pi}{4})$.

- (a) Assuming that $x(t)$ is sampled at a rate of 4000 Hz, find the resulting sampled signal $x[n]$, expressed in terms of apparent frequencies. Does this sampling rate cause any aliasing? Explain.
- (b) Determine the maximum sampling interval T that can be used to sample the signal in part (a) without aliasing.

4.2-17 A sampler with sampling interval $T = 0.1 \text{ ms}$ (10^{-4} s) samples continuous-time sinusoids of the following frequencies:

- | | |
|----------------------------|---------------------------|
| (a) $f = 12.5 \text{ kHz}$ | (b) $f = 8500 \text{ Hz}$ |
| (c) $f = 10 \text{ kHz}$ | (d) $f = 1500 \text{ Hz}$ |
| (e) $f = 32 \text{ kHz}$ | (f) $f = 9600 \text{ Hz}$ |

Determine the apparent frequency of the sampled signal in each case.

4.3-1 Find and sketch the odd and even components of

- | | |
|---|---------------------------------|
| (a) $u[n]$ | (b) $nu[n]$ |
| (c) $\sin(\frac{\pi n}{4})$ | (d) $\cos(\frac{\pi n}{4})$ |
| (e) $\sin(\frac{\pi n}{4})u[n]$ | (f) $\cos(\frac{\pi n}{4})u[n]$ |
| (g) $\cos(\frac{\pi n}{4} + \frac{\pi}{4})$ | |
| (h) $u[n+5] - u[n-5]$ | |

4.3-2 Let $x_e[n]$ and $x_o[n]$ be the even and odd portions of a causal signal $x[n] = x[n]u[n]$.

- (a) Show that $E_{x_e} = E_{x_o} = 0.5E_x$.
- (b) Show that the cross-energy of x_e and x_o is zero, that is

$$\sum_{n=-\infty}^{\infty} x_e[n]x_o^*[n] = 0.$$

4.3-3 State with reason(s) whether the following signals are periodic:

- | | |
|----------------------------------|--------------------------------|
| (a) $\cos(0.5\pi n + 0.2)$ | (b) $e^{j\frac{22}{7}n}$ |
| (c) $\sin(0.5n + \frac{\pi}{3})$ | (d) $e^{-(1+j\frac{\pi}{3})n}$ |
| (e) $\cos(\sqrt{2}\pi n + 1.2)$ | (f) $-6e^{j\frac{\pi}{7}n}$ |

If periodic, determine the period.

4.3-4 Prove that a discrete-time exponential $e^{j\Omega n}$ is periodic only if $\frac{\Omega}{2\pi}$ is a rational number.

4.3-5 Repeat Prob. 4.3-3 for the signals

- (a) $\cos(0.6\pi n + 0.3) + 3 \sin(0.5\pi n + 0.4)$
- (b) $\cos(1.6\pi n + 0.3) + 3 \sin(1.5\pi n + 0.4) + 8 \cos(1.8\pi n - \frac{\pi}{3})$
- (c) $\cos(0.7\pi n + 0.3) + 3 \sin(0.5n + 0.4)$

4.3-6 Find the energies of the signals depicted in Fig. P4.2-4.

4.3-7 If $x[n] = n(u[n-1] - u[n-5])$, sketch signals $x[n]$, $-x[n]$, $x[-n]$, $x[n+1]$, and $3x[n]$, and find their energies.

4.3-8 If the energy of a signal $x[n]$ is E_x , find the energies of the signals $-x[n]$, $x[-n]$, $x[n-m]$, and $cx[n]$.

4.3-9 The periodic replication of a DT energy signal $x[n]$ produces a power signal $\tilde{x}[n]$, given as

$$\tilde{x}[n] = \sum_{m=-\infty}^{\infty} x[n - mN].$$

Using a period of $N = 6$, periodically replicate the signals shown in Fig. P4.2-4. For each case, sketch the signal, and determine its power.

4.3-10 Find the powers of the signals depicted in Fig. P4.3-10.

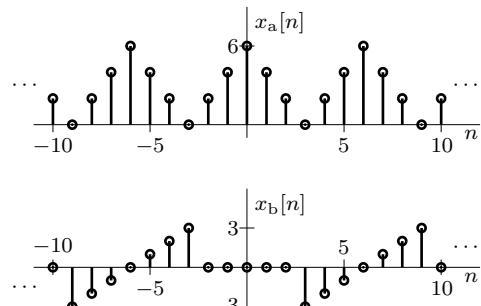


Figure P4.3-10

4.3-11 Sketch the signals $(1)^n$, $(-1)^n$, $u[n]$, and $(-1)^n u[n]$, and find their powers.

4.3-12 Designating the power of a periodic signal $x[n]$ as P_x , find the powers of $-x[n]$, $x[-n]$, $x[n-m]$, and $cx[n]$.

4.3-13 Determine whether the following signals are energy signals, power signals, or neither. In each case, compute both the energy and the power.

- | | |
|------------------------|-------------------------------|
| (a) $\delta[n]$ | (b) $\frac{1}{\sqrt{n}} u[n]$ |
| (c) $\cos(0 \times n)$ | (d) $\cos(2\pi n)$ |

4.3-14 Determine the energy and power for the following signals:

- (a) $x_a[n] = c(0.3)^n$
- (b) $x_b[n] = c(0.3)^{|n|}$
- (c) $x_c[n] = nu[n] - nu[n - 5]$
- (d) $x_d[n] = nu[n]$
- (e) $x_e[n] = [1 - (0.3)^n]u[n]$
- (f) $x_f[n] = n(0.5)^n$

State whether any of the signals are neither an energy signal nor a power signal.

- 4.3-15** Determine the size (power and energy) of the following signals:

- (a) $3 \cos(\frac{\pi}{6}n + \theta)$
- (b) $2 \cos(0.3n + \theta)$

- 4.3-16** Show that the power of signal $X e^{j\frac{2\pi}{N_0}n}$ is $|X|^2$. Using this result, show that the power of signal

$$x[n] = \sum_{k=0}^{N_0-1} X_k e^{jk\Omega_0 n}$$

is

$$P_x = \sum_{k=0}^{N_0-1} |X_k|^2, \quad \text{where } \Omega_0 = \frac{2\pi}{N_0}.$$

Hint: Make use of the following equality:

$$\sum_{k=0}^{N_0-1} e^{j(k-m)\Omega_0 n} = \begin{cases} N_0 & k = m \\ 0 & \text{otherwise} \end{cases}.$$

- 4.4-1** Let $p[n]$ be the population of a certain country at the beginning of the n th year. The birth and death rates of the population during any year are 3.3% and 1.3%, respectively. If $i[n]$ is the total number of immigrants entering the country during the n th year, write the difference equation relating $p[n+1]$, $p[n]$, and $i[n]$, assuming that the immigrants enter the country throughout the year at a uniform rate.

- 4.4-2** For the three digital integrators in Ex. 4.10, derive the outputs $y[n]$ and $y(t)$ for the analog input $x(t) = u(t)$. Use $u(0) = 1$, and let the D/C converter be a simple zero-order hold. How do the outputs compare with the output of a continuous-time integrator?

- 4.4-3** (a) In the same manner as used to obtain Eq. (4.51), derive an approximation of $d^3y(t)/dt^3$ in terms of finite differences. Use Eq. (4.52) to verify but not obtain your result.

- (b) Find a difference equation, including initial conditions, that approximates the behavior of the second-order difference equation

$$\frac{d^2y(t)}{dt^2} + 3 \frac{dy(t)}{dt} + 2y(t) = x(t),$$

where $y(0) = 0$ and $\dot{y}(0) = 3$. Take the sampling interval as $T = 0.05$.

- 4.4-4** A variable, such as a stock market average, may fluctuate (up and down) daily, masking its long-term trend. We can discern such long-term trends by smoothing or averaging the past N values of the variable. For the stock market average, we may consider a 5-day moving average $y[n]$ to be the mean of the past five days' market closing values $x[n]$, $x[n - 1]$, \dots , $x[n - 4]$.

- (a) Write the nonrecursive representation of this system.
- (b) Determine a block realization of this system.

- 4.4-5** (a) Find the recursive representation of the 5-day moving-average system described in Prob. 4.4-4.
- (b) Determine a block realization of this recursive system.

- 4.4-6** For the resistive ladder shown in Fig. P4.4-6, the voltage at the n th node is $v[n]$ ($n = 0, 1, 2, \dots, N$). By inspection, we see that $v[0] = V$ and $v[N] = 0$. Show that the voltage at all other nodes satisfies the second-order difference equation

$$v[n+1] - Av[n] + v[n-1] = 0 \quad A = 2 + \frac{1}{a}.$$

- 4.4-7** Consider the classical problem of supply and demand of some commodity. At time n , let

$$d[n] = \text{demand},$$

$$s[n] = \text{supply},$$

$$\text{and } p[n] = \text{prevailing price}.$$

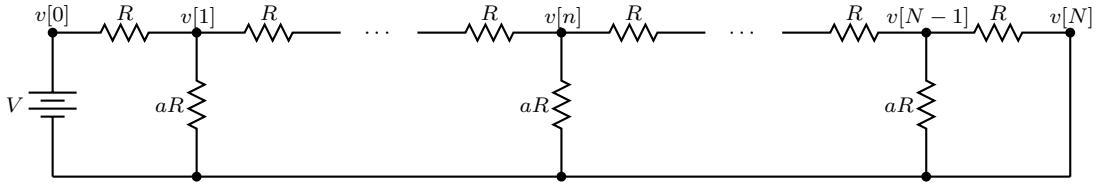


Figure P4.4-6

The demand $d[n]$, which is a function of the price $p[n]$, is given by

$$d[n] = d_0 - ap[n].$$

Because the production of a commodity (corn, for instance) is not instantaneous but takes time to produce, the producer bases production on the price during the previous period. Thus,

$$s[n] = s_0 + bp[n-1],$$

where s_0 is usually a negative constant. Finally, the prevailing price must be such that supply equals demand during any period.

- (a) Construct the model (equation) for the prevailing price during the n th period.
- (b) Determine a block realization of this system.

4.4-8 Consider the problem of modeling national income. At time n , let

$$\begin{aligned} y[n] &= \text{national income}, \\ c[n] &= \text{consumer expenditure}, \\ p[n] &= \text{induced private investment} \\ &\quad (\text{capital equipment to} \\ &\quad \text{increase production}), \\ \text{and } g[n] &= \text{government expenditure}. \end{aligned}$$

In this model, the consumer expenditure during any period is directly proportional to the national income during the previous period. On the other hand, the induced private investment is directly proportional to $c[n] - c[n-1]$, the increase in the consumer spending over the previous period. Finally, the national income is the sum of the consumer spending, the induced private investment, and the government expenditure.

- (a) Construct the model (equation) for the prevailing price during the n th period.

- (b) Determine a block realization of this system.

4.5-1 With input $x[n]$ and output $y[n]$, determine which of the following systems are linear and which are nonlinear:

- (a) $y[n] + 2y[n-1] = x^2[n]$
- (b) $y[n] + 3ny[n-1] = n^2x[n]$
- (c) $y^2[n] + 2y[n-1] = x[n]$
- (d) $y[n] = y[n-1] + \frac{x[n]}{y[n-1]}$
- (e) $3y[n] + 2 = x[n]$
- (f) $y[n] + 2y[n-1] = x[n]x[n-1]$
- (g) $y[n] = x[2-n]$
- (h) $y[n] = \sum_{n=-\infty}^n x[n]$

4.5-2 (a) Show that a system described by the input-output equation $y[n] = x^*[n]$ satisfies the additivity property but not the homogeneity property.

- (b) Show that a system described by the input-output equation $y[n] = x^2[n]/x[n-1]$ satisfies the homogeneity property but not the additivity property.

4.5-3 Determine the unit impulse response of the systems described by

- (a) $y[n] = x[n-1]$ (ideal unit delay)
- (b) $y[n] = x[n+1]$ (ideal unit advance)
- (c) $y[n] = n^2x[n-1]$
- (d) $y[n] = x[-n]$
- (e) $y[n] = x[n/L]$ (L integer)
- (f) $y[n] = x[Mn]$ (M integer)
- (g) $y[n] = \sum_{n=-5}^5 x[n]$
- (h) $y[n] = x[n]x[n-1]$

4.5-4 State with reason(s) whether the following systems are (1) time invariant or time variant, (2) linear or nonlinear, (3) causal or noncausal, (4) stable or unstable, (5) static or dynamic, and (6) invertible or noninvertible.

- (a) $y[n] = x[n - 1]$ (ideal unit delay)
- (b) $y[n] = x[n + 1]$ (ideal unit advance)
- (c) $y[n] = n^2 x[n - 1]$
- (d) $y[n] = nx[n - 1]$
- (e) $y[n] = nx^2[n]$
- (f) $y[n] = x^2[n]$
- (g) $y[n] = x[2 - n]$
- (h) $y[n] = x[n/L]$ (L integer)
- (i) $y[n] = x[Mn]$ (M integer)
- (j) $y[n] = \sum_{n=-5}^5 x[n]$
- (k) $y[n] = x[n]x[n - 1]$

4.5-5 Show that a system specified by the equation $y[n] - y[n - 1] = x[n]$ is BIBO unstable.

4.5-6 Explain with reason(s) whether the following systems are (1) linear or nonlinear, (2) time invariant or time variant, and (3) causal or noncausal.

- (a) an ideal backward difference system defined by the input-output relationship

$$y[n] = x[n] - x[n - 1]$$

- (b) an ideal forward difference system defined by the input-output relationship

$$y[n] = x[n + 1] - x[n]$$

- (c) a moving-average system defined by the input-output relationship

$$y[n] = \frac{1}{K_2 - K_1 + 1} \sum_{k=K_1}^{K_2} x[n - k]$$

- (d) an accumulator defined by the input-output relationship

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]$$

4.6-1 Consider the signal $x[n] = \cos(\Omega n)$, where $\Omega = 4\pi/31$.

- (a) Determine an expansion factor L and compression factor M that result in a signal that reduces the sampling rate F_s of $x[n]$ by 40% to $0.6F_s$. Choose M and L to be coprime.
- (b) Using M and L from part (a), plot the resulting signal at each stage of an implementation that follows Fig. 4.34a.
- (c) Using M and L from part (a), plot the resulting signal at each stage of an implementation that follows Fig. 4.34b.

4.6-2 Suppose that a signal $x[n]$ is sampled at a rate $F_s = 1$ kHz.

- (a) Determine an expansion factor L and compression factor M that result in a signal that reduces the sampling rate by 60% to $0.4F_s$. Choose L and M to be coprime.
- (b) Show that ordering the expander before the compressor requires each component to operate at faster sampling rates than if the compressor is ordered before the expander.

Chapter 5

Time-Domain Analysis of Discrete-Time Systems

In this chapter, we develop time-domain methods for finding the response of LTID systems. Although transform-domain techniques found in later chapters are often more powerful for systems analysis, time-domain methods remain the most common form of systems implementation. To begin our study, we consider hand calculations using iteration (or recursion). Although the technique is extremely simple and applies to all discrete-time systems, it generally does not provide a closed-form solution. Closed-form expressions for the zero-input response (ZIR) and zero-state response (ZSR) are possible using a system's characteristic equation and impulse response, respectively. The zero-state response further requires discrete-time convolution, and several views of this important technique are presented. The chapter concludes with a discussion of stability, insights into system behavior, and a peek at the classical solution to linear difference equations.

5.1 Iterative Solutions to Difference Equations

Let us consider the solution of a causal LTID system represented by a general K th-order difference equation. We shall consider the delay form, as given by Eq. (4.62) and rewritten as

$$y[n] = -a_1y[n-1] - \cdots - a_{K-1}y[n-(K-1)] - a_Ky[n-K] + b_0x[n] + b_1x[n-1] + \cdots + b_{K-1}x[n-(K-1)] + b_Kx[n-K]. \quad (5.1)$$

Thus, $y[n]$, the output at the n th instant, is computed from $2K+1$ pieces of information. These are the past K output values $y[n-1], y[n-2], \dots, y[n-K]$, the past K input values $x[n-1], x[n-2], \dots, x[n-K]$, and the present input value $x[n]$. We can readily compute iteratively or recursively the output values $y[0], y[1], y[2], y[3], \dots$. For instance, to find $y[0]$ we set $n=0$ in Eq. (5.1). The left-hand side is $y[0]$, and the right-hand side contains terms $y[-1], y[-2], \dots, y[-K]$ and the inputs $x[0], x[-1], x[-2], \dots, x[-K]$. To begin this recursion, therefore, we must know the K initial conditions $y[-1], y[-2], \dots, y[-K]$. Knowing these conditions and the input $x[n]$, we can iteratively find the values $y[0], y[1], y[2]$, and so on. As we shall see in the following examples, iterative techniques are well suited to both hand and computer calculation.

▷ Example 5.1 (Iterative Solution to a First-Order Difference Equation)

Using initial condition $y[-1] = 6$ and causal input $x[n] = \cos(n\pi/2)u[n]$, iteratively solve the first-order system given by

$$y[n] - 0.5y[n-1] = x[n] - 0.5x[n-1].$$

Following the form of Eq. (5.1), the system is expressed as

$$y[n] = 0.5y[n-1] + x[n] - 0.5x[n-1]. \quad (5.2)$$

To begin, we set $n = 0$ in this equation and use $y[-1] = 6$, $x[0] = \cos(0) = 1$, and $x[-1] = \cos(-\pi/2)u[-1] = 0$ to obtain

$$\begin{aligned} y[0] &= 0.5y[-1] + x[0] - 0.5x[-1] \\ &= 0.5(6) + 1 - 0.5(0) = 4. \end{aligned}$$

Next, we set $n = 1$ in Eq. (5.2) and use the values $y[0] = 4$ (computed in the $n = 0$ step), $x[1] = \cos(\pi/2) = 0$, and $x[0] = 1$ to obtain

$$y[1] = 0.5(4) + 0 - 0.5(1) = 1.5.$$

Continuing, we set $n = 2$ in Eq. (5.2) and use the values $y[1] = 1.5$ (computed in the previous step), $x[2] = \cos(2\pi/2) = -1$, and $x[1] = 0$ to obtain

$$y[2] = 0.5(1.5) - 1 - 0.5(0) = -0.25.$$

Proceeding in this way iteratively, we obtain

$$\begin{aligned} y[3] &= 0.5(-0.25) + 0 - 0.5(-1) = 0.3750 \\ y[4] &= 0.5(0.3750) + 1 - 0.5(0) = 1.1875 \\ y[5] &= 0.5(1.1875) + 0 - 0.5(1) = 0.0938 \\ y[6] &= 0.5(0.0938) - 1 - 0.5(0) = -0.9531 \\ &\vdots \end{aligned}$$

Computer programs, such as MATLAB, are well adapted to this repetitive process and produce identical results.

```
01 n = (-1:12); x = cos(pi*n/2).* (n>=0);
02 y = zeros(size(n)); y(n== -1) = 6;
03 for ind = find(n>=0),
04     y(ind) = 0.5*y(ind-1)+x(ind)-0.5*x(ind-1);
05 end
06 stem(n,y)
```

In this case, notice that the variable n does not comply with MATLAB's indexing requirements, so the variable `ind` serves to index the vectors `x` and `y`. The output $y[n]$ is depicted in Fig. 5.1a.

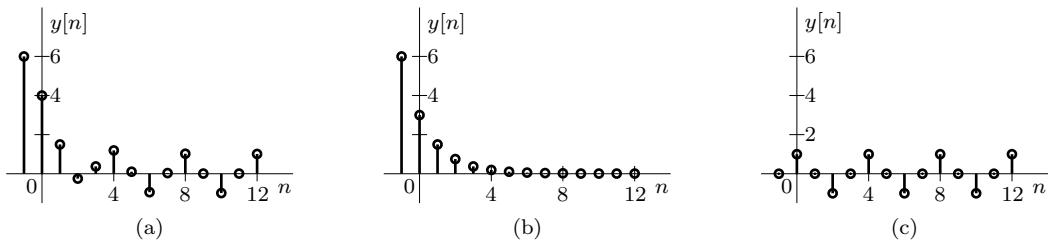


Figure 5.1: Iterative solution to $y[n] - 0.5y[n-1] = x[n] - 0.5x[n-1]$ using $y[-1] = 6$ and $x[n] = \cos(n\pi/2)u[n]$: (a) total response (b) ZIR, and (c) ZSR.

The iterative procedure is also suitable to calculate the zero-input and zero-state responses of the system. The ZIR, shown in Fig. 5.1b, is obtained by changing line 01 to force a zero input.

```
01 n = (-1:12); x = 0.*n;
```

Similarly, the ZSR, shown in Fig. 5.1c, is obtained by changing line 02 to force zero state.

```
02 y = zeros(size(n));
```

The total output (Fig. 5.1a) is just the sum of the ZIR (Fig. 5.1b) and the ZSR (Fig. 5.1c) (see Eq. (4.55)).

Example 5.1 \triangleleft

The iterative method can be applied to a difference equation in delay or advance form. In Ex. 5.1, we considered the former. Let us now apply the iterative method to a second-order equation in advance form.

▷ **Example 5.2 (Iterative Solution to a Second-Order Difference Equation)**

Using causal input $x[n] = nu[n]$ and initial conditions $y[-1] = 2$ and $y[-2] = 1$, iteratively solve the second-order system given by

$$y[n+2] - y[n+1] + 0.24y[n] = x[n+2] - 2x[n+1].$$

Again following Eq. (5.1), the system is expressed as

$$y[n+2] = y[n+1] - 0.24y[n] + x[n+2] - 2x[n+1]. \quad (5.3)$$

Setting $n = -2$ and substituting $y[-1] = 2$, $y[-2] = 1$, $x[0] = 0$, and $x[-1] = 0$, we obtain

$$y[0] = 2 - 0.24(1) + 0 - 0 = 1.76.$$

Setting $n = -1$ in Eq. (5.3) and substituting $y[0] = 1.76$, $y[-1] = 2$, $x[1] = 1$, and $x[0] = 0$, we obtain

$$y[1] = 1.76 - 0.24(2) + 1 - 0 = 2.28.$$

Continuing to iterate, we obtain

$$\begin{aligned} y[2] &= 2.28 - 0.24(1.76) + 2 - 2(1) = 1.86 \\ y[3] &= 1.86 - 0.24(2.28) + 3 - 2(2) = 0.31 \\ y[4] &= 0.31 - 0.24(1.86) + 4 - 2(3) = -2.14 \\ &\vdots \end{aligned}$$

These values are easily confirmed by modifying the MATLAB code found in Ex. 5.1.

```
01 n = (-2:4); x = n.*(n>=0);
02 y = zeros(size(n)); y(n<0) = [1,2];
03 for ind = find(n>=0),
04     y(ind) = y(ind-1)-0.24*y(ind-2)+x(ind)-2*x(ind-1);
05 end
06 y(n>=0)
y = 1.76  2.28  1.86  0.31  -2.14
```

Example 5.2 \triangleleft

Recursive and Nonrecursive Representations

Carefully note the recursive nature of the computations in Exs. 5.1 and 5.2. From the K initial conditions (and input), we first obtain $y[0]$. Then, using $y[0]$ and the previous $K - 1$ initial conditions (along with the input), we find $y[1]$. Next, using $y[0]$ and $y[1]$ along with the initial conditions and input, we obtain $y[2]$, and so on. This method is general and can be applied to a recursive form of difference equation of any order (see Eq. (5.1)). Interestingly, the direct form I (DFI) hardware realization of Eq. (5.1), shown in Fig. 4.32 on page 255, generates its output precisely in this iterative fashion.

When $a_k = 0$ for $k = (1, \dots, K)$, however, Eq. 5.1 becomes

$$y[n] = b_0x[n] + b_1x[n - 1] + \dots + b_Kx[n - K]. \quad (5.4)$$

This is a nonrecursive equation. In this form, determination of the present output $y[n]$ does not require the past values $y[n - 1], y[n - 2], \dots, y[n - K]$, and the output is not computed recursively. Instead, the output is computed directly from the input, as the next example demonstrates.

▷ Example 5.3 (Solution to a Nonrecursive Difference Equation)

As developed in Ex. 4.9, a digital differentiator is represented as

$$y[n] = \frac{1}{T}(x[n] - x[n - 1]). \quad (5.5)$$

Determine the response of this system to (a) $x[n] = u[n]$ and (b) $x[n] = nTu[n]$.

This example requires no recursion or iteration. The output at any instant is determined only by the input values. Since the system is nonrecursive, there is no need for initial conditions. Further, since the system and inputs are causal, the outputs are 0 for $n < 0$. Assuming uniform sampling at $t = nT$, the discrete-time inputs $u[n]$ and $nTu[n]$ would result from sampling the continuous-time signals $u(t)$ and $tu(t)$, respectively.

(a) Response to $x[n] = u[n]$

Substituting $x[n] = u[n]$ into Eq. (5.5) directly produces the result,

$$y[n] = \frac{1}{T}(u[n] - u[n - 1]) = \frac{1}{T}\delta[n].$$

Physical systems, however, do not produce outputs in this all-at-once manner. Rather, hardware tends to compute the output point by point as the input is acquired. To demonstrate, we start at $n = 0$ and substitute $x[0] = 1$ and $x[-1] = 0$ into Eq. (5.5) to obtain

$$y[0] = \frac{1}{T}(1 - 0) = \frac{1}{T}.$$

Setting $n = 1$ in Eq. (5.5) and substituting $x[0] = x[1] = 1$, we obtain

$$y[1] = \frac{1}{T}(1 - 1) = 0.$$

In a similar way, we obtain $y[2] = y[3] = y[4] = \dots = 0$. Taken together, the output is identical to our earlier result, $y[n] = \frac{1}{T}\delta[n]$.

(b) Response to $x[n] = nTu[n]$

Substituting $x[n] = nTu[n]$ into Eq. (5.5) produces the result

$$y[n] = \frac{1}{T}(nTu[n] - (n - 1)Tu[n - 1]) = u[n - 1].$$

This result is also obtained in sequence as follows. Setting $n = 0$ and substituting $x[0] = 0$ and $x[-1] = 0$, we obtain

$$y[0] = 0.$$

Setting $n = 1$ in Eq. (5.5) and substituting $x[1] = T$ and $x[0] = 0$, we obtain

$$y[1] = \frac{1}{T}(T - 0) = 1.$$

Setting $n = 2$ in Eq. (5.5) and substituting $x[1] = 2T$ and $x[0] = T$, we obtain

$$y[2] = \frac{1}{T}(2T - T) = 1.$$

Continuing on, we find that $y[3] = y[4] = y[5] = \dots = 1$. Taken together, the output is thus $y[n] = u[n - 1]$, which matches our earlier result.

When $y[n]$ is passed through a D/C converter, the continuous-time output is a fairly close approximation to a unit step, which is the true derivative of a ramp input. As $T \rightarrow 0$, the output approaches the ideal. See Fig. 4.26.

Example 5.3 

▷ Example 5.4 (Recursive versus Nonrecursive Implementations)

Consider a five-point sum, where the current output is simply a sum of the five most recent input values. Express this system using both nonrecursive and recursive forms, and compare the two implementations.

This system is somewhat unusual in that it can be easily expressed in either a recursive or nonrecursive form. In nonrecursive form, the input and output are related by

$$y[n] = x[n] + x[n - 1] + x[n - 2] + x[n - 3] + x[n - 4].$$

We can find $y[n - 1]$ by replacing n with $n - 1$ in this equation. Then, subtracting $y[n - 1]$ from $y[n]$, we obtain the recursive form as

$$y[n] - y[n - 1] = x[n] - x[n - 5].$$

Clearly, both equations represent the same system. It is easy to see that computation of each value of $y[n]$ from the nonrecursive form requires four additions, whereas the same computation by iteration using the recursive form requires only two additions. The computational economy of the recursive form over the nonrecursive form is significant, an observation that holds true in general. In fact, recursive equations are most often *dramatically* more efficient than their nonrecursive counterparts.

Example 5.4 

Iterative methods are useful to solve difference equations, particularly from the perspective of hardware realization. Despite this fact, a closed-form solution is far more useful in the study of system behavior and its dependence on the input and various system parameters. For this reason, we next develop a systematic procedure to analyze discrete-time systems that is similar to analysis techniques for continuous-time systems.

▷ Drill 5.1 (Iterative Solution to a Difference Equation)

Using the iterative method, find the first three terms of $y[n]$ if

$$y[n + 1] - 2y[n] = x[n].$$

The initial condition is $y[-1] = 10$, and the input $x[n] = 2u[n]$.



▷ **Drill 5.2 (Solution to a Nonrecursive Difference Equation)**

Consider an input $x[n] = \cos(\pi n/2)u[n]$ and a system represented as

$$y[n] = \frac{1}{5}(x[n+2] + x[n+1] + x[n] + x[n-1] + x[n-2]).$$

Determine the output $y[n]$ at times $n = 0$ and $n = 1234$.

△

5.2 Operator Notation

For the sake of compactness and ease of manipulation, it is convenient to represent equations using operator notation. Continuous-time differential equations, for example, can be represented in terms of the operator D , which denotes the operation of differentiation. For discrete-time difference equations, we shall use the operator E to denote the operation of advancing a sequence by one unit. By extension, an advance-by-two operation is represented as $E(E) = E^2$, and so forth. Thus,

$$\begin{aligned} E\{x[n]\} &\equiv x[n+1] \\ E^2\{x[n]\} &\equiv x[n+2] \\ &\vdots \\ E^K\{x[n]\} &\equiv x[n+K]. \end{aligned} \tag{5.6}$$

Systems described in terms of the advance operators E, E^2, \dots are said to be in *advance-operator form*.[†]

To provide an example, the digital differentiator of Ex. (5.3) is represented in advance form as

$$y[n+1] = \frac{1}{T}(x[n+1] - x[n]).$$

Using operator notation, this equation becomes

$$E\{y[n]\} = \frac{1}{T}(E\{x[n]\} - x[n]) = \frac{1}{T}(E - 1)\{x[n]\}.$$

Similarly, the general K th-order difference equation of Eq. (5.1) is expressed in advance form as

$$\begin{aligned} y[n+K] + a_1y[n+(K-1)] + \cdots + a_{K-1}y[n+1] + a_Ky[n] = \\ b_0x[n+K] + b_1x[n+(K-1)] + \cdots + b_{K-1}x[n+1] + b_Kx[n]. \end{aligned} \tag{5.7}$$

In operator notation, Eq. (5.7) becomes

$$A(E)\{y[n]\} = B(E)\{x[n]\}, \tag{5.8}$$

where $A(E)$ and $B(E)$ are the K th-order polynomial operators

$$\begin{aligned} A(E) &= E^K + a_1E^{K-1} + \cdots + a_{K-1}E + a_K \\ \text{and } B(E) &= b_0E^K + b_1E^{K-1} + \cdots + b_{K-1}E + b_K. \end{aligned}$$

[†]While we focus our current discussion on advance-operator form, other operator notations are certainly possible. For example, *delay-operator form* represents a system in terms of the delay operators E^{-1}, E^{-2}, \dots , where E^{-1} represents a unit-delay operation, E^{-2} represents a delay-by-two operation, and so forth.

Since delay form is easily and directly converted to advance form and vice versa, the operator notation of Eq. (5.8) is valid for either the advance form of Eq. (5.7) or the delay form of Eq. (5.1). To provide an example, consider a second-order system described in delay form as

$$y[n] - 2y[n-1] - 7y[n-2] = x[n] + 3x[n-1].$$

Replacing n with $n+2$ yields the advance form,

$$y[n+2] - 2y[n+1] - 7y[n] = x[n+2] + 3x[n+1].$$

In advance-operator form, the system is thus described as

$$(E^2 - 2E - 7) \{y[n]\} = (E^2 + 3E) \{x[n]\}.$$

Deceptive Appearances

Equation (5.8) is Eq. (5.7) (or Eq. (5.1)) expressed in a compact form using operator notation. Despite its deceptive appearance, Eq. (5.8) is not an algebraic equation and should never be treated as such. In other words, $A(E) \{y[n]\}$ does not mean that a polynomial $A(E)$ multiplies $y[n]$. It should be understood in the sense that $A(E)$ operates on $y[n]$. It is dangerous to apply the rules of algebra indiscriminately to equations written in operator form. To highlight these dangers, consider the system of Ex. 5.1 described in advance-operator notation as

$$(E - 0.5) \{y[n]\} = (E - 0.5) \{x[n]\}.$$

This is a first-order system that can support initial conditions, as the ZIR of Fig. 5.1b demonstrates. Algebraic rules, which would allow us to divide both sides of the equation by $E - 0.5$, incorrectly suggest that this system is equivalent to $y[n] = x[n]$, a 0th-order system whose ZIR is always zero. Clearly, the two systems cannot be equivalent. Although operator notation is a convenient and compact way of manipulating difference equations, it is not algebraic in nature.

Response of a Linear Discrete-Time System

We have seen earlier that Eq. (5.7) is a linear equation. From the decomposition property, we have also seen that the general solution of a linear equation is the sum of its zero-input and zero-state responses. Let us revisit these ideas using operator notation.

Recall that Eq. (5.8) represents Eq. (5.7) in operator form. To demonstrate that it is linear, let the inputs $x_1[n]$ and $x_2[n]$ generate the outputs $y_1[n]$ and $y_2[n]$, respectively. From Eq. (5.8) it follows that

$$A(E) \{y_1[n]\} = B(E) \{x_1[n]\} \quad \text{and} \quad A(E) \{y_2[n]\} = B(E) \{x_2[n]\}.$$

Multiplying these equations by k_1 and k_2 , respectively, and then adding yield

$$A(E) \{k_1 y_1[n] + k_2 y_2[n]\} = B(E) \{k_1 x_1[n] + k_2 x_2[n]\}.$$

This equation shows that the input $k_1 x_1[n] + k_2 x_2[n]$ generates the response $k_1 y_1[n] + k_2 y_2[n]$. Therefore, the system of Eq. (5.8) is linear.

Continuing, if $y_{\text{zir}}[n]$ is the zero-input response, then, by definition,

$$A(E) \{y_{\text{zir}}[n]\} = 0.$$

Further, if $y_{\text{zs}}[n]$ is the zero-state response, then $y_{\text{zs}}[n]$ is the solution of

$$A(E) \{y_{\text{zs}}[n]\} = B(E) \{x[n]\}$$

subject to zero initial conditions (zero-state). The addition of these two equations yields

$$A(E) \{y_{\text{zir}}[n] + y_{\text{zs}}[n]\} = B(E) \{x[n]\}.$$

Clearly, $y_{\text{zir}}[n] + y_{\text{zs}}[n]$ is the general solution of Eq. (5.8).

5.3 The Zero-Input Response

The zero-input response $y[n]$ is the system response to the internal conditions (initial conditions) when the input is zero.[†] Thus, $y[n]$ is the solution of Eq. (5.8) with $x[n] = 0$; that is,

$$A(E)\{y[n]\} = 0 \quad (5.9)$$

or

$$(E^K + a_1 E^{K-1} + \cdots + a_{K-1} E + a_K) \{y[n]\} = 0,$$

which means that

$$y[n+K] + a_1 y[n+K-1] + \cdots + a_{K-1} y[n+1] + a_K y[n] = 0.$$

Although we can solve this equation systematically, even a cursory examination points to its solution. This equation states that a linear combination of $y[n]$ and its shifted versions $y[n+k]$ ($k = 1, 2, \dots, K$) is zero *not for some values of n but for all n*. Such a situation is possible *if and only if* $y[n]$ and its shifted versions $y[n+k]$ are of the same form. Only an exponential function γ^n has this property. To see how, notice that

$$E^k \{\gamma^n\} = \gamma^{n+k} = \gamma^k \gamma^n.$$

This equation shows that the shifted exponential γ^{n+k} is just a constant times γ^n . Clearly, γ^{n+k} and γ^n are of the same form. Therefore, a solution of Eq. (5.9) is of the form

$$y[n] = c\gamma^n. \quad (5.10)$$

To determine c and γ , we first note from Eq. (5.6) that

$$\begin{aligned} E\{y[n]\} &= y[n+1] = c\gamma^{n+1} \\ E^2\{y[n]\} &= y[n+2] = c\gamma^{n+2} \\ &\vdots \\ E^K\{y[n]\} &= y[n+K] = c\gamma^{n+K}. \end{aligned}$$

Substituting these results into Eq. (5.9) yields

$$c(\gamma^K + a_1 \gamma^{K-1} + \cdots + a_{K-1} \gamma + a_K) \gamma^n = 0.$$

A nontrivial solution of this equation requires that

$$(\gamma^K + a_1 \gamma^{K-1} + \cdots + a_{K-1} \gamma + a_K) = A(\gamma) = 0. \quad (5.11)$$

Our solution $c\gamma^n$ (Eq. (5.10)) is correct, provided that γ satisfies Eq. (5.11).

Since $A(\gamma)$ is a K th-order polynomial, it has K roots. Assuming that these roots are distinct, Eq. (5.11) can be expressed in factored form as

$$(\gamma - \gamma_1)(\gamma - \gamma_2) \cdots (\gamma - \gamma_K) = A(\gamma) = 0. \quad (5.12)$$

Clearly, Eq. (5.12) has K solutions $\gamma_1, \gamma_2, \dots, \gamma_K$, and therefore, Eq. (5.9) also has K solutions $y_1[n] = c_1 \gamma_1^n, y_2[n] = c_2 \gamma_2^n, \dots, y_K[n] = c_K \gamma_K^n$, where c_1, c_2, \dots, c_K are constants. We can readily show that the general solution $y[n]$ is given by the sum of these K solutions. Briefly, since $y_1[n], y_2[n], \dots, y_K[n]$ are all solutions of Eq. (5.9), we know that

$$A(E)\{y_1[n]\} = 0$$

$$A(E)\{y_2[n]\} = 0$$

\vdots

$$A(E)\{y_K[n]\} = 0.$$

[†]To help avoid confusion, we sometimes subscript the zero-input response as $y_{\text{zir}}[n]$.

Multiplying these equations by constants c_1, c_2, \dots, c_K , respectively, and adding them together yield

$$A(E) \{c_1y_1[n] + c_2y_2[n] + \dots + c_Ky_K[n]\} = 0.$$

This result shows that $c_1y_1[n] + c_2y_2[n] + \dots + c_Ky_K[n]$ is also a solution to Eq. (5.9). Thus,

$$\begin{aligned} y[n] &= c_1y_1[n] + c_2y_2[n] + \dots + c_Ky_K[n] \\ &= c_1\gamma_1^n + c_2\gamma_2^n + \dots + c_K\gamma_K^n, \end{aligned} \quad (5.13)$$

where $\gamma_1, \gamma_2, \dots, \gamma_K$ are the roots of Eq. (5.12), and c_1, c_2, \dots, c_K are constants determined from K auxiliary conditions, generally given in the form of initial conditions.

The polynomial $A(\gamma)$ is called the *characteristic polynomial* of the system, and $A(\gamma) = 0$ is the *characteristic equation* of the system. Moreover, $\gamma_1, \gamma_2, \dots, \gamma_K$, the roots of the characteristic equation, are called *characteristic roots*, *characteristic values*, or *eigenvalues* of the system. The exponentials γ_k^n ($k = 1, 2, \dots, K$) are the *characteristic modes* or *natural modes* of the system. A characteristic mode corresponds to each characteristic root of the system, and the *zero-input response is a linear combination of the characteristic modes of the system*.

The characteristic polynomial and characteristic roots reflect the internal structure of a system. In a sense, they also predestine a system's output. A system is not free to behave in arbitrary ways but is bound by its characteristic polynomial to very particular and restricted behaviors. Unable to escape its inner character, a system has no choice to respond to initial conditions except with a combination of its natural modes.

▷ Example 5.5 (Computing the Zero-Input Response)

Using initial conditions $y[-1] = 0$ and $y[-2] = \frac{25}{4}$, determine the zero-input response of an LTID system described by the difference equation

$$y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2].$$

In operator notation, the system equation is

$$(E^2 - 0.6E - 0.16) \{y[n]\} = 5E^2 \{x[n]\}.$$

The characteristic polynomial is

$$A(\gamma) = \gamma^2 - 0.6\gamma - 0.16 = (\gamma + 0.2)(\gamma - 0.8).$$

The characteristic equation is

$$A(\gamma) = (\gamma + 0.2)(\gamma - 0.8) = 0.$$

Thus, the characteristic roots are $\gamma_1 = -0.2$ and $\gamma_2 = 0.8$, and the zero-input response is

$$y[n] = c_1(-0.2)^n + c_2(0.8)^n.$$

To determine the constants c_1 and c_2 , we set $n = -1$ and -2 into this equation and use the initial conditions $y[-1] = 0$ and $y[-2] = \frac{25}{4}$ to obtain

$$\left. \begin{array}{lcl} (-0.2)^{-1}c_1 + (0.8)^{-1}c_2 & = & 0 \\ (-0.2)^{-2}c_1 + (0.8)^{-2}c_2 & = & \frac{25}{4} \end{array} \right\} \implies \begin{bmatrix} -5 & \frac{5}{4} \\ 25 & \frac{25}{16} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{25}{4} \end{bmatrix}.$$

Using the matrix representation, MATLAB easily computes the constants c_1 and c_2 .

```
01 c = inv([-5 5/4;25 25/16])*[0;25/4]
c =
0.2000
0.8000
```

Therefore, the zero-input response is

$$y[n] = 0.2(-0.2)^n + 0.8(0.8)^n.$$

The reader can verify this solution by computing the first few terms using the iterative method (see Exs. 5.1 and 5.2).

Example 5.5 □

Repeated Roots

So far we have assumed the system to have K distinct characteristic roots $\gamma_1, \gamma_2, \dots, \gamma_K$ with corresponding characteristic modes $\gamma_1^n, \gamma_2^n, \dots, \gamma_K^n$. If two or more roots coincide (repeated roots), it is not enough to simply repeat the characteristic mode multiple times. Rather, additional characteristic modes are produced. Direct substitution shows that if a root γ has multiplicity r (repeats r times), then the characteristic modes corresponding to this root are $\gamma^n, n\gamma^n, n^2\gamma^n, \dots, n^{r-1}\gamma^n$. Thus, if the characteristic equation of a system is

$$A(\gamma) = (\gamma - \gamma_1)^r(\gamma - \gamma_{r+1})(\gamma - \gamma_{r+2}) \cdots (\gamma - \gamma_K),$$

then the zero-input response of the system is

$$y[n] = (c_1 + c_2n + c_3n^2 + \cdots + c_rn^{r-1})\gamma_1^n + c_{r+1}\gamma_{r+1}^n + c_{r+2}\gamma_{r+2}^n + \cdots + c_K\gamma_K^n. \quad (5.14)$$

Although Eq. (5.14) shows the zero-input response for a system with one root repeated r times and the remaining $K - r$ roots all distinct, the result is easily extended to any combination of distinct and repeated roots.

▷ **Example 5.6 (Zero-Input Response for a System with Repeated Roots)**

Using initial conditions $y[-1] = -\frac{1}{3}$ and $y[-2] = -\frac{2}{9}$, determine the zero-input response of an LTID system described by the equation

$$(E^2 + 6E + 9)\{y[n]\} = (2E^2 + 6E)\{x[n]\}.$$

In this case, the characteristic polynomial is $\gamma^2 + 6\gamma + 9 = (\gamma + 3)^2$, and the system has a repeated characteristic root at $\gamma = -3$. The characteristic modes are $(-3)^n$ and $n(-3)^n$. Hence, the zero-input response is

$$y[n] = (c_1 + c_2n)(-3)^n.$$

Similar to Ex. 5.5, we next set $n = -1$ and -2 into this equation and use the initial conditions $y[-1] = -\frac{1}{3}$ and $y[-2] = -\frac{2}{9}$ to obtain the matrix representation

$$\begin{bmatrix} (-3)^{-1} & (-1)(-3)^{-1} \\ (-3)^{-2} & (-1)(-3)^{-2} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{3} \\ \frac{2}{9} \end{bmatrix}.$$

The coefficients c_1 and c_2 are found using MATLAB.

```
01 c = inv([(-3)^(-1) -1*(-3)^(-1); (-3)^(-2) (-2)*(-3)^(-2)])*[-1/3;-2/9]
  c =
    4
    3
```

The zero-input response is thus

$$y[n] = (4 + 3n)(-3)^n.$$

Example 5.6 □

Complex Roots of Real Systems

As in the case of continuous-time systems, the complex roots of a real discrete-time system must occur in pairs of conjugates so that the system equation coefficients are real. Complex roots can be treated exactly as we treat real roots, and this is often the simplest approach. However, just as in the case of continuous-time systems, we can eliminate dealing with complex numbers by exploiting the conjugate pairings to produce a solution in real form.

To begin, we express any complex-conjugate roots γ and γ^* in polar form. If $|\gamma|$ is the magnitude and β is the angle of γ , then

$$\gamma = |\gamma|e^{j\beta} \quad \text{and} \quad \gamma^* = |\gamma|e^{-j\beta}.$$

The zero-input response for these roots is given by

$$\begin{aligned} y[n] &= c_1\gamma^n + c_2(\gamma^*)^n \\ &= c_1|\gamma|^n e^{j\beta n} + c_2|\gamma|^n e^{-j\beta n}. \end{aligned}$$

For a real system, the constants c_1 and c_2 must be conjugates ($c_2 = c_1^*$) so that $y[n]$ is a real function of n . Let

$$c_1 = \frac{c}{2}e^{j\theta} \quad \text{and} \quad c_2 = \frac{c}{2}e^{-j\theta}.$$

Then,

$$\begin{aligned} y[n] &= \frac{c}{2}|\gamma|^n \left(e^{j(\beta n + \theta)} + e^{-j(\beta n + \theta)} \right) \\ &= c|\gamma|^n \cos(\beta n + \theta), \end{aligned} \tag{5.15}$$

where c and θ are constants determined from the auxiliary conditions. This is the real form of the solution.

► Example 5.7 (Zero-Input Response for a System with Complex Roots)

Using initial conditions $y[-1] = 2$ and $y[-2] = 1$, determine and plot the zero-input response of a real LTID system described by the equation

$$(E^2 - 1.5588E + 0.81) \{y[n]\} = (E + 3) \{x[n]\}.$$

The characteristic polynomial of this system is $(\gamma^2 - 1.5588\gamma + 0.81)$. Although the roots can be found using the quadratic formula, it is simpler to use MATLAB.

```
01 gamma = roots([1 -1.5588 0.81])
gamma = 0.7794+0.4500i 0.7794-0.4500i
```

In factored form, the characteristic polynomial is thus $(\gamma - 0.78 - j0.45)(\gamma - 0.78 + j0.45)$. The characteristic roots are $\gamma_1 = 0.78 + j0.45$ and $\gamma_2 = 0.78 - j0.45 = \gamma_1^*$.

Next, we compute the zero-input response in two ways. First, we follow the standard form of Eq. (5.13). Since the roots are complex, this requires us to work with complex numbers. Second, we compute the zero-input response using the real form of Eq. (5.15).

Standard Form

Using Eq. (5.13), the zero-input response is

$$y[n] = c_1 \underbrace{(0.78 + j0.45)^n}_{\gamma_1} + c_2 \underbrace{(0.78 - j0.45)^n}_{\gamma_2}.$$

Next, we set $n = -1$ and -2 in this equation and express the results in matrix form as

$$\begin{bmatrix} \gamma_1^{-1} & \gamma_2^{-1} \\ \gamma_1^{-2} & \gamma_2^{-2} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} y[-1] \\ y[-2] \end{bmatrix}.$$

Substituting $y[-1] = 2$ and $y[-2] = 1$, we use MATLAB to compute the constants c_1 and c_2 .

```
02 c = inv([gamma(1)^(-1) gamma(2)^(-1):gamma(1)^(-2) gamma(2)^(-2)]*([2;1])
c = 1.1538-0.1984i
1.1538+0.1984i
```

Therefore, the zero-input response is

$$y[n] = (1.154 - j0.198)(0.78 + j0.45)^n + (1.154 + j0.198)(0.78 - j0.45)^n.$$

Figure 5.2 displays the result $y[n]$.

```
03 n = 0:20; y = @(n) c(1)*gamma(1).^n + c(2)*gamma(2).^n; stem(n,real(y(n)));
```

Although the response should be real, computer rounding causes MATLAB to produce a tiny imaginary portion to $y[n]$, which is eliminated in line 03 using the `real` command.

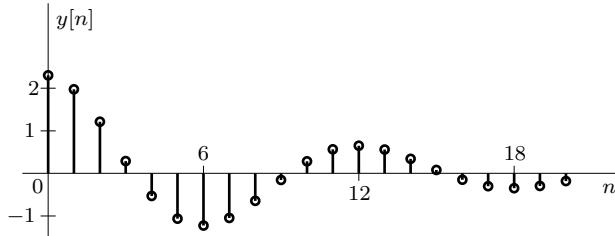


Figure 5.2: ZIR of $(E^2 - 1.5588E + 0.81) \{y[n]\} = (E + 3) \{x[n]\}$ using $y[-1] = 2$ and $y[-2] = 1$.

Real Form

When expressed in polar form, the characteristic roots $0.78 \pm j0.45$ are $0.9e^{\pm j\frac{\pi}{6}}$. Thus, $|\gamma| = 0.9$ and $\beta = \pi/6$, and the zero-input response, according to Eq. (5.15), is given by

$$y[n] = c(0.9)^n \cos\left(\frac{\pi}{6}n + \theta\right).$$

To determine the constants c and θ , we set $n = -1$ and -2 in this equation and substitute the initial conditions $y[-1] = 2$ and $y[-2] = 1$ to obtain

$$2 = \frac{c}{0.9} \cos\left(-\frac{\pi}{6} + \theta\right) = \frac{c}{0.9} \left[\frac{\sqrt{3}}{2} \cos(\theta) + \frac{1}{2} \sin(\theta) \right]$$

and $1 = \frac{c}{(0.9)^2} \cos\left(-\frac{\pi}{3} + \theta\right) = \frac{c}{0.81} \left[\frac{1}{2} \cos(\theta) + \frac{\sqrt{3}}{2} \sin(\theta) \right]$

or

$$\begin{aligned} \frac{\sqrt{3}}{1.8} c \cos(\theta) + \frac{1}{1.8} c \sin(\theta) &= 2 \\ \frac{1}{1.62} c \cos(\theta) + \frac{\sqrt{3}}{1.62} c \sin(\theta) &= 1 \end{aligned} \quad \left. \right\} \quad \Rightarrow \quad \begin{bmatrix} \frac{\sqrt{3}}{1.8} & \frac{1}{1.8} \\ \frac{1}{1.62} & \frac{\sqrt{3}}{1.62} \end{bmatrix} \begin{bmatrix} c \cos(\theta) \\ c \sin(\theta) \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

These are two simultaneous equations in two unknowns $c \cos(\theta)$ and $c \sin(\theta)$. Solving these equations yields

$$c \cos(\theta) = 2.308 \quad \text{and} \quad c \sin(\theta) = -0.397.$$

Dividing $c \sin(\theta)$ by $c \cos(\theta)$ yields

$$\tan(\theta) = \frac{-0.397}{2.308} = \frac{-0.172}{1}.$$

Thus,

$$\theta = \tan^{-1}(-0.172) = -0.17 \text{ rad.}$$

Substituting $\theta = -0.17$ rad into $c \cos(\theta) = 2.308$ yields $c = 2.34$. Thus,

$$y[n] = 2.34(0.9)^n \cos\left(\frac{\pi}{6}n - 0.17\right).$$

When plotted, this result is identical to Fig. 5.2.

Example 5.7 \triangleleft

Zero-Input Response for Nonrecursive Systems

For systems represented by nonrecursive equations, $a_k = 0$ for $k = 1, 2, \dots, K - 1$. Hence, $A(E) = E^K$, and the characteristic equation is $\gamma^K = 0$. The characteristic roots are all zero, resulting in the zero-input response $y[n] \equiv 0$. In such cases, the total response is given by the zero-state component.

▷ Drill 5.3 (Computing the Zero-Input Response)

Find and sketch the zero-input response for each of the following systems. Verify each solution by iteratively computing the first three terms.

- (a) $y[n + 1] - 0.8y[n] = 3x[n + 1]$ with $y[-1] = 10$
- (b) $y[n + 1] + 0.8y[n] = 3x[n + 1]$ with $y[-1] = 10$
- (c) $y[n] + 0.3y[n - 1] - 0.1y[n - 2] = x[n] + 2x[n - 1]$ with $y[-1] = 1$ and $y[-2] = 33$
- (d) $y[n] + 4y[n - 2] = 2x[n]$ with $y[-1] = -\frac{1}{2\sqrt{2}}$ and $y[-2] = \frac{1}{4\sqrt{2}}$

\triangleleft

5.3.1 Insights into the Zero-Input Behavior of a System

By definition, the zero-input response is the system response to its internal conditions, assuming that its input is zero. Understanding this phenomenon provides interesting insight into system behavior. If a system is disturbed momentarily from its rest position (implying that there is no outside disturbance after that moment), the system will not come back to rest instantaneously. In general, it will come back to rest over a period of time and only through a special type of motion that is characteristic of the system.[†]

For example, if we press on an automobile fender momentarily and then release it at $t = 0$, there is, ignoring gravity, no external force on the automobile for $t > 0$. Still, the auto body eventually comes back to its rest (equilibrium) position, but not through any arbitrary motion. It must do so using only a form of response that is sustainable by the system on its own without any external source because the input is zero. Only characteristic modes satisfy this condition. *The system uses a combination of characteristic modes to come back to the rest position while satisfying appropriate boundary (or initial) conditions.* If the shock absorbers of the automobile are in good condition (high damping coefficient), the characteristic modes will be monotonically decaying exponentials, and the auto body will come to rest rapidly with little or no oscillation. In contrast, for poor shock absorbers (low damping coefficients), the characteristic modes will be exponentially decaying sinusoids, and the body will come to rest through oscillatory motion.

To provide another example, when a series *RC* circuit with an initial charge on the capacitor is shorted, the capacitor will start to discharge exponentially through the resistor. This response of the *RC* circuit is caused entirely by its internal conditions and is sustained by this system without the aid of any external input. The exponential current waveform is therefore the characteristic mode of this *RC* circuit. Although we give here two continuous-time examples, the same observations hold for discrete-time systems as well.

[†]This assumes that the system will eventually come back to its original rest (or equilibrium) position.

Mathematically, we know that *any combination of characteristic modes can be sustained by the system alone without requiring an external input*. Let us demonstrate this fact with a first-order system represented by the equation

$$(E - 0.5) \{y[n]\} = x[n].$$

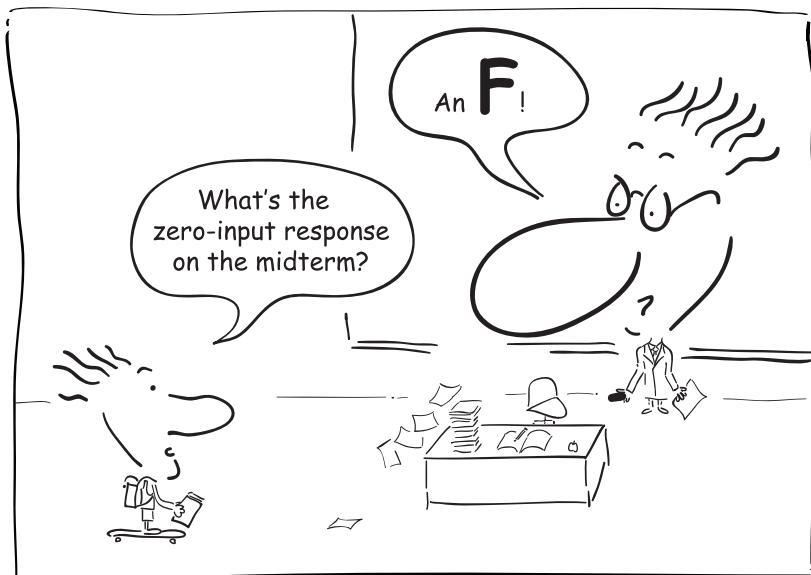
This system has a single characteristic root $\gamma_1 = 0.5$, and the characteristic mode is $(0.5)^n u[n]$. We now verify that the response $y[n] = c(0.5)^n$ can be sustained by this system without any input (zero input). The input $x[n]$ required to generate the response $y[n] = (0.5)^n$ is given by

$$\begin{aligned} x[n] &= y[n+1] - 0.5y[n] \\ &= c(0.5)^{n+1} - 0.5c(0.5)^n \\ &= 0.5c(0.5)^n - 0.5c(0.5)^n \\ &= 0. \end{aligned}$$

Clearly, the characteristic mode $y[n] = c(0.5)^n$ is sustained by this DT system on its own, without the necessity of an external input.

The Resonance Phenomenon

We have seen that any signal consisting of a system's characteristic modes is sustained by the system on its own; the system offers no obstacle to such signals. Imagine what happens if we actually drive a system with an external input that is one of its characteristic modes. This is like pouring gasoline on a fire in a dry forest or hiring an alcoholic to taste liquor. An alcoholic will gladly do the job without pay. Think what will happen if he is paid by the amount of liquor he tastes! The system response to characteristic modes is likewise naturally very high. We call this behavior the *resonance phenomenon*. An intelligent discussion of this important phenomenon requires an understanding of the zero-state response; for this reason, we postpone this topic until Sec. 5.8.6.



It's easy to determine the zero-input response.

5.4 The Unit Impulse Response

In this section, we develop time-domain methods to determine the response of an LTID system to the unit impulse input $\delta[n]$. The unit impulse response of an LTID system is very important because every input can be expressed as a sum of impulses of the form $\delta[n - m]$ (see Eq. (4.10)). Hence, a knowledge of the unit impulse response enables us to compute the response of an LTID system to any arbitrary input. Although the time-domain methods of this section are relatively simple, the next chapter presents another, much simpler method that utilizes the z -transform.

Consider a K th-order system specified like Eq. (5.8) as

$$A(E) \{y[n]\} = B(E) \{x[n]\}.$$

The unit impulse response $h[n]$ is the solution of this equation for the input $\delta[n]$ with all the initial conditions zero; that is,

$$A(E) \{h[n]\} = B(E) \{\delta[n]\} \quad (5.16)$$

subject to initial conditions

$$h[-1] = h[-2] = \dots = h[-K] = 0.$$

To determine $h[n]$, Eq. (5.16) can be solved iteratively or in a closed form. The following example demonstrates the iterative approach.

▷ Example 5.8 (Iterative Determination of the Impulse Response)

Iteratively find and then plot the unit impulse response $h[n]$ of a system described by the equation

$$y[n] - 0.6y[n - 1] - 0.16y[n - 2] = 5x[n].$$

To determine the unit impulse response, we let input $x[n] = \delta[n]$ and output $y[n] = h[n]$ to obtain

$$h[n] - 0.6h[n - 1] - 0.16h[n - 2] = 5\delta[n],$$

subject to zero initial state $h[-1] = h[-2] = 0$.

Setting $n = 0$ in this equation yields

$$h[0] - 0.6(0) - 0.16(0) = 5(1) \implies h[0] = 5.$$

Next, setting $n = 1$ and using $h[0] = 5$, we obtain

$$h[1] - 0.6(5) - 0.16(0) = 5(0) \implies h[1] = 3.$$

Similarly, setting $n = 2$ and using $h[1] = 3$ and $h[0] = 5$, we obtain

$$h[2] - 0.6(3) - 0.16(5) = 5(0) \implies h[2] = 2.6.$$

Continuing this way, we can determine any number of terms of $h[n]$. MATLAB performs these iterative calculations efficiently, the results of which are presented in Fig. 5.3.

```
01 n = (-2:16); delta = (n==0); h = zeros(size(n));
02 for ind = find(n>=0),
03     h(ind) = 0.6*h(ind-1)+0.16*h(ind-2)+5*delta(ind);
04 end
05 stem(n,h)
```

Unfortunately, the iterative approach does not yield a closed-form expression for $h[n]$. Nevertheless, determining a few values of $h[n]$ can be useful in determining the closed-form solution, as the following development shows.

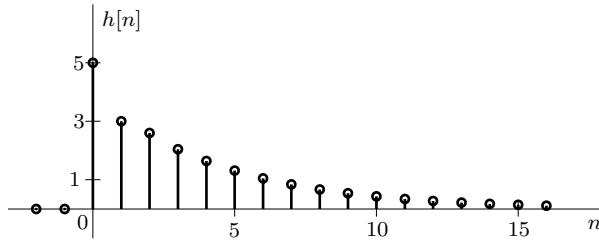


Figure 5.3: Impulse response $h[n]$ of $y[n] - 0.6y[n - 1] - 0.16y[n - 2] = 5x[n]$.

Example 5.8 \triangleleft

5.4.1 Closed-Form Solution of the Impulse Response

Recall that $h[n]$ is the system response to input $\delta[n]$, which is 0 for $n > 0$. We know that when the input is 0, only the characteristic modes can be sustained by the system. Therefore, $h[n]$ must be made up of characteristic modes for $n > 0$. Due to the $\delta[n]$ input, the response may have some nonzero value at $n = 0$ so that $h[n]$ can be expressed as[†]

$$h[n] = A_0\delta[n] + y_c[n]u[n],$$

where $y_c[n]$ is a linear combination of the characteristic modes. Substituting this expression into Eq. (5.8) yields $A(E)\{A_0\delta[n] + y_c[n]u[n]\} = B(E)\{\delta[n]\}$. Since $y_c[n]$ is made up of characteristic modes, $A(E)\{y_c[n]u[n]\} = 0$ ($n \geq 0$), and we are left with $A(E)\{A_0\delta[n]\} = B(E)\{\delta[n]\}$, or

$$A_0(\delta[n+K] + a_1\delta[n+(K-1)] + \cdots + a_K\delta[n]) = b_0\delta[n+K] + \cdots + b_K\delta[n].$$

Setting $n = 0$ into this equation and recognizing that $\delta[K] = 0$ for all $K \neq 0$, we find that

$$A_0a_K = b_K \implies A_0 = \frac{b_K}{a_K}.$$

Therefore, as long as $a_K \neq 0$,

$$h[n] = \frac{b_K}{a_K}\delta[n] + y_c[n]u[n], \quad (5.17)$$

We discuss the special case when $a_K = 0$ later in this section. The K unknown coefficients in $y_c[n]$ are determined from K values of $h[n]$ ($n \geq 0$). Fortunately, it is a straightforward task to determine K values of $h[n]$ iteratively, as demonstrated in Ex. 5.8. Once, we compute the K values $h[0], h[1], h[2], \dots, h[K-1]$ iteratively, we set $n = 0, 1, 2, \dots, K-1$ in Eq. (5.17) to determine the K unknowns in $y_c[n]$. The following example clarifies the process.

▷ **Example 5.9 (Closed-Form Representation of the Impulse Response)**

Determine a closed-form representation of the unit impulse response $h[n]$ for the system in Ex. 5.8, which is specified by the equation

$$y[n] - 0.6y[n - 1] - 0.16y[n - 2] = 5x[n].$$

[†]Since $h[n]$ consists of characteristic modes only for $n > 0$, the characteristic mode terms in $h[n]$ start at $n = 1$. To reflect this behavior, they are naturally expressed in terms of $u[n-1]$. But because $u[n-1] = u[n] - \delta[n]$, we use the more convenient $u[n]$ and let the impulse $A_0\delta[n]$ absorb the difference.

In the advance form, the system is $y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2]$, which leads to the operator notation of

$$(E^2 - 0.6E - 0.16) \{y[n]\} = 5E^2 \{x[n]\}.$$

The characteristic polynomial is

$$\gamma^2 - 0.6\gamma - 0.16 = (\gamma + 0.2)(\gamma - 0.8),$$

and the characteristic modes are $(-0.2)^n$ and $(0.8)^n$. Therefore,

$$y_c[n] = c_1(-0.2)^n + c_2(0.8)^n.$$

Since $a_K = -0.16$ and $b_K = 0$, $A_0 = 0$, and according to Eq. (5.17), the impulse response is

$$h[n] = (c_1(-0.2)^n + c_2(0.8)^n)u[n]. \quad (5.18)$$

To determine c_1 and c_2 , we need to find two values of $h[n]$ iteratively. Example 5.8 performs this step and establishes $h[0] = 5$ and $h[1] = 3$. Using these values and setting $n = 0$ and 1 in Eq. (5.18), we obtain

$$\begin{array}{rcl} c_1 + c_2 & = & 5 \\ -0.2c_1 + 0.8c_2 & = & 3 \end{array} \left. \right\} \implies \begin{bmatrix} 1 & 1 \\ -0.2 & 0.8 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}.$$

Using the matrix representation, MATLAB computes the constants c_1 and c_2 .

```
01 c = inv([1 1;-0.2 0.8])*[5;3]
c =
    1
    4
```

Therefore,

$$h[n] = [(-0.2)^n + 4(0.8)^n] u[n].$$

The plot of this expression is identical to Fig. 5.3.

Example 5.9 ◀

Accommodating Systems with $a_K = 0$

The impulse response $h[n]$ is represented with Eq. (5.17) only when $a_K \neq 0$. When $a_K = 0$, the form of $h[n]$ changes slightly. Let R designate the number of consecutive coefficients including a_K that are 0; that is, $a_K = a_{K-1} = \dots = a_{K-(R-1)} = 0$ and $a_{K-R} \neq 0$. In this case, $A(E)$ can be expressed as $E^R \hat{A}(E)$, and using $x[n] = \delta[n]$ and $y[n] = h[n]$, Eq. (5.8) becomes

$$E^R \hat{A}(E) \{h[n]\} = B(E) \{\delta[n]\} = B(E) \{E^R \delta[n-R]\} = E^R B(E) \{\delta[n-R]\}.$$

Hence,

$$\hat{A}(E) \{h[n]\} = B(E) \{\delta[n-R]\}.$$

In this case, the input vanishes not for $n > 0$ but for $n > R$. Therefore, the response consists not only of the zero-input term but also impulses at $n = 0, 1, \dots, R$. In general, even for $a_K \neq 0$,

$$h[n] = A_0 \delta[n] + A_1 \delta[n-1] + \dots + A_R \delta[n-R] + y_c[n]u[n]. \quad (5.19)$$

Basically, we get an extra impulse function (besides $A_0 \delta[n]$) for every characteristic root that is 0. We can determine the unknowns A_0, A_1, \dots, A_R and the $K-R$ coefficients in $y_c[n]$ from the $K+1$ initial values $h[0], h[1], \dots, h[K]$, determined as usual from the iterative solution of the equation $A(E) \{h[n]\} = B(E) \{\delta[n]\}$. The reason that $y_c[n]$ has $K-R$ rather than K coefficients is that R of the modes are necessarily 0. The next example demonstrates the procedure.

▷ **Example 5.10 (Determination of the Impulse Response When $a_K = 0$)**

Determine the impulse response $h[n]$ of a system described by the equation

$$(E^3 + E^2) \{y[n]\} = x[n].$$

In this case, $a_K = 0$, so we determine $h[n]$ from Eq. (5.19). Since $A(E) = E^3 + E^2$, there are $K = 3$ characteristic roots: one at -1 and two at 0 . Only the nonzero characteristic root shows up in $y_c[n]$. Noting that $R = 2$, the impulse response is therefore

$$h[n] = A_0\delta[n] + A_1\delta[n - 1] + A_2\delta[n - 2] + c_1(-1)^n u[n]. \quad (5.20)$$

To determine the coefficients A_0 , A_1 , A_2 , and c_1 , we require $K + 1 = 4$ values of $h[n]$ ($n \geq 0$), which we obtain iteratively as in Ex. 5.8.

```

01 n = (-3:3); delta = (n==0); h = zeros(size(n));
02 for ind = find(n>=0),
03     h(ind) = -h(ind-1)+delta(ind-3);
04 end
05 h(n>=0)
ans = 0 0 0 1

```

Expressing Eq. (5.20) in matrix form using $n = 0, 1, 2$, and 3 yields

$$\begin{bmatrix} 1 & 0 & 0 & (-1)^0 \\ 0 & 1 & 0 & (-1)^1 \\ 0 & 0 & 1 & (-1)^2 \\ 0 & 0 & 0 & (-1)^3 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ c_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Using the matrix representation, MATLAB computes the needed coefficients.

```

06 coef = inv([1 0 0 1;0 1 0 -1;0 0 1 1;0 0 0 -1])*([0;0;0;1])
coef =
    1
    -1
    1
    -1

```

Therefore,

$$h[n] = \delta[n] - \delta[n - 1] + \delta[n - 2] - (-1)^n u[n].$$

Example 5.10 ▷

Impulse Response of a Nonrecursive System

Although Eq. (5.20) can be used to determine the impulse response of a nonrecursive system, there is a much simpler and more direct method. Because $a_1 = \dots = a_K = 0$ for nonrecursive systems, the general delay form of the system equation is, according to Eq. (5.4),

$$y[n] = b_0x[n] + b_1x[n - 1] + \dots + b_Kx[n - K].$$

Letting $x[n] = \delta[n]$ and $y[n] = h[n]$ in this equation, the impulse response is therefore

$$h[n] = b_0\delta[n] + b_1\delta[n - 1] + \dots + b_K\delta[n - K] = \sum_{k=0}^K b_k\delta[n - k]. \quad (5.21)$$

Such a nonrecursive system has no natural modes, and thus its impulse response has no $y_c[n]$ term.

▷ **Example 5.11 (Impulse Response of a Nonrecursive System)**

Find the impulse response of a nonrecursive LTID system described by the equation

$$y[n] = x[n] + x[n - 1] + x[n - 2] + x[n - 3] + x[n - 4].$$

Following Eq. (5.21), we obtain the answer instantly as

$$h[n] = \delta[n] + \delta[n - 1] + \delta[n - 2] + \delta[n - 3] + \delta[n - 4].$$

Example 5.11 ◀

Impulse Response Duration: FIR and IIR Systems

Systems are commonly classified according to the duration of their impulse responses. A *finite impulse response (FIR) system* has an impulse response that has finite length, which means that $h[n]$ is 0 outside some finite interval. In Eq. (5.1), if the coefficients $a_k = 0$ for $k = 1, 2, 3, \dots, K$, then the system is an FIR system because, as shown in Eq. (5.21), the number of nonzero terms in $h[n]$ is a finite number, $K + 1$. Such a system uses only the present and past K input values. For this reason, FIR systems are said to have a finite memory of K samples.

Conversely, an *infinite impulse response (IIR) system* has an impulse response that has infinite length. In Eq. (5.1), if some coefficients $a_k \neq 0$, then the system is usually, although not always, IIR. For causal IIR systems, the impulse response may asymptotically approach 0 as $n \rightarrow \infty$, but it never actually reaches 0. The system of Ex. 5.8 is an example of an IIR system.

Notice that the classification of a system as FIR or IIR is not the same as classifying a system as recursive or nonrecursive. The latter classification is based on whether a system is implemented recursively or nonrecursively, not on the impulse response characteristics. The FIR system of Ex. 5.4, for instance, can be either recursive or nonrecursive depending on its implementation. Still, FIR systems are most commonly nonrecursive, and IIR systems are most commonly recursive.

▷ **Drill 5.4 (Determining the Impulse Response)**

Find the impulse response $h[n]$ of the LTID systems specified by the following equations:

- (a) $y[n+2] - 5y[n+1] + 6y[n] = 8x[n+1] - 19x[n]$
- (b) $y[n+2] - 4y[n+1] + 4y[n] = 2x[n+2] - 2x[n+1]$
- (c) $y[n] = x[n] - 2x[n-1]$

◀

5.5 The Zero-State Response

The zero-state response $y[n]$ is the system response to an input $x[n]$ when the system is in zero state. In this section, we shall assume that systems are in zero state unless mentioned otherwise, so the zero-state response will be the total response of the system. Here we follow a procedure parallel to that used in the continuous-time case by expressing an arbitrary input $x[n]$ as a sum of impulse components.

Any DT signal $x[n]$ can be represented as a sum of weighted impulse functions. The signal $x[n]$ shown in Fig. 5.4a, for example, is comprised of weighted impulse components including those depicted in Figs. 5.4b through 5.4f. The component of $x[n]$ at $n = m$ is $x[m]\delta[n - m]$, and $x[n]$ is

the sum of all these components summed from $m = -\infty$ to ∞ . Therefore,

$$\begin{aligned} x[n] &= x[0]\delta[n] + x[1]\delta[n-1] + x[2]\delta[n-2] + \dots \\ &\quad + x[-1]\delta[n+1] + x[-2]\delta[n+2] + \dots \\ &= \sum_{m=-\infty}^{\infty} x[m]\delta[n-m]. \end{aligned}$$

This is just the DT sampling property of Eq. (4.10).

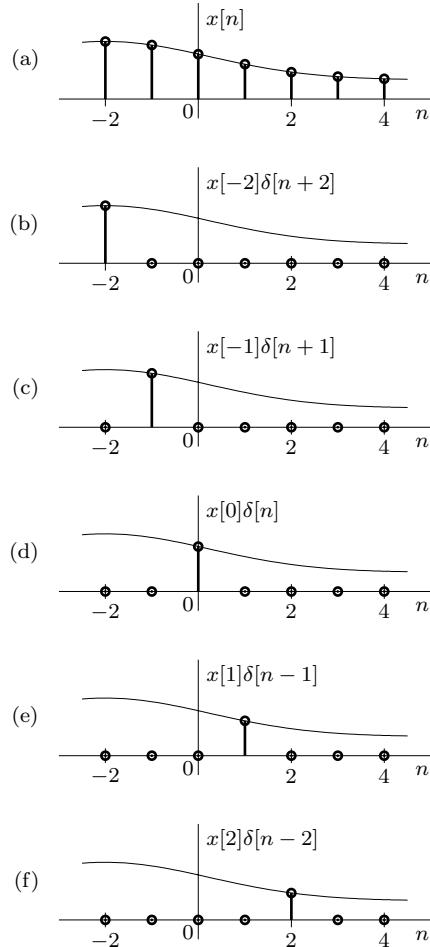


Figure 5.4: Representing a signal $x[n]$ as a sum of weighted impulse functions.

For a linear and time-invariant system, knowing the system response to $\delta[n]$ allows us to determine the system response to an arbitrary input by summing the system response to various impulse components. Let $h[n]$ be the system response to impulse input $\delta[n]$, a relationship we designate with an arrow directed from input to output as

$$\delta[n] \implies h[n].$$

When a system is time-invariant, this relationship generalizes to

$$\delta[n-m] \implies h[n-m].$$

When a system is also linear, the scaling property ensures that

$$x[m]\delta[n-m] \implies x[m]h[n-m],$$

and the additivity property ensures that

$$\underbrace{\sum_{m=-\infty}^{\infty} x[m]\delta[n-m]}_{x[n]} \implies \underbrace{\sum_{m=-\infty}^{\infty} x[m]h[n-m]}_{y[n]}.$$

From the DT sampling property, we recognize the left-hand side of this equation as $x[n]$. The right-hand side, therefore, is the system response $y[n]$ to input $x[n]$. Therefore,[†]

$$y[n] = \sum_{m=-\infty}^{\infty} x[m]h[n-m]. \quad (5.22)$$

Referring to Eq. (4.59), we recognize the summation on the right-hand side of Eq. (5.22) as the *convolution sum* of $x[n]$ and $h[n]$, which is represented symbolically by $x[n] * h[n]$.

In deriving Eq. (5.22), we only assume the system to be linear and time invariant. There are no other restrictions on either the input signal or the system. In our applications, almost all the input signals are causal, and a majority of the systems are also causal. These restrictions further simplify the limits of the sum in Eq. (5.22). If the input $x[n]$ is causal, $x[m] = 0$ for $m < 0$. Similarly, if the system is causal (i.e., if $h[n]$ is causal), then $h[n-m] = 0$ when $n-m < 0$ or $m > n$. Therefore, if $x[n]$ and $h[n]$ are both causal, the product $x[m]h[n-m] = 0$ for $m < 0$ and for $m > n$, and Eq. (5.22) simplifies to

$$y[n] = \sum_{m=0}^n x[m]h[n-m]. \quad (5.23)$$

We shall evaluate the convolution sum first by an analytical method and later with graphical aids. Table 5.1 provides a selection of sums that are commonly encountered in convolution calculations [1].

1.	$\sum_{m=p}^n r^m = \frac{r^p - r^{n+1}}{1-r}$	$r \neq 1$
2.	$\sum_{m=0}^n m = \frac{n(n+1)}{2}$	
3.	$\sum_{m=0}^n m^2 = \frac{n(n+1)(2n+1)}{6}$	
4.	$\sum_{m=0}^n mr^m = \frac{r+[n(r-1)-1]r^{n+1}}{(r-1)^2}$	$r \neq 1$
5.	$\sum_{m=0}^n m^2 r^m = \frac{r[(1+r)(1-r^n)-2n(1-r)r^n-n^2(1-r)^2r^n]}{(r-1)^3}$	$r \neq 1$

Table 5.1: A selection of useful sums.

▷ Example 5.12 (Analytic Computation of the Convolution Sum)

Analytically determine the DT convolution $y[n] = x[n] * h[n]$ for

$$x[n] = (0.8)^n u[n] \quad \text{and} \quad h[n] = (0.3)^n u[n].$$

[†]In deriving this result, we assume a linear and time-invariant system. If the system is linear and time variant, then the system response to input $\delta[n-m]$ cannot be expressed as $h[n-m]$ but instead has the form $h[n, m]$. Using this form, Eq. (5.22) is modified as

$$y[n] = \sum_{m=-\infty}^{\infty} x[m]h[n, m].$$

We have

$$y[n] = \sum_{m=-\infty}^{\infty} x[m]h[n-m].$$

Note that

$$x[m] = (0.8)^m u[m] \quad \text{and} \quad h[n-m] = (0.3)^{n-m} u[n-m].$$

Both $x[n]$ and $h[n]$ are causal. Therefore, according to Eq. (5.23),

$$\begin{aligned} y[n] &= \sum_{m=0}^n x[m]h[n-m] \\ &= \sum_{m=0}^n (0.8)^m u[m] (0.3)^{n-m} u[n-m]. \end{aligned}$$

In the preceding summation, m lies between 0 and n ($0 \leq m \leq n$). Therefore, if $n \geq 0$, then both m and $n - m \geq 0$, so $u[m] = u[n-m] = 1$. If $n < 0$, m is negative, and $u[m] = 0$. Therefore, the preceding equation becomes

$$y[n] = \begin{cases} \sum_{m=0}^n (0.8)^m (0.3)^{n-m} & n \geq 0 \\ 0 & n < 0 \end{cases}$$

or, more compactly,

$$y[n] = (0.3)^n \sum_{m=0}^n \left(\frac{0.8}{0.3}\right)^m u[n].$$

This is a geometric progression with common ratio $(0.8/0.3)$. Using entry 1 of Table 5.1,

$$\begin{aligned} y[n] &= (0.3)^n \frac{1 - (0.8/0.3)^{n+1}}{(1 - 0.8/0.3)} u[n] \\ &= 2 [(0.8)^{n+1} - (0.3)^{n+1}] u[n]. \end{aligned}$$

Example 5.12 ◀

▷ Drill 5.5 (Analytic Computation of the Convolution Sum)

Analytically determine the DT convolution $y[n] = x[n] * h[n]$ for $x[n] = (0.8)^n u[n]$ and $h[n] = u[n]$. How does $y[n]$ change if $h[n] = u[n-1]$ rather than $u[n]$?

◀

5.5.1 Convolution Sum Properties

A number of useful properties accompany the convolution sum, which is similar in structure to the convolution integral. Not surprisingly, the properties of the convolution sum are similar to those of the convolution integral. We enumerate the most important properties here without proofs, which are straightforward in most cases (see Prob. 5.5-1).

1. Commutative Property:

$$x[n] * h[n] = h[n] * x[n].$$

2. Distributive Property:

$$x[n] * (h_1[n] + h_2[n]) = x[n] * h_1[n] + x[n] * h_2[n].$$

3. Associative Property:

$$x[n] * (h_1[n] * h_2[n]) = (x[n] * h_1[n]) * h_2[n].$$

4. Shifting Property:

If

$$x[n] * h[n] = y[n],$$

then

$$x[n-m] * h[n-p] = y[n-m-p].$$

The shifting property highlights a deficiency of the compact notation $y[n] = x[n]*h[n]$: shifting signal $y[n]$ by m , which involves replacing n with $n-m$, would seem to imply that $y[n-m] = x[n-m]*h[n-m]$, but this is not so! To overcome this deficiency, convolution is sometimes represented in an alternate and arguably more correct form $(x * h)[n]$.

5. Convolution with an Impulse:

$$x[n] * \delta[n] = x[n].$$

6. Width and Length Properties:

If $x[n]$ and $h[n]$ have finite widths of W_x and W_h units, respectively, then the width of $x[n]*h[n]$ is $W_x + W_h$ units. Alternately, the width property may be stated in terms of lengths. If $x[n]$ and $h[n]$ have finite lengths of L_x and L_h , respectively, then the length of $x[n]*h[n]$ is $L_x + L_h - 1$ elements. Notice that the width of a DT signal is defined as one less than its length (number of elements). For instance, the signal $u[n] - u[n-10]$ has 10 elements (length of 10) but has a width of only 9 units. The unit impulse $\delta[n]$ has a length of 1 and a width of 0.

Certain convolution sums occur frequently. Tables of convolution sums, such as Table 5.2, eliminate the need for boring and repetitive calculations; desired results are simply read directly from the table. For example, the convolution in Ex. 5.12 can be read directly from entry 8 of Table 5.2 as

$$(0.8)^n u[n] * (0.3)^n u[n] = \frac{(0.8)^{n+1} - (0.3)^{n+1}}{0.8 - 0.3} u[n] = 2[(0.8)^{n+1} - (0.3)^{n+1}] u[n].$$

As the next example demonstrates, a combination of properties and tables allows an extended class of convolution problems to be solved with reduced effort.

▷ **Example 5.13 (Convolution Using Properties and Tables)**

For the input $x[n] = 4^{-n}u[n]$, find the zero-state response $y[n]$ of an LTID system described by the equation

$$y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2].$$

The input can be expressed as $x[n] = 4^{-n}u[n] = (1/4)^n u[n] = (0.25)^n u[n]$. The unit impulse response of this system, obtained in Ex. 5.9, is

$$h[n] = [(-0.2)^n + 4(0.8)^n] u[n].$$

Therefore,

$$y[n] = x[n] * h[n] = (0.25)^n u[n] * [(-0.2)^n u[n] + 4(0.8)^n u[n]].$$

Using the distributive property, we obtain

$$y[n] = (0.25)^n u[n] * (-0.2)^n u[n] + (0.25)^n u[n] * 4(0.8)^n u[n].$$

Using entry 8 in Table 5.2, we obtain

$$\begin{aligned} y[n] &= \left[\frac{(0.25)^{n+1} - (-0.2)^{n+1}}{0.25 - (-0.2)} + 4 \frac{(0.25)^{n+1} - (0.8)^{n+1}}{0.25 - 0.8} \right] u[n] \\ &= (2.22 [(0.25)^{n+1} - (-0.2)^{n+1}] - 7.27 [(0.25)^{n+1} - (0.8)^{n+1}]) u[n] \\ &= [-5.05(0.25)^{n+1} - 2.22(-0.2)^{n+1} + 7.27(0.8)^{n+1}] u[n]. \end{aligned}$$

Recognizing that $\gamma^{n+1} = \gamma(\gamma^n)$, the zero-state response is thus

$$\begin{aligned} y[n] &= [-1.26(0.25)^n + 0.444(-0.2)^n + 5.82(0.8)^n] u[n] \\ &= [-1.26(4)^{-n} + 0.444(-0.2)^n + 5.82(0.8)^n] u[n]. \end{aligned}$$

Example 5.13 \triangleleft

$x[n]$	$h[n]$	$x[n] * h[n]$
1. $x[n]$	$\delta[n - k]$	$x[n - k]$
2. $u[n]$	$u[n]$	$(n + 1)u[n]$
3. $u[n]$	$n u[n]$	$\frac{n(n+1)}{2} u[n]$
4. $nu[n]$	$nu[n]$	$\frac{n(n-1)(n+1)}{6} u[n]$
5. $u[n]$	$\gamma^n u[n]$	$\left(\frac{1-\gamma^{n+1}}{1-\gamma}\right) u[n]$
6. $nu[n]$	$\gamma^n u[n]$	$\left(\frac{\gamma(\gamma^{n-1})+n(1-\gamma)}{(1-\gamma)^2}\right) u[n]$
7. $\gamma^n u[n]$	$\gamma^n u[n]$	$(n + 1)\gamma^n u[n]$
8. $\gamma_1^n u[n]$	$\gamma_2^n u[n]$	$\left(\frac{\gamma_1^{n+1}-\gamma_2^{n+1}}{\gamma_1-\gamma_2}\right) u[n] \quad \gamma_1 \neq \gamma_2$
9. $\gamma_1^n u[n]$	$n\gamma_2^n u[n]$	$\frac{\gamma_1\gamma_2}{(\gamma_1-\gamma_2)^2} \left(\gamma_2^n + \frac{\gamma_2-\gamma_1}{\gamma_1} n\gamma_2^n - \gamma_1^n\right) u[n] \quad \gamma_1 \neq \gamma_2$
10. $\gamma_1^n u[n]$	$\gamma_2^n u[-n - 1]$	$\frac{\gamma_1}{\gamma_2-\gamma_1} \gamma_1^n u[n] + \frac{\gamma_2}{\gamma_2-\gamma_1} \gamma_2^n u[-n - 1] \quad \gamma_2 > \gamma_1 $
11. $ \gamma_1 ^n u[n]$	$ \gamma_2 ^n \cos(\beta n + \theta) u[n]$	$\frac{1}{R} [\gamma_2 ^{n+1} \cos[\beta(n+1)+\theta-\phi] - \gamma_1 ^{n+1} \cos(\theta-\phi)] u[n]$ $R = \sqrt{ \gamma_1 ^2 - 2 \gamma_1 \gamma_2 \cos(\beta) + \gamma_2 ^2}$ $\phi = \tan^{-1} \left[\frac{ \gamma_2 \sin(\beta)}{ \gamma_2 \cos(\beta) - \gamma_1 } \right]$

Table 5.2: Selected convolution sums.

Response to Multiple Inputs

Multiple inputs to LTI systems can be treated by applying the superposition principle. Each input is considered separately, with all other inputs assumed to be 0. The sum of all these individual system responses constitutes the total system output when all the inputs are applied simultaneously.

Response to Complex Inputs

We shall now show that for an LTID system with real $h[n]$, if the input and the output are expressed in terms of their real and the imaginary parts, then the real part of the input generates the real part of the response, and the imaginary part of the input generates the imaginary part of the response.

If the input is $x[n] = x_r[n] + jx_i[n]$, where $x_r[n]$ and $x_i[n]$ are the real and imaginary parts of $x[n]$, then for real $h[n]$,

$$y[n] = h[n] * (x_r[n] + jx_i[n]) = h[n] * x_r[n] + jh[n] * x_i[n] = y_r[n] + jy_i[n],$$

where $y_r[n]$ and $y_i[n]$ are the real and the imaginary parts of $y[n]$. Using the right-directed arrow notation to indicate an input and output pair, this same result is expressed as follows. If

$$x[n] = x_r[n] + jx_i[n] \implies y[n] = y_r[n] + jy_i[n],$$

then

$$x_r[n] \implies y_r[n] \quad \text{and} \quad x_i[n] \implies y_i[n]. \quad (5.24)$$

From this perspective, the response to complex inputs is just a special case of the response to multiple inputs.

▷ Drill 5.6 (Convolution Using Properties and Tables)

Use Table 5.2 and the properties of the convolution sum to compute the following:

- (a) $(0.8)^{n+1}u[n] * u[n]$
- (b) $n3^{-n}u[n] * (0.2)^n u[n - 1]$
- (c) $(e^{-n}u[n] - j\delta[n + 1]) * 2^{-n}u[n]$

△

5.5.2 Graphical Procedure for the Convolution Sum

Depending on the signals involved, it can be difficult to solve the convolution sum using a purely analytic approach. Followed systematically, a graphical approach clarifies convolution through visual means. Recall that the convolution of signals $x[n]$ and $h[n]$ is given by

$$y[n] = x[n] * h[n] = \sum_{m=-\infty}^{\infty} x[m]h[n-m].$$

Following this expression, the graphical method first plots $x[m]$ and $h[n-m]$ as functions of m (not n). For each value of n , the product $x[m]h[n-m]$ is formed and then summed over m to obtain $y[n]$. Procedurally, the convolution operation can be performed as follows:

1. Plot $x[m]$ and $h[n-m]$ as functions of m . To obtain $h[n-m]$, first invert $h[m]$ about the vertical axis ($m=0$) to obtain $h[-m]$. Next, shift $h[-m]$ by n to obtain $h[n-m]$.
2. Starting with n large and negative, multiply $x[n]$ and $h[n-m]$, and add all the products, either manually or analytically, to obtain $y[n]$.
3. Repeat the procedure for each value of n over the range $-\infty$ to ∞ . Observe and exploit structures that enable simultaneous computation over entire regions of n .

Since $x[n] * h[n] = h[n] * x[n]$, the roles of $x[n]$ and $h[n]$ are easily reversed. In other words, we can invert and shift either $x[n]$ or $h[n]$, whichever is most convenient.

We shall demonstrate by an example the graphical procedure for finding the convolution sum. Although both the functions in this example are causal, the procedure is applicable to the general case.

▷ **Example 5.14 (Graphical Procedure for the Convolution Sum)**

Using the graphical approach, repeat Ex. 5.12 and determine the DT convolution $y[n] = x[n] * h[n]$ for

$$x[n] = (0.8)^n u[n] \quad \text{and} \quad h[n] = (0.3)^n u[n].$$

To begin, we plot both $x[m]$ and $h[m]$, as shown in Figs. 5.5a and 5.5b. These plots are identical to plots of $x[n]$ and $h[n]$ except that m replaces n . In this case, there is no particular difference in complexity between the two signals, and we arbitrarily choose to follow the procedure using $x[m]$ and $h[n-m]$. To obtain $h[n-m]$, we first reflect $h[m]$ to obtain $h[-m]$, as shown in Fig. 5.5c. Next, we shift $h[-m]$ by n to obtain $h[n-m]$, as shown in Fig. 5.5d. Expressed mathematically, we see that

$$x[m] = (0.8)^m u[m] \quad \text{and} \quad h[n-m] = (0.3)^{n-m} u[n-m].$$

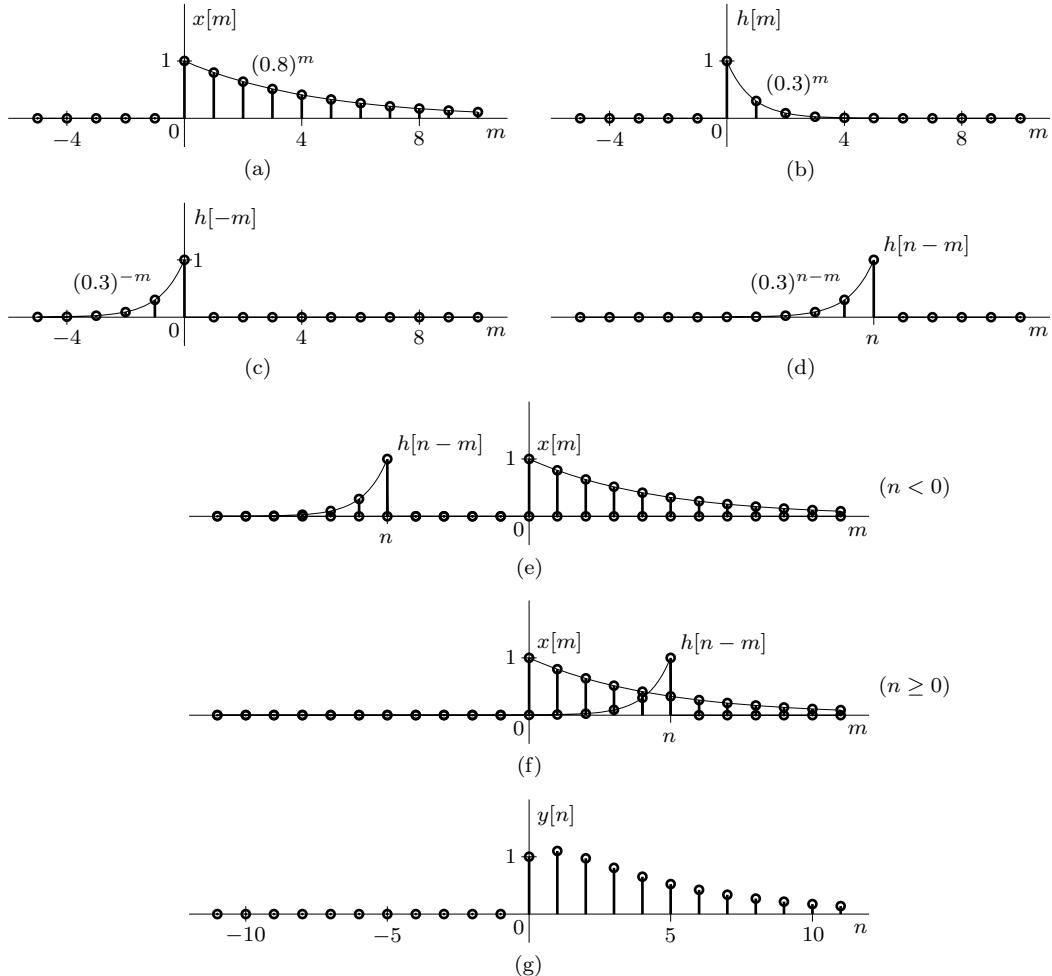


Figure 5.5: Graphical procedure for the convolution $y[n] = x[n] * h[n]$.

For $n < 0$, there is no overlap between $x[m]$ and $h[n - m]$, as shown in Fig. 5.5e. Thus,

$$y[n] = 0 \quad \text{for } n < 0.$$

Figure 5.5f shows the general situation for $n \geq 0$. The two functions $x[m]$ and $h[n - m]$ overlap over the interval $0 \leq m \leq n$. Therefore, for $n \geq 0$,

$$\begin{aligned} y[n] &= \sum_{m=0}^n x[m]h[n - m] \\ &= \sum_{m=0}^n (0.8)^m (0.3)^{n-m} \\ &= (0.3)^n \sum_{m=0}^n \left(\frac{0.8}{0.3}\right)^m. \end{aligned}$$

Using entry 1 of Table 5.1,

$$y[n] = 2 [(0.8)^{n+1} - (0.3)^{n+1}] \quad \text{for } n \geq 0.$$

Combining the results for $n < 0$ and $n \geq 0$ yields

$$y[n] = 2 [(0.8)^{n+1} - (0.3)^{n+1}] u[n].$$

This result, shown in Fig. 5.5g, agrees with the earlier result of Ex. 5.12.

Example 5.14 \triangleleft

▷ Drill 5.7 (Graphical Procedure for the Convolution Sum)

For $x[n] = (0.8)^n u[n - 1]$ and $h[n] = u[n + 3]$, determine $x[n] * h[n]$ using the graphical procedure, and then plot the result.

\triangleleft

An Alternative Graphical Procedure: The Sliding-Tape Method

It is not necessary to plot signals in the conventional way to enjoy the benefits of the graphical procedure. The sliding-tape method is an alternate form of the graphical procedure demonstrated in Fig. 5.5. The only difference is that this method presents signals as sequences of numbers on tapes rather than graphical plots. This algorithm is particularly convenient when the sequences $x[n]$ and $h[n]$ are short or when they are available only in graphical form. Further, the sliding-tape method can facilitate an array representation of convolution, as presented in Prob. 5.5-23. The following example demonstrates the sliding-tape procedure.

▷ Example 5.15 (Sliding-Tape Procedure for the Convolution Sum)

Using the sliding-tape approach, determine the DT convolution $y[n] = x[n] * h[n]$ using the signals $x[n] = n(u[n + 2] - u[n - 4])$ and $h[n] = u[n]$, which are both shown in Fig. 5.6.

We begin by representing $x[m]$ and $h[m]$ as number sequences on tapes, as shown in Figs. 5.7a and 5.7b. Since the variable m does not appear on these tapes, it is important to identify a reference point (usually $m = 0$, shown shaded). Without such a reference point, the result $y[n]$ may incorrectly shift to one side or the other.

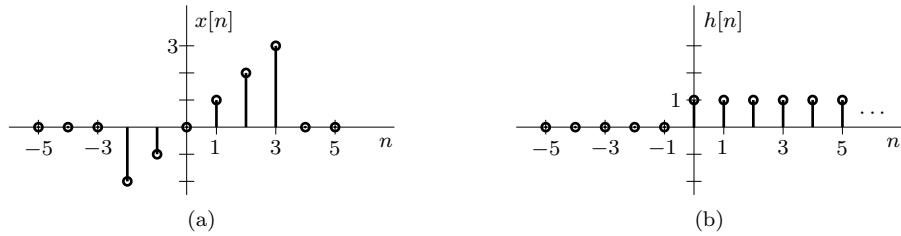


Figure 5.6: Signals $x[n] = n(u[n+2] - u[n-4])$ and $h[n] = u[n]$.

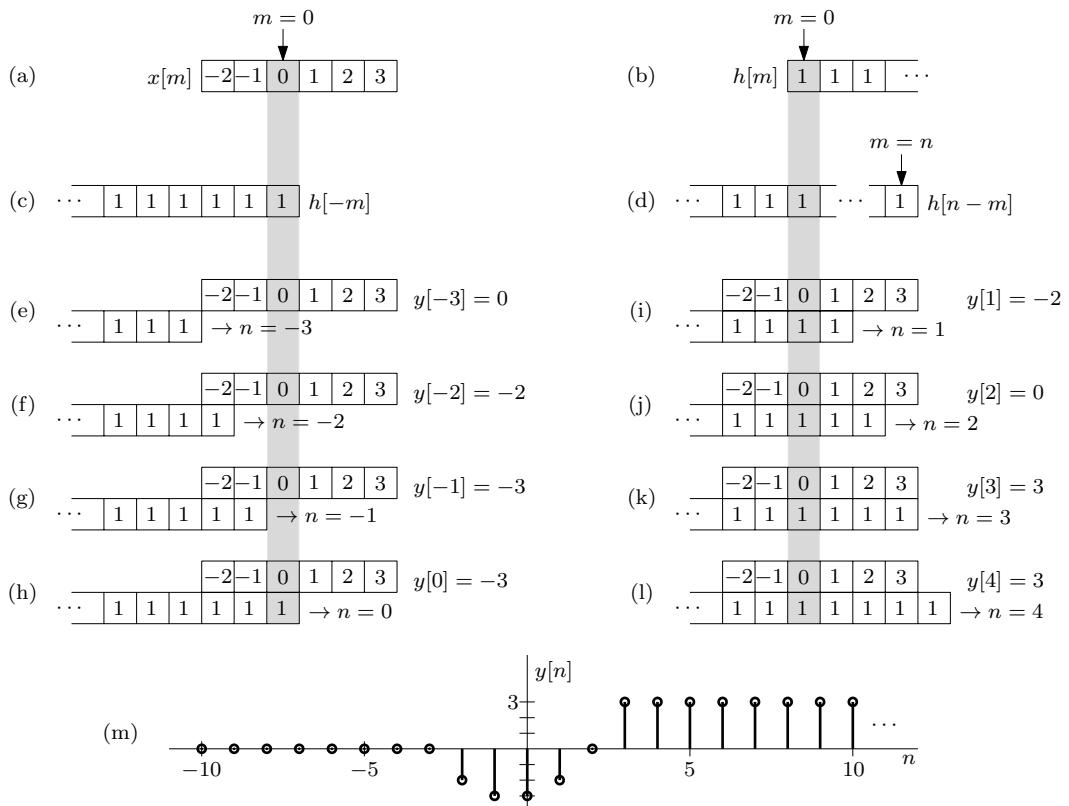


Figure 5.7: Sliding-tape method for the convolution $y[n] = x[n] * h[n]$.

The $h[-m]$ tape of Fig. 5.7c is obtained by inverting the $h[m]$ tape about the origin ($m = 0$). Shifting the inverted tape by n slots yields $h[n-m]$, as shown in Fig. 5.7d. Figures 5.7a through 5.7d are analogous to Figs. 5.5a through 5.5d of the graphical approach.

To obtain $y[n] = x[n] * h[n]$, we slide the $h[n - m]$ tape along the $x[m]$ tape, multiply the values of adjacent slots, and add all the products. For $n \leq -3$, the $x[m]$ and $h[n - m]$ tapes do not overlap, and $y[n]$ is therefore 0. Figure 5.7e illustrates the $n = -3$ case, just before the $h[n - m]$ tape begins to overlap the $x[m]$ tape. At $n = -2$, the two tapes begin to overlap (Fig. 5.7f). In this case,

$$y[-2] = (-2 \times 1) = -2.$$

At $n = -1$ (Fig. 5.7g), two slots overlap, and

$$y[-1] = (-2 \times 1) + (-1 \times 1) = -3.$$

Continuing this process (Figs. 5.7h through 5.7k),

$$\begin{aligned}y[0] &= (-2 \times 1) + (-1 \times 1) + (0 \times 1) = -3, \\y[1] &= (-2 \times 1) + (-1 \times 1) + (0 \times 1) + (1 \times 1) = -2, \\y[2] &= (-2 \times 1) + (-1 \times 1) + (0 \times 1) + (1 \times 1) + (2 \times 1) = 0, \\ \text{and } y[3] &= (-2 \times 1) + (-1 \times 1) + (0 \times 1) + (1 \times 1) + (2 \times 1) + (3 \times 1) = 3.\end{aligned}$$

As Fig. 5.7l demonstrates, the $x[m]$ tape is fully covered by the $h[n-m]$ for $n \geq 4$. Thus, $y[n] = 3$ for $n \geq 4$. Viewed in sequence, Figs. 5.7e through 5.7l show $h[n-m]$ sliding forward, one slot at a time, to produce $y[n]$, one value of n at a time. Figure 5.7m shows a plot of $y[n]$.

Example 5.15 \triangleleft

▷ Drill 5.8 (Sliding-Tape Procedure for the Convolution Sum)

For $x[n] = (3 - |n|)(u[n+3] - u[n-4])$ and $h[n] = u[-n+4] - u[-n-2]$, use the sliding-tape procedure of Ex. 5.15 to determine and plot $y[n] = x[n] * h[n]$. Verify the convolution width property.

\triangleleft

Computer-Based Convolution

Various computer software packages support discrete-time convolution. Some are capable of generating closed-form analytic solutions, and others, like MATLAB, simply convolve particular sequences. While convenient from a computational viewpoint, computer-based convolution generally fails to give a proper understanding of the convolution mechanism. As the next example shows, computer-based solutions are particularly effective when convolving two finite-length sequences, although certain other cases are also possible.

▷ Example 5.16 (Convolution with MATLAB)

Consider the signals $x[n] = n(u[n+2] - u[n-4])$ and $h[n] = u[n]$, which are both shown in Fig. 5.6. Use MATLAB to compute $x[n] * x[n]$ and $x[n] * h[n]$.

The signal $x[n]$, whose region of support is $-2 \leq n \leq 3$, is a finite-duration signal of length $L_x = 6$. To compute $x[n] * x[n]$, we define a length-6 vector \mathbf{x} and then use the `conv` command, which convolves two finite-duration DT signals.

```
01 x = [-2,-1,0,1,2,3]; conv(x,x)
ans = 4 4 1 -4 -10 -16 -5 4 10 12 9
```

The length-11 result accurately represents the shape of $x[n] * x[n]$ and is readily verified using the sliding-tape method. What is missing, however, is the region of support for this signal, a detail that MATLAB's `conv` command does not provide. Fortunately, the starting point of the result is easy to determine: simply add the starting points of each signal being convolved (see Prob. 5.5-18). Since $x[n]$ starts at $n = -2$, $x[n] * x[n]$ starts at $n = -4$. Figure 5.8a shows the result.

```
02 n = (-4:-4+length(x)-1+length(x)-1); stem(n,conv(x,x));
```

The convolution $x[n]*h[n]$ involves an infinite-duration signal $h[n] = u[n]$. In general, MATLAB's `conv` command cannot properly convolve infinite-duration signals. This is not too surprising since computers themselves cannot store the infinite-length vector needed to represent such signals. For special cases, such as the convolution of right-sided signals, `conv` can correctly compute a portion of the convolution. To see how, let us represent $h[n]$ with its first 10 values and then plot the resulting convolution. Remember, since $x[n]$ starts at $n = -2$ and $h[n]$ starts at $n = 0$, the result starts at $n = -2$.

```
03 h = ones(1,10); n = (-2:-2+length(x)-1+length(h)-1); stem(n,conv(x,h));
```

The result is shown in Fig. 5.8b. Compared with the true result of Fig. 5.7m, we see that the first 10 values are correct, while the remaining values ($n \geq 8$, shown shaded) are not. The accuracy of the result is limited by the 10-value accuracy with which we represent $h[n]$.

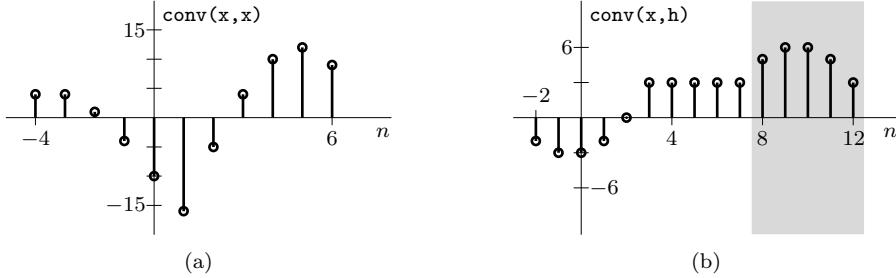


Figure 5.8: Computer convolutions involving $x[n] = n(u[n + 2] - u[n - 4])$ and $h[n] = u[n]$.

Example 5.16 ◀

Polynomial Multiplication by Convolution

One useful application of discrete-time convolution is the multiplication of constant-coefficient polynomials. Let $A(x)$ and $B(x)$ be K th- and L th-order polynomials defined as

$$A(x) = a_0x^K + a_1x^{K-1} + \cdots + a_{K-1}x + a_K$$

and $B(x) = b_0x^L + b_1x^{L-1} + \cdots + b_{L-1}x + b_L.$

The coefficients of the product $A(x)B(x)$ are obtained by convolving the respective coefficient vectors,

$$[a_0, a_1, \dots, a_{K-1}, a_K] * [b_0, b_1, \dots, b_{L-1}, b_L].$$

We demonstrate the technique with an example.

▷ Example 5.17 (Polynomial Multiplication by Convolution)

Expand $(x^2 + 2x + 1)(3x + 4)$ by hand and by using DT convolution. What is the expansion of $(x + 2 + x^{-1})(3x^4 + 4x^3)$?

By hand calculation, the desired polynomial expansion is

$$\begin{aligned} (x^2 + 2x + 1)(3x + 4) &= 3x^3 + 6x^2 + 3x + 4x^2 + 8x + 4 \\ &= 3x^3 + 10x^2 + 11x + 4. \end{aligned}$$

Next, we use MATLAB to compute the convolution $[1, 2, 1] * [3, 4]$.

```
01 conv([1 2 1],[3 4])
ans = 3 10 11 4
```

The vector of coefficients produced through convolution clearly matches the coefficients of the original polynomial expansion, which confirms that $(x^2 + 2x + 1)(3x + 4) = 3x^3 + 10x^2 + 11x + 4$.

The expansion of $(x + 2 + x^{-1})(3x^4 + 4x^3)$ can be obtained by the same convolution. To see how, notice that $(x + 2 + x^{-1})(3x^4 + 4x^3) = x^2(x^2 + 2x + 1)(3x + 4)$. Thus,

$$\begin{aligned}(x + 2 + x^{-1})(3x^4 + 4x^3) &= x^2(3x^3 + 10x^2 + 11x + 4) \\ &= 3x^5 + 10x^4 + 11x^3 + 4x^2.\end{aligned}$$

Since $(x^2 + 2x + 1)(3x + 4)$ and $(x + 2 + x^{-1})(3x^4 + 4x^3)$ share the same coefficient vectors, so do their respective expansions; it is only the powers of the polynomials that differ.

Example 5.17 \triangleleft

▷ **Drill 5.9 (Polynomial Multiplication by Convolution)**

Use DT convolution to expand $(x^4 + 2x^2 + 3)^2(4x - 2 + 3x^{-1})$.

\triangleleft

The Convolution Sum Is a Nonrecursive Representation

Earlier we discussed the recursive solution of a difference equation. Convolution is another method of solving a linear difference equation. The convolution sum in Eq. (5.22) is really a representation of the system in a nonrecursive form. The system response represented by the convolution sum in Eq. (5.22) does not use recursion, that is, previous output values, to compute the current output value. The output is given by adding the weighted input terms only. The weighting function is the impulse response.[†]

As an example, consider the system of Ex. 5.4, which has a nonrecursive representation of

$$y[n] = x[n] + x[n - 1] + x[n - 2] + x[n - 3] + x[n - 4]. \quad (5.25)$$

Example 5.4 shows that this system can also be represented in recursive form as

$$y[n] - y[n - 1] = x[n] - x[n - 5].$$

The impulse response $h[n]$ for this system can be found either from the nonrecursive form or from the recursive form. However, finding $h[n]$ from the nonrecursive form is much simpler, as seen from Eq. (5.21) and also found in Ex. 5.11 as

$$h[n] = \delta[n] + \delta[n - 1] + \delta[n - 2] + \delta[n - 3] + \delta[n - 4].$$

The system response to an arbitrary input $x[n]$ is given by

$$\begin{aligned}y[n] &= x[n] * h[n] \\ &= x[n] * (\delta[n] + \delta[n - 1] + \delta[n - 2] + \delta[n - 3] + \delta[n - 4] + \delta[n - 5]) \\ &= x[n] + x[n - 1] + x[n - 2] + x[n - 3] + x[n - 4].\end{aligned}$$

But this is precisely the nonrecursive form of system representation, as seen from Eq. (5.25). To reiterate, the convolution sum may be viewed as a nonrecursive representation of the system.

5.5.3 Interconnected Systems

It is common practice to build complex systems from combinations of simpler systems. Two forms of interconnection are common: parallel connections (Fig. 5.9a) and cascade connections (Fig. 5.9b). Let us investigate how these interconnections impact overall system function.

[†]The weighting function is the inverted and shifted impulse response, to be exact.

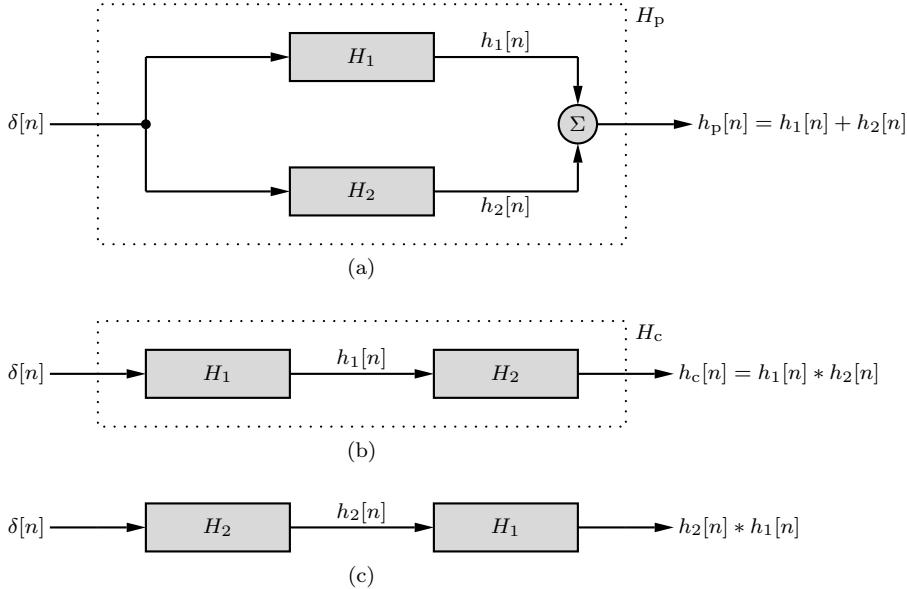


Figure 5.9: System interconnections: (a) parallel, (b) cascade, and (c) reverse-order cascade.

To begin, consider two LTID systems H_1 and H_2 with impulse responses $h_1[n]$ and $h_2[n]$, respectively. Further, let us assume that interconnecting these systems, as shown in Fig. 5.9, does not load them. This means that the impulse response of either H_1 or H_2 remains unchanged whether observed when these systems are unconnected or when they are interconnected.

Figure 5.9a shows these two systems connected in parallel. Applying an impulse as the input, each system responds with its own impulse response, which are then added. Thus, the composite parallel system H_p has an impulse response $h_p[n] = h_1[n] + h_2[n]$. This result also follows from the distributive property of the convolution sum.

If systems H_1 and H_2 are connected in cascade, as shown in Fig. 5.9b, we see that the impulse response of the composite cascade system H_c is $h_c[n] = h_1[n] * h_2[n]$. This result also follows from the associate property of the convolution sum. Moreover, because of the commutative property, $h_1[n] * h_2[n] = h_2[n] * h_1[n]$, and the systems commute. In other words, the order of the systems in the cascade is not important, at least theoretically.[†] Thus, the cascade of H_2 followed by H_1 (Fig. 5.9c) behaves identically to the cascade of H_1 followed by H_2 (Fig. 5.9b).

Inverse Systems

Suppose that an LTID system with impulse response $h[n]$ is connected in cascade with its inverse, which has an impulse response $h_i[n]$. The impulse response of the cascade system is $h[n] * h_i[n]$. We also know that the cascade of a system with its inverse is an identity system whose output is the same as the input. In other words, the unit impulse response of the cascade is a unit impulse $\delta[n]$. Consequently,

$$h[n] * h_i[n] = \delta[n]. \quad (5.26)$$

Because order is unimportant to convolution, we also see that $h[n]$ is the inverse of $h_i[n]$. That is, $h[n]$ and $h_i[n]$ are inverses of each other.

▷ Example 5.18 (Inverse Systems)

Show that the inverse of a discrete-time accumulator is a first-order backward difference system.

[†]Because of physical limitations, sensitivities, and imperfections such as quantization, cascade order can affect performance in real-world implementations. See Ch. 8.

A discrete-time accumulator is analogous to a continuous-time integrator, and a backward difference system is analogous to a differentiator. Intuitively, it is little surprise that the two systems are therefore inverses.

As seen in Ex. 4.7, an accumulator system is specified by

$$y[n] = \sum_{k=-\infty}^n x[k].$$

It is easy to see from this equation that the impulse response of an accumulator is given by

$$h[n] = \sum_{k=-\infty}^n \delta[k] = u[n].$$

As seen in Ex. 4.9, a first-order backward difference system is specified by

$$y[n] = x[n] - x[n-1].$$

Designating this as the inverse system, the impulse response is

$$h_i[n] = \delta[n] - \delta[n-1].$$

The impulse response of the two systems in cascade is

$$h[n] * h_i[n] = u[n] * (\delta[n] - \delta[n-1]) = u[n] - u[n-1] = \delta[n].$$

Clearly, a backward difference system is the inverse of an accumulator (and vice versa).

Example 5.18 ◀

System Response to $\sum_{k=-\infty}^n x[k]$

Figure 5.10a shows a cascade of two LTID systems: a system H with impulse response $h[n]$, followed by an accumulator. Figure 5.10b shows a cascade of the same two systems in reverse order: an accumulator followed by H . In Fig. 5.10a, if the input $x[n]$ to H results in the output $y[n]$, then the output of the accumulator is $\sum_{k=-\infty}^n y[k]$. In Fig. 5.10b, the output of the accumulator is the sum $\sum_{k=-\infty}^n x[k]$, and using the linearity property, the output of system H is $\sum_{k=-\infty}^n y[k]$, which is identical to the output in Fig. 5.10a. Hence, it follows that

$$\text{if } x[n] \implies y[n], \quad \text{then } \sum_{k=-\infty}^n x[k] \implies \sum_{k=-\infty}^n y[k]. \quad (5.27)$$

If we let $x[n] = \delta[n]$ in Eq. (5.27), then $y[n] = h[n]$. Now using the fact that $\sum_{k=-\infty}^n \delta[k] = u[n]$, we obtain $s[n]$, the unit step response of an LTID system with impulse response $h[n]$, given by

$$s[n] = \sum_{k=-\infty}^n h[k]. \quad (5.28)$$

The reader can readily prove the inverse relationship,

$$h[n] = s[n] - s[n-1]. \quad (5.29)$$

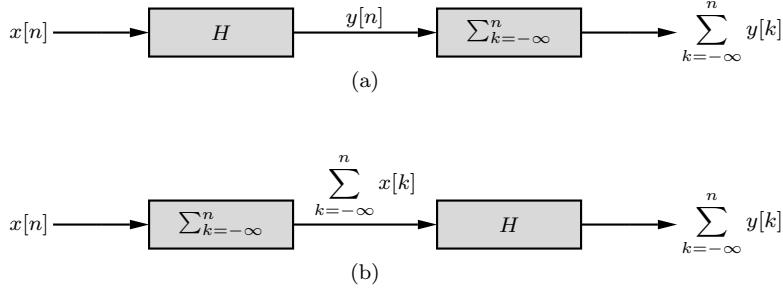


Figure 5.10: Cascade connections with an accumulator system.

5.5.4 LTID System Response to an Everlasting Exponential z^n

Everlasting exponentials are very special functions for linear, time-invariant systems. In Sec. 1.6.1, we showed that there exists one signal for which the response of an LTIC system is the same as the input within a multiplicative constant. The response of an LTIC system to an everlasting exponential input e^{st} is $H(s)e^{st}$, where $H(s)$ is the system transfer function. We now show that for an LTID system, the same role is played by an everlasting exponential z^n . The system response $y[n]$ in this case is given by

$$\begin{aligned} y[n] &= h[n] * z^n \\ &= \sum_{m=-\infty}^{\infty} h[m]z^{n-m} \\ &= z^n \sum_{m=-\infty}^{\infty} h[m]z^{-m}. \end{aligned}$$

For causal $h[n]$, the right-hand sum limits would range from 0 to ∞ . In any case, the sum $\sum_{m=-\infty}^{\infty} h[m]z^{-m}$ is a function of z and not n . Let us denote this sum, when it converges, by $H(z)$. Thus,

$$y[n] = H(z)z^n = H(z)x[n]\Big|_{x[n]=z^n}, \quad (5.30)$$

where

$$H(z) = \sum_{m=-\infty}^{\infty} h[m]z^{-m}. \quad (5.31)$$

Equation (5.30) is valid only for those values of z for which the sum on the right-hand side of Eq. (5.31) exists (converges). For a given z , notice that $H(z)$ is a constant. Thus, the input and the output are the same (within a multiplicative constant) when the input is an everlasting exponential z^n .

The function $H(z)$ is called the *transfer function* of the system, and it is a function of the complex variable z . A rearrangement of Eq. (5.30) leads to an alternative definition of the transfer function as

$$H(z) = \frac{\text{output signal}}{\text{input signal}}\Big|_{\substack{\text{input} = \text{everlasting exponential}}} = \frac{y[n]}{x[n]}\Big|_{x[n]=z^n}. \quad (5.32)$$

The transfer function is defined for, and is meaningful to, LTID systems only. It does not exist for nonlinear or time-varying systems in general.

For a system specified by Eq. (5.8), the transfer function is given by

$$H(z) = \frac{B(z)}{A(z)}. \quad (5.33)$$

This follows readily by considering an everlasting input $x[n] = z^n$. According to Eq. (5.32), the output is $y[n] = H(z)z^n$. Substitution of this $x[n]$ and $y[n]$ into Eq. (5.8) yields

$$A(E)\{H(z)z^n\} = H(z) A(E)\{z^n\} = B(E)\{z^n\}.$$

Moreover,

$$E^k\{z^n\} = z^{n+k} = z^k z^n.$$

Hence,

$$A(E)\{z^n\} = A(z)z^n \quad \text{and} \quad B(E)\{z^n\} = B(z)z^n.$$

Consequently,

$$H(z)A(z)z^n = B(z)z^n,$$

and

$$H(z) = \frac{B(z)}{A(z)}.$$

We stress again that in this discussion we are talking about the everlasting exponential, which starts at $n = -\infty$, not the causal exponential $z^n u[n]$, which starts at $n = 0$.

LTID System Response to an Everlasting Exponential $e^{j\Omega n}$

The everlasting exponential $e^{j\Omega n}$ is just a special case of the everlasting exponential z^n . Consequently, using Eqs. (5.30) and (5.31), the LTID system response to an everlasting exponential $e^{j\Omega n}$ is

$$y[n] = H(e^{j\Omega})e^{j\Omega n} = H(e^{j\Omega})x[n]\Big|_{x[n]=e^{j\Omega n}}, \quad (5.34)$$

where

$$H(e^{j\Omega}) = \sum_{m=-\infty}^{\infty} h[m]e^{-j\Omega m}. \quad (5.35)$$

Later in Ch. 6 we shall see that Eq. (5.35) is the discrete-time Fourier transform (DTFT) of the impulse response and that $H(e^{j\Omega})$ is the *frequency response* of the system.

▷ Drill 5.10 (Transfer Functions of LTID Systems)

Determine the transfer function of the following systems:

- (a) a digital differentiator given by $y[n+1] = \frac{1}{T}(x[n+1] - x[n])$
- (b) a unit-delay system given by $y[n] = x[n-1]$

▷

5.6 Total Response

From the decomposition property of Eq. 4.55, we know that the total response $y[n]$ of an LTID system is the sum of the zero-input response $y_{\text{zir}}[n]$ and the zero-state response $y_{\text{zsr}}[n]$. The zero-input response is a linear combination of the characteristic modes, which are determined from the characteristics roots of the system equation. From the system equation, we also determine the impulse response $h[n]$. Knowing $h[n]$ and the input $x[n]$, we find the zero-state response as the convolution of $x[n]$ and $h[n]$. For a difference equation such as Eq. (5.1) with unique roots γ_k , the total response is thus

$$\underbrace{y[n]}_{\text{total response}} = \underbrace{\sum_{k=1}^K c_k \gamma_k^n}_{\text{zero-input response}} + \underbrace{x[n] * h[n]}_{\text{zero-state response}}.$$

The constants c_1, c_2, \dots, c_K in the zero-input response are determined from the K initial conditions of the system.

▷ **Example 5.19 (Total Response of an LTID System)**

Using the initial conditions $y[-1] = 0$ and $y[-2] = \frac{25}{4}$ and the input $x[n] = 4^{-n}u[n]$, determine and sketch the total response $y[n]$ of an LTID system described by the equation

$$y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2].$$

From Ex. 5.5, we know the zero-input response using $y[-1] = 0$ and $y[-2] = \frac{25}{4}$ is

$$y_{\text{zir}}[n] = 0.2(-0.2)^n + 0.8(0.8)^n.$$

From Ex. 5.13, we know the zero-state response to input $x[n] = 4^{-n}u[n]$ is

$$y_{\text{zsir}}[n] = [-1.26(4)^{-n} + 0.444(-0.2)^n + 5.82(0.8)^n] u[n].$$

Adding the ZIR and ZSR together, the total response is

$$\begin{aligned} y[n] &= y_{\text{zsir}}[n] + y_{\text{zir}}[n] \\ &= \underbrace{[-1.26(4)^{-n} + 0.444(-0.2)^n + 5.82(0.8)^n]}_{\text{zero-state response}} u[n] + \underbrace{0.2(-0.2)^n + 0.8(0.8)^n}_{\text{zero-input response}}. \end{aligned}$$

For $n \geq 0$, this simplifies to

$$y[n] = -1.26(4)^{-n} + 0.644(-0.2)^n + 6.62(0.8)^n.$$

The total response is depicted in Fig. 5.11.

```
01 y = @(n) (-1.26*(4).^( -n)+0.644*(-0.2).^n+6.62*(0.8).^n).* (n>=0);
02 n = (0:20); stem (n,y(n));
```

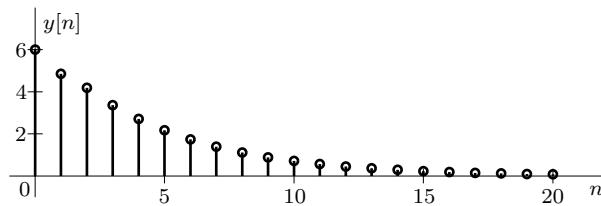


Figure 5.11: Total response of $y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2]$ using input $x[n] = 4^{-n}u[n]$ and initial conditions $y[-1] = 0$ and $y[-2] = \frac{25}{4}$.

Example 5.19 ◀

5.7 System Stability

Chapter 4 introduces the concepts of external (BIBO) stability and internal (asymptotic) stability for discrete-time systems. In this section, we introduce simple criteria to assess both types of stability.

5.7.1 External (BIBO) Stability

External stability is based on how a system responds to external inputs. For LTID systems, the system response to an input $x[n]$ is given by the zero-state response

$$\begin{aligned} y[n] &= h[n] * x[n] \\ &= \sum_{m=-\infty}^{\infty} h[m]x[n-m]. \end{aligned}$$

The magnitude (bound) of the output is thus

$$\begin{aligned} |y[n]| &= \left| \sum_{m=-\infty}^{\infty} h[m]x[n-m] \right| \\ &\leq \sum_{m=-\infty}^{\infty} |h[m]| |x[n-m]|. \end{aligned}$$

For a system that is BIBO stable, any bounded input produces a bounded output. If $x[n]$ is bounded, then $|x[n-m]| < K_x < \infty$, and

$$|y[n]| \leq K_x \sum_{m=-\infty}^{\infty} |h[m]|.$$

Clearly, the output is bounded if the summation on the right-hand side is bounded or, equivalently, if

$$\sum_{n=-\infty}^{\infty} |h[n]| < K_y < \infty. \quad (5.36)$$

Therefore, an LTID system is BIBO stable if its impulse response $h[n]$ is absolutely summable. Otherwise, it is unstable. This is a sufficient condition for BIBO stability. We can show that this is also a necessary condition (see Prob. 5.7-5).

5.7.2 Internal (Asymptotic) Stability

For LTID systems, as in the case of LTIC systems, internal stability is defined in terms of the zero-input response of the system. Also known as *asymptotic stability*, *zero-input stability*, and *stability in the Lyapunov sense*, internal stability reflects the internal structure of a system and is more informative and powerful than external stability.

For an LTID system specified by a difference equation in the form of Eq. (5.7) (or Eq. (5.8)), the zero-input response consists of the characteristic modes of the system. It is the behavior of these modes that determines the system's internal stability. The mode corresponding to a non-repeated characteristic root γ is γ^n . To be more general, let γ be complex so that

$$\gamma = |\gamma|e^{j\beta} \quad \text{and} \quad \gamma^n = |\gamma|^n e^{j\beta n}.$$

Since the magnitude of $e^{j\beta n}$ is always unity regardless of the value of n , the magnitude of γ^n is $|\gamma|^n$. Therefore,

if $|\gamma| < 1$, then $\gamma^n \rightarrow 0$ as $n \rightarrow \infty$;

if $|\gamma| > 1$, then $\gamma^n \rightarrow \infty$ as $n \rightarrow \infty$;

and if $|\gamma| = 1$, then $|\gamma^n| = 1$ for all n .

These results can be grasped more effectively in terms of the location of characteristic roots in the complex plane. Figure 5.12 shows a circle of unit radius centered at the origin in a complex plane.

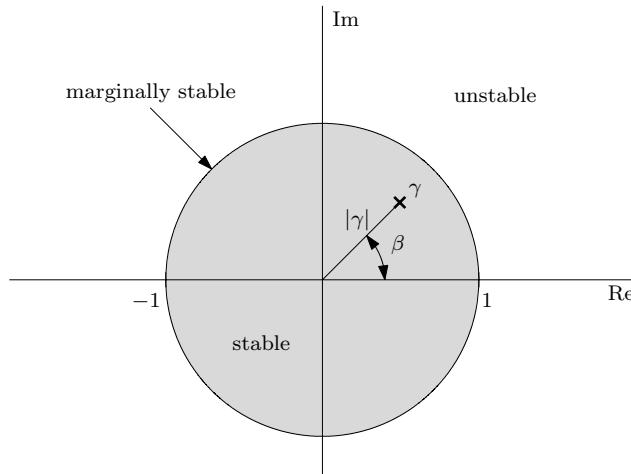


Figure 5.12: Characteristic root locations and system stability.

Assuming a causal system, our discussion shows that if all characteristic roots of the system lie inside the unit circle, $|\gamma_k| < 1$ for all k , then the system is asymptotically stable. On the other hand, if even one characteristic root lies outside the unit circle, then the system is unstable. If none of the characteristic roots lies outside the unit circle, but some simple (non-repeated) roots lie on the circle itself, then the system is marginally stable. If repeated roots lie on the unit circle, then the system is unstable. The reason is that the characteristic mode for a repeated root is of the form $n^{r-1}\gamma^n$, and if $|\gamma| = 1$, then $|n^{r-1}\gamma^n| = n^{r-1} \rightarrow \infty$ as $n \rightarrow \infty$. Note, however, that repeated roots inside the unit circle do not cause instability. Figure 5.13 shows a representative selection of characteristic roots and their corresponding characteristic modes.

Although the development of discrete-time systems is parallel to that of continuous-time systems, we notice that the boundary demarcating stability in DT systems is the unit circle rather than the ω -axis for CT systems. The reason for this difference lies in the form of the characteristic modes. In continuous-time systems we typically express characteristic modes as $e^{\lambda_k t}$. In discrete-time systems we choose (for convenience) the form γ_k^n . Had we chosen this form to be $e^{\lambda_k n}$, where $\gamma_k = e^{\lambda_k}$, then, plotting λ_k , the imaginary axis again would demarcate stability and instability.

To summarize:

1. A causal LTID system is asymptotically stable if and only if all the characteristic roots are inside the unit circle. The roots may be simple or repeated.
2. A causal LTID system is marginally stable if and only if there are no roots outside the unit circle, and there are non-repeated roots on the unit circle.
3. A causal LTID system is unstable if and only if at least one root is outside the unit circle or there are repeated roots on the unit circle or both.

Relationship between BIBO Stability and Asymptotic Stability

For LTID systems, the relation between the two types of stabilities is similar to that for LTIC systems. Consider a causal system specified by Eq. (5.7). We can determine internal stability from the locations of the system's characteristic roots γ_k , and we can determine external stability based on whether or not the impulse response $h[n]$ is absolutely summable. However, since $h[n]$ is comprised of terms like $c_k \gamma_k^n u[n]$, the characteristic roots γ_k also directly impact external stability.

If all the characteristic roots of a system are inside the unit circle (internally stable), then the

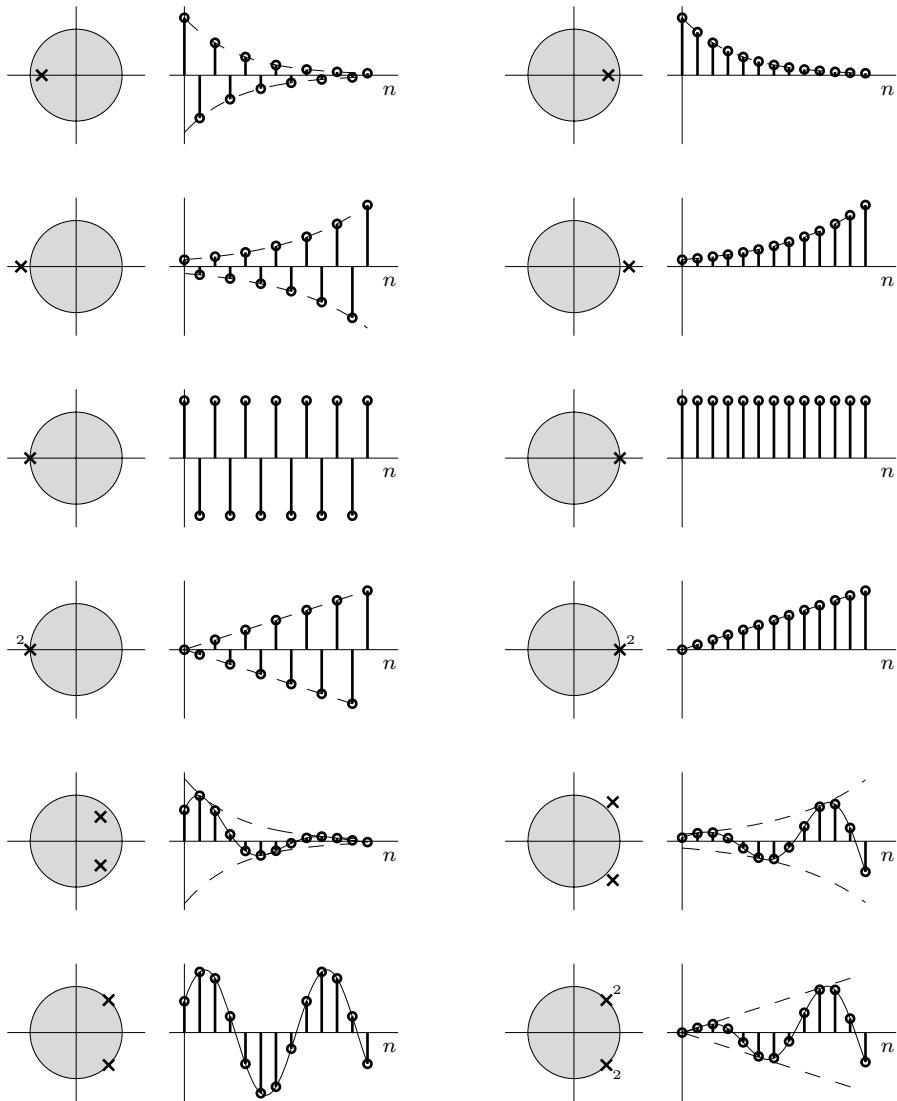


Figure 5.13: Various characteristic roots and corresponding characteristic modes.

terms $c_k \gamma_k^n u[n]$ converge, $h[n]$ is absolutely summable, and the system is BIBO stable.[†] However, a root outside the unit circle (internally unstable) does not necessarily make a system BIBO unstable; the coefficient c_k weighting this mode in the impulse response might be 0. Fortunately, this situation is quite uncommon, and most asymptotically unstable systems are also BIBO unstable. A BIBO unstable system is never asymptotically stable.

In summary, although an asymptotically stable system is necessarily BIBO stable, the converse

[†]If a characteristic root γ_k is inside the unit circle, then the corresponding mode γ_k^n is absolutely summable for $n \geq 0$. This conclusion follows for $|\gamma_k| < 1$ from the fact that (see entry 1 of Table 5.1)

$$\sum_{n=-\infty}^{\infty} |\gamma_k^n| u[n] = \sum_{n=0}^{\infty} |\gamma_k|^n = \frac{1}{1 - |\gamma_k|}.$$

In contrast, if γ_k lies on or outside the unit circle, then γ_k^n is not absolutely summable. These conclusions are valid also for modes of the form $n^r \gamma_k^n$.

is not always true.[†] The stability picture portrayed by the external description is inferior to the internal description of stability. As the following example shows, BIBO (external) stability does not guarantee internal (asymptotic) stability.

▷ **Example 5.20 (BIBO Stability Does Not Guarantee Asymptotic Stability)**

Two first-order LTID systems H_1 and H_2 are connected in a non-loading cascade, as shown in Fig. 5.14. The impulse responses of these systems are $h_1[n]$ and $h_2[n]$, respectively, given by

$$h_1[n] = 4\delta[n] - 3(0.5)^n u[n] \quad \text{and} \quad h_2[n] = 2^n u[n].$$

Determine the internal and external stability of the composite cascade system H_c .

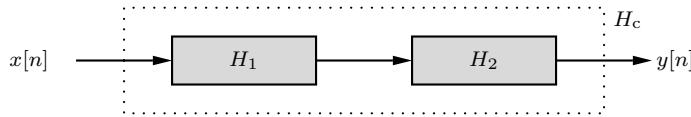


Figure 5.14: A cascade connection of two systems H_1 and H_2 .

The composite system impulse response $h_c[n]$ is given by

$$\begin{aligned} h_c[n] &= h_1[n] * h_2[n] = h_2[n] * h_1[n] \\ &= 2^n u[n] * (4\delta[n] - 3(0.5)^n u[n]) \\ &= 4(2)^n u[n] - 3 \left(\frac{2^{n+1} - (0.5)^{n+1}}{2 - 0.5} \right) u[n] \\ &= (0.5)^n u[n]. \end{aligned}$$

If only the input and the output terminals are accessible on the composite cascade system (a black-box arrangement), a measurement from these external terminals would show that the impulse response of the system is $(0.5)^n u[n]$, without any hint of the fact that the system is harboring a dangerous unstable system within.

The composite system is BIBO stable because its impulse response, $(0.5)^n u[n]$, is absolutely summable. However, the system H_2 is asymptotically unstable because its characteristic root, 2, lies outside the unit circle. This system will eventually burn out (or saturate) because of the unbounded characteristic response generated by intended or unintended initial conditions, no matter how small. Thus, we see that although H_c is BIBO stable, it is asymptotically unstable.

This system is *uncontrollable* because the mode $2^n u[n]$ of H_2 cannot be controlled by the input. If the positions of H_1 and H_2 are interchanged (H_2 followed by H_1), the composite system becomes controllable but *unobservable* because the mode $2^n u[n]$ of H_2 is no longer observable at the output but is now controllable by the input. The system is still asymptotically unstable and BIBO stable. This example shows that BIBO stability does not necessarily ensure asymptotic stability when a system is either uncontrollable, unobservable, or both. The internal and external stability descriptions of a system are equivalent only when the system is both controllable and observable.[‡] In such a case, BIBO stability means that the system is asymptotically stable and vice versa.

Example 5.20 ◀

Fortunately, uncontrollable or unobservable systems are not common in practice. Henceforth, in determining system stability, we shall assume that, unless otherwise mentioned, systems are both

[†]By corollary, an asymptotically unstable system is not necessarily BIBO unstable.

[‡]The topics of controllability and observability are best studied within the context of state-space analysis, a topic outside the scope of this text. See [1] for an introductory treatment.

controllable and observable. In such cases, the internal and external descriptions of systems are equivalent.

▷ **Example 5.21 (Determining System Stability)**

Determine the internal and external stabilities of the systems specified by the following equations. In each case, plot the characteristic roots in the complex plane. Each system is both controllable and observable.

- (a) $y[n+2] + 2.5y[n+1] + y[n] = x[n+1] - 2x[n]$
- (b) $y[n] - y[n-1] + 0.21y[n-2] = 2x[n-1] + 3x[n-2]$
- (c) $y[n+3] + 2y[n+2] + \frac{3}{2}y[n+1] + \frac{1}{2}y[n] = x[n+1]$
- (d) $(E^2 - E + 1)^2 \{y[n]\} = (3E + 1) \{x[n]\}$

(a) The characteristic polynomial is

$$\gamma^2 + 2.5\gamma + 1 = (\gamma + 0.5)(\gamma + 2).$$

The characteristic roots are -0.5 and -2 , as shown in Fig. 5.15a. Because $| -2 | > 1$, the root -2 lies outside the unit circle, and the system is asymptotically unstable and BIBO unstable.

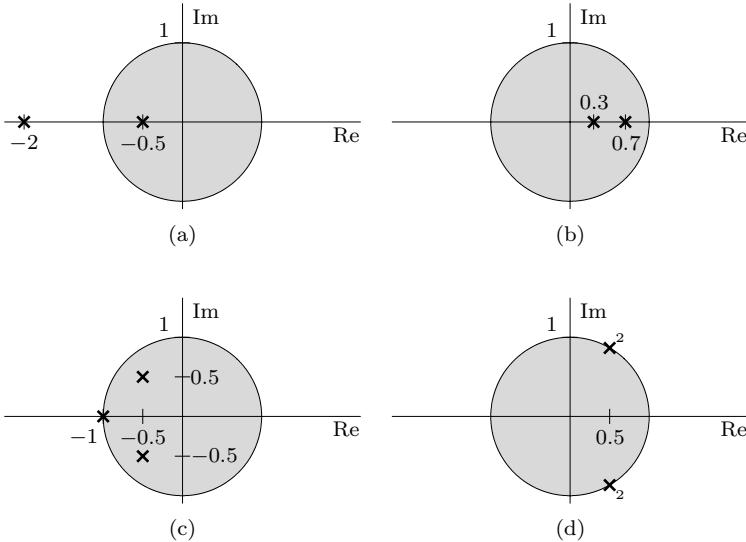


Figure 5.15: Characteristic root locations for Ex. 5.21.

(b) The characteristic polynomial is

$$\gamma^2 - \gamma + 0.21 = (\gamma - 0.3)(\gamma - 0.7).$$

As shown in Fig. 5.15b, the characteristic roots of 0.3 and 0.7 both lie inside the unit circle. The system is asymptotically stable and BIBO stable.

(c) The characteristic polynomial is

$$\gamma^3 + 2\gamma^2 + \frac{3}{2}\gamma + \frac{1}{2} = (\gamma + 1)(\gamma^2 + \gamma + \frac{1}{2}) = (\gamma + 1)(\gamma + 0.5 - j0.5)(\gamma + 0.5 + j0.5).$$

The three characteristic roots are -1 and $-0.5 \pm j0.5$ (Fig. 5.15c). One of the characteristic roots is on the unit circle, and the remaining two roots are inside the unit circle. The system is BIBO unstable but marginally stable.

(d) The characteristic polynomial is

$$(\gamma^2 - \gamma + 1)^2 = \left(\gamma - \frac{1}{2} - j\frac{\sqrt{3}}{2} \right)^2 \left(\gamma - \frac{1}{2} + j\frac{\sqrt{3}}{2} \right)^2.$$

The characteristic roots are $\frac{1}{2} \pm j\frac{\sqrt{3}}{2} = 1e^{\pm j\frac{\pi}{3}}$ repeated twice, and they lie on the unit circle (Fig. 5.15d). The system is BIBO unstable and asymptotically unstable.

Example 5.21 ◀

▷ Drill 5.11 (Determining System Stability)

Determine the internal and external stabilities of the systems specified by the following equations. In each case, plot the characteristic roots in the complex plane. Each system is both controllable and observable.

- (a) $(E + 1)(E^2 + 4E + 5) \{y[n]\} = 3E \{x[n]\}$
- (b) $(E^2 - 2E + 1)(E + 0.5) \{y[n]\} = (E^2 + 2E + 3) \{x[n]\}$

◀

5.8 Intuitive Insights into System Behavior

This section attempts to provide an understanding of what determines system behavior. Because of its intuitive nature, the following discussion is more or less qualitative. We show that the most important attributes of a system are its characteristic roots or characteristic modes because they determine not only the zero-input response but also the entire behavior of the system.

5.8.1 Dependence of System Behavior on Characteristic Modes

Recall that the zero-input response of a system consists of the system's characteristic modes. For a stable system, these characteristic modes decay exponentially and eventually vanish. This behavior may give the impression that these modes do not substantially affect system behavior in general and system response in particular. This impression is totally wrong! We shall now see that the system's characteristic modes leave their imprint on every aspect of the system behavior. *We may compare the system's characteristic modes (or roots) to a seed that eventually dissolves in the ground; however, the plant that springs from it is totally determined by the seed. The imprint of the seed exists on every cell of the plant.*

In order to understand this interesting phenomenon, recall that the characteristic modes of a system are very special to that system because it can sustain these signals without the application of an external input. In other words, the system offers a free ride and ready access to these signals. Now imagine what happens if we actually drive the system with an input having the form of a characteristic mode. We expect the system to respond strongly (the resonance phenomenon). If the input is not exactly a characteristic mode but is close to such a mode, we still expect the system response to be strong. However, if the input is very different from any of the characteristic modes, we expect the system to respond poorly. We now show that these intuitive deductions are indeed true.

Although correlation offers a formal measure of similarity between signals, we shall take a simpler approach here. Let us restrict the system's inputs to exponentials of the form z^n , where z is generally

a complex number. The similarity of two exponential signals z^n and γ^n is established by the closeness of z and γ . If the difference $|z - \gamma|$ is small, then the signals are similar; if $|z - \gamma|$ is large, then the signals are dissimilar.

Now consider a first-order system with a single characteristic mode γ^n and the input z^n . The impulse response of this system is then given by $c\gamma^n$, where the exact value of the constant c is not important for this qualitative discussion.[†] The system response $y[n]$ is given by

$$\begin{aligned} y[n] &= h[n] * x[n] \\ &= c\gamma^n u[n] * z^n u[n]. \end{aligned}$$

From Table 5.2, we obtain

$$y[n] = \frac{c[z^n - \gamma^n]}{z - \gamma} u[n]. \quad (5.37)$$

Clearly, if the input z^n is similar to γ^n , $z - \gamma$ is small, and the system response is large. *The closer the input $x[n]$ is to the characteristic mode, the stronger the system response becomes.* In contrast, if the input is very different from the natural mode, $z - \gamma$ is large, and the system responds poorly. This is precisely what we set out to prove.

We have proved the preceding assertion for a single-mode (first-order) system. It can be generalized to a K th-order system, which has K characteristic modes. The impulse response $h[n]$ of such a system is a linear combination of its K modes. Therefore, if $x[n]$ is similar to any one of the modes, the corresponding response will be high; if it is similar to none of the modes, the response will be small. Clearly, the characteristic modes are very influential in determining the system response to a given input.

It would be tempting to conclude on the basis of Eq. (5.37) that if the input is identical to the characteristic mode, so that $z = \gamma$, then the response goes to infinity. Remember, however, that if $z = \gamma$, the numerator on the right-hand side of Eq. (5.37) also goes to 0. We shall study this complex behavior, or resonance phenomenon, later in this section.

We shall now show that *mere inspection of the impulse response $h[n]$ (which is composed of characteristic modes) reveals a great deal about the system behavior.*

5.8.2 Response Time of a System: The System Time Constant

Like human beings, systems have a certain response time. In other words, when an input (stimulus) is applied to a system, a certain amount of time elapses before the system fully responds to that input. This time lag or response time is called the system *time constant*. As we shall see, a system's time constant (in samples) is equal to W_h , the width of its impulse response $h[n]$.

An input $\delta[n]$ to a system has a width of zero ($W_\delta = 0$). The response of a system to an input $\delta[n]$ is the impulse response $h[n]$, which has some width W_h . Clearly, the system requires a time equivalent of W_h samples to respond fully to input $\delta[n]$, and we are justified in viewing W_h as the system's response time or time constant. We arrive at the same conclusion via another argument. The system output is a convolution of the input with $h[n]$. If an input is a pulse of width W_x , then, according to the width property of convolution, the output pulse width is $W_x + W_h$. This conclusion shows that the system requires a time equivalent of the width W_h to respond fully to any input. *The system time constant indicates how fast a system responds. A system with a relatively small time constant is a fast system that responds quickly to an input. A system with a relatively large time constant is a sluggish system that cannot respond well to rapidly varying signals.*

For systems represented by recursive difference equations, the width of the impulse response $h[n]$ is ∞ because the characteristic modes approach 0 asymptotically as $n \rightarrow \infty$. However, beyond some value of n , $h[n]$ becomes negligible. It is therefore necessary to use some suitable measure of the impulse response's effective (or equivalent) width.

[†]There may also be an impulse at the origin. We shall ignore this term because its presence merely results in an additional output component that is of the form of the input. This term will not affect the comparison of the outputs.

There is no single satisfactory definition of equivalent signal width applicable to every situation. For the situation depicted in Fig. 5.16, a reasonable definition of the equivalent width of $h[n]$ may be taken as the width $W_{\hat{h}}$ of the rectangular pulse $\hat{h}[n]$ (shown shaded for emphasis). The rectangular pulse $\hat{h}[n]$, which can be called as an *effective impulse response*, is formed such that the sum of all its elements is as close as possible to the sum of all the elements of $h[n]$. The height of $\hat{h}[n]$ is chosen to be equal to that of $h[n]$ at some suitable instant $n = m$. In Fig. 5.16, m is chosen as the instant at which $h[n]$ is maximum. Noting that the number of elements in a DT signal exceeds its width by 1, the effective width satisfies[†]

$$(W_{\hat{h}} + 1)h[m] = \sum_{n=-\infty}^{\infty} h[n].$$

Solving this equation for effective width $W_{\hat{h}}$ yields

$$W_{\hat{h}} = \frac{\sum_{n=-\infty}^{\infty} h[n]}{h[m]} - 1. \quad (5.38)$$

The time constant for a lowpass infinite impulse response (IIR) system is normally well approximated by $W_{\hat{h}}$.

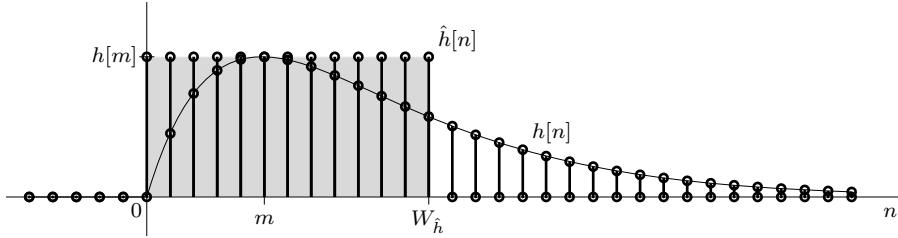


Figure 5.16: Effective width of an impulse response.

Consider an example of a system with a single characteristic root γ that is real, positive, and within the unit circle. The corresponding impulse response is

$$h[n] = c\gamma^n u[n].$$

The maxima of $h[n]$ occurs at the origin ($m = 0$) with value $h(0) = c$. Therefore, according to Eq. (5.38), the effective width (time constant) is

$$W_{\hat{h}} = \frac{1}{c} \sum_{n=0}^{\infty} c\gamma^n - 1 = \frac{1}{1-\gamma} - 1 = \frac{\gamma}{1-\gamma}.$$

For the multimode case, $h[n]$ is a weighted sum of the system's characteristic modes, and $W_{\hat{h}}$ is a weighted average of the time constants associated with the K modes of the system.

A Subtle Dilemma

The width of a physical DT sequence is always an integer value. However, Eq. (5.38) rarely produces an integer result for $W_{\hat{h}}$, and a subtle dilemma arises. Do we keep the non-integer width, or do we round the result to a nearby integer? In truth, it doesn't really matter. The system time constant is an imperfect measure to begin with, and it is primarily useful for approximate and qualitative assessments of system behavior. The pulse $\hat{h}[n]$, which serves as only a rough approximation of the system $h[n]$, rarely needs to correspond to a physical DT signal, so a non-integer value of $W_{\hat{h}}$ need not concern us.

[†]This definition is satisfactory when $h[n]$ is a single, mostly positive (or mostly negative) pulse. Such systems are lowpass systems. This definition should not be applied indiscriminately to all systems.

5.8.3 Time Constant and Rise Time of a System

The system time constant may also be viewed from a different perspective. The unit step response $s[n]$ of a system is the convolution of $u[n]$ with $h[n]$. Let the impulse response $h[n]$ be a rectangular pulse of width $W_h = 3$, as shown in Fig. 5.17. This assumption simplifies the discussion yet gives satisfactory results for a qualitative discussion. The operation $u[n] * h[n] = s[n]$ is illustrated in Fig. 5.17. Note that the output does not rise from 0 to the final value instantaneously as the input does; instead, the output takes $W_h = 3$ samples to accomplish this. Hence, W_r , the system rise time (in samples), is equal to the system time constant.[†] That is,

$$W_r = W_h. \quad (5.39)$$

This result and Fig. 5.17 show clearly that a system generally does not respond to an input instantaneously. Instead, it takes time equivalent of W_h samples for the system to respond fully.

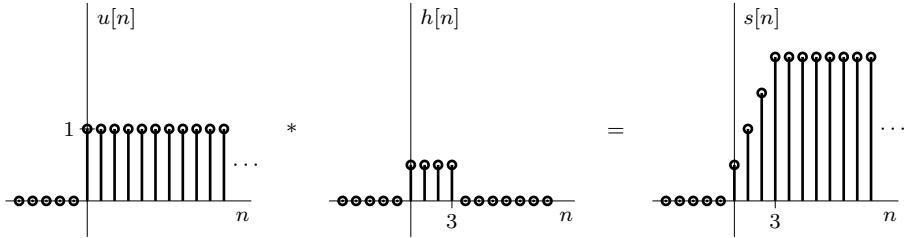


Figure 5.17: Rise time of a system.

5.8.4 Time Constant and Filtering

A larger time constant implies a sluggish system because the system takes a longer time to respond fully to an input. Such a system cannot respond effectively to rapid variations in the input. In contrast, a smaller time constant indicates that a system is capable of responding to rapid variations in the input. Thus, there is a direct connection between a system's time constant and its filtering properties.

A system with a large time constant does not respond well to high-frequency inputs, implying that it suppresses rapidly varying (high-frequency) sinusoids and therefore acts as a lowpass filter. We shall now show that the cutoff frequency Ω_c of a lowpass filter has an inverse relationship to its time constant W_h .

To demonstrate this inverse relationship, let us determine the system response to a sinusoidal input $x[n]$ by convolving this input with a rectangular impulse response $h[n]$, which is depicted in Fig. 5.18a. The time constant of this system is given by W_h , the width of the impulse response. Figures 5.18b and 5.18c show the process of convolving $h[n]$ with sinusoidal inputs of two different frequencies. The sinusoid in Fig. 5.18b has a relatively high frequency, while the frequency of the sinusoid in Fig. 5.18c is low. Recall that the convolution of $x[n]$ and $h[n]$ is equal to the sum of the product $x[m]h[n-m]$. The regions of summation are shown shaded in Figs. 5.18b and 5.18c for the two cases. For the high-frequency sinusoid (Fig. 5.18b), it is clear that the sum of $x[m]h[n-m]$ is very small because its positive and negative samples nearly cancel each other out. In this case the output $y[n]$ remains periodic but has a rather small amplitude. This happens when the period of the input sinusoid is much smaller than the system time constant W_h . In contrast, for the low-frequency sinusoid, the period of the sinusoid is larger than W_h , and the cancellations in the sum

[†]Because of varying definitions of rise time, the reader may find different results in the literature. The qualitative and intuitive nature of this discussion should always be kept in mind.

of $x[m]h[n - m]$ are less effective. Consequently, the output $y[n]$ is much larger, as depicted in Fig. 5.18c.

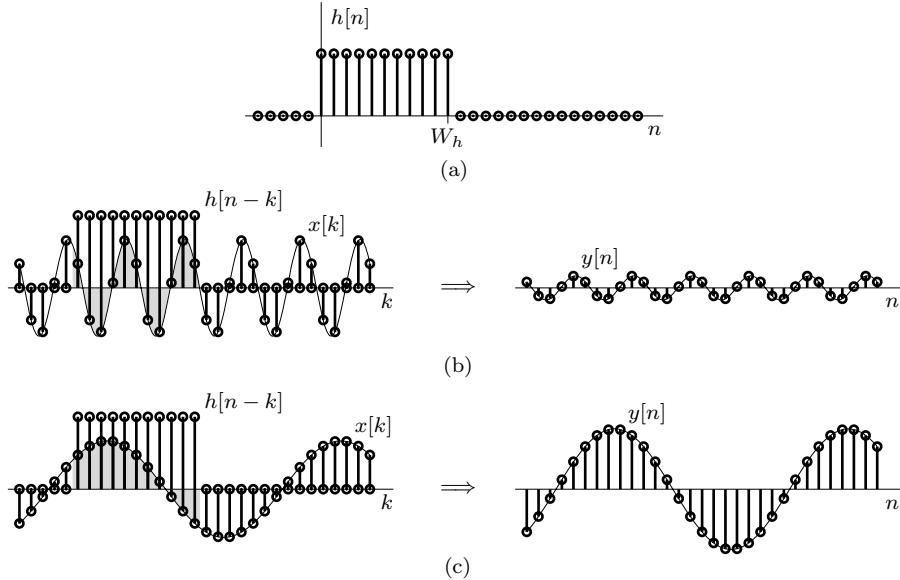


Figure 5.18: Time constant and filtering.

Between these two possible extremes in system behavior, a transition point occurs when the period of the sinusoid is equal to the system time constant W_h . The frequency at which this transition occurs is known as the *cutoff frequency* Ω_c of the system. Thus, the cutoff frequency period is of the order of W_h . This proves our contention that a larger time constant implies a lower bandwidth system, and a smaller time constant leads to a higher bandwidth system. Although these conclusions are derived for an idealized (rectangular) impulse response, its implications are valid for lowpass LTID systems in general since such systems are reasonably approximated by effective impulse responses $\hat{h}[n]$ that are rectangular.

5.8.5 Time Constant and Pulse Dispersion

In general, the transmission of a pulse through an LTID system causes pulse dispersion (or spreading). Therefore, the output pulse is generally wider than the input pulse. Earlier we saw that if an input $x[n]$ is a pulse of width W_x , then W_y , the width of the output $y[n]$, is

$$W_y = W_x + W_h. \quad (5.40)$$

This result shows that an input pulse spreads out (disperses) as it passes through a system. Since W_h is also the system's time constant or rise time, the amount of spread in the pulse is equal to the time constant (or rise time) of the system.

This discussion (Secs. 5.8.2 through 5.8.5) shows that the system time constant determines much of a system's behavior: its filtering characteristics, rise time, pulse dispersion, and so on. In turn, the time constant is determined by the system's characteristic roots. Clearly, the characteristic roots and their relative amounts in the impulse response $h[n]$ determine the behavior of a system.

5.8.6 The Resonance Phenomenon

Finally, we come to the fascinating phenomenon of resonance. As we have mentioned earlier, this phenomenon is observed when the input signal is identical or very similar to a characteristic mode

of the system. For the sake of simplicity and clarity, we consider a first-order system that has only a single mode, γ^n . Let the impulse response of this system be

$$h[n] = c\gamma^n u[n],$$

and let the input be

$$x[n] = (\gamma - \epsilon)^n u[n].$$

The system response $y[n]$ is then given by

$$y[n] = c\gamma^n u[n] * (\gamma - \epsilon)^n u[n].$$

From Table 5.2, we obtain

$$y[n] = \frac{c [\gamma^{n+1} - (\gamma - \epsilon)^{n+1}]}{\epsilon} u[n]. \quad (5.41)$$

Now, as $\epsilon \rightarrow 0$, both the numerator and the denominator of Eq. (5.41) approach 0. Applying L'Hôpital's rule yields

$$\lim_{\epsilon \rightarrow 0} y[n] = c(n+1)\gamma^n u[n]. \quad (5.42)$$

Clearly, the response does not go to infinity as $\epsilon \rightarrow 0$, but it acquires a factor $n+1$, which approaches ∞ as $n \rightarrow \infty$. If γ is inside the unit circle, γ^n decays faster than $n+1$, and $y[n] \rightarrow 0$ as $n \rightarrow \infty$. The resonance phenomenon in this case is present, but its manifestation is aborted by the signal's own exponential decay.

This discussion shows that *resonance is a cumulative phenomenon*, not instantaneous. It builds up linearly with n .[†] When the mode decays exponentially, the signal decays at a rate too fast for resonance to counteract the decay; as a result, the signal vanishes before resonance has a chance to build it up. However, if the mode decays at a rate less than $1/(n+1)$, we should see the resonance phenomenon clearly. This specific condition is possible when $|\gamma| = 1$ (γ lies on the unit circle). In this case,

$$\gamma = e^{j\Omega},$$

and Eq. (5.42) becomes

$$y[n] = c(n+1)e^{j\Omega n} u[n]. \quad (5.43)$$

Here, the response magnitude does go to infinity linearly with n .

For a real system, if $\gamma = e^{j\Omega}$ is a root, then $\gamma^* = e^{-j\Omega}$ must also be a root, and the impulse response is of the form $(ce^{j\Omega n} + c^*e^{-j\Omega n}) u[n] = 2|c| \cos(\Omega n + \theta) u[n]$. The response of this system to input $\cos(\Omega n) u[n]$ is $2|c| \cos(\Omega n + \theta) u[n] * \cos(\Omega n) u[n]$. The reader can show that this convolution contains a term of the form $(n+1) \cos(\Omega n) u[n]$. The resonance phenomenon is clearly visible. The component $n \cos(\Omega n) u[n]$ of the response increases linearly with n , eventually reaching ∞ , as indicated in Fig. 5.19.

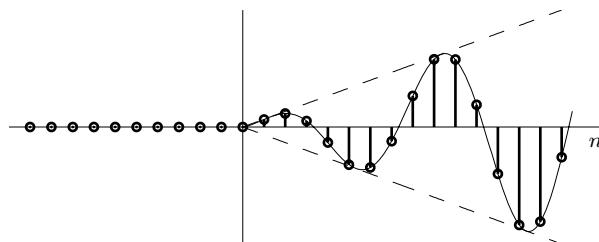


Figure 5.19: Resonant component in response to a sinusoidal input.

[†]If the characteristic root in question repeats r times, the resonance effect increases as n^{r-1} . As long as γ is within the unit circle, even $n^{r-1}\gamma^n \rightarrow 0$ as $n \rightarrow \infty$ for any value of r .

Recall that when $\gamma = e^{j\Omega}$, the system is marginally stable. As we have indicated, the full effect of resonance cannot be seen for an asymptotically stable system; only in a marginally stable system does the resonance phenomenon boost the system's response to infinity when the system's input is a characteristic mode. But even in an asymptotically stable system, we see a manifestation of resonance if its characteristic roots are close to the unit circle, such as when $\gamma = (1 - \delta)e^{\pm j\Omega}$ and δ is very small positive number. In such cases, we can show that the system response to the input $e^{j\Omega n}$ or the sinusoid $\cos(\Omega n)$ is very large.[†] The response drops off rapidly as the input signal frequency moves away from Ω or if δ becomes larger. This frequency-selective behavior can be studied more profitably using frequency-domain analysis. For this reason, we postpone full discussion of this subject until the next chapter.

Importance of the Resonance Phenomenon

The resonance phenomenon is important because it allows us to design frequency-selective systems by choosing their characteristic roots properly. Lowpass, bandpass, highpass, and bandstop filters are all examples of frequency-selective networks. In mechanical systems, the inadvertent presence of resonance can cause signals of such tremendous magnitudes that the system may fall apart. A musical note (periodic vibrations) of proper frequency can shatter a glass if the frequency is matched to a characteristic root of the glass, which acts as a mechanical system. Similarly, a company of soldiers marching in step across a bridge amounts to applying a periodic force to the bridge. If the frequency of this input force happens to be near a characteristic root of the bridge, the bridge may respond (vibrate) violently and collapse, even though it would have been strong enough to carry many soldiers marching out of step. A case in point is the Tacoma Narrows Bridge failure of 1940. This bridge, nicknamed “Galloping Gertie” due to the motions of its center span in even light winds, collapsed in a mild gale on November 7, 1940, less than four months after opening. The collapse occurred not because of the wind’s brute force but because the frequency of wind-generated vortices, which matched the natural frequencies (characteristic roots) of the bridge, caused resonance.

Because of the great damage that may occur, mechanical resonance is generally something to be avoided, especially in structures or vibrating mechanisms. If an engine with periodic force (such as piston motion) is mounted on a platform, the platform with its mass and springs should be designed so that their characteristic roots are not close to the engine’s frequency of vibration. Proper design of this platform not only helps avoid resonance but also attenuates vibrations if the system roots are placed far away from the frequency of vibration.

5.9 Classical Solution of Linear Difference Equations

As we have seen, the sum of the zero-input and zero-state responses is the solution to a linear difference equation. The classical method obtains the same solution from the sum of the natural and forced components.

The ZIR and ZSR both include characteristic modes of the system, and the ZSR includes additional terms that reflect the input to the system. When all the characteristic mode terms in the total response are lumped together, the resulting component is the *natural response*. The *forced response* is made up of the remaining non-characteristic modes. If $y_c[n]$ and $y_\phi[n]$ denote the natural and the forced response, respectively, then the total response $y[n]$ is given by

$$y[n] = \underbrace{y_c[n]}_{\text{modes}} + \underbrace{y_\phi[n]}_{\text{non-modes}} . \quad (5.44)$$

[†]This follows directly from Eq. (5.41) with $\gamma = e^{j\Omega}$ and $\epsilon = \delta e^{j\Omega}$.

▷ **Example 5.22 (Natural and Forced Responses from ZIR and ZSR)**

Using the initial conditions $y[-1] = 0$ and $y[-2] = \frac{25}{4}$ and the input $x[n] = 4^{-n}u[n]$, determine the forced and natural responses of an LTID system described by the equation

$$y[n+2] - 0.6y[n+1] - 0.16y[n] = 5x[n+2].$$

From Ex. 5.5, we know this second-order system has two characteristic modes: $(-0.2)^n$ and $(0.8)^n$. Using the result derived in Ex. 5.19 and grouping characteristic and non-characteristic modes separately, the forced and natural responses are

$$y[n] = \underbrace{-1.26(4)^{-n}}_{\text{forced response}} + \underbrace{0.644(-0.2)^n + 6.62(0.8)^n}_{\text{natural response}} \quad (n \geq 0).$$

Example 5.22 ◀

Finding the Natural and Forced Responses

Because the total response $y_c[n] + y_\phi[n]$ is a solution of Eq. (5.8), we have

$$A(E) \{y_c[n] + y_\phi[n]\} = B(E) \{x[n]\}. \quad (5.45)$$

But since $y_c[n]$ is made up of characteristic modes,

$$A(E) \{y_c[n]\} = 0. \quad (5.46)$$

Just like the ZIR, the natural response must be a linear combination of characteristic modes. The constants c_k that weight each mode are determined from suitable auxiliary conditions usually given as $y[0], y[1], \dots, y[K-1]$.

Let us clarify the reason that we use *auxiliary conditions* $y[0], y[1], \dots, y[K-1]$ rather than the initial conditions $y[-1], y[-2], \dots, y[-K]$. As we have seen, the initial conditions $y[-1], y[-2], \dots, y[-K]$ yield the ZIR, not the natural response (which also includes mode terms produced by the input). At $n = -1, -2, \dots, -K$, only the zero-input component exists, and these initial conditions can be applied to the zero-input component only. In the classical method, the zero-input and zero-state components cannot be separated. Consequently, the initial conditions must be applied to the total response, which begins at $n = 0$.[†] Hence, we need auxiliary conditions for $n \geq 0$, the first K of which are $y[0], y[1], \dots, y[K-1]$. If we are given the initial conditions $y[-1], y[-2], \dots, y[-K]$, we can derive the auxiliary conditions $y[0], y[1], \dots, y[K-1]$ using the iterative procedure.

We now turn our attention to the forced response. Substitution of Eq. (5.46) into Eq. (5.45) yields

$$A(E) \{y_\phi[n]\} = B(E) \{x[n]\}. \quad (5.47)$$

The forced response $y_\phi[n]$ satisfies this equation and, by definition, contains only non-mode terms. To determine the forced response, we shall use the method of undetermined coefficients, where we assume that the input (forcing function) follows one of the forms given in Table 5.3.[‡] The unknown coefficients can be determined by substituting $y_\phi[n]$ into Eq. (5.47) and equating coefficients of similar terms. We shall illustrate the classical method using several examples.

[†]Minor modifications are required for inputs that start at times other than $n = 0$.

[‡]To make them causal, the inputs and outputs in Table 5.3 should technically include $u[n]$ terms. However, since we are only interested in the output for $n \geq 0$, the inclusion of $u[n]$ multipliers makes no difference and would only serve as a distraction.

Input $x[n]$	Forced Response $y_\phi[n]$
1. ξ^n	$b\xi^n$
2. $\cos(\beta n + \theta)$	$b\cos(\beta n + \phi)$
3. $\left(\sum_{m=0}^M \alpha_m n^m\right) \xi^n$	$\left(\sum_{m=0}^M b_m n^m\right) \xi^n$
4. ξ^n and $\xi =$ an unrepeated characteristic root	$bn\xi^n$

Table 5.3: Forced responses for selected inputs.

▷ **Example 5.23 (Classical Solution to a Linear Difference Equation)**

Using the classical method, solve

$$(E^2 - 5E + 6)\{y[n]\} = (E - 5)\{x[n]\}$$

if the input $x[n] = (5 + 3n)u[n]$ and the auxiliary conditions are $y[0] = 4$ and $y[1] = 13$.

The characteristic equation is

$$\gamma^2 - 5\gamma + 6 = (\gamma - 2)(\gamma - 3) = 0.$$

Therefore, the natural response is

$$y_c[n] = c_1(2)^n + c_2(3)^n.$$

To find the form of the forced response $y_\phi[n]$ for $n \geq 0$, we use pair 3 of Table 5.3 with $\xi = 1$ and $M = 1$. This yields

$$y_\phi[n] = b_1 n + b_0.$$

Therefore,

$$y_\phi[n+1] = b_1(n+1) + b_0 \quad \text{and} \quad y_\phi[n+2] = b_1(n+2) + b_0.$$

Also, for $n \geq 0$,

$$x[n] = 3n + 5 \quad \text{and} \quad x[n+1] = 3(n+1) + 5.$$

Substitution of these results in Eq. (5.47) yields

$$b_1(n+2) + b_0 - 5(b_1(n+1) + b_0) + 6(b_1n + b_0) = 3(n+1) + 5 - 5(3n+5)$$

or

$$2b_1n + 2b_0 - 3b_1 = -12n - 17.$$

Comparison of similar terms on the two sides yields

$$\begin{aligned} 2b_1 &= -12 \\ 2b_0 - 3b_1 &= -17 \end{aligned} \quad \left. \right\} \implies \begin{aligned} b_1 &= -6 \\ b_0 &= -\frac{35}{2} \end{aligned}.$$

Therefore,

$$y_\phi[n] = -6n - \frac{35}{2}.$$

For $n \geq 0$, the total response is

$$y[n] = y_c[n] + y_\phi[n] = c_1(2)^n + c_2(3)^n - 6n - \frac{35}{2}.$$

To determine c_1 and c_2 , we set $n = 0$ and 1 and substitute $y[0] = 4$ and $y[1] = 13$ to obtain

$$\begin{aligned} c_1 + c_2 - \frac{35}{2} &= 4 \\ 2c_1 + 3c_2 - \frac{47}{2} &= 13 \end{aligned} \quad \left\{ \Rightarrow \begin{array}{l} c_1 = 28 \\ c_2 = -\frac{13}{2} \end{array} \right..$$

Therefore,

$$y[n] = \underbrace{28(2)^n}_{y_c[n]} - \frac{13}{2}(3)^n + \underbrace{\left(-6n - \frac{35}{2}\right)}_{y_\phi[n]}.$$

Example 5.23 □

▷ Example 5.24 (Classical Solution to $\sum_{m=0}^n m^2$.)

Using the classical method, determine the sum

$$y[n] = \sum_{m=0}^n m^2. \quad (5.48)$$

To begin, we find an appropriate difference equation that has $y[n]$ as the response. From Eq. (5.48), we observe that $y[n+1] = y[n] + (n+1)^2$. Hence,

$$y[n+1] - y[n] = (n+1)^2. \quad (5.49)$$

This is the equation we seek. For this first-order difference equation, we need one auxiliary condition, the value of $y[n]$ at $n = 0$. From Eq. (5.48), it follows that $y[0] = 0$. Thus, we seek the solution of Eq. (5.49) subject to the auxiliary condition $y[0] = 0$.

The characteristic equation of Eq. (5.49) is $\gamma - 1 = 0$, the characteristic root is $\gamma = 1$, and the characteristic mode is $c(1)^n u[n] = cu[n]$. For $n \geq 0$, the natural response is clearly the constant c .

Using pair 3 of Table 5.3 with $\xi = 1$ and $M = 2$, we see that an input $x[n] = (n+1)^2 = n^2 + 2n + 1$ normally produces the forced response $b_2 n^2 + b_1 n + b_0$. However, since the constant b_0 matches the characteristic mode of this system, the correct form of the forced response is

$$y_\phi[n] = b_2 n^3 + b_1 n^2 + b_0 n.$$

Using this result,

$$E \{y_\phi[n]\} = y_\phi[n+1] = b_2(n+1)^3 + b_1(n+1)^2 + b_0(n+1).$$

From Eq. (5.47), we obtain

$$(E - 1) \{y_\phi[n]\} = n^2 + 2n + 1$$

or

$$[b_2(n+1)^3 + b_1(n+1)^2 + b_0(n+1)] - [b_2 n^3 + b_1 n^2 + b_0 n] = n^2 + 2n + 1.$$

Equating the coefficients of similar powers, we obtain

$$b_0 = \frac{1}{6}, \quad b_1 = \frac{1}{2}, \quad \text{and} \quad b_2 = \frac{1}{3}.$$

Hence

$$y[n] = c + \frac{2n^3 + 3n^2 + n}{6} = c + \frac{n(n+1)(2n+1)}{6}.$$

Setting $n = 0$ in this equation and using the auxiliary condition $y[0] = 0$, we find that $c = 0$. Thus,

$$y[n] = \sum_{m=0}^n m^2 = \frac{n(n+1)(2n+1)}{6}.$$

Checking our result, we see that this result correctly matches pair 3 of Table 5.1.

Example 5.24 ◀

Additional Insights for Exponential Inputs

Consider an exponential input $x[n] = \xi^n$ applied to a linear difference equation specified by $A(E)\{y[n]\} = B(E)\{x[n]\}$ (Eq. (5.8)). As long as ξ does not equal a characteristic mode, we know from Table 5.3 that the forced response is an exponential of the same form,

$$y_\phi[n] = b\xi^n.$$

We now show that the coefficient b is easily obtained as

$$b = H(\xi) = \frac{B(\xi)}{A(\xi)}. \quad (5.50)$$

To prove Eq. (5.50), notice that

$$E^k\{x[n]\} = x[n+k] = \xi^{n+k} = \xi^k \xi^n \quad \text{and} \quad E^k\{y_\phi[n]\} = y_\phi[n+k] = b\xi^{n+k} = b\xi^k \xi^n.$$

Thus,

$$B(E)\{x[n]\} = B(\xi)\xi^n \quad \text{and} \quad A(E)\{y_\phi[n]\} = bA(\xi)\xi^n,$$

and Eq. (5.8) reduces to

$$bA(\xi)\xi^n = B(\xi)\xi^n.$$

Solving this expression yields Eq. (5.50). We again stress that this result is valid only if ξ is not a characteristic root of the system.

Observe that an exponential input ξ^n includes a wide variety of signals, such as a constant, a sinusoid, and an exponentially growing or decaying sinusoid. For example, the input $e^{\pm j\Omega n}$ is an exponential ξ^n with $\xi = e^{\pm j\Omega}$. Hence,

$$y_\phi[n] = H(e^{j\Omega})e^{j\Omega n} = \frac{B(e^{j\Omega})}{A(e^{j\Omega})}e^{j\Omega n}.$$

Along these lines, we can show that

$$x[n] = \cos(\Omega n + \theta) \implies y_\phi[n] = |H(e^{j\Omega})| \cos[\Omega n + \theta + \angle H(e^{j\Omega})]. \quad (5.51)$$

Readers are encouraged to work out the details.

▷ **Example 5.25 (Forced Response to an Exponential)**

For the input $x[n] = (3)^n u[n]$, determine the forced response $y_\phi[n]$ of a system specified by the equation

$$(E^2 - 3E + 2)\{y[n]\} = (E + 2)\{x[n]\}.$$

In this case,

$$H(\xi) = \frac{B(\xi)}{A(\xi)} = \frac{\xi + 2}{\xi^2 - 3\xi + 2}.$$

Since $\xi = 3$ is not a characteristic root of the system, the forced response to input $(3)^n u[n]$ is

$$y_\phi[n] = H(3)(3)^n u[n] = \frac{3 + 2}{(3)^2 - 3(3) + 2} (3)^n u[n] = \frac{5}{2} (3)^n u[n].$$

Example 5.25 ◀

▷ Drill 5.12 (Forced Response to a Sinusoid)

For the input $x[n] = \cos(2n + \pi/3)u[n]$, determine the forced response $y_\phi[n]$ of a system specified by the equation

$$(E^2 - E + 0.16) \{y[n]\} = (E + 0.32) \{x[n]\}.$$

◀

Assessment of the Classical Method

The classical method is relatively simple compared with finding the response as a sum of the zero-input and zero-state components. Convolution, for example, is not required with the classical method. Unfortunately, the classical method has a serious drawback because it yields the total response, which cannot be separated into components arising from the internal conditions and the external input. In the study of systems, it is important to be able to express the system response to an input $x[n]$ as an explicit function of $x[n]$. This is not possible in the classical method. Moreover, the classical method is restricted to a certain class of inputs; it cannot be applied to any input. Given these drawbacks, the classical method is generally not preferred in the study of DT signals and systems.

5.10 Summary

This chapter discusses the time-domain analysis of LTID (linear, time-invariant, discrete-time) systems. This analysis parallels that of LTIC systems, with minor differences. Discrete-time systems are commonly described by difference equations, which are conveniently expressed using operator notation as $A(E) \{y[n]\} = B(E) \{x[n]\}$. For a K th-order system, K initial conditions must be specified for a unique solution to an input starting at $n = 0$. The iterative approach is a conceptually simple approach to analyze LTID systems, both recursive and nonrecursive, although it rarely delivers a closed-form solution.

A closed-form solution is possible by summing the zero-input and zero-state responses of a system. The zero-input response (ZIR) is how the system responds to internal conditions when the input is 0. The zero-input response is comprised of the characteristic modes of the system, which are determined from the roots of the characteristic equation $A(\gamma) = 0$. An unpeated root γ produces a characteristic mode γ^n , while a root γ repeated r times has r characteristic modes: $\gamma^n, n\gamma^n, \dots, n^{r-1}\gamma^n$. The coefficients that weight the K modes of the ZIR are determined from the K initial conditions of the system.

The duration of the unit impulse $\delta[n]$ is one sample at $n = 0$. For almost all time, an input of $\delta[n]$ is like having no input at all. Consequently, the unit impulse response $h[n]$ of an LTID system to input $\delta[n]$ is comprised primarily of the system's characteristic modes. Based on the duration of

$h[n]$, discrete-time systems are commonly classified as either infinite impulse response (IIR) or finite impulse response (FIR) systems.

The zero-state response (ZSR) is how a system responds to an external input with zero initial conditions (zero state). Since any DT input can be represented as a sum of weighted impulse functions, the ZSR of a LTID system is a sum of weighted impulse response functions. This sum, called the convolution sum, is similar in structure and properties to the convolution integral used in the study of LTIC systems. For an arbitrary input $x[n]$, the impulse response $h[n]$ of an LTID system allows us to determine the ZSR as the convolution $x[n] * h[n]$.

Complex systems are often constructed from interconnections of simpler systems. Assisted by the properties of DT convolution, the behaviors of parallel and cascade connections of LTID systems are easily established. The convolution sum also helps demonstrate that the response of an LTID system to an everlasting exponential input z^n is, within a multiplicative constant, the same signal. Thus, the response of an LTID system to z^n is $H(z)z^n$, where $H(z)$ is the transfer function of the system.

Stability greatly influences system behavior. A system is externally, or bounded-input bounded-output (BIBO), stable if and only if every bounded input produces a bounded output. Otherwise, the system is BIBO unstable. Internal stability is determined based on the locations of a system's characteristic roots. An asymptotically stable system has all roots inside the unit circle. An asymptotically unstable system has at least one root outside the unit circle, repeated roots on the unit circle, or both. A marginally stable system, which is neither asymptotically stable nor unstable, has unrepeated roots on the unit circle and possibly other roots inside the unit circle. An asymptotically stable system is always BIBO stable. The converse is not necessarily true.

Characteristic roots not only determine internal stability, but they also dictate every aspect of system behavior. Response and rise times, filtering characteristics, and pulse dispersion are all related to a system's characteristic roots. A system responds strongly to inputs that are close to its characteristic roots. If an input matches a characteristic root, resonance occurs, and a new (possibly unstable) mode appears.

Difference equations of LTID systems can also be solved by the classical method, where the response is obtained as a sum of the natural and forced components. These are not the same as the zero-input and zero-state components, although they satisfy the same equations, respectively. Although simple, this method is applicable to a restricted class of input signals and cannot provide the separate zero-input and zero-state responses. These limitations diminish its value considerably in the theoretical study of systems.

Reference

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.

Problems

- 5.1-1** A realization of a digital integrator that uses a backward difference approximation is represented as

$$y[n] - y[n-1] = Tx[n],$$

where T is the sampling interval. Assuming zero initial condition $y[-1] = 0$, solve this equation iteratively when the input $x[n]$ is

(a) $\delta[n]$ (b) $u[n]$

- 5.1-2** A digital integrator that uses a trapezoidal approximation is characterized by the difference equation

$$y[n] - y[n-1] = \frac{T}{2} (x[n] + x[n-1]),$$

where T is the sampling interval. Assuming zero initial condition $y[-1] = 0$, solve this equation iteratively when the input $x[n]$ is

(a) $\delta[n]$ (b) $u[n]$

Comment on the performance of this system as an integrator.

- 5.1-3** Solve iteratively (first three terms only)

- (a) $y[n+1] - 0.5y[n] = 0$ with $y[-1] = 10$
- (b) $y[n+1] + 2y[n] = x[n+1]$ with $x[n] = e^{-n}u[n]$ and $y[-1] = 0$
- (c) $y[n] + \frac{1}{4}y[n-1] + \frac{1}{16}y[n-2] = x[n]$ with $x[n] = 100u[n]$ and $y[-1] = y[-2] = 0$

- 5.1-4** Using $y[-1] = -25$ and $y[-2] = 0$, iteratively solve (first three terms only)

$$y[n] - 0.6y[n-1] - 0.16y[n-2] = 0.$$

- 5.1-5** Using $x[n] = (3)^n u[n]$, $y[-1] = 3$, and $y[-2] = 2$, iteratively determine (first three terms only) the total response, ZIR, and ZSR for

$$\begin{aligned} y[n+2] + 3y[n+1] + 2y[n] = \\ x[n+2] + 3x[n+1] + 3x[n]. \end{aligned}$$

- 5.1-6** Using $x[n] = (3)^{-n}u[n]$, $y[-1] = 2$, and $y[-2] = 3$, iteratively determine (first three terms only) the total response, ZIR, and ZSR for

$$y[n] + 2y[n-1] + y[n-2] = 2x[n] - x[n-1].$$

- 5.1-7** A time-varying parameter system to generate binomial coefficients (see Ex. 4.12) is, for $n = 1, 2, \dots, N$, characterized by the first-order difference equation

$$y[n] - \frac{N-n+1}{n}y[n-1] = 0.$$

Using $y[0] = {}^N_0 = 1$, iteratively solve this equation.

- 5.2-1** Express the systems of Prob. 5.1-3 using operator notation. In each case, clearly identify both $A(E)$ and $B(E)$.

- 5.2-2** Express $y[n] - 0.6y[n-1] - 0.16y[n-2] = 0$ using operator notation.

- 5.2-3** Express $y[n+1] - y[n] = x[n] + x[n-1]$ in operator notation. Identify both $A(E)$ and $B(E)$.

- 5.3-1** A person deposits a \$10,000 lottery prize at $n = -1$ in a bank savings account and makes no further deposits. The bank offers an annual percent yield (interest rate) of 12% per year (or $1 - (1.12)^{1/12}$ per month). Find the savings account balance $y[n]$, where n indicates the n th month.

- 5.3-2** Find the output of the digital integrator in Prob. 5.1-1 when the input $x[n] = 0$ and the initial condition is $y[-1] = 1$. Repeat the problem for the integrator in Prob. 5.1-2. Comment on the two results.

- 5.3-3** Using $y[-1] = 0$ and $y[-2] = 1$, solve

$$y[n+2] + 3y[n+1] + 2y[n] = 0.$$

- 5.3-4** Using $y[-1] = 1$ and $y[-2] = 1$, solve

$$y[n+2] + 2y[n+1] + y[n] = 0.$$

- 5.3-5** Using $y[-1] = 1$ and $y[-2] = 0$, solve

$$y[n+2] - 2y[n+1] + 2y[n] = 0.$$

- 5.3-6** Find $v[n]$, the voltage at the N th node of the resistive ladder depicted in Fig. P4.4-6 (page 267), if $V = 100$ volts and $a = 2$. The difference equation for $v[n]$ is given in Prob. 4.4-6, and the auxiliary conditions are $v[0] = 100$ and $v[N] = 0$.

5.4-1 Find the unit impulse response $h[n]$ for each of the following systems:

- (a) $y[n+1] + 2y[n] = x[n]$
- (b) $y[n+1] + 2y[n] = x[n+1]$
- (c) $(E^2 - 6E + 9)y[n] = Ex[n]$
- (d) $y[n] - 6y[n-1] + 25y[n-2] = 2x[n] - 4x[n-1]$

5.4-2 Consider the digital integrator in Prob. 5.1-1.

- (a) Realize the system using gain, delay, and summing blocks. Apply $\delta[n]$ at the input of the realization, and find the resulting impulse response $h[n]$ by inspection.
- (b) Determine the unit impulse response $h[n]$ by using the methods of Sec. 5.4.

5.4-3 Repeat Prob. 5.4-2 for the digital integrator in Prob. 5.1-2.

5.4-4 Repeat Prob. 5.4-2 for the savings account system characterized in Prob. 5.3-1.

5.4-5 Repeat Prob. 5.4-2 for a digital differentiator characterized by

$$y[n] = \frac{1}{T} (x[n] - x[n-1]),$$

where T is the sampling interval. Is this system recursive or nonrecursive in nature?

5.4-6 Find and sketch the unit impulse response $h[n]$ for the following systems:

- (a) an ideal backward difference system defined by the input-output relationship

$$y[n] = x[n] - x[n-1]$$

- (b) an ideal forward difference system defined by the input-output relationship

$$y[n] = x[n+1] - x[n]$$

- (c) a moving-average system defined by the input-output relationship

$$y[n] = \frac{1}{K_2 - K_1 + 1} \sum_{k=K_1}^{K_2} x[n-k]$$

(d) an accumulator defined by the input-output relationship

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]$$

5.5-1 Prove the six convolution sum properties enumerated in Sec. 5.5.1.

- 5.5-2** (a) Find the zero-state response $y[n]$ of an LTID system with input $x[n] = e^{-n}u[n]$ and impulse response $h[n]$ given by $(-2)^n u[n]$.
- (b) Repeat part (a) using impulse response $h[n] = \frac{1}{2} (\delta[n] - (-2)^n) u[n]$.
- (c) We showed in Eq. (5.28) that

$$s[n] = \sum_{k=-\infty}^n h[k].$$

To derive this result, we used the LTID system property that if $x[n] \Rightarrow y[n]$, then $\sum x[n] \Rightarrow \sum y[n]$. Derive this result without using this property.

5.5-3 Find the zero-state response $y[n]$ of an LTID system if the input $x[n]$ and the unit impulse response $h[n]$ are

- (a) $x[n] = (3)^{n+2}u[n]$ and $h[n] = [(2)^n + 3(-5)^n]u[n]$
- (b) $x[n] = (3)^{-n}u[n]$ and $h[n] = 3n(2)^n u[n]$

5.5-4 Find the zero-state response $y[n]$ of an LTID system if the input is $x[n] = (2)^n u[n]$ and the impulse response is

$$h[n] = (3)^n \cos\left(\frac{\pi}{3}n - 0.5\right) u[n].$$

5.5-5 Derive entries 2, 3, and 4 in Table 5.2. Table 5.1 may be useful.

5.5-6 Derive entries 5, 6, and 7 in Table 5.2. Table 5.1 may be useful.

5.5-7 Derive entries 8, 9, and 10 in Table 5.2. Table 5.1 may be useful.

5.5-8 For the digital differentiator specified in Ex. 5.3, find the zero-state response $y[n]$ for the following inputs:

- | | |
|---------------------|-----------------------------|
| (a) $u[n]$ | (b) $u[n] - u[n-10]$ |
| (c) $Tnu[n]$ | (d) $T^2 n^2 u[n]$ |

5.5-9 For the digital integrator of Prob. 5.1-1, find the zero-state response $y[n]$ for the following inputs:

- (a) $u[n]$
- (b) $u[n] - u[n - 10]$
- (c) $\delta[n]$
- (d) $Tnu[n]$

5.5-10 Repeat Prob. 5.5-9 for the digital integrator of Prob. 5.1-2.

5.5-11 For an LTID system with impulse response $h[n] = (0.5)^n u[n]$, find the zero-state response $y[n]$ for the following inputs:

- (a) $2^n u[n]$
- (b) $2^{(n-3)} u[n]$
- (c) $2^{-n} u[n]$
- (d) $2^n u[n - 2]$

5.5-12 Two subsystems H_1 and H_2 have impulse responses $h_1[n] = (1 + 2^{-n})u[n]$ and $h_2[n] = u[n] - u[n - 5]$, respectively.

- (a) Determine the overall impulse response h_p of the parallel connection of these subsystems, shown in Fig. P5.5-12a.
- (b) Determine the overall impulse response h_c of the cascade connection of these subsystems, shown in Fig. P5.5-12b.

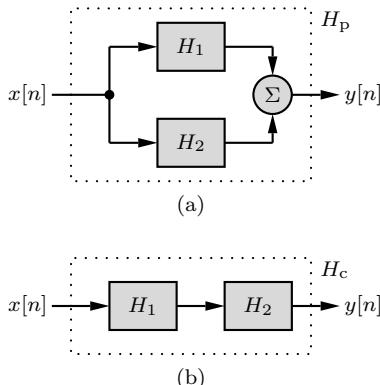


Figure P5.5-12

5.5-13 Repeat Prob. 5.5-12 if the impulse responses of the two systems are $h_1[n] = e^{-n}(u[n] - u[n - 5])$ and $h_2[n] = e^n(u[n] - u[n - 5])$.

5.5-14 Two subsystems H_1 and H_2 are connected in parallel and cascade, as shown

in Fig. P5.5-12. Let system H_1 have step response $s_1[n]$ and system H_2 have step response $s_2[n]$.

- (a) In terms of $s_1[n]$ and $s_2[n]$, what is the step response $s_p[n]$ of the parallel connection of these systems?
- (b) In terms of $s_1[n]$ and $s_2[n]$, what is the step response $s_c[n]$ of the cascade connection of these systems?

5.5-15 (a) Prove Eq. (5.29).

(b) Show that

$$x[n] = \sum_{m=-\infty}^{\infty} (x[m] - x[m-1]) u[n-m].$$

5.5-16 The impulse response of a causal, time-variant system is given by

$$h[n, m] = ma^n u[n - m],$$

where $h[n, m]$ defines the system response to an unit impulse $\delta[n - m]$.

- (a) Find the system response to the input $x[n] = u[n] - u[n - 11]$.
- (b) Find the system response to the input $x[n] = u[n]$.

5.5-17 Starting at $n = 0$, a person deposits \$500 at the beginning of every month with the exception of month $n = 4$, when instead of depositing \$500, she withdraws \$1,000 (a withdrawal is just a negative deposit). Find $y[n]$ if the interest rate is 1% per month. Because the deposit starts at $n = 0$, the initial condition $y[-1] = 0$.

5.5-18 Let right-sided signals $x[n]$ and $h[n]$ start at times $n = N_x$ and N_h , respectively. Show that $x[n] * h[n]$ starts at $N_x + N_h$.

5.5-19 Using the sliding-tape algorithm, show that

- (a) $u[n] * u[n] = (n + 1)u[n]$
- (b) $(u[n] - u[n - m]) * u[n] = (n + 1)u[n] - (n - m + 1)u[n - m]$

5.5-20 Using the sliding-tape algorithm, find $x[n] * h[n]$ for the signals shown in Fig. P5.5-20.

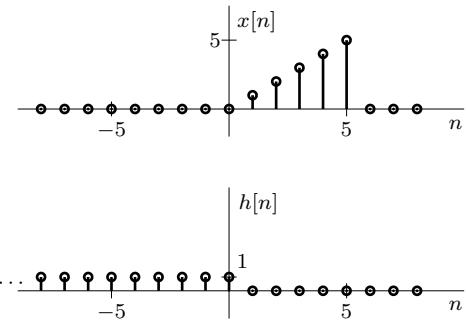


Figure P5.5-20

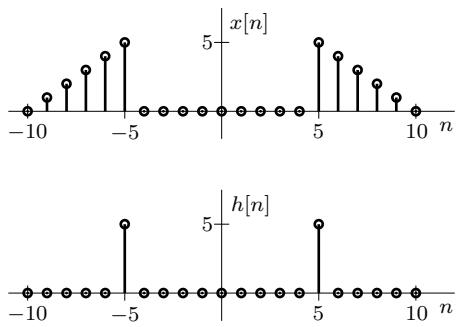


Figure P5.5-21

5.5-21 Repeat Prob. 5.5-20 for the signals shown in Fig. P5.5-21.

5.5-22 Repeat Prob. 5.5-20 for the signals depicted in Fig. P5.5-22. Compare this result with the result obtained if $x[n] = h[n] = u[n+3] - u[n-4]$.

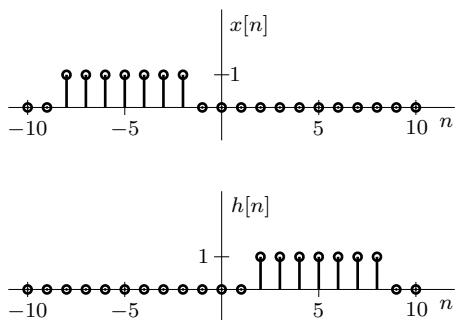


Figure P5.5-22

5.5-23 The convolution sum in Eq. (5.23) can be expressed in a matrix form as

$$\mathbf{y} = \mathbf{H}\mathbf{x},$$

where

$$\mathbf{y} = \begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ y[n] \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[n] \end{bmatrix}$$

and

$$\mathbf{H} = \begin{bmatrix} h[0] & 0 & \cdots & 0 \\ h[1] & h[0] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ h[n] & h[n-1] & \cdots & h[0] \end{bmatrix}.$$

Notice that matrix \mathbf{H} is a lower-triangular matrix (all elements above the main diagonal are zero). Knowing $h[n]$ and the output $y[n]$, we can determine the input $x[n]$ as

$$\mathbf{x} = \mathbf{H}^{-1}\mathbf{y}.$$

This operation is the reverse of the convolution and is known as *deconvolution*. Moreover, knowing $x[n]$ and $y[n]$, we can determine $h[n]$. This can be done by expressing the matrix equation as $n+1$ simultaneous equations in terms of $n+1$ unknowns $h[0], h[1], \dots, h[n]$. These equations can readily be solved iteratively. Thus, we can synthesize a system that yields a certain output $y[n]$ for a given input $x[n]$.

- (a) Design a system (i.e., determine $h[n]$) that will yield the output sequence $(8, 12, 14, 15, 15.5, \dots)$ for the input sequence $(1, 1, 1, 1, 1, \dots)$.
- (b) For a system with the impulse response sequence $(1, 2, 4, \dots)$, the output sequence is $(1, 7/3, 43/9, \dots)$. Determine the input sequence.

5.5-24 The sliding-tape method is conceptually quite valuable in understanding the convolution mechanism. With it, we can verify that DT convolution of Eq. (5.23) can be performed from an array using the sets $x[0], x[1], x[2], \dots$ and $h[0], h[1], h[2], \dots$, as depicted in Fig. P5.5-24. The (i, j) th element (element in the i th row and j th column) is given by $x[i]h[j]$. We add the elements of the array along diagonals to produce $y[n] = x[n] * h[n]$. For example, if we sum the elements corresponding to the first

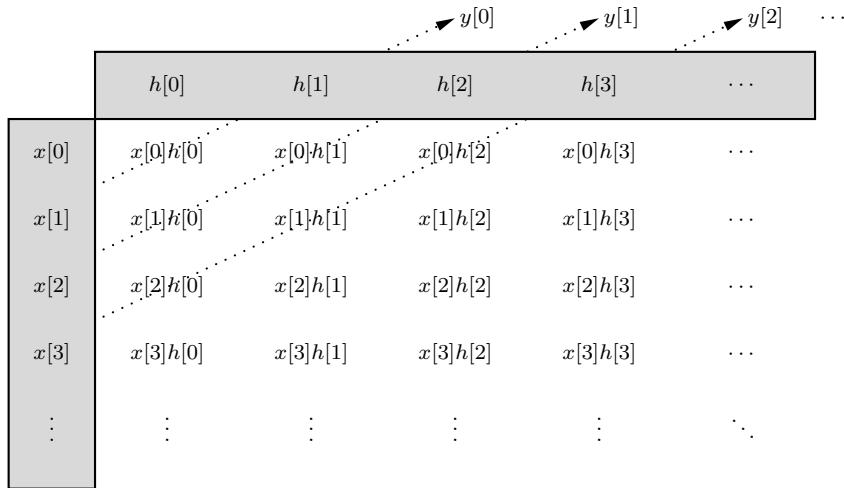


Figure P5.5-24

diagonal of the array, we obtain $y[0]$. Similarly, if we sum along the second diagonal, we obtain $y[1]$, and so on. Draw the array for the signals $x[n]$ and $h[n]$ in Ex. 5.15, and find $x[n] * h[n]$.

- 5.5-25** Find the transfer function $H(z)$ for the systems described by the following input-output relationships:

- (a) $y[n] = 2y[n-1] - y[n-2] + 2x[n] - x[n-1] + 3x[n-2]$
- (b) $y[n+2] + 0.5y[n+1] - 0.8y[n] = x[n+1] - 3x[n]$
- (c) $y[n] = x[n] + x[n-1] + x[n-2] + x[n-3] + x[n-4]$
- (d) $y[n+1] - y[n] = x[n+1] - x[n-4]$

Are the transfer functions in parts (c) and (d) related? Explain.

- 5.5-26** Find the transfer function $H(z)$ for the systems described by the following impulse response functions:

- (a) $h[n] = \delta[n] + 4\delta[n-2] - 2\delta[n-4]$
- (b) $h[n] = \gamma^n u[n]$, where $|\gamma| < 1$
- (c) $h[n] = \delta[n] + \delta[n-1] + \delta[n-2]$
- (d) $h[n] = h[n-1] + \delta[n] - \delta[n-3]$

Are the transfer functions in parts (c) and (d) related? Explain.

- 5.5-27** Find the transfer function $H(z)$ for a moving-average system specified by the input-output relationship

$$y[n] = \frac{1}{7} \sum_{k=0}^6 x[n-k].$$

- 5.5-28** Find the transfer function $H(z)$ for an accumulator system specified by a general input-output relationship

$$y[n] = \sum_{k=-\infty}^n x[n].$$

- 5.6-1** Using $y[-1] = 10$ and input $x[n] = e^{-n}u[n]$, find the total response of a system specified by the equation

$$y[n+1] + 2y[n] = x[n+1].$$

- 5.6-2** To pay off a loan of M dollars in N payments using a fixed monthly payment of P dollars, show that

$$P = \frac{rM}{1 - (1+r)^{-N}},$$

where r is the monthly interest rate.

Hint: Let the monthly payments of P dollars start at $n = 1$. Then, consider the loan as the initial condition $y_0[0] = -M$ and the input as $x[n] = Pu[n-1]$. The loan balance is the sum of the zero-input component (due to the initial condition) and the

zero-state component $h[n] * x[n]$. Alternatively, consider the loan as an input $-M$ at $n = 0$; that is, $-M\delta[n]$. The total input is then $x[n] = -M\delta[n] + Pu[n - 1]$, and the loan balance is $h[n]*x[n]$. Because the loan is paid off in N payments, set $y[N] = 0$.

- 5.6-3** A person receives an automobile loan of \$30,000 from a bank at an interest rate of 0.75% per month. His monthly payment is \$650, with the first payment due one month after he receives the loan. Compute the number of payments required to pay off the loan. Note that the last payment may not be exactly \$650.

Hint: Follow the procedure in Prob. 5.6-2 to determine the balance $y[n]$. To determine the number of payments, solve $y[K] = 0$ for K . In general, K will not be an integer. The number of payments N is the largest integer $\leq K$, and the residual payment is $|y[N]|$.

- 5.7-1** Each of the following equations specifies an LTID system. Determine whether these systems are asymptotically stable, unstable, or marginally stable. Determine also whether they are BIBO stable or unstable.

- (a) $y[n+2] + 0.6y[n+1] - 0.16y[n] = x[n+1] - 2x[n]$
- (b) $(E^2 + 1)(E^2 + E + 1) \{y[n]\} = E \{x[n]\}$
- (c) $(E - 1)^2(E + \frac{1}{2}) \{y[n]\} = (E + 2) \{x[n]\}$
- (d) $y[n] + 2y[n-1] + 0.96y[n-2] = 2x[n-1] + 3x[n-3]$
- (e) $(E^2 - 1)(E^2 + 1) \{y[n]\} = x[n]$

- 5.7-2** Repeat Prob. 5.7-1 for the digital integrator described in Prob. 5.1-2.

- 5.7-3** Repeat Prob. 5.7-1 for the systems described in Prob. 5.4-6.

- 5.7-4** Discuss the stability (asymptotic, marginal, and BIBO) of a K th-order, causal, nonrecursive LTID system described by Eq. (5.4).

- 5.7-5** In Sec. 5.7.1, we showed that for BIBO stability in an LTID system, it is sufficient for its impulse response to satisfy Eq. (5.36). Show that this is also a necessary condition for the system to be BIBO stable. In

other words, show that if Eq. (5.36) is not satisfied, there exists a bounded input that produces unbounded output.

Hint: Assume that a system exists for which $h[n]$ violates Eq. (5.36), yet its output is bounded for every bounded input. Establish the contradiction in this statement by considering an input $x[n]$ defined by $x[n_1 - m] = 1$ when $h[m] > 0$ and $x[n_1 - m] = -1$ when $h[m] < 0$, where n_1 is some fixed integer.

- 5.8-1** Consider a system with impulse response $h[n] = -n(0.5)^n u[n]$. Determine this system's time constant, rise time, and pulse dispersion. Identify, if possible, an input $x[n]$ that will cause this system to resonate.

- 5.8-2** Repeat Prob. 5.8-1 for the system described by $h[n] = (0.9)^n u[n] + 2(-.3)^n u[n]$.

- 5.8-3** Repeat Prob. 5.8-1 for the system described by $h[n] = 2u[n] - u[n-5] - u[n-10]$.

- 5.9-1** For input $x[n] = e^{-n} u[n]$ and auxiliary condition $y[0] = 1$, use the classical method to solve

$$y[n+1] + 2y[n] = x[n+1].$$

- 5.9-2** For input $x[n] = e^{-n} u[n]$ and initial condition $y[-1] = 0$, use the classical method to solve

$$y[n] + 2y[n-1] = x[n-1].$$

Hint: You will have to determine the auxiliary condition $y[0]$ using the iterative method.

- 5.9-3** An input $x[n] = (3)^n$ is applied to a system described by

$$\begin{aligned} y[n+2] + 3y[n+1] + 2y[n] = \\ x[n+2] + 3x[n+1] + 3x[n]. \end{aligned}$$

- (a) Use the classical method and auxiliary conditions $y[0] = 1$ and $y[1] = 3$ to solve this problem.
- (b) Use the classical method and initial conditions $y[-1] = y[-2] = 1$ to solve this problem. Hint: Determine $y[0]$ and $y[1]$ iteratively.

- 5.9-4** For input $x[n] = 3^{-n}u[n]$ and auxiliary conditions $y[0] = 2$ and $y[1] = -\frac{13}{3}$, use the classical method to solve

$$y[n] + 2y[n-1] + y[n-2] = 2x[n] - x[n-1].$$

- 5.9-5** Using the classical method, find the following sums:

(a) $\sum_{k=0}^n k$ (b) $\sum_{k=0}^n k^3$

- 5.9-6** Using the classical method, solve

$$(E^2 - E + 0.16) \{y[n]\} = E \{x[n]\}$$

with the input $x[n] = (0.2)^n u[n]$ and the auxiliary conditions $y[0] = 1$ and $y[1] = 2$. Hint: The input is a natural mode of the system.

- 5.9-7** Using the classical method, solve

$$(E^2 - E + 0.16) \{y[n]\} = E \{x[n]\}$$

with the input $x[n] = \cos(\frac{\pi n}{2} + \frac{\pi}{3})u[n]$ and the initial conditions $y[-1] = y[-2] = 0$. Hint: Find $y[0]$ and $y[1]$ iteratively.

- 5.9-8** Use classical method to find the unit step response of the system in Ex. 5.9. Next, use Eq. (5.29) to determine the impulse response $h[n]$ for this system.

Chapter 6

Discrete-Time Fourier Analysis

This chapter treats frequency-domain analysis by developing the discrete-time Fourier transform (DTFT) and the spectral representation of discrete-time signals. The development, properties, and applications of the DTFT closely parallel those for the continuous-time Fourier transform. In fact, the DTFT and the CTFT are closely related, a fact that we can use to our advantage in many applications. Discrete-time Fourier analysis provides an important tool for the study of discrete-time signals and systems. A frequency-domain perspective is particularly useful to better understand the digital processing of analog signals and digital resampling techniques. In preparation for the next chapter, this chapter concludes by generalizing the DTFT to the z -transform.

6.1 The Discrete-Time Fourier Transform

The *continuous-time Fourier transform* (CTFT), sometimes simply called the *Fourier transform* (FT), is a tool to represent an aperiodic continuous-time signal in terms of its frequency components, thereby providing a spectral representation of the signal. We now develop a similar tool, the *discrete-time Fourier transform* (DTFT), to represent an aperiodic discrete-time signal in terms of its frequency components, which also leads to a spectral representation of the signal. The principal difference between the two types of transforms is that the former represents signals with continuous-time sinusoids or exponentials, while the latter uses discrete-time sinusoids or exponentials.

One possible way to develop the continuous-time Fourier transform is to start with the Fourier series representation of periodic signals and then, letting the period go to infinity, extend the results to aperiodic signals. This approach, considered in Sec. 1.8 of Ch. 1 for CT signals, is intuitive and provides a physical appreciation of the spectral representation. Although we could follow the same intuitive treatment to develop the DTFT, we shall avoid this redundancy and instead take a direct approach to derive the DTFT. Later, we show that the discrete-time Fourier series, which is used to analyze periodic discrete-time signals, is a special case of the DTFT.

We define $X(\Omega)$, the discrete-time Fourier transform of $x[n]$, as

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}. \quad (6.1)$$

This *analysis equation* of the DTFT identifies the spectral components of $x[n]$. It provides a frequency-domain description of $x[n]$. Since $X(\Omega)$ is constructed exclusively from 2π -periodic functions $e^{-j\Omega n}$, it too is a 2π -periodic function of Ω . Often, $X(\Omega)$ is referred to as the *spectrum* of $x[n]$.

To find the inverse relationship, or inverse DTFT, we change the dummy variable n to m in

Eq. (6.1), multiply both sides by $e^{j\Omega n}/2\pi$, and then integrate over the interval $-\pi \leq \Omega < \pi$ as

$$\begin{aligned} \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega &= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left(\sum_{m=-\infty}^{\infty} x[m] e^{-j\Omega m} \right) e^{j\Omega n} d\Omega \\ &= \sum_{m=-\infty}^{\infty} x[m] \left(\frac{1}{2\pi} \int_{-\pi}^{\pi} e^{j\Omega(n-m)} d\Omega \right). \end{aligned}$$

Using the identity $\frac{1}{2\pi} \int_{-\pi}^{\pi} e^{j\Omega(n-m)} d\Omega = \delta[n-m]$, we can write this equation as[†]

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega = \sum_{m=-\infty}^{\infty} x[m] \delta[n-m].$$

According to the sampling property of the impulse (Eq. (4.10)), the right-hand side is $x[n]$, and we obtain the desired inverse relationship we seek,

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega. \quad (6.2)$$

This *synthesis equation* of the DTFT represents a signal $x[n]$ as a continuous sum (integral) of complex exponentials, each of which is weighted by the signal spectrum $X(\Omega)$.

Equations (6.1) and (6.2) define the DTFT pair. We call $X(\Omega)$ the (*direct*) discrete-time Fourier transform (DTFT) of $x[n]$. Conversely, $x[n]$ is the *inverse* discrete-time Fourier transform (IDTFT) of $X(\Omega)$. This can be represented as

$$X(\Omega) = \text{DTFT}\{x[n]\} \quad \text{and} \quad x[n] = \text{IDTFT}\{X(\Omega)\}$$

or, more compactly, as

$$x[n] \iff X(\Omega).$$

Physical Appreciation of the Discrete-Time Fourier Transform

The physical interpretation of $X(\Omega)$, the DTFT of $x[n]$, is very similar to that of $X(\omega)$, the continuous-time Fourier transform of a signal $x(t)$. Referring to Eq. (6.2) for the DTFT and Eq. (1.74) for the CTFT, we see that the two synthesis equations are essentially identical, which means that the spectra $X(\Omega)$ and $X(\omega)$ serve equivalent roles.

In understanding any aspect of the Fourier transform, we should remember that the Fourier representation is a way of expressing a signal $x[n]$ as a sum of everlasting discrete-time exponentials (or sinusoids). The Fourier spectrum of a signal indicates the relative magnitudes and phases of the discrete-time exponentials (or sinusoids) required to synthesize $x[n]$. We can reinforce these ideas by expressing Eq. (6.2) as

$$\begin{aligned} x[n] &= \lim_{\Delta\Omega \rightarrow 0} \sum_{k\Delta\Omega=-\pi}^{\pi} \left(X(k\Delta\Omega) \frac{\Delta\Omega}{2\pi} \right) e^{j(k\Delta\Omega)n} \\ &= \lim_{\Delta\Omega \rightarrow 0} \sum_{k\Delta\Omega=-\pi}^{\pi} (X(k\Delta\Omega) \Delta\mathcal{F}) e^{j(k\Delta\Omega)n}. \end{aligned}$$

[†]To prove this identity, let $k = n - m$ so that $k = 0$ when $n = m$ and k is a nonzero integer when $n \neq m$. For integer k , we thus have

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} e^{j\Omega k} d\Omega = \frac{1}{j2\pi k} e^{j\Omega k} \Big|_{-\pi}^{\pi} = \frac{1}{j2\pi k} [e^{j\pi k} - e^{-j\pi k}] = \text{sinc}(k) = \delta[k].$$

The last step follows from the fact that $\text{sinc}(k) = \frac{\sin(\pi k)}{\pi k} = 0$ when k is any nonzero integer ($m \neq n$) and $\text{sinc}(k) = \frac{\sin(\pi k)}{\pi k} = 1$ when $k = 0$ ($m = n$).

Thus, the Fourier integral appearing on the right-hand side of Eq. (6.2) can be interpreted as a sum of exponentials of the form $e^{j(k\Delta\Omega)n}$. The amplitude of the exponential $e^{j(k\Delta\Omega)n}$ is $X(k\Delta\Omega)\Delta\mathcal{F}$, and we sum all these discrete-time exponentials over the frequency band from $-\pi$ to π to synthesize $x[n]$. We are basically expressing an arbitrary signal $x[n]$ as a sum of its exponential (or sinusoidal) components. Thus, the function $X(\Omega)$ found in Eq. (6.1) acts as a spectral function, which indicates the relative amounts of the various exponential (or sinusoidal) components of $x[n]$.

▷ **Example 6.1 (DTFT of a Rectangular Pulse)**

Determine the DTFT $X(\Omega)$ of a discrete-time rectangular pulse with odd length L_x ,

$$x[n] = u[n + (L_x - 1)/2] - u[n - (L_x + 1)/2].$$

Sketch $x[n]$ and $X(\Omega)$ for $L_x = 9$.

The pulse $x[n]$, illustrated in Fig. 6.1a for $L_x = 9$, is an L_x -point rectangular window function. Using Eq. (6.1),

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} = \sum_{n=-\frac{L_x-1}{2}}^{\frac{L_x-1}{2}} (e^{-j\Omega})^n.$$

This is a geometric progression with a common ratio $e^{-j\Omega}$, and according to entry 1 of Table 5.1,

$$\begin{aligned} X(\Omega) &= \frac{e^{j\Omega\frac{L_x-1}{2}} - e^{-j\Omega\frac{L_x+1}{2}}}{1 - e^{-j\Omega}} \\ &= \frac{e^{-j\Omega/2} \left(e^{j\frac{L_x}{2}\Omega} - e^{-j\frac{L_x}{2}\Omega} \right)}{e^{-j\Omega/2}(e^{j\Omega/2} - e^{-j\Omega/2})} \\ &= \frac{\sin(L_x\Omega/2)}{\sin(\Omega/2)}. \end{aligned} \quad (6.3)$$

Observe that $X(\Omega)$ is a 2π -periodic function. Figure 6.1b shows the result for $L_x = 9$, $X(\Omega) = \sin(4.5\Omega)/\sin(0.5\Omega)$. As the plot of $X(\Omega)$ makes clear, $x[n]$ is comprised primarily of relatively low-frequency components.

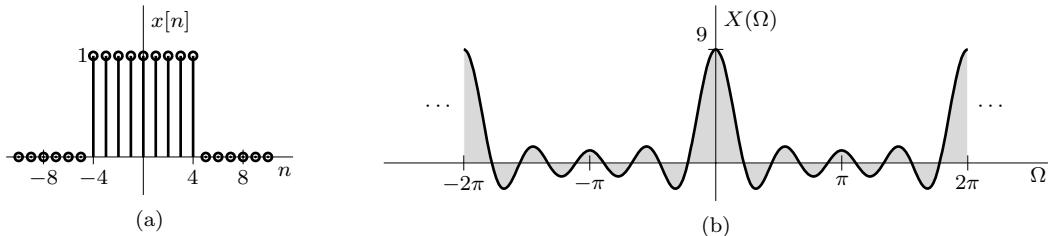


Figure 6.1: (a) Length-9 rectangular pulse $x[n]$ and (b) its DTFT $X(\Omega)$.

Example 6.1 ◀

▷ **Example 6.2 (IDTFT of a Rectangular Pulse)**

Determine the IDTFT $x[n]$ of the 2π -periodic rectangular pulse train described over the fundamental band $|\Omega| \leq \pi$ by $X(\Omega) = \Pi(\frac{\Omega}{2B})$, where $B \leq \pi$. Sketch $X(\Omega)$ and $x[n]$ for $B = \pi/4$.

Using $B = \pi/4$, the pulse train $X(\Omega)$ is illustrated in Fig. 6.2a. Notice that the width of each pulse is $2B$. Further, using Eq. (6.2), the IDTFT is

$$\begin{aligned}
x[n] &= \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{jn\Omega} d\Omega \\
&= \frac{1}{2\pi} \int_{-B}^B e^{jn\Omega} d\Omega \\
&= \frac{1}{j2\pi n} e^{jn\Omega} \Big|_{-B}^B \\
&= \frac{\sin(Bn)}{\pi n} \\
&= \frac{B}{\pi} \operatorname{sinc}\left(\frac{Bn}{\pi}\right).
\end{aligned} \tag{6.4}$$

Figure 6.2b depicts the signal $x[n]$ for the case $B = \pi/4$.

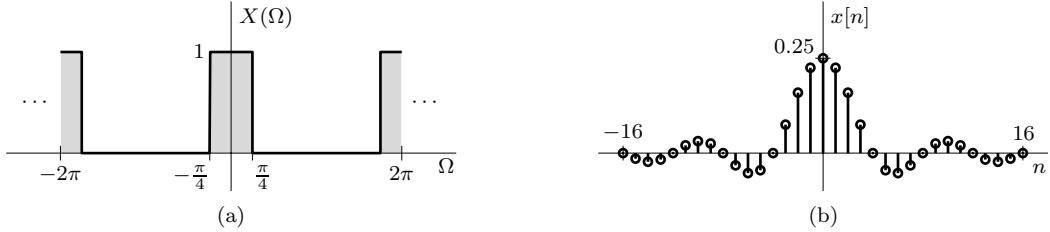


Figure 6.2: (a) Width $\pi/2$ rectangular pulse train $X(\Omega)$ and (b) its IDTFT $x[n]$.

Example 6.2 ◀

Magnitude and Phase Spectra

The DTFT provides the spectrum $X(\Omega)$ of a discrete-time signal $x[n]$. The spectrum $X(\Omega)$ is generally complex and is represented in polar form as

$$X(\Omega) = |X(\Omega)| e^{j\angle X(\Omega)}. \tag{6.5}$$

In Eq. (6.5), the quantities $|X(\Omega)|$ and $\angle X(\Omega)$ are both real functions of Ω and represent the magnitude spectrum and the phase spectrum, respectively.

As in the continuous-time case, a real signal $x[n]$ has a conjugate symmetric spectrum; that is, $X(\Omega) = X^*(-\Omega)$. This can be readily proved from Eq. (6.1). Thus, for real $x[n]$,

$$\underbrace{|X(\Omega)| e^{j\angle X(\Omega)}}_{X(\Omega)} = \underbrace{|X(-\Omega)| e^{-j\angle X(-\Omega)}}_{X^*(-\Omega)},$$

which means that

$$|X(\Omega)| = |X(-\Omega)| \quad \text{and} \quad \angle X(\Omega) = -\angle X(-\Omega). \tag{6.6}$$

In other words, a real signal $x[n]$ has an even magnitude spectrum $|X(\Omega)|$ and an odd phase spectrum $\angle X(\Omega)$. The Fourier transform and the Fourier series both exhibit these exact same symmetries.

Existence of the DTFT

Because $|e^{-j\Omega n}| = 1$, it follows from Eq. (6.1) that the existence of $X(\Omega)$ is guaranteed if $x[n]$ is absolutely summable; that is, $X(\Omega)$ exists if

$$\sum_{n=-\infty}^{\infty} |x[n]| < \infty. \quad (6.7)$$

The absolute summability of $x[n]$ is a sufficient, although not necessary, condition for the existence of the DTFT representation. This condition also guarantees uniform convergence.

Much like the CTFT, the DTFT exists for a broader class of signals beyond those that are absolutely summable. This is fortunate since a number of useful DT signals do not satisfy Eq. (6.7), such as sinc functions and sinusoids. In such cases, however, the DTFT may exist in only a generalized sense or with weaker-than-uniform convergence. A signal with finite energy, for example, has

$$E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 \leq \left(\sum_{n=-\infty}^{\infty} |x[n]| \right)^2 < \infty. \quad (6.8)$$

Clearly, finite energy is a weaker condition than absolute summability. In other words, not all finite-energy signals are absolutely summable. Still, one can show that the DTFT of finite-energy signals is guaranteed to exist, although it converges in the mean rather than uniformly.[†] For example, the sinc function $x[n] = \text{sinc}[n]$ is not absolutely summable, thus violating Eq. (6.7), but it has finite energy, thus satisfying Eq. (6.8). Consequently, the DTFT of $\text{sinc}[n]$ exists, although it converges in the mean and not uniformly.

If generalized functions such as $\delta(\Omega)$ are permitted, then we can even find the DTFT of some signals that violate both Eq. (6.7) and Eq. (6.8).[‡] Bounded-amplitude power signals, for example, are neither finite energy nor absolutely summable, yet they have DTFTs, at least in a generalized sense. To provide two examples, the DTFT of $x[n] = 1$ is $X(\Omega) = 2\pi\delta(\Omega)$ over the fundamental band $|\Omega| \leq \pi$, and the DTFT of $e^{j\Omega_0 n}$ is $2\pi\delta(\Omega - \Omega_0)$ over the same band (see Ex. 6.5).

Despite our best hopes and efforts, however, some signals remain forever intransigent and refuse to be characterized by the DTFT. For example, an exponentially growing signal $\gamma^n u[n]$, where $|\gamma| > 1$, violates both Eq. (6.7) and Eq. (6.8) and does not have a DTFT, even in a generalized sense. That the DTFT does not exist for such signals simply indicates that it takes more than a sum of sinusoids to represent the signal. To analyze such signals requires a more powerful transform, such as the z -transform discussed in the next chapter.

► Example 6.3 (Magnitude and Phase Spectra of a Causal Exponential)

Determine the magnitude and phase spectra of the causal exponential $x[n] = \gamma^n u[n]$.

From Eq. (6.1), we see that

$$X(\Omega) = \sum_{n=0}^{\infty} \gamma^n e^{-j\Omega n} = \sum_{n=0}^{\infty} (\gamma e^{-j\Omega})^n.$$

[†]Convergence in the mean implies that

$$\lim_{N \rightarrow \infty} \int_{-\pi}^{\pi} \left| X(\Omega) - \sum_{n=-N}^N x[n] e^{-j\Omega n} \right|^2 d\Omega = 0.$$

[‡]A note of caution: the continuous-time $\delta(\cdot)$ (a generalized function) should not be confused with the discrete-time $\delta[\cdot]$ (an ordinary function). The former has a unit area concentrated at one point, the origin, whereas the latter has a unit value at the origin.

This is an infinite geometric series with a common ratio $\gamma e^{-j\Omega}$. As long as $|\gamma e^{-j\Omega}| = |\gamma| < 1$, the sum converges to (see Table 5.1)

$$X(\Omega) = \frac{1}{1 - \gamma e^{-j\Omega}} = \frac{1}{1 - \gamma \cos(\Omega) + j\gamma \sin(\Omega)}.$$

For $|\gamma| < 1$, the magnitude and phase spectra are thus

$$|X(\Omega)| = \frac{1}{\sqrt{1 + \gamma^2 - 2\gamma \cos(\Omega)}} \quad \text{and} \quad \angle X(\Omega) = -\tan^{-1} \left(\frac{\gamma \sin(\Omega)}{1 - \gamma \cos(\Omega)} \right).$$

If $|\gamma| > 1$, then $X(\Omega)$ does not converge.

Figure 6.3 shows $x[n] = \gamma^n u[n]$ and its spectra for $\gamma = 0.8$. Observe that the magnitude spectrum $|X(\Omega)|$ is an even function and the phase spectrum $\angle X(\Omega)$ is an odd function. Further, the frequency spectra are continuous and 2π -periodic functions of Ω . It is entirely sufficient to specify or display the spectra of DT signals over a frequency interval of only 2π , such as the fundamental frequency range $[-\pi, \pi]$.

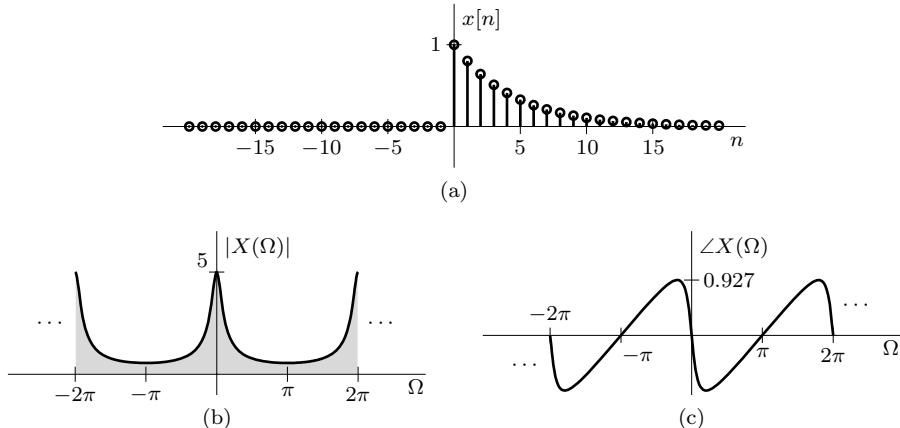


Figure 6.3: Causal exponential and its spectra: (a) $x[n] = (0.8)^n u[n]$, (b) $|X(\Omega)|$, and (c) $\angle X(\Omega)$.

Example 6.3 ◁

▷ Example 6.4 (Magnitude and Phase Spectra of an Anti-Causal Exponential)

Determine the magnitude and phase spectra of the anti-causal exponential $y[n] = -\gamma^n u[-n-1]$.

The anti-causal exponential $y[n] = -\gamma^n u[-n-1]$, shown in Fig. 6.4 for $\gamma = \frac{1}{0.8}$, is really just a left-sided version of the causal exponential $x[n] = \gamma^n u[n]$, shown in Fig. 6.3a for $\gamma = 0.8$. Given the strong resemblance between these two signals, we expect to find similarity in their spectra as well.

From Eq. (6.1), we see that

$$Y(\Omega) = \sum_{n=-\infty}^{-1} -\gamma^n e^{-j\Omega n} = - \sum_{n=-\infty}^{-1} (\gamma e^{-j\Omega})^n.$$

This is an infinite geometric series with a common ratio $\gamma e^{-j\Omega}$, just like in Ex. 6.3. Unlike Ex. 6.3, however, which had sum limits from 0 to ∞ , this sum has limits of $-\infty$ to -1 . Only when $|\gamma e^{-j\Omega}| =$

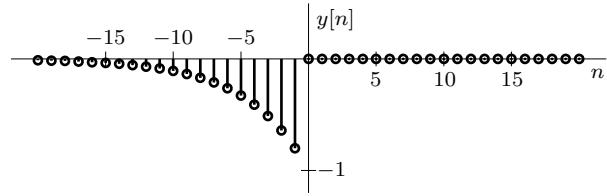


Figure 6.4: Anti-causal exponential $y[n] = -\gamma^n u[-n-1]$ for $\gamma = 1/0.8$.

$|\gamma| > 1$ does the sum converge to (see Table 5.1)

$$Y(\Omega) = -\frac{0-1}{1-\gamma e^{-j\Omega}} = \frac{1}{1-\gamma \cos(\Omega) + j\gamma \sin(\Omega)}.$$

For $|\gamma| > 1$, the magnitude and phase spectra are thus

$$|Y(\Omega)| = \frac{1}{\sqrt{1+\gamma^2 - 2\gamma \cos(\Omega)}} \quad \text{and} \quad \angle Y(\Omega) = -\tan^{-1} \left(\frac{\gamma \sin(\Omega)}{1-\gamma \cos(\Omega)} \right).$$

If $|\gamma| < 1$, then $Y(\Omega)$ does not converge.

Notice that the DTFT for this signal is identical to that of $x[n] = \gamma^n u[n]$ (see Ex. 6.3). Yet there is no ambiguity in determining the IDTFT of $\frac{1}{1-\gamma e^{-j\Omega}}$ because of the restrictions on the value of γ in each case. If $|\gamma| < 1$, then the inverse transform is $\gamma^n u[n]$. For $|\gamma| > 1$, it is $-\gamma^n [-n-1]$.

Example 6.4 □

▷ **Drill 6.1 (Magnitude and Phase Spectra of a Two-Sided Exponential)**

By direct calculation, determine the DTFT of the two-sided exponential $x[n] = \gamma^{|n|}$. Using $\gamma = 0.8$, sketch the resulting spectrum $X(\Omega)$.

□

6.1.1 The Nature of Fourier Spectra

We now discuss several important features of the discrete-time Fourier transform and the spectra associated with it.

Fourier Spectra are Continuous Functions of Ω

Those who are visiting for the first time the brave new world of discrete-time often think that everything here must be discrete. This is not true. Although $x[n]$ is a discrete-time signal, its DTFT $X(\Omega)$ is a continuous function of Ω for the simple reason that Ω is a continuous variable, which can take any value over a continuous interval from $-\infty$ to ∞ .

Fourier Spectra are 2π -Periodic Functions of Ω

We have observed earlier that a DTFT $X(\Omega)$ is a periodic spectrum with period 2π . We now show this formally. From Eq. (6.1), it follows that

$$X(\Omega + 2\pi) = \sum_{n=-\infty}^{\infty} x[n] e^{-j(\Omega+2\pi)n} = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n} \underbrace{e^{-j2\pi n}}_1 = X(\Omega).$$

Clearly, the spectrum $X(\Omega)$ is a continuous and periodic function of Ω with period 2π . Nothing can deprive the DTFT of its 2π -periodicity, a feature that is built into the very definition of the DTFT. How do we explain this intuitively? Recall from Ch. 4 that discrete-time frequency is bandlimited to frequencies in the range $-\pi \leq \Omega < \pi$ (fundamental band). Indeed, this is the reason that Eq. (6.2) synthesizes $x[n]$ by integrating only over the fundamental band. Frequencies outside this band can be viewed merely as the frequencies in the fundamental band called by other names. The same frequency repeats at an interval of 2π . Hence, the spectrum must repeat periodically with period 2π . Because discrete-time signals are, in a real sense, bandlimited to the fundamental range, it is not necessary to represent or display signal spectra beyond the fundamental range, although we often do so.

Generalized Limits of Integration for the DTFT Synthesis Equation

We know the spectrum $X(\Omega)$ is 2π -periodic. Moreover, $e^{j\Omega n}$ is also 2π -periodic because $e^{j\Omega n} = e^{j(\Omega+2\pi)n}$. Hence, the integrand $X(\Omega)e^{j\Omega n}$ of the DTFT synthesis equation (Eq. (6.2)) is also 2π -periodic. Now, the area under a periodic function over one period is always the same no matter where we start integrating. In other words, the integral on the right-hand side of Eq. (6.2) over the range $q \leq \Omega < q + 2\pi$ is the same regardless of the value of q . Thus, Eq. (6.2) can be generalized as

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega)e^{j\Omega n} d\Omega, \quad (6.9)$$

where the symbol $\int_{2\pi}$ indicates integral over a range of 2π starting at any value. We may use a range 0 to 2π , π to 3π , or any other interval that is 2π in width.

6.1.2 Obtaining the DTFT from the CTFT

We have already noted the similarity of the CTFT and DTFT synthesis equations, and we now make an observation that allows us to derive a DTFT pair from a CTFT pair. A comparison of Eq. (6.2) with Eq. (1.74) shows that the two Fourier integrals are identical when the CTFT $X(\omega)$ is bandlimited, either naturally or following frequency scaling, to $|\omega| \leq \pi$. Thus, when $x(t) \iff X(\omega)$, where $X(\omega)$ is bandlimited to $|\omega| \leq \pi$, the same pair is also a DTFT pair if we substitute n for t in $x(t)$ and Ω for ω in $X(\omega)$. This gives the DTFT in the fundamental band $|\Omega| \leq \pi$. To find the DTFT for all Ω , we repeat this spectrum periodically with period 2π . The following example clarifies this idea.

▷ Example 6.5 (Obtaining the DTFT from the CTFT)

For each of the following signals, determine the DTFT from the CTFT.

(a) $x[n] = \frac{B}{\pi} \operatorname{sinc}\left(\frac{Bn}{\pi}\right)$	(b) $x[n] = 1$
(c) $x[n] = e^{j\Omega_0 n}$	(d) $x[n] = \cos(\Omega_0 n)$

(a) We examine the CTFT Table 1.1 to see if any pair has a bandlimited $X(\omega)$ with the corresponding $x(t)$ in the form of $\operatorname{sinc}(Bt/\pi)$. Fortunately, pair 8 makes our day. This pair is given by

$$\frac{B}{\pi} \operatorname{sinc}\left(\frac{Bt}{\pi}\right) \iff \Pi\left(\frac{\omega}{2B}\right).$$

The CTFT in this pair is bandlimited to $|\omega| \leq B$. If we choose $0 < B \leq \pi$, then $X(\omega)$ is bandlimited to $|\omega| \leq \pi$. Now, $x[n]$ is obtained by letting $t = n$ in $x(t)$ and $\omega = \Omega$ in $X(\omega)$. Hence, for $0 < B \leq \pi$, we have the DTFT pair in the fundamental band as

$$\frac{B}{\pi} \operatorname{sinc}\left(\frac{Bn}{\pi}\right) \iff \Pi\left(\frac{\Omega}{2B}\right), \quad |\Omega| \leq \pi.$$

To find the DTFT for all Ω , we repeat the spectrum periodically with period 2π to obtain

$$\frac{B}{\pi} \operatorname{sinc}\left(\frac{Bn}{\pi}\right) \iff \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{2B}\right), \quad 0 < B \leq \pi.$$

This is the same DTFT pair found in Ex. 6.2.

(b) From pair 13 of CTFT Table 1.1, we see that a dc constant $x(t) = 1$ has a bandlimited spectrum $X(\omega) = 2\pi\delta(\omega)$. This pair is given by

$$1 \iff 2\pi\delta(\omega).$$

We obtain the desired $x[n]$ by letting $t = n$ in $x(t)$ and $\omega = \Omega$ in $X(\omega)$. This results in $x[n] = 1$ for all n and $X(\Omega) = 2\pi\delta(\Omega)$. Hence, we have the DTFT pair

$$1 \iff 2\pi\delta(\Omega), \quad |\Omega| \leq \pi.$$

This describes $X(\Omega)$ in the fundamental band only. The DTFT over all Ω is given by

$$1 \iff 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k). \quad (6.10)$$

(c) Pair 16 from CTFT Table 1.1 shows that $x(t) = e^{j\omega_0 t}$ has a spectrum $X(\omega)$ that is bandlimited to π if $\omega_0 \leq \pi$. This pair is given by

$$e^{j\omega_0 t} \iff 2\pi\delta(\omega - \omega_0).$$

We obtain the desired pair by letting $t = n$ in $x(t)$ and $\omega = \Omega$ in $X(\omega)$. As long as $|\Omega_0| < \pi$, this results in the DTFT pair

$$e^{j\Omega_0 n} \iff 2\pi\delta(\Omega - \Omega_0), \quad |\Omega| \leq \pi.$$

This describes $X(\Omega)$ in the fundamental band only. Over the entire range of Ω , the DTFT pair is given by

$$e^{j\Omega_0 n} \iff 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k). \quad (6.11)$$

(d) Pair 17 from CTFT Table 1.1 shows that $x(t) = \cos(\omega_0 t)$ has a spectrum $X(\omega)$ that is bandlimited to π if $|\omega_0| \leq \pi$. This pair is given by

$$\cos(\omega_0 t) \iff \pi[\delta(\omega - \omega_0) + \delta(\omega + \omega_0)].$$

We obtain the desired pair by letting $t = n$ in $x(t)$ and $\omega = \Omega$ in $X(\omega)$. Assuming $|\Omega_0| < \pi$, this results in

$$\cos(\Omega_0 n) \iff \pi[\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)], \quad |\Omega| \leq \pi.$$

This describes $X(\Omega)$ in the fundamental band only. The DTFT over all Ω is given by

$$\cos(\Omega_0 n) \iff \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k).$$

Example 6.5 \triangleleft

▷ Drill 6.2 (Verification of DTFT Pairs)

Verify the DTFT pair of Eq. (6.10) by taking the inverse DTFT of $X(\Omega)$ to obtain $x[n]$. In the same way, verify the DTFT pair of Eq. (6.11).

\triangleleft

▷ **Drill 6.3 (DTFT Pairs for Generalized Sinusoids)**

Show that

$$\cos(\Omega_0 n + \theta) \iff \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) e^{j\theta} + \delta(\Omega + \Omega_0 - 2\pi k) e^{-j\theta}.$$

What is the DTFT of the generalized sinusoid $\sin(\Omega_0 n + \theta)$? △

6.1.3 DTFT Tables and the Nuisance of Periodicity

Table 6.1 gives a selection of commonly encountered DTFT pairs. As its nature requires, each DTFT in this table is 2π -periodic. The DTFTs in pairs 1–7 have periodicity built into the expression $X(\Omega)$. The remaining pairs, however, require periodic replication in the form of an infinite sum to achieve periodicity. These infinite sums make the DTFTs appear dreadfully complex compared with the gentle CTFTs of Table 1.1. Such complexity diminishes the intuitive nature of the spectra. Further, it is inconvenient to deal with such unwieldy expressions. How many of us would be enthused about convolving two periodic spectra?

$x[n]$	$X(\Omega)$	
1. $\delta[n - k]$	$e^{-jk\Omega}$	Integer k
2. $\gamma^n u[n]$	$\frac{e^{j\Omega}}{e^{j\Omega} - \gamma}$	$ \gamma < 1$
3. $-\gamma^n u[-n - 1]$	$\frac{e^{j\Omega}}{e^{j\Omega} - \gamma}$	$ \gamma > 1$
4. $\gamma^{ n }$	$\frac{1 - \gamma^2}{1 - 2\gamma \cos(\Omega) + \gamma^2}$	$ \gamma < 1$
5. $n\gamma^n u[n]$	$\frac{\gamma e^{j\Omega}}{(e^{j\Omega} - \gamma)^2}$	$ \gamma < 1$
6. $ \gamma ^n \cos(\Omega_0 n + \theta) u[n]$	$\frac{e^{j\Omega} [e^{j\Omega} \cos(\theta) - \gamma \cos(\Omega_0 - \theta)]}{e^{j2\Omega} - 2 \gamma \cos(\Omega_0) e^{j\Omega} + \gamma ^2}$	$ \gamma < 1$
7. $u[n] - u[n - L_x]$	$\frac{\sin(L_x \Omega / 2)}{\sin(\Omega / 2)} e^{-j\Omega(L_x - 1)/2}$	
8. $\frac{B}{\pi} \text{sinc}\left(\frac{Bn}{\pi}\right)$	$\sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{2B}\right)$	
9. $\frac{B}{2\pi} \text{sinc}^2\left(\frac{Bn}{2\pi}\right)$	$\sum_{k=-\infty}^{\infty} \Lambda\left(\frac{\Omega - 2\pi k}{2B}\right)$	
10. 1	$2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k)$	
11. $u[n]$	$\frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k)$	
12. $e^{j\Omega_0 n}$	$2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k)$	
13. $\cos(\Omega_0 n)$	$\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k)$	
14. $\sin(\Omega_0 n)$	$\frac{\pi}{j} \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) - \delta(\Omega + \Omega_0 - 2\pi k)$	
15. $\cos(\Omega_0 n) u[n]$	$\frac{e^{j2\Omega} - e^{j\Omega} \cos(\Omega_0)}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1} + \frac{\pi}{2} \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k)$	
16. $\sin(\Omega_0 n) u[n]$	$\frac{e^{j\Omega} \sin(\Omega_0)}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1} + \frac{\pi}{2j} \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) - \delta(\Omega + \Omega_0 - 2\pi k)$	

Table 6.1: Selected discrete-time Fourier transform pairs.

Beware the Red Herring

Carefully inspected, this forbidding look of Table 6.1 is only a façade. In reality, the DTFT is almost as easy to use as the CTFT. The basic reason for the intimidating appearance of the DTFT is its periodic extension. But recall that, in a basic sense, discrete-time signals are bandlimited to $|\Omega| \leq \pi$. The periodic nature of the spectra is merely to satisfy a mathematical obligation. As we shall see, it is a red herring that can be easily circumvented in most applications. If we rewrite Table 6.1 over only the fundamental band, as shown in Table 6.2, the DTFT expressions become almost as simple as those of the CTFT. For example, pair 10 becomes $1 \iff 2\pi\delta(\Omega)$, pair 12 becomes $e^{j\Omega_0 n} \iff 2\pi\delta(\Omega - \Omega_0)$, pair 13 reduces to $\cos(\Omega_0 n) \iff \pi[\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)]$, and so on. The first seven pairs have built-in periodicity. Hence, their expressions in the fundamental band do not simplify any further. The remaining expressions are parallel to those for the CTFT.

$x[n]$	$X(\Omega)$ for $-\pi \leq \Omega < \pi$	
1. $\delta[n - k]$	$e^{-jk\Omega}$	Integer k
2. $\gamma^n u[n]$	$\frac{e^{j\Omega}}{e^{j\Omega} - \gamma}$	$ \gamma < 1$
3. $-\gamma^n u[-n - 1]$	$\frac{e^{j\Omega}}{e^{j\Omega} - \gamma}$	$ \gamma > 1$
4. $\gamma^{ n }$	$\frac{1 - \gamma^2}{1 - 2\gamma \cos(\Omega) + \gamma^2}$	$ \gamma < 1$
5. $n\gamma^n u[n]$	$\frac{\gamma e^{j\Omega}}{(e^{j\Omega} - \gamma)^2}$	$ \gamma < 1$
6. $ \gamma ^n \cos(\Omega_0 n + \theta) u[n]$	$\frac{e^{j\Omega} [e^{j\Omega} \cos(\theta) - \gamma \cos(\Omega_0 - \theta)]}{e^{j2\Omega} - 2 \gamma \cos(\Omega_0) e^{j\Omega} + \gamma ^2}$	$ \gamma < 1$
7. $u[n] - u[n - L_x]$	$\frac{\sin(L_x \Omega / 2)}{\sin(\Omega / 2)} e^{-j\Omega(L_x - 1)/2}$	
8. $\frac{B}{\pi} \text{sinc}\left(\frac{Bn}{\pi}\right)$	$\Pi\left(\frac{\Omega}{2B}\right)$	$0 < B \leq \pi$
9. $\frac{B}{2\pi} \text{sinc}^2\left(\frac{Bn}{2\pi}\right)$	$\Lambda\left(\frac{\Omega}{2B}\right)$	$0 < B \leq \pi$
10. 1	$2\pi\delta(\Omega)$	
11. $u[n]$	$\frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi\delta(\Omega)$	
12. $e^{j\Omega_0 n}$	$2\pi\delta(\Omega - \Omega_0)$	$ \Omega_0 < \pi$
13. $\cos(\Omega_0 n)$	$\pi [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)]$	$ \Omega_0 < \pi$
14. $\sin(\Omega_0 n)$	$\frac{\pi}{j} [\delta(\Omega - \Omega_0) - \delta(\Omega + \Omega_0)]$	$ \Omega_0 < \pi$
15. $\cos(\Omega_0 n) u[n]$	$\frac{e^{j2\Omega} - e^{j\Omega} \cos(\Omega_0)}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1} + \frac{\pi}{2} [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)]$	$ \Omega_0 < \pi$
16. $\sin(\Omega_0 n) u[n]$	$\frac{e^{j\Omega} \sin(\Omega_0)}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1} + \frac{\pi}{2j} [\delta(\Omega - \Omega_0) - \delta(\Omega + \Omega_0)]$	$ \Omega_0 < \pi$

Table 6.2: Selected DTFT pairs using the fundamental band.

What is more important, however, is that we can perform practically all DTFT operations using these simpler fundamental-band DTFT expressions and then apply a 2π -periodic extension to the results thus obtained to satisfy the mathematical imperative. The rule is, if an operation does not entail aliasing (frequency spilling beyond $\pm\pi$), we can safely operate entirely within the fundamental band. The main advantage of using the clumsier periodic spectral expressions is that they automatically take care of aliasing that may arise during an operation. In contrast, when we use only simple fundamental-band expressions, if aliasing does arise in an operation, the resulting spectrum spills beyond $\pm\pi$. We need some procedure to take care of this problem. For example, we can simply repeat such a spectrum (with spilling beyond $\pm\pi$) with 2π periodicity and add

the overlapping portions of the spectrum. This procedure is quite straightforward and avoids the operation under question being carried out using clumsy periodic expressions. Alternatively, we can fold the spilled spectrum back at $\pm\pi$ to take care of aliasing. This frequency folding procedure, however, may not be so convenient in the case of complex spectra. We shall explain both these procedures and the benefits of using fundamental-band expressions by several examples in the next section.

DTFT of Finite-Duration Signals

Consider a finite-duration signal $x[n]$ that begins at time $n = n_1$ and ends at time $n = n_2$. In this case, the DTFT is determined through a finite sum

$$X(\Omega) = \sum_{n=n_1}^{n_2} x[n]e^{-j\Omega n}. \quad (6.12)$$

As the next example shows, this form is particularly advantageous for computer-based computation of the DTFT.

▷ Example 6.6 (DTFT of a Finite-Duration Signal)

A signal $x[n]$ equals $[1, 2, -1, -2, 0, 2, 1, -2, -1]$ for $-4 \leq n \leq 4$ and is otherwise 0. Sketch the DTFT of this signal.

Since the signal $x[n]$ has finite duration, the DTFT is easily computed by various computational packages, such as MATLAB.

```
01 x = [1 2 -1 -2 0 2 1 -2 -1]; n2 = 4; Omega = linspace(-pi,pi,501);
02 X = @Omega polyval(x,exp(1j*Omega))./exp(1j*Omega*n2);
03 subplot(211); plot(Omega,abs(X(Omega))); subplot(212); plot(Omega,angle(X(Omega)));
```

The resulting magnitude and phase spectra are shown in Figs. 6.5a and 6.5b, respectively.

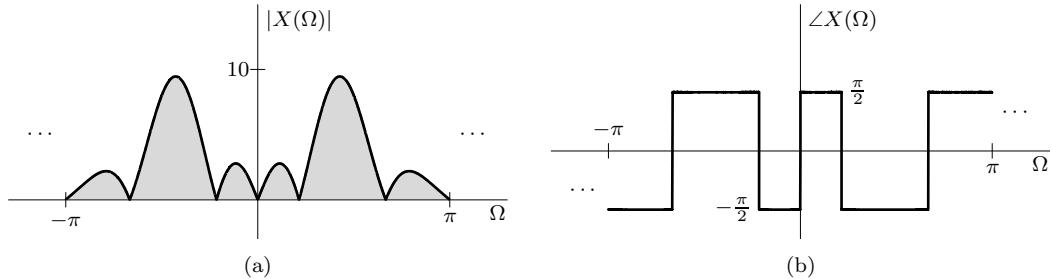


Figure 6.5: DTFT of a finite-duration signal.

To understand how line 02 computes the DTFT, we first write Eq. (6.12) in the alternate form of

$$\begin{aligned} X(\Omega) &= \frac{1}{e^{j\Omega n_2}} \sum_{n=n_1}^{n_2} x[n]e^{j\Omega(n_2-n)} \\ &= \left(x[n_1]e^{j\Omega(n_2-n_1)} + \cdots + x[n_2-1]e^{j\Omega} + x[n_2] \right) / e^{j\Omega n_2}. \end{aligned}$$

Here, the DTFT is computed using a weighted sum of descending powers of $e^{j\Omega}$, which is exactly the form required of the `polyval` command.

Example 6.6 ▶

6.2 Properties of the DTFT

Just as in the case of the CTFT, DTFT properties provide many useful insights and simplify the analysis of discrete-time signals and systems. We have already seen a close connection between the CTFT and the DTFT, a connection that is explored in greater depth in Sec. 6.4. For this reason, the properties of the DTFT are very similar to those for the CTFT, as the following discussion shows.

6.2.1 Duality

The DTFT does not possess a formal duality property like Eq. (1.80) for the CTFT. The reason is easy to explain: the roles of $x[n]$ and $X(\Omega)$ cannot be interchanged since one is a function of a discrete variable and the other is a function of a continuous variable. Still, the essence of duality is often found in the DTFT. For example, time shifting a DT signal $x[n]$ results in multiplying $X(\Omega)$ by a complex exponential. The dual is also true, that shifting $X(\Omega)$ causes multiplication of $x[n]$ by a complex exponential. We shall see other examples, such as a multiplication-convolution duality, as we proceed through the DTFT properties.

6.2.2 Linearity Property

The discrete-time Fourier transform is linear; that is, if

$$x[n] \iff X(\Omega) \quad \text{and} \quad y[n] \iff Y(\Omega),$$

then, for any constants a and b ,

$$ax[n] + by[n] \iff aX(\Omega) + bY(\Omega). \quad (6.13)$$

The proof is trivial, and this result can be extended to any finite sum.

6.2.3 Complex-Conjugation Property

From Eq. (6.1), we obtain the DTFT of $x^*[n]$ as

$$\text{DTFT}\{x^*[n]\} = \sum_{n=-\infty}^{\infty} x^*[n]e^{-j\Omega n} = \left(\sum_{n=-\infty}^{\infty} x[n]e^{j\Omega n} \right)^* = X^*(-\Omega).$$

Using this result, the complex-conjugation property states that

$$\text{if } x[n] \iff X(\Omega), \text{ then } x^*[n] \iff X^*(-\Omega). \quad (6.14)$$

Like the CTFT case, several interesting observations arise from the complex-conjugation property:

1. If $x[n]$ is real, then $X(\Omega)$ is conjugate symmetric. That is, if $x[n] = x^*[n]$, then $X(\Omega) = X^*(-\Omega)$, which also ensures that $|X(\Omega)|$ is even and $\angle X(\Omega)$ is odd.
2. If $x[n]$ is imaginary, then $X(\Omega)$ is conjugate antisymmetric. That is, if $x[n] = -x^*[n]$, then $X(\Omega) = -X^*(-\Omega)$, which also ensures that $|X(\Omega)|$ is odd and $\angle X(\Omega)$ is even.
3. If $y[n]$ is the real part of some signal $x[n]$, then its transform $Y(\Omega)$ is the conjugate-symmetric portion of $X(\Omega)$. That is, if $y[n] = \text{Re}\{x[n]\} = \frac{1}{2}x[n] + \frac{1}{2}x^*[n]$, then $Y(\Omega) = X_{\text{cs}}(\Omega) = \frac{1}{2}X(\Omega) + \frac{1}{2}X^*(-\Omega)$.
4. If $y[n]$ is the imaginary part of some signal $x[n]$, then its transform $Y(\Omega)$ is $1/j$ times the conjugate-antisymmetric portion of $X(\Omega)$. That is, if $y[n] = \text{Im}\{x[n]\} = \frac{1}{2j}x[n] - \frac{1}{2j}x^*[n]$, then $Y(\Omega) = \frac{1}{j}X_{\text{ca}}(\Omega) = \frac{1}{2j}X(\Omega) - \frac{1}{2j}X^*(-\Omega)$.

6.2.4 Time Scaling and the Time-Reversal Property

Unlike the CT case, it is a somewhat tricky business to time scale a DT signal. As we saw in Ch. 4, the operations of upsampling and downsampling constitute forms of time scaling for DT signals, although both are restricted to integer scale factors. Beyond these restrictions, upsampling and downsampling can produce distinctly different results depending on the nature of the DT signal and the details of any filters (interpolation or decimation) that are used. For example, although downsampling always causes a loss of data samples, it does not always produce a loss or distortion of information. To give another example, ideal interpolation filters produce different results than practical interpolation filters. Given all these factors, it is just not possible to establish a simple time-scaling property for DT signals. Later in Sec. 6.6, when we have developed a set of suitable tools, we explore several important cases of digital resampling (time scaling) from a frequency-domain perspective.

Despite the complexities of general scaling, a simple DTFT property does exist for the special case of reversal. This property states that

$$\text{if } x[n] \iff X(\Omega), \text{ then } x[-n] \iff X(-\Omega). \quad (6.15)$$

This equation serves as both the time-reversal property and the frequency-reversal property. As with the complex-conjugation property, Eq. (6.15) leads to some interesting observations:

1. If $x[n]$ is even, then $X(\Omega)$ is even. That is, if $x[n] = x[-n]$, then $X(\Omega) = X(-\Omega)$.
2. If $x[n]$ is odd, then $X(\Omega)$ is odd. That is, if $x[n] = -x[-n]$, then $X(\Omega) = -X(-\Omega)$.
3. If $y[n]$ is the even part of some signal $x[n]$, then its transform $Y(\Omega)$ is the even portion of $X(\Omega)$. That is, if $y[n] = x_e[n] = \frac{1}{2}x[n] + \frac{1}{2}x[-n]$, then $Y(\Omega) = X_e(\Omega) = \frac{1}{2}X(\Omega) + \frac{1}{2}X(-\Omega)$.
4. If $y[n]$ is the odd part of some signal $x[n]$, then its transform $Y(\Omega)$ is the odd portion of $X(\Omega)$. That is, if $y[n] = x_o[n] = \frac{1}{2}x[n] - \frac{1}{2}x[-n]$, then $Y(\Omega) = X_o(\Omega) = \frac{1}{2}X(\Omega) - \frac{1}{2}X(-\Omega)$.

▷ Example 6.7 (Using the Time-Reversal Property)

In Table 6.1, derive pair 4 using pair 2 and the time-reversal property of Eq. (6.15).

Pair 2 of Table 6.1 states that

$$\gamma^n u[n] \iff \frac{e^{j\Omega}}{e^{j\Omega} - \gamma}, \quad |\gamma| < 1.$$

Applying Eq. (6.15) yields

$$\gamma^{-n} u[-n] \iff \frac{e^{-j\Omega}}{e^{-j\Omega} - \gamma}, \quad |\gamma| < 1.$$

Moreover, summing $\gamma^n u[n]$ and $\gamma^{-n} u[-n]$ nearly yields $\gamma^{|n|}$, except for an extra impulse at $n = 0$. Hence,

$$\gamma^{|n|} = \gamma^n u[n] + \gamma^{-n} u[-n] - \delta[n].$$

Invoking the linearity property and substituting the DTFT of each piece yield pair 4 of Table 6.1,

$$\text{DTFT}\{\gamma^{|n|}\} = \frac{e^{j\Omega}}{e^{j\Omega} - \gamma} + \frac{e^{-j\Omega}}{e^{-j\Omega} - \gamma} - 1 = \frac{1 - \gamma^2}{1 - 2\gamma \cos(\Omega) + \gamma^2}, \quad |\gamma| < 1.$$

Example 6.7 ▷

▷ **Drill 6.4 (Using the Time-Reversal Property)**

In Table 6.2, derive pair 13 using pair 15 and the time-reversal property of Eq. (6.15).

△

6.2.5 Time-Shifting Property

The time-shifting property states that

$$\text{if } x[n] \iff X(\Omega), \text{ then } x[n - m] \iff X(\Omega)e^{-j\Omega m} \text{ for integer } m. \quad (6.16)$$

To prove this property, we directly substitute $x[n - m]$ into Eq. (6.1) to obtain

$$x[n - m] \iff \sum_{n=-\infty}^{\infty} x[n-m]e^{-j\Omega n} = \sum_{k=-\infty}^{\infty} x[k]e^{-j\Omega[k+m]} = e^{-j\Omega m} \sum_{k=-\infty}^{\infty} x[k]e^{-j\Omega k} = e^{-j\Omega m}X(\Omega).$$

This result shows that *delaying a signal by m samples does not change its magnitude spectrum. The phase spectrum, however, is changed by $-\Omega m$.* This added phase is a linear function of Ω with slope $-m$. We emphasize that the time-shifting property of Eq. (6.16) applies only for integer values of m . Still, as we shall later see, the spectrum $X(\Omega)e^{-j\Omega m}$ has an intriguing and useful interpretation for non-integer m as well.

A Physical Explanation of Linear Phase

Time delay in a signal causes a linear phase shift in its spectrum. The heuristic explanation of this result is exactly parallel to that for continuous-time signals given in Sec. 1.9.5 (see Fig. 1.45 on page 57).

▷ **Example 6.8 (Using the Time-Shifting Property)**

Use the time-shifting property to determine the DTFT of $x[n] = \frac{1}{4}\text{sinc}\left(\frac{n-2}{4}\right)$.

As shown in Fig. 6.6a, $x[n]$ is a sinc function that is right shifted by 2 units from the origin. Thus, its spectrum should only differ from the non-shifted sinc by an additional linear phase component of -2Ω .

To determine the DTFT $X(\Omega)$, we substitute $B = \pi/4$ into pair 8 of Table 6.1 to obtain

$$\frac{1}{4}\text{sinc}\left(\frac{n}{4}\right) \iff \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{\pi/2}\right).$$

Applying the time-shifting property of Eq. (6.16) with a shift of $m = 2$ yields

$$\frac{1}{4}\text{sinc}\left(\frac{n-2}{4}\right) \iff \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{\pi/2}\right) e^{-j2\Omega}. \quad (6.17)$$

The resulting magnitude and phase spectra are shown in Figs. 6.6b and 6.6c, respectively. Compared with the spectrum of the unshifted sinc (Fig. 6.2), the only difference is an added phase component of -2Ω .

To demonstrate an alternate technique, we now solve the problem in the fundamental band and periodically repeat the results to satisfy the mathematical requirement that discrete-time signals have periodic spectra. To begin, we substitute $B = \pi/4$ into pair 8 of Table 6.2 to obtain

$$\frac{1}{4}\text{sinc}\left(\frac{n}{4}\right) \iff \Pi\left(\frac{\Omega}{\pi/2}\right), \quad |\Omega| \leq \pi.$$

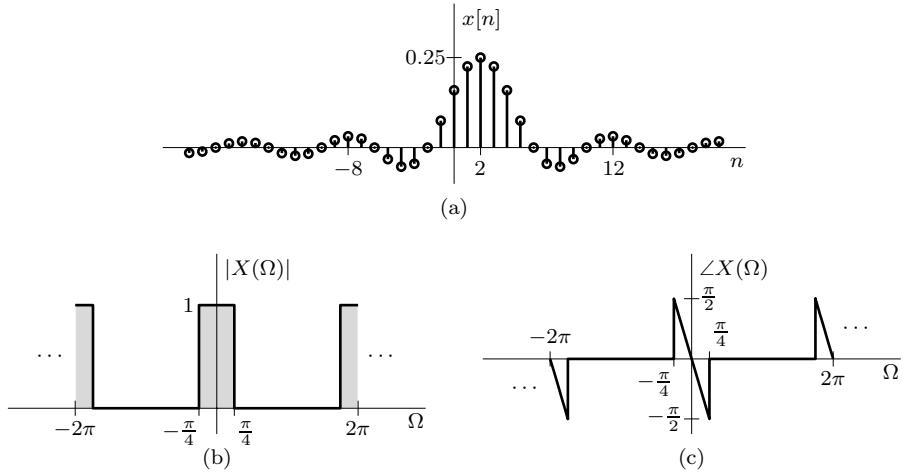


Figure 6.6: The effects of time shifting: (a) $x[n]$, (b) $|X(\Omega)|$, and (c) $\angle X(\Omega)$.

Applying the time-shifting property of Eq. (6.16) with a shift of $m = 2$ yields

$$\frac{1}{4} \text{sinc}\left(\frac{n-2}{4}\right) \iff \Pi\left(\frac{\Omega}{\pi/2}\right) e^{-j2\Omega}, \quad |\Omega| \leq \pi.$$

Repeating the spectrum periodically with period 2π yields

$$\frac{1}{4} \text{sinc}\left(\frac{n-2}{4}\right) \iff \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{\pi/2}\right) e^{-j2(\Omega - 2\pi k)}.$$

This result is identical to Eq. (6.17) because $e^{\pm j2\pi k} = 1$ for all integer values of k . Hence, the term representing the phase in the final result is equal to $e^{-j2\Omega}$.

Example 6.8 \triangleleft

▷ Drill 6.5 (Using the Time-Shifting Property)

Use pair 7 of Table 6.1 and the time-shifting property to verify the result in Eq. (6.3). \triangleleft

Frequency-Shifting Property

The frequency-shifting property states that

$$\text{if } x[n] \iff X(\Omega), \text{ then } x[n]e^{j\Omega_0 n} \iff X(\Omega - \Omega_0). \quad (6.18)$$

This property is the dual of the time-shifting property, and it follows directly from Eq. (6.1):

$$x[n]e^{j\Omega_0 n} \iff \sum_{n=-\infty}^{\infty} x[n]e^{j\Omega_0 n}e^{-j\Omega n} = \sum_{n=-\infty}^{\infty} x[n]e^{-j(\Omega - \Omega_0)n} = X(\Omega - \Omega_0).$$

From this result, it follows that

$$x[n]e^{-j\Omega_0 n} \iff X(\Omega + \Omega_0).$$

Adding this pair to the pair in Eq. (6.18), we obtain

$$x[n] \cos(\Omega_0 n) \iff \frac{1}{2} [X(\Omega - \Omega_0) + X(\Omega + \Omega_0)]. \quad (6.19)$$

This is the *modulation property*. Under normal use, modulation is used to spectrally shift a signal. Note that if the bandwidth of $X(\Omega)$ is B , then the bandwidth of the modulated signal $x[n] \cos(\Omega_0 n)$ is not necessarily $2B$ because of the possibility of aliasing, as illustrated in upcoming Ex. 6.9.

Multiplying both sides of Eq. (6.18) by $e^{j\theta}$, we obtain

$$x[n] e^{j(\Omega_0 n + \theta)} \iff X(\Omega - \Omega_0) e^{j\theta}.$$

Using this pair, we can generalize the modulation property as

$$x[n] \cos(\Omega_0 n + \theta) \iff \frac{1}{2} [X(\Omega - \Omega_0) e^{j\theta} + X(\Omega + \Omega_0) e^{-j\theta}]. \quad (6.20)$$

▷ Example 6.9 (Exploring Modulation)

A signal $x[n] = \text{sinc}(n/4)$ modulates a carrier $\cos(\Omega_0 n)$. Using the periodic expression for $X(\Omega)$ from Table 6.1, find and sketch the spectrum of the modulated signal $x[n] \cos(\Omega_0 n)$ for

$$(a) \quad \Omega_0 = \frac{\pi}{2} \quad (b) \quad \Omega_0 = 0.85\pi$$

(a) For $x[n] = \text{sinc}(n/4)$, we find its DTFT $X(\Omega)$ from pair 8 of Table 6.1 as

$$X(\Omega) = 4 \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{0.5\pi}\right).$$

Figure 6.7a depicts $X(\Omega)$. From the modulation property of Eq. (6.19), we obtain

$$x[n] \cos(\pi n/2) \iff 2 \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - \pi/2 - 2\pi k}{0.5\pi}\right) + \Pi\left(\frac{\Omega + \pi/2 - 2\pi k}{0.5\pi}\right).$$

Figure 6.7b shows $X(\Omega)$ shifted by $\pi/2$, and Fig. 6.7c shows $X(\Omega)$ shifted by $-\pi/2$. The spectrum of the modulated signal is obtained by adding these two shifted spectra and multiplying by half, as shown in Fig. 6.7d. In this case, the bandwidth of $x[n] \cos(\pi n/2)$ is twice that of $x[n]$, no aliasing occurs, and the original shape of $X(\Omega)$ remains intact. The modulation operation achieves the desired goal of spectral shifting without distortion of the underlying signal. Hence, we can reconstruct the signal $x[n]$ from the modulated signal $x[n] \cos(\pi n/2)$.

(b) Figure 6.8a shows $X(\Omega)$, which is the same as that in Fig. 6.7a. For $\Omega_0 = 0.85\pi$, the modulation property of Eq. (6.19) yields

$$x[n] \cos(0.85\pi n) \iff 2 \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 0.85\pi - 2\pi k}{0.5\pi}\right) + \Pi\left(\frac{\Omega + 0.85\pi - 2\pi k}{0.5\pi}\right).$$

Figures 6.8b and 6.8c show $X(\Omega)$ shifted by $\pm 0.85\pi$. The spectrum of the modulated signal is obtained by adding these two shifted (and overlapping) spectra and multiplying by half, as shown in Fig. 6.8d.

In this case, shifting by $\pm 0.85\pi$ causes the original baseband spectrum to spill outside the fundamental band, and aliasing occurs. The original shape of $X(\Omega)$ does not remain intact, and it is not possible to reconstruct the signal $x[n]$ from the modulated signal $x[n] \cos(0.85\pi n)$. Because of

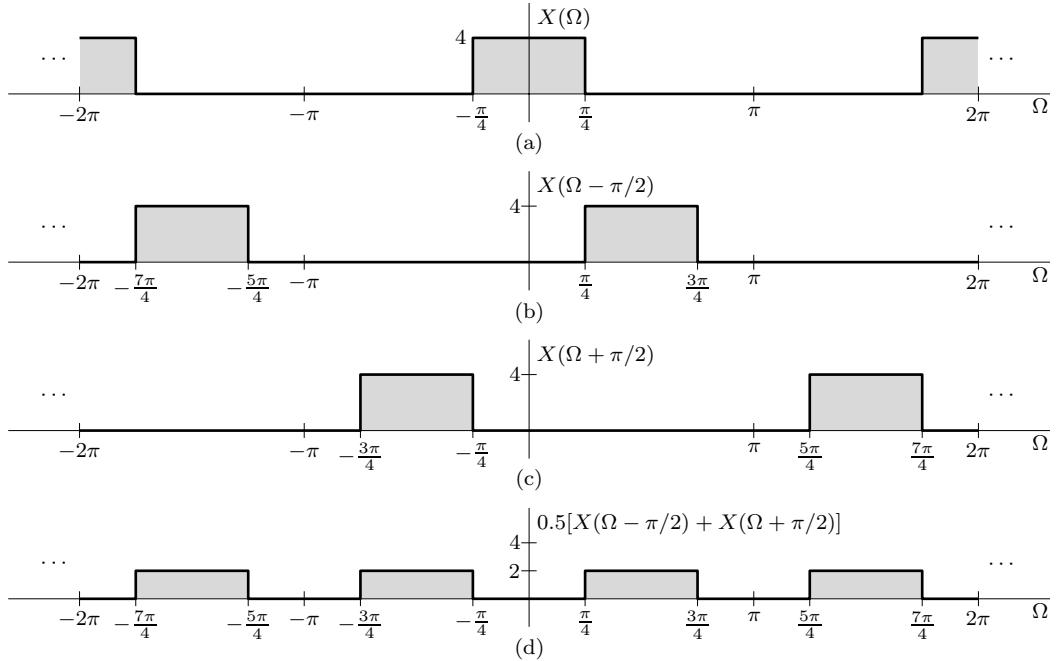


Figure 6.7: An example of modulation where no aliasing occurs.

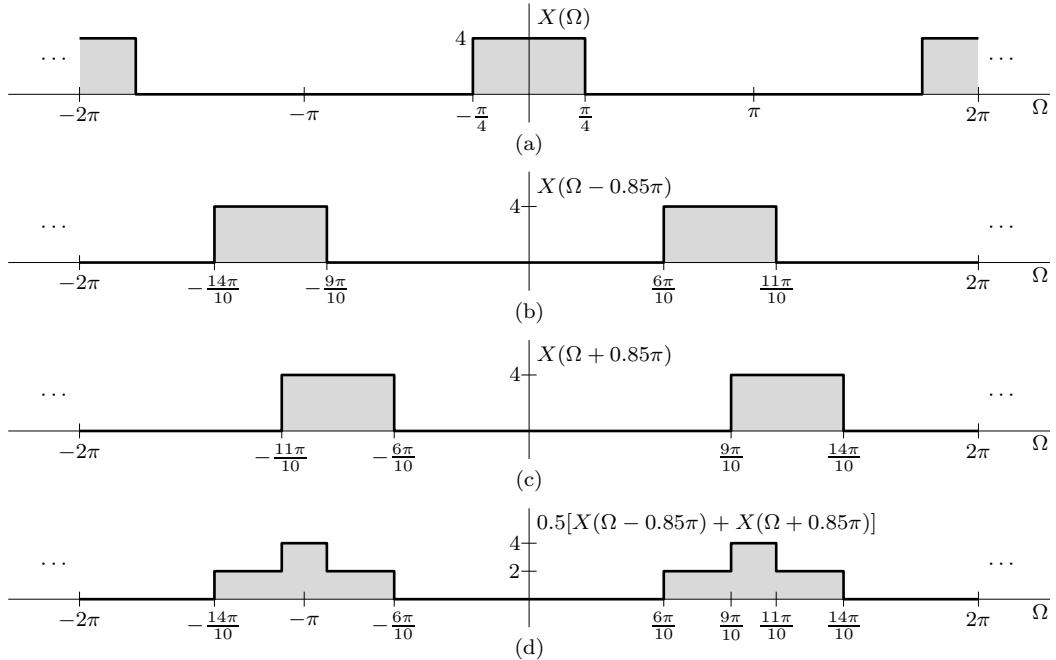


Figure 6.8: An example of modulation where aliasing does occur.

aliasing, the modulation operation does not achieve the desired purpose of spectra shifting, and the 0.4π bandwidth of $x[n] \cos(0.85\pi n)$ is less than twice the $\pi/4$ bandwidth of $x[n]$. In this example,

to realize spectral shifting without aliasing requires $\Omega_0 \leq \pi - \pi/4 = 0.75\pi$.

Example 6.9 □

▷ **Example 6.10 (Exploring Modulation Using the Fundamental Band)**

Solve Ex. 6.9 in the fundamental band by using the expression for $X(\Omega)$ from Table 6.2 and then applying a 2π -periodic extension of the final spectra.

(a) For $x[n] = \text{sinc}(n/4)$, we find its DTFT $X(\Omega)$ from pair 8 of Table 6.2 as

$$X(\Omega) = 4\Pi\left(\frac{\Omega}{0.5\pi}\right), \quad |\Omega| \leq \pi.$$

Figure 6.9a shows $X(\Omega)$ over the fundamental band. From the modulation property of Eq. (6.19), we obtain

$$x[n] \cos(\pi n/2) \iff 2 \left[\Pi\left(\frac{\Omega - \pi/2}{0.5\pi}\right) + \Pi\left(\frac{\Omega + \pi/2}{0.5\pi}\right) \right], \quad |\Omega| \leq \pi.$$

Here, the spectrum $0.5X(\Omega)$ is shifted by $\pm\pi/2$, as shown in Fig. 6.9b. Observe that these shifted components are still in the fundamental band. To represent the spectrum valid for all Ω , we periodically repeat this spectrum with period 2π , the result of which is identical to that in Fig. 6.7d.

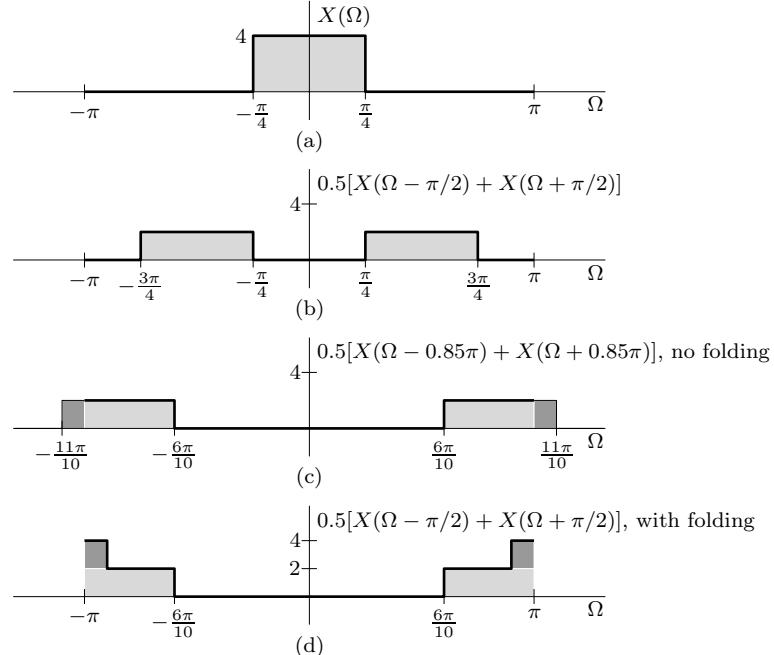


Figure 6.9: Exploring modulation using fundamental-band descriptions.

(b) From the modulation property of Eq. (6.19) we obtain

$$x[n] \cos(0.85\pi n) \iff 2 \left[\Pi\left(\frac{\Omega - 0.85\pi}{0.5\pi}\right) + \Pi\left(\frac{\Omega + 0.85\pi}{0.5\pi}\right) \right], \quad |\Omega| \leq \pi.$$

The spectrum $0.5X(\Omega)$ is now shifted by $\pm 0.85\pi$, as shown in Fig. 6.9c. As shown with dark shading, observe that this spectrum spills outside the fundamental band $\pm\pi$, indicating aliasing. How do we handle this situation, knowing that spectrum outside the fundamental band is not permitted? Two approaches are common: we can repeat the spectrum in Fig. 6.9c periodically with period 2π , adding the spectra where they overlap, or we can fold any spilled spectrum back inside the fundamental band. In this case, the spectrum is real, and it is easiest to fold the spilled spectrum (dark gray) back into the fundamental band, as shown in Fig. 6.9d. To obtain the spectrum valid for all Ω , we just repeat the spectrum in Fig. 6.9d with period 2π , the result of which is identical to Fig. 6.8d. We obtain the same result if we repeat the spectrum in Fig. 6.9c periodically with period 2π and add the spectra where they overlap.

This example illustrates the simplicity of dealing with spectra in the fundamental band rather than using the expressions for the entire range of Ω . The fundamental band can offer a more intuitive approach to problems. Difficulties occur only when an operation leads to aliasing. In such cases, we can still operate in the fundamental band by using periodic replication or frequency folding. Although the folding approach is relatively easy to visualize and perform graphically, especially when the spectra are real, it may not be simple or appropriate for complex spectra or spectra that lack conjugate symmetry. In such cases, we can always apply the standard approach of repeating the spectrum periodically and adding the overlapping portions.

Example 6.10 ◀

▷ **Drill 6.6 (Using the Frequency-Shifting Property)**

In Table 6.1, derive pairs 12 and 13 using pair 10 and the frequency-shifting property.

◀

6.2.6 Frequency-Differentiation Property

Since $x[n]$ is a discrete-time signal, there are no time-domain integration or differentiation properties for the DTFT. However, since $X(\Omega)$ is a function of the continuous variable Ω , there is a frequency-differentiation property that states

$$\text{if } x[n] \iff X(\Omega), \text{ then } -jnx[n] \iff \frac{dX(\Omega)}{d\Omega}. \quad (6.21)$$

This property follows immediately by differentiating both sides of Eq. (6.1) with respect to Ω .

▷ **Example 6.11 (Using the Frequency-Differentiation Property)**

In Table 6.1, derive pair 5 using pair 2 and the frequency-differentiation property of Eq. (6.21).

Pair 2 states that

$$\gamma^n u[n] \iff \frac{e^{j\Omega}}{e^{j\Omega} - \gamma}, \quad |\gamma| < 1.$$

Applying Eq. (6.21), we obtain

$$n\gamma^n u[n] \iff j \frac{d}{d\Omega} \left\{ \frac{e^{j\Omega}}{e^{j\Omega} - \gamma} \right\} = \frac{\gamma e^{j\Omega}}{(e^{j\Omega} - \gamma)^2}, \quad |\gamma| < 1.$$

This result exactly matches pair 5 in Table 6.1.

Example 6.11 ◀

6.2.7 Time-Domain and Frequency-Domain Convolution Properties

Let us begin with two DTFT pairs

$$x[n] \iff X(\Omega) \quad \text{and} \quad y[n] \iff Y(\Omega).$$

The time-domain convolution property states that

$$x[n] * y[n] = \sum_{m=-\infty}^{\infty} x[m]y[n-m] \iff X(\Omega)Y(\Omega). \quad (6.22)$$

Using this property, a simple frequency-domain multiplication replaces the cumbersome time-domain convolution operation. This elegant result forms the basis for the frequency-domain analysis of LTID systems.

The dual of the time-domain convolution property is the frequency-domain convolution property, which states that

$$x[n]y[n] \iff \frac{1}{2\pi} X(\Omega) \circledast Y(\Omega) = \frac{1}{2\pi} \int_{2\pi} X(\lambda)Y(\Omega - \lambda) d\lambda. \quad (6.23)$$

Recall from Ch. 1 (page 68) that the symbol \circledast denotes circular (or periodic) convolution. Circular convolution is more sensible than linear convolution in Eq. (6.23) since both $X(\Omega)$ and $Y(\Omega)$ are 2π -periodic functions.

Both convolution properties are readily proven. To prove the frequency-domain convolution property, for instance, we observe that

$$x[n]y[n] \iff \sum_{n=-\infty}^{\infty} x[n]y[n]e^{-j\Omega n} = \sum_{n=-\infty}^{\infty} y[n] \left[\frac{1}{2\pi} \int_{2\pi} X(\lambda)e^{j\lambda n} d\lambda \right] e^{-j\Omega n}.$$

Interchanging the order of summation and integration, we obtain Eq. (6.23) as

$$x[n]y[n] \iff \frac{1}{2\pi} \int_{2\pi} X(\lambda) \left[\sum_{n=-\infty}^{\infty} y[n]e^{-j(\Omega-\lambda)n} \right] d\lambda = \frac{1}{2\pi} \int_{2\pi} X(\lambda)Y(\Omega - \lambda) d\lambda.$$

▷ Example 6.12 (DTFT of an Accumulator)

Given $x[n] \iff X(\Omega)$, show that the DTFT of an accumulator is given by

$$\sum_{k=-\infty}^n x[k] \iff X(\Omega) \frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi X(0) \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k).$$

To begin, we recognize that an accumulator can be represented using convolution as

$$\sum_{k=-\infty}^n x[k] = \sum_{m=-\infty}^{\infty} x[m]u[n-m] = x[n] * u[n].$$

Hence, from the time-domain convolution property of Eq. (6.22) and pair 11 of Table 6.1, it follows that

$$\sum_{k=-\infty}^n x[k] = x[n] * u[n] \iff X(\Omega) \left(\frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k) \right), \quad |\Omega| \leq \pi.$$

Because of 2π -periodicity, $X(0) = X(2\pi k)$. Moreover, $X(\Omega)\delta(\Omega - 2\pi k) = X(2\pi k)\delta(\Omega - 2\pi k) = X(0)\delta(\Omega - 2\pi k)$. Hence,

$$\sum_{k=-\infty}^n x[k] \iff X(\Omega) \frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi X(0) \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k).$$

Because of the simple nature of this example, we solved it directly using the periodic expression for the DTFT of $u[n]$ instead of the fundamental-band expression. The reader is encouraged to solve the problem in the fundamental band and then use a 2π -periodic extension of the result.

Example 6.12 ◀

▷ Example 6.13 (Using the Frequency-Domain Convolution Property)

Letting $x[n] = \frac{B}{\pi} \text{sinc}\left(\frac{Bn}{\pi}\right)$, find and sketch the spectrum $Y(\Omega)$ of the signal $y[n] = x^2[n]$ for

$$(a) \quad 0 < B \leq \frac{\pi}{2} \quad (b) \quad \frac{\pi}{2} < B \leq \pi$$

As usual, this example can be solved directly using the spectrum $X(\Omega)$ for the entire range of Ω , or it can be solved in the fundamental band and the result then extended periodically. We shall use the latter approach and demonstrate the ease of solving this problem in the fundamental band.

From pair 8 of Table 6.2, we have, assuming $B \leq \pi$,

$$\frac{B}{\pi} \text{sinc}\left(\frac{Bn}{\pi}\right) \iff \Pi\left(\frac{\Omega}{2B}\right), \quad |\Omega| \leq \pi.$$

To find the DTFT of $x^2[n]$, we shall convolve $\Pi(\Omega/2B)$ with itself, multiply by $1/2\pi$, and then extend the result periodically.

For convenience, we shall label the fundamental-band spectrum as $\hat{X}(\Omega)$. Thus

$$\hat{X}(\Omega) = \Pi\left(\frac{\Omega}{2B}\right).$$

We now convolve this spectrum with itself. Choosing the interval of integration from $-\pi$ to π , we obtain

$$\hat{X}(\Omega) * \hat{X}(\Omega) = \int_{-\pi}^{\pi} \hat{X}(\lambda) \hat{X}(\Omega - \lambda) d\lambda.$$

Figure 6.10a shows $\hat{X}(\lambda)$ as a function of λ . To obtain $\hat{X}(-\lambda)$, we invert $\hat{X}(\lambda)$. Because of symmetry about the vertical axis, $\hat{X}(-\lambda) = \hat{X}(\lambda)$. Now shift $\hat{X}(-\lambda)$ by Ω to obtain $\hat{X}(\Omega - \lambda)$, as shown in Fig. 6.10b. For $-2B \leq \Omega < 0$, the area under the product of $\hat{X}(\lambda)$ and $\hat{X}(\Omega - \lambda)$ is $2B + \Omega$ (linearly increasing with Ω), as shown in Fig. 6.10c. For $0 \leq \Omega \leq 2B$, the area is $2B - \Omega$ (linearly decreasing with Ω), as shown in Fig. 6.10d. Taken together, the resulting convolution is a triangle function given as

$$\hat{X}(\Omega) * \hat{X}(\Omega) = 2B \Lambda\left(\frac{\Omega}{4B}\right).$$

(a) For $0 < B \leq \pi/2$, the triangle function $\Lambda(\Omega/4B)$ is entirely within the fundamental band. There is no aliasing to complicate the picture. As shown in Fig. 6.10e, the desired spectrum is thus

$$x^2[n] \iff \frac{1}{2\pi} \hat{X}(\Omega) * \hat{X}(\Omega) = \frac{B}{\pi} \Lambda\left(\frac{\Omega}{4B}\right), \quad |\Omega| \leq \pi.$$

The spectral expression valid for all Ω is obtained by 2π -periodic extension of this result, as

$$y[n] = x^2[n] \iff Y(\Omega) = \frac{B}{\pi} \sum_{k=-\infty}^{\infty} \Lambda\left(\frac{\Omega - 2\pi k}{4B}\right).$$

(b) Let us now consider the case where $\pi/2 < B \leq \pi$. Procedurally, this case remains the same as that in part (a). The only difference is that because $\pi/2 < B \leq \pi$, the convolved spectrum spills outside the fundamental band, as shown in Fig. 6.10f. This spectral spill, which indicates aliasing,

requires repair. In this case, a simple repair is to fold the spilled spectrum about $\Omega = \pi$, as indicated in Fig. 6.10g, and then repeat the resulting spectrum periodically with period 2π . Another method is to apply a 2π -periodic repetition of the spectrum in Fig. 6.10f (without folding) and add the overlapping portions of the spectrum. This, of course, results in the same spectrum as in Fig. 6.10g.

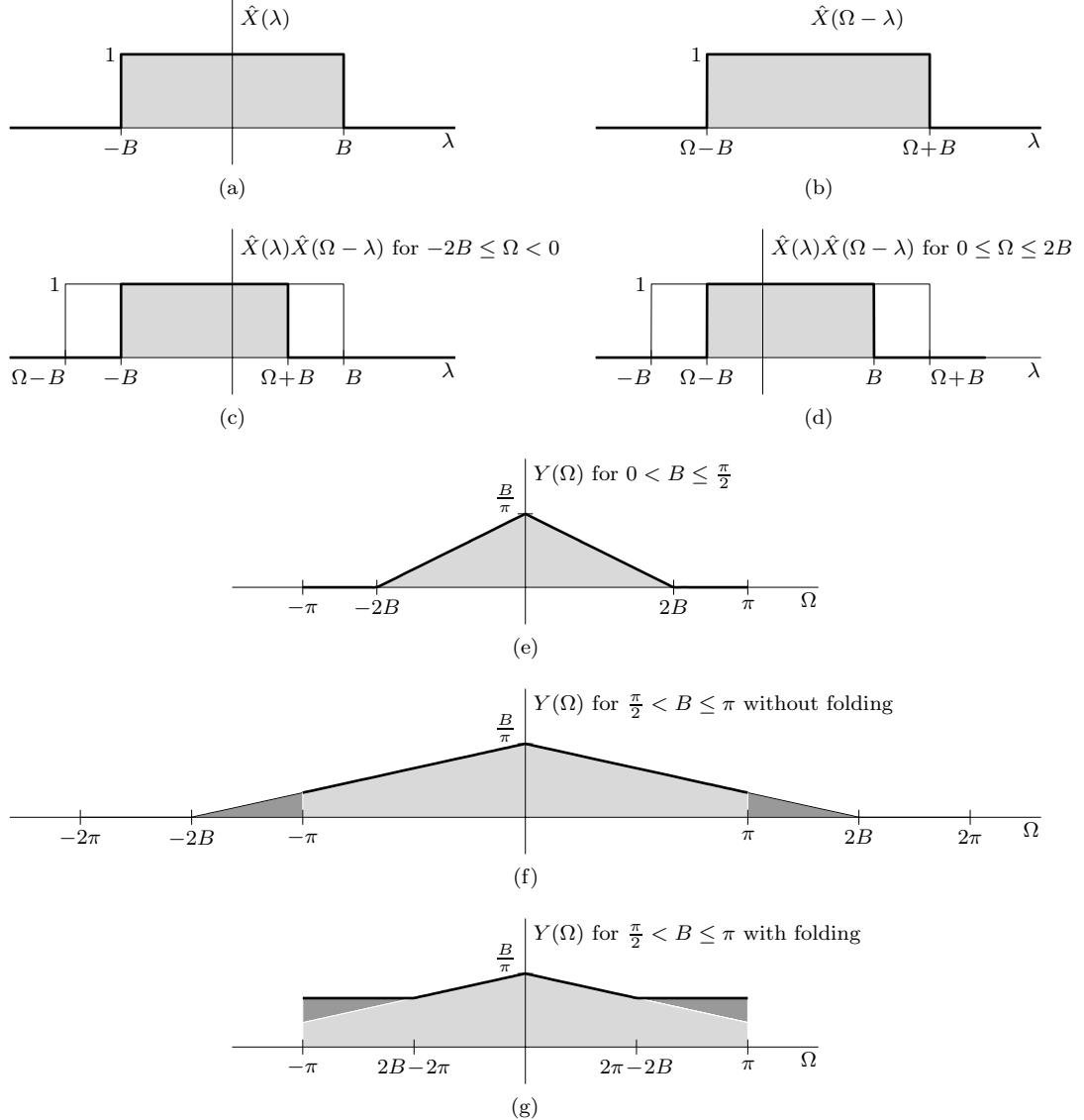


Figure 6.10: Application of the frequency-domain convolution property using fundamental-band expressions.

Example 6.13 ◀

Example 6.13 makes the same point as does Ex. 6.10. Had we convolved the periodic spectra instead of the fundamental-band spectra, the procedure would have been nightmarishly complicated due to multiple overlapping segments, especially when $B > \pi/2$. The reader should try to sample the flavor of this approach. The point we are making is that to perform periodic convolution, only the

fundamental-band spectrum is necessary. The periodic extension of the result automatically takes care of any aliasing, if it exists. The caveats for complex signals are mentioned in the concluding remarks of Ex. 6.10.

Width of Periodic Spectra

We know that all DT signals have periodic spectra. As a result of this periodicity, the widths of these spectra are technically infinite. More usefully, however, we can speak of the width of a DT signal's spectrum by restricting our attention to the fundamental band. With such a perspective and assuming no aliasing, the width of the spectrum resulting from the (periodic) convolution of two spectra is the sum of their individual widths, a property we have seen before with linear convolution (both CT and DT). Should aliasing occur, however, this width property is no longer guaranteed to hold. The reader may verify these behaviors for the cases in Ex. 6.13.

▷ **Drill 6.7 (Using the Frequency-Domain Convolution Property)**

In Table 6.1, use the frequency-domain convolution property to derive pair 15 from pairs 11 and 13, assuming $|\Omega_0| < \pi$.

△

6.2.8 Correlation and the Correlation Property

Quite similar to the correlation between CT functions presented in Sec. 1.9.8, the correlation between DT functions $x[\cdot]$ and $y[\cdot]$ is defined as

$$\rho_{x,y}[l] = \sum_{n=-\infty}^{\infty} x[n+l]y^*[n]. \quad (6.24)$$

The correlation function of Eq. (6.24) provides a measure of similarity between two DT signals as a function of the time lag l between them. When the signals $x[n]$ and $y[n]$ are distinct from one another, $\rho_{x,y}[l]$ is termed the *cross-correlation function* between $x[n]$ and $y[n]$. If $y[n] = x[n]$, then the correlation function $\rho_{x,x}[l]$ is termed the *autocorrelation function*. The correlation function $\rho_{x,y}[l]$ is easily expressed in terms of DT convolution as

$$\rho_{x,y}[l] = x[l] * y^*[-l]. \quad (6.25)$$

Applying the complex-conjugation, time-reversal, and time-domain convolution properties yields the correlation property

$$\rho_{x,y}[l] = x[l] * y^*[-l] \iff X(\Omega)Y^*(\Omega). \quad (6.26)$$

As in the CT case, we stress that correlation order is important. That is, $\rho_{x,y}[l]$ does not generally equal $\rho_{y,x}[l]$.

A Frequency-Domain Representation of Signal Energy

Setting $x[n] = y[n]$ and $l = 0$ in Eq. (6.24) yields signal energy

$$\rho_{x,x}[0] = \sum_{n=-\infty}^{\infty} x[n]x^*[n] = \sum_{n=-\infty}^{\infty} |x[n]|^2 = E_x. \quad (6.27)$$

Since $\rho_{x,x}[l]$ has DTFT $X(\Omega)X^*(\Omega)$, it can be synthesized using Eq. (6.2) as

$$\rho_{x,x}[l] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega)X^*(\Omega)e^{j\Omega l} d\Omega = \frac{1}{2\pi} \int_{-\pi}^{\pi} |X(\Omega)|^2 e^{j\Omega l} d\Omega.$$

Substituting $l = 0$ and combining with Eq. (6.27), we obtain

$$E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{2\pi} |X(\Omega)|^2 d\Omega. \quad (6.28)$$

Equation (6.28) is *Parseval's theorem* for the discrete-time Fourier transform, and it shows that $|X(\Omega)|^2$ serves as the *energy spectral density* of $x[n]$.

▷ **Example 6.14 (Using Parseval's Theorem)**

Use Parseval's theorem to find the energy of $x[n] = \text{sinc}(Bn/\pi)$.

From pair 8 of Table 6.2, the fundamental-band spectrum of $x[n]$ is

$$\text{sinc}\left(\frac{Bn}{\pi}\right) \iff \frac{\pi}{B} \Pi\left(\frac{\Omega}{2B}\right), \quad |\Omega| \leq \pi.$$

From Parseval's theorem in Eq. (6.28), we have

$$E_x = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{\pi^2}{B^2} \left[\Pi\left(\frac{\Omega}{2B}\right) \right]^2 d\Omega.$$

Recognizing that $\Pi(\Omega/2B) = 1$ over $|\Omega| \leq B$ and is 0 otherwise, the preceding integral yields

$$E_x = \frac{1}{2\pi} \left(\frac{\pi^2}{B^2} \right) (2B) = \frac{\pi}{B}.$$

Example 6.14 ◀

Table 6.3 summarizes the discrete-time Fourier transform properties just discussed and provides a side-by-side comparison with the corresponding properties of the Fourier transform. In most cases, the properties of the DTFT and the FT are nearly identical.

6.3 LTID System Analysis by the DTFT

From Ch. 4, we know that the (zero-state) output $y[n]$ of a linear, time-invariant, discrete-time (LTID) system is the convolution of its impulse response $h[n]$ and the input $x[n]$,

$$y[n] = x[n] * h[n].$$

Here, convolution provides a time-domain characterization of a system's input-output behavior.

Letting $x[n] \iff X(\Omega)$, $y[n] \iff Y(\Omega)$, and $h[n] \iff H(\Omega)$, it follows from the convolution property of Eq. (6.22) that

$$Y(\Omega) = X(\Omega)H(\Omega). \quad (6.29)$$

Similar to continuous-time systems, this frequency-domain characterization of a system's input-output behavior is often preferred to the time-domain characterization since it replaces convolution with multiplication. Figure 6.11 summarizes these input-output relationships and emphasizes that, in most cases, $H(\Omega)$ serves every bit as well as $h[n]$ to characterize LTID system behavior.

For Eq. (6.29) to hold, $y[n]$, $x[n]$, and $h[n]$ must be (discrete-time) Fourier transformable, which is to say that $Y(\Omega)$, $X(\Omega)$, and $H(\Omega)$ must all exist. Thus, Eq. (6.29) does not apply to (asymptotically) unstable systems, but it does apply to BIBO stable and, with one exception, marginally stable systems. For marginally stable systems, if the input $x[n]$ contains a finite-amplitude sinusoid of the

Discrete-Time Fourier Transform	Fourier Transform
<p>Synthesis: $x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega$</p> <p>Analysis: $X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n}$</p> <p>Duality:</p> <p>Linearity: $ax[n] + by[n] \iff aX(\Omega) + bY(\Omega)$</p> <p>Complex Conjugation: $x^*[n] \iff X^*(-\Omega)$</p> <p>Scaling and Reversal: (see Sec. 6.6) $x[-n] \iff X(-\Omega)$</p> <p>Shifting: $x[n - m] \iff X(\Omega) e^{-j\Omega m}$ $x[n] e^{j\Omega_0 n} \iff X(\Omega - \Omega_0)$</p> <p>Differentiation: $-jnx[n] \iff \frac{d}{d\Omega} X(\Omega)$</p> <p>Time Integration:</p> <p>Convolution: $x[n] * y[n] \iff X(\Omega) Y(\Omega)$ $x[n]y[n] \iff \frac{1}{2\pi} X(\Omega) \circledast Y(\Omega)$</p> <p>Correlation: $\rho_{x,y}[l] = x[l] * y^*[-l] \iff X(\Omega) Y^*(\Omega)$</p> <p>Parseval's: $E_x = \sum_{n=-\infty}^{\infty} x[n] ^2 = \frac{1}{2\pi} \int_{2\pi} X(\Omega) ^2 d\Omega$</p>	<p>Synthesis: $x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega$</p> <p>Analysis: $X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt$</p> <p>Duality: if $x(t) \iff X(\omega)$, then $X(t) \iff 2\pi x(-\omega)$</p> <p>Linearity: $ax(t) + by(t) \iff aX(\omega) + bY(\omega)$</p> <p>Complex Conjugation: $x^*(t) \iff X^*(-\omega)$</p> <p>Scaling and Reversal: $x(at) \iff \frac{1}{ a } X\left(\frac{\omega}{a}\right)$ $x(-t) \iff X(-\omega)$</p> <p>Shifting: $x(t - t_0) \iff X(\omega) e^{-j\omega t_0}$ $x(t)e^{j\omega_0 t} \iff X(\omega - \omega_0)$</p> <p>Differentiation: $\frac{d}{dt} x(t) \iff j\omega X(\omega)$ $-jtx(t) \iff \frac{d}{d\omega} X(\omega)$</p> <p>Time Integration: $\int_{-\infty}^t x(\tau) d\tau \iff \frac{X(\omega)}{j\omega} + \pi X(0) \delta(\omega)$</p> <p>Convolution: $x(t) * y(t) \iff X(\omega) Y(\omega)$ $x(t)y(t) \iff \frac{1}{2\pi} X(\omega) * Y(\omega)$</p> <p>Correlation: $\rho_{x,y}(\tau) = x(\tau) * y^*(-\tau) \iff X(\omega) Y^*(\omega)$</p> <p>Parseval's: $E_x = \int_{-\infty}^{\infty} x(t) ^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) ^2 d\omega$</p>

Table 6.3: Properties of the DTFT and the FT.

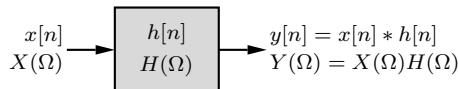


Figure 6.11: Basic block diagram of an LTID system.

system's natural frequency, which leads to unbounded resonance, then the output is not Fourier transformable, and Eq. (6.29) does not hold.[†]

Let us further examine the role of the system frequency response $H(\Omega)$. Equation (6.29) shows

[†]As we shall discuss in Ch. 7, the z -transform is more versatile than the DTFT and is capable of analyzing all kinds of LTID systems whether stable, unstable, or marginally stable. The z -transform can also handle exponentially growing inputs. Compared with the z -transform, the DTFT in system analysis is clumsier. Hence, the z -transform is preferable to the DTFT in LTID system analysis except in filtering, where the DTFT is generally better suited.

that the output signal frequency spectrum is the product of the input signal frequency spectrum and the frequency response of the system. Alternatively, this equation is expressed as

$$|Y(\Omega)| = |X(\Omega)| |H(\Omega)| \quad \text{and} \quad \angle Y(\Omega) = \angle X(\Omega) + \angle H(\Omega). \quad (6.30)$$

Here, $|H(\Omega)|$ is called the *magnitude response* and $\angle H(\Omega)$ is the *phase response* of the system. Equation (6.30) shows that the output magnitude spectrum is the product of the input magnitude spectrum and the magnitude response of the system. The output phase spectrum is the sum of the input phase spectrum and the phase response of the system.

The multiplicative nature of the frequency-domain relationship of Eq. (6.29) conveys a filtering perspective of system behavior. The spectrum $X(\Omega)$ represents the amplitudes and phases of various exponential (sinusoidal) components of the input. When transmitted through an LTID system, each component is individually scaled by the frequency response $H(\Omega)$ and then combined to produce the output $Y(\Omega)$. To better understand this concept, we see from Eq. (5.34) that an LTID system response to an everlasting exponential $e^{j\Omega n}$ is $H(e^{j\Omega})e^{j\Omega n}$, where $H(e^{j\Omega})$, as seen in Eq. (5.35), is really the DTFT of $h[n]$, that is, $H(\Omega)$. Hence, the response of an LTID system to an everlasting input $e^{j\Omega n}$ is represented using a directed arrow notation as

$$e^{j\Omega n} \xrightarrow{} H(\Omega)e^{j\Omega n}. \quad (6.31)$$

Using the DTFT synthesis equation to generalize the input and then invoking the linearity property, we obtain

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega \xrightarrow{} \frac{1}{2\pi} \int_{2\pi} X(\Omega) H(\Omega) e^{j\Omega n} d\Omega = y[n].$$

In other words, the output $y[n]$ is a scaled sum of the responses to all the component frequencies of the input. Just as in Eq. (6.29), $X(\Omega)$ is the input spectrum, and $Y(\Omega)$ is the output spectrum, given by $X(\Omega)H(\Omega)$.

▷ Example 6.15 (LTID System Analysis by the DTFT)

An LTID system is specified by the equation

$$y[n] - 0.5y[n-1] = x[n].$$

Find $H(\Omega)$, the frequency response of this system, and determine the zero-state response $y[n]$ if the input $x[n] = (0.8)^n u[n]$.

Let $x[n] \iff X(\Omega)$ and $y[n] \iff Y(\Omega)$. The DTFT of the system equation yields

$$(1 - 0.5e^{-j\Omega})Y(\Omega) = X(\Omega).$$

According to Eq. (6.29),

$$H(\Omega) = \frac{Y(\Omega)}{X(\Omega)} = \frac{1}{1 - 0.5e^{-j\Omega}} = \frac{e^{j\Omega}}{e^{j\Omega} - 0.5}.$$

Also, $x[n] = (0.8)^n u[n]$. Hence,

$$X(\Omega) = \frac{e^{j\Omega}}{e^{j\Omega} - 0.8}$$

and

$$Y(\Omega) = X(\Omega)H(\Omega) = \frac{(e^{j\Omega})^2}{(e^{j\Omega} - 0.8)(e^{j\Omega} - 0.5)} = \frac{1}{(1 - 0.8e^{-j\Omega})(1 - 0.5e^{-j\Omega})}.$$

To invert $Y(\Omega)$, we first use MATLAB to perform the needed partial fraction expansion. Either of the MATLAB commands `residue` (for positive powers of $e^{j\Omega}$) or `residuez` (for negative powers of $e^{j\Omega}$) will work, although the latter is more convenient for this case.

```
01 [r,p,k] = residuez(1,poly([0.8,0.5]))
r = 2.6667 -1.6667
p = 0.8000 0.5000
k = []
```

Thus,

$$Y(\Omega) = \frac{8/3}{1 - 0.8e^{-j\Omega}} + \frac{-5/3}{1 - 0.5e^{-j\Omega}} = \frac{8e^{j\Omega}/3}{e^{j\Omega} - 0.8} + \frac{-5e^{j\Omega}/3}{e^{j\Omega} - 0.5}. \quad (6.32)$$

Of course, this partial fraction can also be computed manually. Using a modified partial fraction expansion, we have

$$\frac{Y(\Omega)}{e^{j\Omega}} = \frac{e^{j\Omega}}{(e^{j\Omega} - 0.8)(e^{j\Omega} - 0.5)} = \frac{8/3}{e^{j\Omega} - 0.8} + \frac{-5/3}{e^{j\Omega} - 0.5}.$$

Multiplying both sides by $e^{j\Omega}$ produces the result of Eq. (6.32). A complete treatment of partial fraction expansions can be found in [1].

Using Table 6.1, the inverse DTFT of Eq. (6.32) is thus

$$y[n] = \left[\frac{8}{3}(0.8)^n - \frac{5}{3}(0.5)^n \right] u[n].$$

The analysis of LTID systems using the DTFT is similar to the analysis of LTIC systems using the CTFT. As explained earlier, this method does not work for unstable systems or for marginally stable systems where the input causes unbounded resonance. We shall not belabor this method further because it is clumsier and more restrictive than the z -transform method discussed in Ch. 7.

Example 6.15 ◀

Frequency Response from a Difference Equation

The frequency response $H(\Omega)$ is the DTFT of the system's impulse response $h[n]$. As Ex. 6.15 informally shows, we can also determine $H(\Omega)$ directly from an LTID system's difference equation without having to explicitly know the impulse response. More generally, consider a general LTID system described by a difference equation in delay form as (see Eq. (4.61))

$$\sum_{k=0}^K a_k y[n-k] = \sum_{l=0}^K b_l x[n-l]. \quad (6.33)$$

Let the DTFTs of $x[n]$ and $y[n]$ be $X(\Omega)$ and $Y(\Omega)$, respectively. From the time-shifting property, we obtain $x[n-m] \iff X(\Omega)e^{-jm\Omega}$ and $y[n-k] \iff Y(\Omega)e^{-jk\Omega}$. Using these facts, we can take the DTFT of the preceding difference equation to obtain

$$\sum_{k=0}^K a_k Y(\Omega)e^{-jk\Omega} = \sum_{l=0}^K b_l X(\Omega)e^{-jl\Omega}$$

or

$$\left(\sum_{k=0}^K a_k e^{-jk\Omega} \right) Y(\Omega) = \left(\sum_{l=0}^K b_l e^{-jl\Omega} \right) X(\Omega).$$

Because $H(\Omega) = Y(\Omega)/X(\Omega)$, we obtain

$$H(\Omega) = \frac{Y(\Omega)}{X(\Omega)} = \frac{\sum_{l=0}^K b_l e^{-jl\Omega}}{\sum_{k=0}^K a_k e^{-jk\Omega}}. \quad (6.34)$$

▷ **Example 6.16 (System Response to a Sinusoid)**

Determine the zero-state response $y[n]$ to input $x[n] = \cos(3\pi n/10)$ of an LTID system described by the difference equation

$$y[n] - 1.2728y[n-1] + 0.81y[n-2] = \frac{1}{40} (x[n] + 2x[n-1] + 2x[n-2] + 2x[n-3] + x[n-4]).$$

Since the input is a sinusoid of frequency $\Omega = 3\pi/10$, the output is also a sinusoid of the same frequency, modified in gain and phase according to the frequency response $H(\Omega)$. Following Eq. (6.34), MATLAB readily computes the gain and phase.

```
01 Omega = 3*pi/10; A = [1 -1.2728 0.81 0 0]; B = [1 2 2 2 1]/40;
02 H = @(Omega) polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
03 [abs(H(Omega)),angle(H(Omega))]
ans = 0.3598 2.8335
```

For line 02 to properly compute $H(\Omega)$, the coefficient vectors A and B must be the same length, which is why line 01 pads vector A with two zeros (see Prob. 6.3-19). Using these results, the system output is thus

$$y[n] = 0.3598 \cos(3\pi n/10 + 2.8335).$$

Example 6.16 ◀

6.3.1 Distortionless Transmission

In several applications where digital signals are passed through LTID systems, we require that the output waveform be a replica of the input waveform. As in the continuous-time case, transmission is said to be distortionless if the input $x[n]$ and the output $y[n]$ satisfy the condition

$$y[n] = |a|x[n - n_g]. \quad (6.35)$$

Here, the delay n_g (in samples) is assumed to be an integer, and the scale factor $|a|$ is forced real and positive for convenience. The discrete-time Fourier transform of Eq. (6.35) yields

$$Y(\Omega) = |a|X(\Omega)e^{-j\Omega n_g}.$$

Since $Y(\Omega) = X(\Omega)H(\Omega)$, the frequency response of a distortionless system is therefore[†]

$$H(\Omega) = |a|e^{-j\Omega n_g}. \quad (6.36)$$

It follows from this equation that

$$|H(\Omega)| = |a| \quad \text{and} \quad \angle H(\omega) = -\Omega n_g. \quad (6.37)$$

Thus, for distortionless transmission, the magnitude response $|H(\Omega)|$ must be constant, and the phase response $\angle H(\Omega)$ must be a linear function of Ω with slope $-n_g$, as shown in Fig. 6.12. Similar to the CT case, Eqs. (6.36) and (6.37) describe an *ideal linear phase* (ILP) frequency response. Observe that an ideal delay of n_g samples has ILP characteristics with gain $|a| = 1$ (see Eq. (6.16)).

[†]If n_g is not an integer, the function $H(\Omega) = |a|e^{-j\Omega n_g}$ is not 2π -periodic in Ω and, therefore, is not a valid frequency response for any DT system. It is possible, however, to define $H(\Omega) = |a|e^{-j\Omega n_g}$ over the fundamental band $|\Omega| < \pi$ and then periodically replicate the result. When n_g is not an integer, such a system can produce an output that is quite different looking than the input, although the input and output spectra are basically equivalent. Such a system seems distortionless in the frequency domain but not the time domain. Additional insights into this case are provided by the Sec. 6.4 discussion of non-integer shifts.

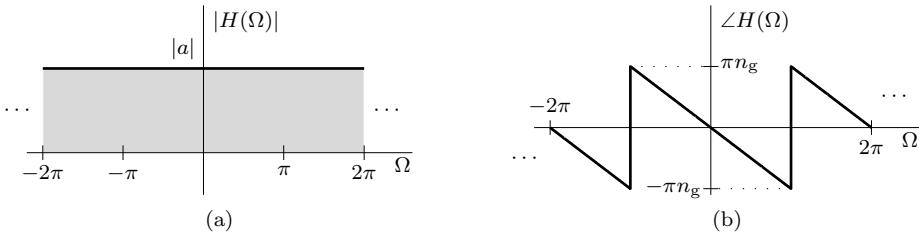


Figure 6.12: Distortionless transmission: (a) magnitude response and (b) phase response.

Measure of Delay Variation

It is often thought (erroneously) that flatness of magnitude response $|H(\Omega)|$ alone can guarantee signal quality. However, a system may have a flat magnitude response and yet distort a signal beyond recognition if the phase response is not linear (n_g not constant over the band of interest). Example 2.4 on page 95 details one such example.

Thus, in addition to a constant magnitude response, distortionless transmission requires that the system possess *linear phase* characteristics. In practice, many systems have phase characteristics that are only approximately linear. A convenient way of judging phase linearity is to plot the negative of the slope of $\angle H(\Omega)$ as a function of frequency. This slope, which is a constant for an ideal linear phase system but a function of Ω in the general case, is expressed as

$$n_g(\Omega) = -\frac{d}{d\Omega} \angle H(\Omega). \quad (6.38)$$

Termed “group delay” or “envelope delay,” $n_g(\Omega)$ plays an important role in bandpass signal transmission, as the following discussion shows.

Distortionless Transmission in Bandpass Systems

As shown in Eq. (6.37), distortionless transmission generally requires a flat magnitude response and a linear phase response that passes through the origin. As in the case of continuous-time systems, these distortionless transmission conditions can be relaxed for discrete-time bandpass systems. In particular, distortionless bandpass transmission requires flat magnitude response and linear phase response (constant n_g), *but only over the passband of interest*. Further, the phase response need not pass through the origin. Thus, for distortionless bandpass transmission, the frequency response is similar to that in Eq. (6.36) except that it need be satisfied only over the passband, and the phase offset ϕ_0 need not be zero, as shown in Fig. 6.13. Over the (fundamental) bands of interest, we thus describe $H(\Omega)$ as

$$H(\Omega) = \begin{cases} |a|e^{j(\phi_0 - \Omega n_g)} & \Omega_c - B \leq \Omega \leq \Omega_c + B \\ |a|e^{j(-\phi_0 - \Omega n_g)} & -\Omega_c - B \leq \Omega \leq -\Omega_c + B \end{cases}, \quad (6.39)$$

where it is assumed that $0 < \Omega_c - B < \Omega_c + B < \pi$. The system of Eq. (6.39) is said to have *generalized linear phase* (GLP) in contrast to the ideal linear phase (ILP) characteristics in Fig. 6.12. Most practical systems satisfy the conditions of Eq. (6.39), approximately, at least over a very small band.

To demonstrate that Eq. (6.39) describes distortionless transmission for a bandpass signal, we follow a proof that is nearly identical to the CT case in Sec. 2.2.2. A modulated signal $x_{bp}[n] = x[n] \cos(\Omega_c n)$ is a bandpass signal, where $\cos(\Omega_c n)$ is the carrier and $x[n]$, which is a real lowpass signal of bandwidth B , is the envelope of $x_{bp}[n]$.[†] The bandwidth of the bandpass (modulated) signal $x[n]$ is $2B$, and the center frequency of the spectrum $X(\Omega)$ is $\pm\Omega_c$.

[†]As in the CT case, the envelope of $x_{bp}[n]$ is well defined only when B , the bandwidth of the envelope, is much smaller than the carrier Ω_c .

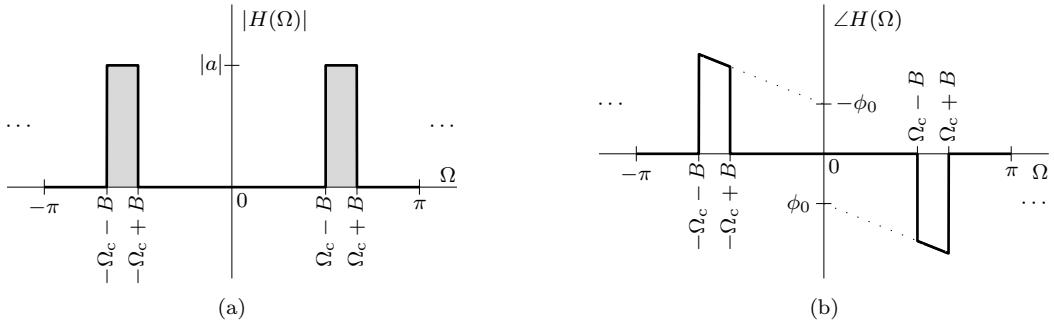


Figure 6.13: Generalized linear phase characteristics for $|\Omega| \leq \pi$: (a) $|H(\Omega)|$ and (b) $\angle H(\Omega)$.

We shall now show that the transmission of $x_{\text{bp}}[n]$ through $H(\Omega)$ results in distortionless transmission of the envelope $x[n]$ while the carrier phase changes by ϕ_0 relative to this envelope. To simplify our discussion, we first write $x_{\text{bp}}[n]$ as

$$x_{\text{bp}}[n] = \operatorname{Re} \{ x[n] e^{j\Omega_c n} \}.$$

Since the system is real, Eq. (5.24) ensures that the output is given as

$$y_{\text{bp}}[n] = \operatorname{Re} \{ H \{ x[n] e^{j\Omega_c n} \} \}.$$

To find $y_{\text{bp}}[n]$, we only need to determine $H \{ x[n] e^{j\Omega_c n} \}$, the system response to $x[n] e^{j\Omega_c n}$. The frequency-shifting property of Eq. (6.18) yields the DTFT of $x[n] e^{j\Omega_c n}$ as $X(\Omega - \Omega_c)$, and the system response to this input is $H(\Omega)X(\Omega - \Omega_c)$. Because $X(\Omega - \Omega_c)$ is 0 for $-\pi < \Omega < 0$, we express this result in the fundamental band using Eq. (6.39) as

$$H(\Omega)X(\Omega - \Omega_c) = |a| e^{j(\phi_0 - \Omega n_g)} X(\Omega - \Omega_c).$$

Using both the time-shifting and frequency-shifting properties of Eqs. (6.16) and (6.18), the inverse discrete-time Fourier transform yields

$$|a| e^{j\phi_0} X(\Omega - \Omega_c) e^{-j\Omega n_g} \iff |a| e^{j\phi_0} x[n - n_g] e^{j\Omega_c(n - n_g)}.$$

Taking the real portion, the bandpass output $y_{\text{bp}}(t)$ is thus[†]

$$y_{\text{bp}}[n] = |a| x[n - n_g] \cos [\Omega_c(n - n_g) + \phi_0]. \quad (6.40)$$

Here, n_g , the envelope (or group) delay, is the negative slope of $\angle H(\Omega)$ at Ω_c . The output envelope $x[n - n_g]$ is a delayed version of the input envelope $x[n]$, and its shape is not affected by extra phase ϕ_0 of the carrier. Observe that the output $y[n]$ is basically the input $x[n]$ delayed by n_g , except that the output carrier acquires an extra phase ϕ_0 . Equation (6.40) is also expressed in alternate form as

$$y_{\text{bp}}[n] = |a| x[n - n_g] \cos [\Omega_c(n - n_p)],$$

where $n_p = -\angle H(\Omega_c)/\Omega_c$ is called the *phase delay*. In physical systems, the quantities n_g , ϕ_0 , and n_p are generally functions of Ω .

[†]A more general bandpass signal with spectrum centered at $\Omega = \Omega_c$ can be represented as $x_{\text{bp}}[n] = x[n] \cos(\Omega_c n + \theta[n])$, where $x[n]$ is the envelope of the signal, and $\theta[n]$ is a time-varying phase. When this signal is passed through the system in Eq. (6.39), the system output is given by (see Prob. 6.3-17) $y_{\text{bp}}[n] = |a| x[n - n_g] \cos(\Omega_c(n - n_g) + \phi_0 + \theta[n - n_g])$.

A Caution

Recall that the phase response associated with the magnitude response may have jump discontinuities when the frequency response changes sign. Jump discontinuities also arise because of the use of principal value for phase. In computing the group delay (Eq. (6.38)), we generally ignore these jump discontinuities.

6.3.2 Ideal and Realizable Filters

Ideal DT filters allow distortionless transmission of a certain band of frequencies (passband) and suppress all the remaining frequencies (stopband). Figure 6.14 displays the magnitude responses of ideal lowpass, highpass, bandpass, and bandstop filters. At first glace, these responses may seem different from their CT counterparts of Fig. 2.14. These differences arise from the 2π -periodicity that is implicit in all DT systems. However, when viewed over the fundamental band, which is the only band that really matters, the DT responses of Fig. 6.14 look identical to the CT responses of Fig. 2.14. As in the CT case, it is a simple matter to transform a lowpass DT filter to any of the other three basic types. Thus, for the time being, we shall concentrate solely on lowpass filters. Most of our subsequent discussions and observations about lowpass filters apply to highpass, bandpass, and bandstop filters as well.

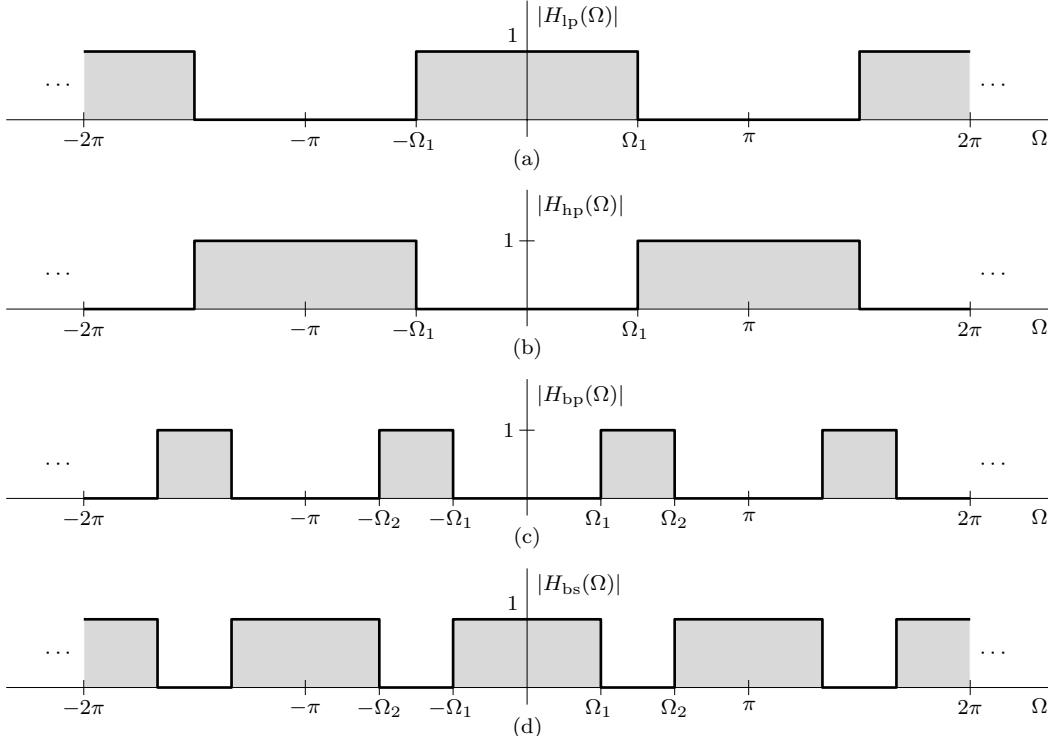


Figure 6.14: Ideal DT filter magnitude responses: (a) lowpass, (b) highpass, (c) bandpass, and (d) bandstop.

In a perfect world, an ideal filter passes desired input frequencies without delay, which is to say that the filter has a zero phase response. As discussed in Sec. 2.3, the causality requirement of realizable filters makes such expectations impractical. Some delay in the output is inevitable to achieve acceptable response characteristics while also meeting the constraint of causality (realizability). The linear phase characteristics of distortionless transmission provide a fixed amount of delay over all

frequencies of interest, an acceptable compromise for most DT filter applications. Ideal DT filters therefore permit linear phase, and although some delay is acquired, the output is still classified as distortionless.

Figures 6.15a and 6.15b illustrate the magnitude and phase characteristics of an ideal lowpass filter. The phase response is linear over the passband of interest and has a slope of $-n_g$, which results in a delay of n_g samples for all its input components of frequencies below Ω_1 rad/sample. Therefore, if the input is a signal $x[n]$ bandlimited to Ω_1 , then the output $y[n]$ is $x[n]$ delayed by n_g ; that is, if $x[n]$ bandlimited to Ω_1 , then $y[n] = x[n - n_g]$.

Over the fundamental band, this filter's frequency response can be expressed as $\Pi(\Omega/2\Omega_1) e^{-j\Omega n_g}$. Hence, $H(\Omega)$ valid for all Ω is[†]

$$H(\Omega) = \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{2\Omega_1}\right) e^{-j\Omega n_g}.$$

The unit impulse response $h[n]$ of this filter, obtained using pair 8 of Table 6.1 and the time-shifting property, is

$$h[n] = \frac{\Omega_1}{\pi} \operatorname{sinc}\left[\frac{\Omega_1(n - n_g)}{\pi}\right]. \quad (6.41)$$

Figure 6.15c shows $h[n]$ for $\Omega_1 = \pi/3$ and $n_g = 12$. By definition, the impulse response is the output in response to $\delta[n]$, an input that begins (and ends) at $n = 0$. Because the sinc function has infinite duration, we see that regardless of how large the delay n_g is, $h[n]$ cannot be rendered causal (realizable). Thus, despite the assistance of linear phase and the delay it provides, it is not possible to realize an ideal lowpass frequency response. The same conclusion is true for ideal highpass, bandpass, and bandstop filters as well.

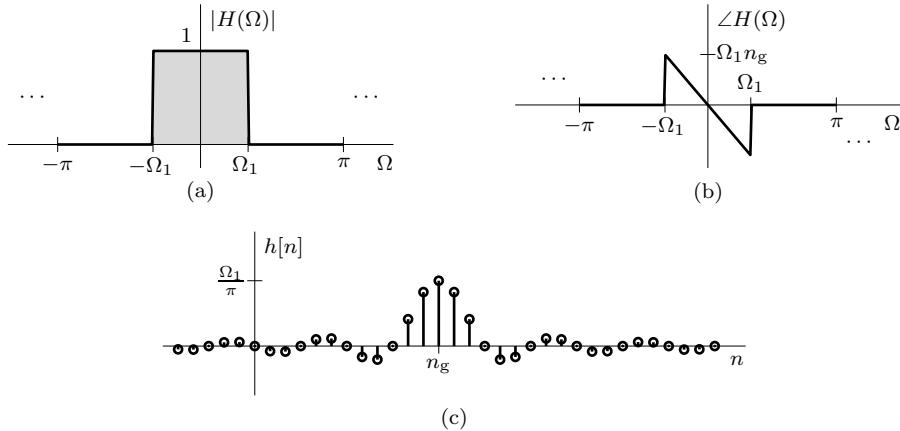


Figure 6.15: Ideal LPF ($\Omega_1 = \pi/3$ and $n_g = 12$): (a) $|H(\Omega)|$, (b) $\angle H(\Omega)$, and (c) $h[n]$.

To obtain a realizable filter, we must approximate the ideal impulse response $h[n]$ with some causal impulse response $\hat{h}[n]$. One simple and practical approach is to simply multiply $h[n]$ by a (shifted) rectangular window according to[‡]

$$\hat{h}[n] = h[n] \Pi\left(\frac{n - n_g}{2n_g^+}\right). \quad (6.42)$$

[†]Since kn_g is an integer, we write $e^{-j(\Omega-2\pi k)n_g}$ in the simpler and equivalent form of $e^{-j\Omega n_g}$.

[‡]We use n_g^+ in Eq. (6.42) to ensure that $\Pi(\cdot)$ is 1 at $n = 0$ and $2n_g$ rather than 0.5.

Removing the tails of $h[n]$ renders it causal, but it also distorts the desired ideal frequency response. To illustrate, let us apply Eq. (6.42) to the impulse response of Fig. 6.15c ($\Omega_1 = \pi/3$ and $n_g = 12$). MATLAB is next used to perform the necessary calculations.

```

01 Omega1 = pi/3; ng = 12; n = -5:30; Omega = linspace(-pi,pi,5001);
02 hhat = @(n) Omega1/pi*sinc(Omega1*(n-ng)/pi).*((n>=0)&(n<=2*ng));
03 Hhat = @(Omega) polyval(hhat(0:2*ng),exp(1j*Omega))./exp(1j*Omega*(2*ng));
04 subplot(311); stem(n,hhat(n));
05 subplot(312); plot(Omega,abs(Hhat(Omega)));
06 subplot(313); plot(Omega,unwrap(angle(Hhat(Omega))));
```

The resulting impulse response $\hat{h}[n]$ is causal (Fig. 6.16a), but the magnitude response $|\hat{H}(\Omega)|$ (Fig. 6.16b) is degraded, particularly near the filter transition bands. In this case, the phase response $\angle\hat{H}(\Omega)$ (Fig. 6.16c) remains linear. As the delay n_g is increased, the frequency response $\hat{H}(\Omega)$ more closely approximates the ideal. The price of accuracy is increased delay. Chapter 8 details this and many other digital filter design methods.

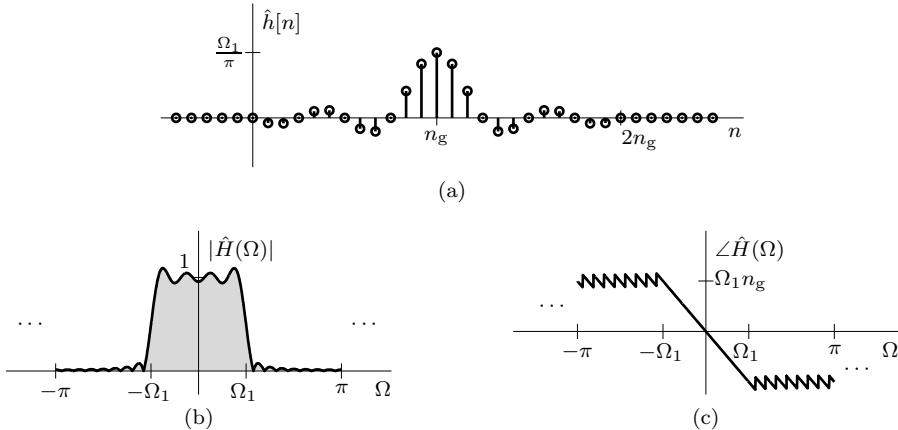


Figure 6.16: Realizable LPF ($\Omega_1 = \pi/3$ and $n_g = 12$): (a) $\hat{h}[n]$, (b) $|\hat{H}(\Omega)|$, and (c) $\angle\hat{H}(\Omega)$.

▷ Drill 6.8 (Practical LPF Design)

Using a delay of $n_g = 5$ and a cutoff frequency of $\Omega_1 = 2\pi/3$, use Eqs. (6.41) and (6.42) to design a realizable discrete-time lowpass filter. Plot the impulse response $\hat{h}[n]$, the magnitude response $|\hat{H}(\Omega)|$, and the phase response $\angle\hat{H}(\Omega)$.

△

6.4 Connection between the DTFT and the CTFT

Chapter 3 presents procedures to travel between the continuous-time and discrete-time worlds. Sampling generates a discrete-time signal from a continuous-time signal (a CT-to-DT process), and ideal interpolation (Eq. (3.13)) reconstructs a continuous-time signal from its sample values (a DT-to-CT process). Such conversions between the continuous-time and discrete-time domains can be extremely useful, even for purely digital systems that never once produce a continuous-time signal. The connection between the CT and DT domains aids in the analysis of A/D and D/A conversion and also in the digital processing of analog signals. It provides insights into the spectra of downsampled, upsampled, and interpolated signals. It allows us to compute the Fourier transform

of continuous-time signals numerically using digital methods such as the DTFT and the efficient fast Fourier transform (FFT). In addition, it allows us to interpret the meaning of a non-integer shift of a DT signal. Let us now connect the DTFT and the CTFT to further explore these ideas.

Before proceeding, we first address a possible source of confusion. Like many engineering texts, we represent the CTFT of $x(t)$ and the DTFT of $x[n]$ both using the same notation $X(\cdot)$. Normally, the two utilize different variables and do not occur together, so there is no ambiguity in meaning. In this section, however, the CTFT and the DTFT both occur in the same equations, and we cannot rely on the argument variables to distinguish transform type. To help avoid confusion, in this section we shall subscript all CTFT pairs with a “c.” Thus, $X(\omega T)$ is a DTFT and $X_c(\Omega/T)$ is a CTFT, despite that the former is written in terms of ω and the latter in terms of Ω .

Consider a continuous-time signal $x_c(t)$ that has sample values $x[n]$ spaced every T seconds. In this way, the n th sample of $x_c(t)$ at $t = nT$ is equal to $x[n]$, or

$$x_c(nT) = x[n].$$

Figure 6.17a shows one such signal $x_c(t)$, and Fig. 6.17b shows its spectrum $X_c(\omega)$. For simplicity, $x_c(t)$ is shown to be bandlimited with a bandwidth $B \leq \pi/T$ rad/s (1/2T Hz).

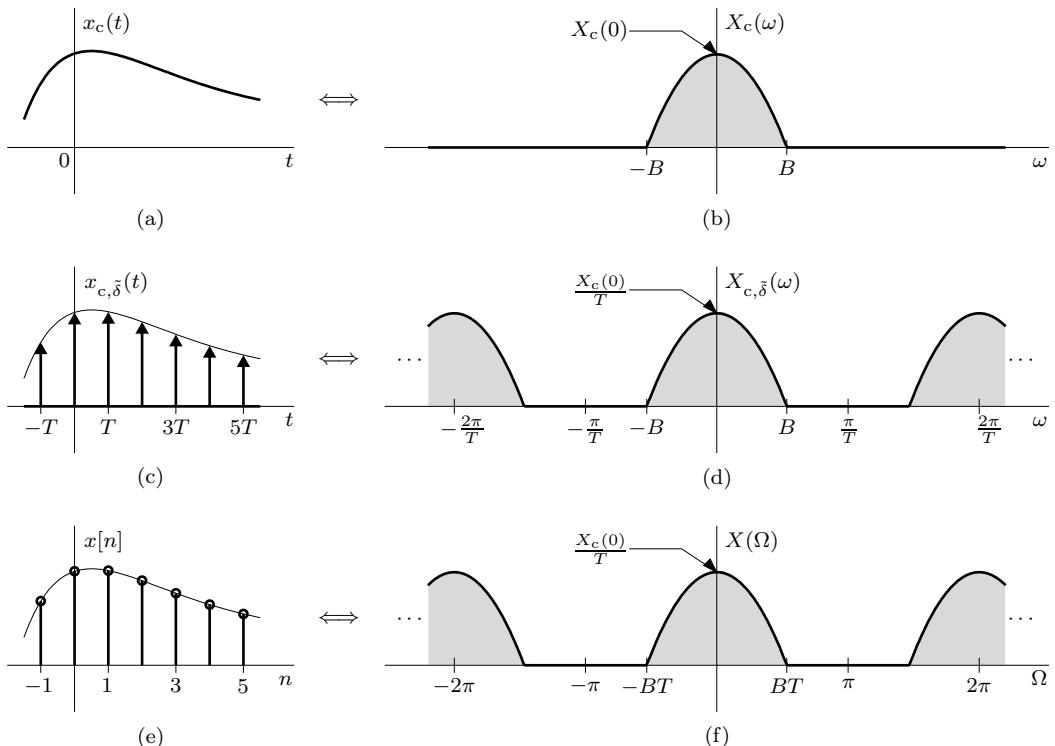


Figure 6.17: Connection between the DTFT and the CTFT.

Because $x_c(nT) = x[n]$, we can use Eq. (3.3) to express $x_{c,\tilde{\delta}}(t)$ (Fig. 6.17c), the impulse-sampled version of $x_c(t)$, as

$$x_{c,\tilde{\delta}}(t) = \sum_{n=-\infty}^{\infty} x[n]\delta(t - nT).$$

Because $\delta(t - nT) \iff e^{-jn\omega T}$, the CT Fourier transform of the preceding equation yields

$$X_{c,\tilde{\delta}}(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-jn\omega T}.$$

In Sec. 3.1, we have shown that $X_{c,\tilde{\delta}}(\omega)$ is $\frac{1}{T}X_c(\omega)$ repeating periodically with period $\omega_s = 2\pi/T$, as illustrated in Fig. 6.17d. Notice that the right-hand side of the equation for $X_{c,\tilde{\delta}}(\omega)$ is just $X(\Omega)$, the DTFT of $x[n]$, with Ω replaced by ωT . Thus,

$$X_{c,\tilde{\delta}}(\omega) = X(\Omega)|_{\Omega=\omega T} = X(\omega T) \quad \text{and} \quad X(\Omega) = X_{c,\tilde{\delta}}(\omega)\Big|_{\omega=\Omega/T} = X_{c,\tilde{\delta}}(\Omega/T). \quad (6.43)$$

Equation (6.43) confirms the equivalence of impulse sampling and point sampling. Within a frequency scale factor of T , the CTFT of an impulse-sampled signal (Fig. 6.17d) is identical to the DTFT of the signal's point samples (Fig. 6.17f). Since Ω equals ωT , $\omega = 2\pi/T$ in Fig. 6.17d corresponds to $\Omega = 2\pi$ in Fig. 6.17f. Further noting that $\omega_s = 2\pi/T$, we combine Eq. (6.43) with Eq. (3.6) from Ch. 3 to obtain

$$X(\omega T) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c(\omega - k\omega_s) \quad \text{and} \quad X(\Omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\Omega - 2\pi k}{T}\right). \quad (6.44)$$

This is the relationship that we seek. From Eq. (6.44), which connects the DTFT to the CTFT, we see that the DTFT of a sequence $x[n]$ equals a scaled periodic replication of the CTFT of *any* CT signal $x_c(t)$ that has samples $x_c(nT) = x[n]$. Shown graphically, we see that the DTFT of Fig. 6.17f is a scaled and periodic replication of the CTFT of Fig. 6.17b.

▷ Example 6.17 (Connecting the DTFT to the CTFT)

Using the CT signal $x_c(t) = \text{sinc}(t)$, sketch the corresponding sampled signal $x[n]$ and its spectrum $X(\Omega)$ using sampling intervals of (a) $T = 1/2$, (b) $T = 1$, and (c) $T = 3/2$.

From pair 8 of Table 1.1, we know that the spectrum of $x_c(t) = \text{sinc}(t)$ is the simple gate function $X_c(\omega) = \Pi(\frac{\omega}{2\pi})$, which has radian bandwidth $B = \pi$ rad/s. Both $x_c(t)$ and $X_c(\omega)$ are shown in Fig. 6.18.

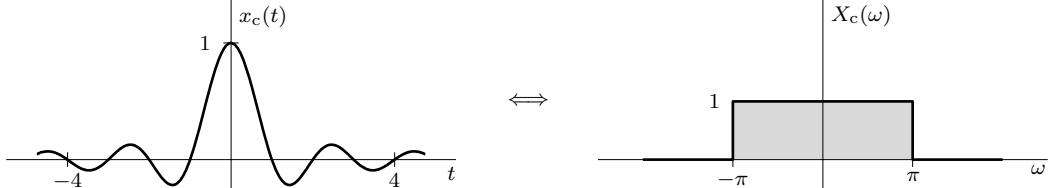


Figure 6.18: Signal $x_c(t) = \text{sinc}(t)$ and its spectrum $X(\omega)$.

(a) $T = 1/2$

As shown in Fig. 6.19a, a sampling interval of $T = 1/2$ produces a DT signal $x[n]$ that captures the essential character of $x_c(t)$. The corresponding spectrum $X(\Omega)$, found using Eq. (6.44), is constructed of the replicates $\frac{1}{T}X_c\left(\frac{\Omega - 2\pi k}{T}\right)$. Compared with the original spectrum $X_c(\omega)$, each replicate is compressed and amplified by a factor of $1/T = 2$. The π -width of each replicate (1/2 of the original's $2B = 2\pi$ width) easily fits into the fundamental band, so no aliasing occurs. As a result, no information is lost during sampling, and the sampled signal's spectrum $X(\Omega)$ closely resembles the original, at least over the fundamental band.

(b) $T = 1$

A sampling interval of $T = 1$ produces a DT delta function $x[n] = \delta[n]$, as shown in Fig. 6.19b. We expect an impulse function to possess a flat spectrum, and this is precisely the result of Eq. (6.44) when the replicates are added together. Notice that the replicates that comprise $X(\Omega)$ just touch one another but do not overlap. Clearly, increasing the sampling interval beyond $T = 1$ will result

in destructive aliasing.

(c) $T = 3/2$

As shown in Fig. 6.19c, a sampling interval of $T = 3/2$ produces a DT signal $x[n]$ that is an undersampled version of $x_c(t)$. In this case, the sampling interval is so wide that important information about $x_c(t)$ is lost. The spectrum $X(\Omega)$, again found using Eq. (6.44), is comprised of overlapping replicates. This overlap (aliasing) distorts the spectrum $X(\Omega)$ and makes recovery of $x_c(t)$ impossible.

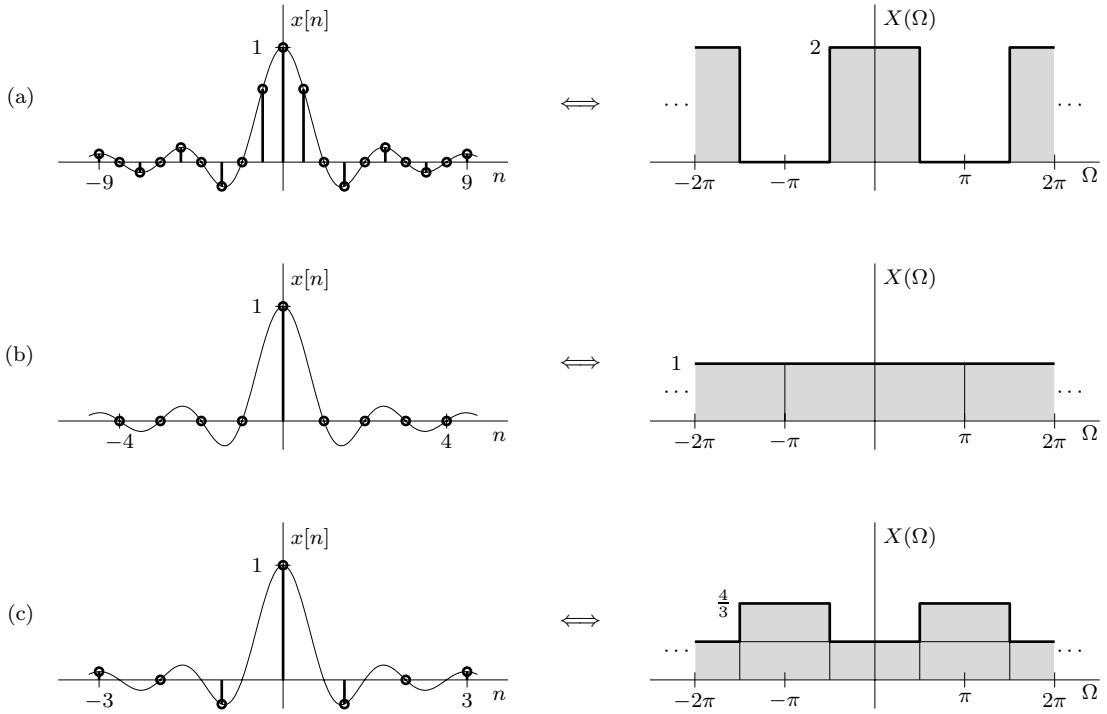


Figure 6.19: Sampled signal $x[n]$ and its spectrum $X(\Omega)$ for (a) $T = 1/2$, (b) $T = 1$, and (c) $T = 3/2$.

Example 6.17 ◀

It may seem impossibly strange that Eq. (6.44) holds for an infinite number of signals $x_c(t)$. To understand this apparent mystery, remember that $x_c(t)$ must satisfy $x_c(nT) = x[n]$, which is to say that although many choices of $x_c(t)$ exist, only their samples must match $x[n]$. Thus, although each possible $x_c(t)$ has a unique CTFT $X_c(\omega)$, the sum $\sum_{k=-\infty}^{\infty} X_c(\omega - k\omega_s)$ never changes; each spectral copy in the periodic replication overlaps in a very special way so that the sum remains unchanged. Of the infinite choices for $x_c(t)$, the *bandlimited interpolation* of $x[n]$ is arguably the most useful. Let us see what makes this choice so special.

Bandlimited Interpolation of $x[n]$

The *bandlimited interpolation* of $x[n]$ is constructed according to Eq. (3.13) as

$$x_c(t) = \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t - nT}{T}\right). \quad (6.45)$$

As seen in Ch. 3, this construction results in a bandlimited signal with bandwidth no greater than $1/2T$ Hz or π/T rad/s. For a given T , the signal $x_c(t)$, as defined in Eq. (6.45), is the *smallest bandwidth signal* whose samples are $x[n]$.[†]

Equation (6.44) represents a 2π -periodic repetition of the spectrum $X_c(\cdot)$, as shown in Fig. 6.17f. Now, if $x_c(t)$ is a bandlimited interpolation of $x[n]$, it is bandlimited to $|\omega| \leq \pi/T$. Hence, $X_c(\Omega/T)$ is bandlimited to $|\Omega| \leq \pi$ (fundamental band), and the replications in Eq. (6.44) do not overlap. Over the fundamental band, we therefore have

$$X(\Omega) = \frac{1}{T} X_c\left(\frac{\Omega}{T}\right) \quad \text{and} \quad X_c(\omega) = T X(\omega T), \quad |\Omega| = |\omega T| \leq \pi. \quad (6.46)$$

Equation (6.46) shows that when $x_c(t)$ is the bandlimited interpolation of $x[n]$, then, in the fundamental band, the DTFT and CTFT are (within a scale factor) frequency-scaled versions of one another. Because $\Omega = \omega T$, frequencies Ω and ω are related by a scaling factor T . Hence, the bandwidth of $x[n]$ (in cycles/sample) is T times the bandwidth of $x_c(t)$ (in Hz). In fact, Eq. (6.46) is the underlying basis of Sec. 6.1.2, where we find the DTFT from the CTFT.

We shall now consider some applications of Eqs. (6.44) and (6.46).

▷ Example 6.18 (Computing the DTFT from a Bandlimited CTFT)

Consider a bandlimited signal $x_c(t)$ with spectrum

$$X_c(\omega) = \begin{cases} \frac{100}{\omega^2 + 10^4} & |\omega| < 50\pi \\ 0 & \text{otherwise} \end{cases}.$$

For $T = 0.01$, determine the DTFT $X(\Omega)$ of $x[n] = x_c(nT)$. Discuss what happens if instead $T = 0.1$.

For $T = 0.01$, the radian folding frequency is $\omega_s/2 = \pi/T = 100\pi$. Since the 50π bandwidth of $x_c(t)$ is less than the 100π folding frequency, no aliasing occurs, and we can write the DTFT $X(\Omega)$ directly using Eq. (6.46) as $X(\Omega) = \frac{1}{T} X_c\left(\frac{\Omega}{T}\right)$. Thus, over the fundamental band $\Omega \leq \pi$,

$$X(\Omega) = \begin{cases} \frac{1}{\Omega^2 + 1} & |\Omega| < \pi/2 \\ 0 & \text{otherwise} \end{cases}.$$

By changing T from 0.01 to 0.1, the radian folding frequency decreases tenfold to 10π . Thus, although $x_c(t)$ is bandlimited, it is not sufficiently bandlimited to prevent aliasing. To obtain the $X(\Omega)$, the DTFT of the now aliased signal, we must instead use Eq. (6.44), which, due to the multiple overlapping spectra, is substantially more complex.

Example 6.18 □

Non-Integer Shift

The shifting property of Eq. (6.16) is applicable, and meaningful, only for integer values of the shift m because the sequence $x[n]$ can be shifted only by an integer number of samples. What, then, is the inverse DTFT of the fundamental band expression $X(\Omega)e^{-j\Omega m}$ when m is non-integer? This question can be readily resolved by considering the continuous-time signal $x_c(t)$, which we take as the bandlimited interpolation of $x[n]$. Let

$$Y(\Omega) = X(\Omega)e^{-j\Omega m}, \quad |\Omega| \leq \pi. \quad (6.47)$$

[†]To prove this, let us assume that there exists another signal $x'_c(t)$ that satisfies $x'_c(nT) = x[n]$ and has bandwidth smaller than that of $x_c(t)$. Since both signals have bandwidth $\leq 1/2T$ Hz, their difference $x_c(t) - x'_c(t)$ also has bandwidth $\leq 1/2T$ Hz. Hence, we can reconstruct signal $x_c(t) - x'_c(t)$ using its samples $x_c(nT) - x'_c(nT)$ in Eq. (6.45). But because $x_c(nT) = x'_c(nT) = x[n]$, we obtain $x'_c(t) - x_c(t) = 0$, and $x_c(t) = x'_c(t)$. Hence, $x_c(t)$ is unique with the smallest bandwidth for a given T .

We shall find $y[n]$, the IDTFT of $Y(\Omega)$, when m is non-integer.

If $x_c(t)$ is the bandlimited interpolation of $x[n]$ and $y_c(t)$ is the bandlimited interpolation of the desired sequence $y[n]$, then the samples of $x_c(t)$ at intervals of T seconds are $x[n]$ and the samples of $y_c(t)$ at intervals of T are $y[n]$. We now show that $y_c(t)$ is just a time-shifted version of $x_c(t)$, given by $y_c(t) = x_c(t - mT)$. Observe from Eqs. (6.46) and (6.47) that

$$Y_c(\omega) = T Y(\omega T) = T X(\omega T) e^{-j\omega T m} = X_c(\omega) e^{-j\omega T m}, \quad |\omega T| \leq \pi.$$

Applying the time-shifting property of Eq. (1.87) for continuous-time signals, we have

$$y_c(t) = x_c(t - mT).$$

This shows that $y_c(t)$ is the signal $x_c(t)$ delayed by mT seconds. Therefore, $y[n]$ can be interpreted as samples of the delayed signal $x_c(t - mT)$. To illustrate graphically, Fig. 6.20a shows a DT signal $x[n]$ and its bandlimited interpolation $x_c(t)$. Using Eq. (6.47) and $m = 1.5$, Fig. 6.20b shows the corresponding DT signal $y[n]$, which is just samples of the bandlimited interpolation $y_c(t) = x_c(t - 1.5T)$.

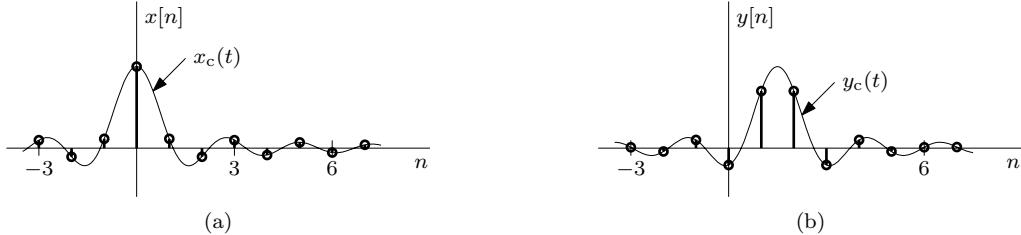


Figure 6.20: Interpretation of time-shifting property for non-integer values of shift.

As Fig. 6.20 demonstrates, when m is non-integer, the sample values $y[n]$ are totally different from those in $x[n]$. When m is integer, however, the results match the shifting property of Eq. (6.16), and $y[n]$ is an exact copy of $x[n]$ shifted by m units. To summarize,

$$\text{if } Y(\Omega) = X(\Omega)e^{-jm\Omega} \text{ for } |\Omega| \leq \pi, \quad \text{then } y[n] = y_c(nT) = x_c(nT - mT), \quad (6.48)$$

where $x_c(t)$ and $y_c(t)$ are the bandlimited interpolations of $x[n]$ and $y[n]$, respectively.

▷ Example 6.19 (Non-Integer Shift)

Given a signal $x[n] = \delta[n]$ with spectrum $X(\Omega) = 1$, determine $y[n]$ if $Y(\Omega) = X(\Omega)e^{-jm\Omega}$ for $|\Omega| \leq \pi$. Here, m is any real number, not necessarily an integer.

Using the definition of the IDTFT (Eq. (6.2)), we have

$$\begin{aligned} y[n] &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\Omega) e^{-jm\Omega} e^{jn\Omega} d\Omega \\ &= \frac{1}{2\pi} \left(\frac{e^{j(n-m)\pi} - e^{-j(n-m)\pi}}{j(n-m)} \right) \\ &= \text{sinc}(n - m). \end{aligned}$$

The same result, however, can be obtained more simply. Using Eq. (6.45) to recognize that $x_c(t) = \text{sinc}(t/T)$ is the bandlimited interpolation of $x[n] = \delta[n]$, we know from Eq. (6.48) that

$$y[n] = x_c(nT - mT) = \text{sinc}(n - m).$$

In fact, this result demonstrates that a DT system $H(\Omega) = e^{-jm\Omega}$ for $|\Omega| \leq \pi$ has an impulse response $h[n] = \text{sinc}(n - m)$.

Example 6.19 ▶

Using Samples to Compute the CTFT

For bandlimited signals, Eq. (6.46) shows that the CTFT is easily computed from the DTFT. Since it involves summation rather than integration, the DTFT (Eq. (6.1)) is easier to compute than the CTFT (Eq. (1.75)), particularly if using a digital computing device such as a computer. Thus, a signal's samples and the DTFT provide an efficient method to find the CTFT of bandlimited signals.

Consider a continuous-time, bandlimited signal $x_c(t)$. We shall associate its samples at intervals of T seconds with a discrete-time signal $x[n]$, that is,

$$x_c(nT) = x[n].$$

If the bandwidth of $x_c(t)$ is B Hz, then $T \leq 1/2B$ ensures that the sampling rate meets the Nyquist criterion. Combining Eqs. (6.46) and (6.1), we have

$$X_c(\omega) = \begin{cases} TX(\omega T) = T \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega T n} & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}. \quad (6.49)$$

Equation (6.49) shows that the CTFT $X_c(\omega)$ is T times the DTFT $X(\Omega)$ at $\Omega = \omega T$, which is computed directly through a summation of the (scaled) samples $x[n]$. This relationship permits us to use modern computer techniques to compute the continuous-time Fourier transform. Chapter 9 extends and develops these ideas further.

6.5 Digital Processing of Analog Signals

Many, if not most, books on discrete-time systems adopt a rather loose terminology so that “analog” is taken to mean “continuous-time” and “digital” is taken to mean “discrete-time.” As explained in Sec. 1.1, strictly speaking, they are not the same things. In practice, however, continuous-time systems are most likely to be analog, and discrete-time systems are most likely to be digital. In keeping with such conventions, we take “digital processing of analog signals” to mean the processing of continuous-time signals using discrete-time systems.

Continuous-time signals can be processed by discrete-time systems as shown in Fig. 6.21a. This system requires two interfaces: a continuous-to-discrete (C/D) converter and a discrete-to-continuous (D/C) converter. The C/D converter samples a continuous-time (“analog”) input signal $x_c(t)$ to convert it into a discrete-time (“digital”) signal $x[n]$, which is applied to an LTI discrete-time system with frequency response $H(\Omega)$. The discrete-time output $y[n]$ of $H(\Omega)$ is converted into a continuous-time signal $y_c(t)$ by the D/C converter. In this manner, the overall system $\hat{H}_c(\omega)$ in Fig. 6.21a is equivalent to (or closely approximates) the LTI continuous-time system $H_c(\omega)$ shown in Fig. 6.21b. The sampling rates in both the C/D and D/C converters in Fig. 6.21a are assumed to be identical, although it is not necessary.

An Implicit Assumption: Bandlimited Signals Sampled above Nyquist

When an analog signal is processed through its samples, it is important to remember that the signal must be bandlimited, and the sampling rate must be no less than the Nyquist rate. If these conditions are not satisfied, then the samples do not convey the full information about the signal, and the system in Fig. 6.21a cannot be equivalent to the LTIC system in Fig. 6.21b.[†] Thus, unless otherwise stated,

[†]There are exceptions. For example, if the signal processing requires suppression of the higher end of its spectrum, we can tolerate distortion in this portion of the spectrum (which will be suppressed anyway). Hence, sub-Nyquist sampling can be tolerated in such cases. See Ex. 6.22.

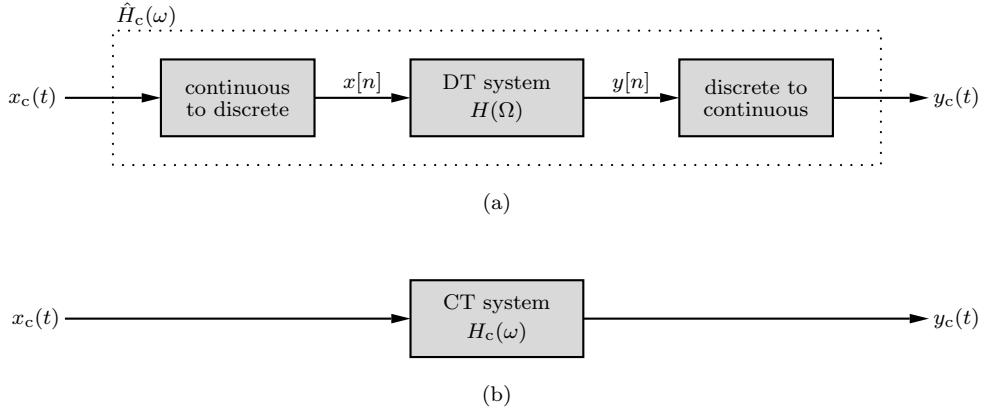


Figure 6.21: Realization of an analog filter using digital means.

our future discussions implicitly assume that signals to be processed are bandlimited and sampled at least at the Nyquist rate. Such an assumption is effectively achieved by placing at the input an anti-aliasing filter with cutoff frequency that is half the sampling rate.

6.5.1 A Mathematical Representation

We shall now characterize, in both the time and frequency domains, the input-output relationships of the ideal C/D and D/C converters shown in Fig. 6.21a. These interfaces may also be referred to as A/D and D/A converters, respectively, for the reasons explained earlier. Using these characterizations, we then develop a mathematical representation of the digital processing of analog signals depicted in Fig. 6.21.

Time-Domain Characterization of Ideal C/D and D/C Converters

A C/D converter samples a bandlimited input $x_c(t)$ at intervals of T seconds. Its output is the discrete-time signal $x[n]$ whose n th sample value is identical to the n th sample of $x_c(t)$, that is,

$$x[n] = x_c(nT).$$

This is the time-domain characterization (relating output to input) of an ideal C/D converter.

The discrete-time signal $x[n]$ acts as the input to the discrete-time filter $H(\Omega)$. The output $y[n]$ of $H(\Omega)$ is then converted by a D/C converter to a continuous-time signal $y_c(t)$, which, using the interpolation formula of Eq. (3.13), is given as

$$y_c(t) = \sum_{n=-\infty}^{\infty} y[n] \operatorname{sinc}\left(\frac{t - nT}{T}\right).$$

This is the time-domain characterization (relating output to input) of an ideal D/C converter. As shown in Ch. 3, the D/C converter functions as an ideal lowpass filter, and its output $y_c(t)$ is the bandlimited interpolation of $y[n]$ (see Sec. 6.4).

Frequency-Domain Characterization of Ideal C/D and D/C Converters

Let the Fourier transform of $x_c(t)$ be $X_c(\omega)$ and the DTFT of $x[n]$ be $X(\Omega)$. Because of our implicit assumption (bandlimited signals sampled above Nyquist), $x_c(t)$ is the bandlimited interpolation of $x[n] = x_c(nT)$, and Eq. (6.46) yields

$$X(\Omega) = \frac{1}{T} X_c\left(\frac{\Omega}{T}\right), \quad |\Omega| \leq \pi. \quad (6.50)$$

This is the frequency-domain characterization (relating output to input) of an ideal C/D converter.

As for the D/C case, because $y_c(t)$ is the bandlimited interpolation of $y[n]$, it follows that $y[n] = y_c(nT)$, and according to Eq. (6.46),

$$Y_c(\omega) = TY(\omega T), \quad |\omega T| \leq \pi.$$

Recognizing that $Y_c(\omega)$ is bandlimited to π/T , we obtain

$$Y_c(\omega) = \begin{cases} TY(\omega T) & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}. \quad (6.51)$$

This is the frequency-domain characterization (relating output to input) of an ideal D/C converter.

Relating $H_c(\omega)$, $\hat{H}_c(\omega)$, and $H(\Omega)$

The system in Fig. 6.21a acts as a bandlimited analog system $\hat{H}_c(\omega)$. Typically, our goal is for that system to equal, as closely as possible, some CT system $H_c(\omega)$, as shown in Fig. 6.21b. That is, we seek

$$\hat{H}_c(\omega) \approx H_c(\omega).$$

Since $\hat{H}_c(\omega)$ is inherently bandlimited, we know this approximation cannot result in equality whenever the desired CT system $H_c(\omega)$ is non-bandlimited.

We now relate the responses $\hat{H}_c(\omega)$ and $H(\Omega)$. From Fig. 6.21a, we know that

$$Y(\Omega) = X(\Omega)H(\Omega).$$

Substituting Eq. (6.50) and setting $\Omega = \omega T$, we obtain

$$Y(\omega T) = \frac{1}{T} X_c(\omega)H(\omega T).$$

Using Eq. (6.51), we next obtain

$$Y_c(\omega) = \begin{cases} X_c(\omega)H(\omega T) & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}.$$

Moreover, as seen from Fig. 6.21a, $Y_c(\omega) = X_c(\omega)\hat{H}_c(\omega)$. Hence,

$$\hat{H}_c(\omega) = \begin{cases} H(\omega T) & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}. \quad (6.52)$$

Conversely,

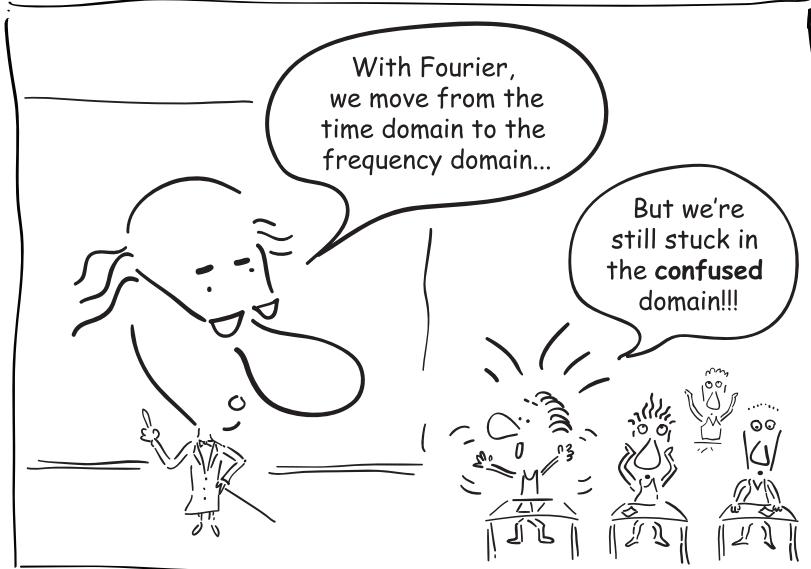
$$H(\Omega) = \hat{H}_c(\Omega/T), \quad |\Omega| \leq \pi. \quad (6.53)$$

These are quite satisfying results. Besides the frequency scaling implicit in $\Omega = \omega T$, the overall CT response of Fig. 6.21a identically matches the DT system response, at least for $|\Omega| = |\omega T| \leq \pi$. By further extension, if $\hat{H}_c(\omega)$ equals (or closely matches) some desired CT system $H_c(\omega)$, then, within a frequency scaling factor, the DT response $H(\Omega)$ also equals (or closely matches) the desired CT response, at least in the fundamental band.

At this point, we might wonder whether or not the restriction $|\omega T| \leq \pi$ means that digital processing is restricted to processing only low-frequency analog signals. Certainly not! Although the bandwidth over which the signals can be processed is restricted, by adequately increasing the sampling rate (reducing the sampling interval T), we can achieve any processing bandwidth we desire. We must remember, however, that Eqs. (6.52) and (6.53) are derived assuming that *the input analog signal is bandlimited and the sampling rate is no less than the Nyquist rate*.

What happens if the input $x_c(t)$ to the system in Fig. 6.21a is not a bandlimited signal or if the sampling rate is below the Nyquist rate? It suffers the same fate as that of any undersampled

signal. The sampling process inherently bandlimits a signal to $\omega = \pi/T$ rad/s ($F_s/2$ Hz) by folding its spectrum at $F_s/2$ Hz (half the sampling frequency).[†] Such folding distorts the spectrum because of overlapping spectral components (aliasing), as shown in Fig. 3.13. For example, if a signal of bandwidth 550 Hz is sampled at a rate 1000 Hz, then sampling bandlimits the signal to $F_s/2 = 500$ Hz by folding any components from 500 to 550 Hz into the 450- to 500-Hz range. Instead of the original spectrum, it is this distorted spectrum, bandlimited to 500 Hz (π/T rad/s), that is now processed by the digital filter in Fig. 6.21a.



The time domain, the frequency domain, and the confused domain.

6.5.2 Time-Domain Criterion: The Impulse Invariance Method

Substituting a desired CT response $H_c(\cdot)$ for $\hat{H}_c(\cdot)$ in Eq. (6.53) yields

$$H(\Omega) = H_c(\Omega/T), \quad |\Omega| \leq \pi. \quad (6.54)$$

This is a *frequency-domain criterion* for designing a digital system $H(\Omega)$ (or composite system $\hat{H}_c(\omega)$) to imitate a CT system $H_c(\omega)$. Let us translate this result to the time domain and obtain an equivalent design criterion in terms of impulse response functions.

Since $\hat{H}_c(\omega)$ is necessarily bandlimited, equivalence between $\hat{H}_c(\omega)$ and $H_c(\omega)$ is possible only for bandlimited $H_c(\omega)$. Thus, we proceed assuming that the desired system $H_c(\omega)$ is bandlimited, in which case the system impulse response $h_c(t)$ is given by

$$h_c(t) = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} H_c(\omega) e^{j\omega t} d\omega.$$

If the system in Fig. 6.21b is equivalent to the system in Fig. 6.21a, then $H_c(\omega) = \hat{H}_c(\omega) = H(\omega T)$ over the band $|\omega| \leq \pi/T$, and the preceding equation becomes

$$h_c(t) = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} H(\omega T) e^{j\omega t} d\omega.$$

[†]It may have multiple foldings about $F_s/2$ and 0, as explained in Ch. 3.

The n th sample of $h_c(t)$ is $h_c(nT)$, which can be found by setting $t = nT$ in this equation. In addition, we change the variable of integration according to $\Omega = \omega T$ to obtain

$$h_c(nT) = \frac{1}{2\pi T} \int_{-\pi}^{\pi} H(\Omega) e^{j\Omega n} d\Omega = \frac{1}{T} h[n],$$

where $h[n]$ is the unit impulse response of the digital system $H(\Omega)$. Therefore, we can design the impulse response of our digital system using the impulse response of our desired CT system according to

$$h[n] = Th_c(nT). \quad (6.55)$$

This strategy, known as the *impulse invariance method*, is the time-domain equivalent of Eq. (6.54). We stress again that *this relationship is effective only if $H_c(\omega)$ is bandlimited and the sampling interval T is no greater than the Nyquist interval*. The impulse invariance method is not an appropriate design technique to implement CT systems that are not at least approximately bandlimited, such as true highpass and bandstop filters.

Comments

Based on the preceding results, we make the following series of observations:

- When we let $h[n] = Th_c(nT)$, the spectrum $H(\Omega)$ is identical to the spectrum of the sampled continuous-time signal $Th_c(nT)$ (or $Th_c(t)\delta(t)$), except for a frequency-scaling factor T . Since sampling in the time domain results in a periodic replication in the frequency domain,

$$H(\Omega) = \sum_{k=-\infty}^{\infty} H_c\left(\frac{\Omega - 2\pi k}{T}\right).$$

This is just like Eq. (6.44) and can also be seen from Figs. 6.17d and 6.17f.

- Combining the first observation with Eq. (6.52), the CT frequency response $\hat{H}_c(\omega)$ of the system in Fig. 6.21a is the same as the spectrum $H_c(\omega)$ repeating periodically with period F_s Hz (as shown in Fig. 6.17d) and then lowpass filtered with cutoff frequency $F_s/2$ Hz.
- If $H_c(\omega)$ is bandlimited to frequency $\leq F_s/2$, then the frequency response $\hat{H}_c(\omega)$ exactly equals $H_c(\omega)$, as desired.

What happens if $H_c(\omega)$ is not bandlimited to $F_s/2$ Hz? This is a question of great interest because all practical $H_c(\omega)$ are non-bandlimited. Although we shall leave the details of this topic for a later chapter, we can readily appreciate a qualitative answer: if $H_c(\omega)$ is not bandlimited, then the samples $Th_c(nT)$ result in aliasing, which, in turn, distorts the realized frequency response $\hat{H}_c(\omega)$.

▷ Example 6.20 (Bandlimited Analog Delay)

Using the minimum possible sampling frequency, design a DT system for Fig. 6.21a that delays a bandlimited ($B \leq 10$ kHz) CT input $x_c(t)$ by $\tau = 250 \mu\text{s}$. Sketch the frequency responses $H(\Omega)$ and $\hat{H}_c(\omega)$. Explain how closely the system compares with an ideal analog delay $\tau = 250 \mu\text{s}$. Using a hypothetical bandlimited ($B \leq 10$ kHz) signal $x_c(t)$ and its spectrum $X_c(\omega)$, sketch the signals and spectra at the input of the C/D converter, the input of $H(\Omega)$, the output of $H(\Omega)$, and the output of the D/C converter.

Because the input signal bandwidth $B \leq 10$ kHz, Nyquist requires a sampling interval $T \leq 1/2B = 50 \mu\text{s}$. To achieve a DT system with minimum sampling frequency, we choose $T = 50 \mu\text{s}$, which corresponds to a sampling frequency of $F_s = 20$ kHz. The required delay of $250 \mu\text{s}$ is thus $\tau = 5T$. As seen from Eq. (2.6), this delay is produced by a CT system with frequency response

$$H_c(\omega) = e^{-j\omega\tau} = e^{-j5\omega T}.$$

According to Eq. (6.54), we obtain the DT system frequency response as

$$H(\Omega) = H_c(\Omega/T) = e^{-j5\Omega}, \quad |\Omega| \leq \pi. \quad (6.56)$$

Further, following Ex. 6.19, the DT system impulse response is

$$h[n] = \text{sinc}(n - 5) = \delta[n - 5].$$

Thus, our DT system for $\tau = 5T$ is just a simple DT delay. Such a system is causal and extremely simple to implement.

Although Eq. (6.56) describes $H(\Omega)$ over the fundamental band only, it is a complete expression because $e^{-j5\Omega}$ happens to be 2π -periodic (the fundamental period is $2\pi/5$). The magnitude response and the phase response are thus

$$|H(\Omega)| = 1 \quad \text{and} \quad \angle H(\Omega) = -5\Omega.$$

The magnitude response is constant, and the phase response is a linear function of Ω with slope -5 , as shown in Fig. 6.22a.

Using this digital delay $H(\Omega) = e^{-j5\Omega}$, the system in Fig. 6.21a acts as a bandlimited analog delay with frequency response (Eq. (6.52))

$$\hat{H}_c(\omega) = \begin{cases} e^{-j5\omega T} & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}.$$

Substituting $T = 50 \times 10^{-6}$, we obtain, for $|\omega| \leq 20000\pi$,

$$\hat{H}_c(\omega) = e^{-j\omega/4000}, \quad |\hat{H}_c(\omega)| = 1, \quad \text{and} \quad \angle \hat{H}_c(\omega) = -\omega/4000.$$

Over the range $-20000\pi < \omega \leq 20000\pi$, the magnitude response is constant, and the phase response is a linear function of ω with slope $-1/4000$, as shown in Fig. 6.22b. The frequency response is zero beyond this range. Thus, $\hat{H}_c(\omega)$ functions as an ideal bandlimited analog delay of $250 \mu\text{s}$. Over the bandwidth of interest, there is no difference between this response and an ideal analog delay of $250 \mu\text{s}$.

To understand how a signal progresses through this system (Fig. 6.21a), let us consider a hypothetical input signal $x_c(t)$ and its 10-kHz bandlimited spectrum $X_c(\omega)$, as shown in Fig. 6.22c. In the interest of simplicity and clarity, we have set the phase spectrum to 0. A nonzero phase function is accommodated by simply adding it to the resulting phase spectrum at each stage. The signal at the output of the C/D converter is a discrete-time signal $x[n] = x_c(nT)$ with periodic spectrum $X(\Omega)$, as shown in Fig. 6.22d. Since $X_c(\omega)$ is bandlimited, the DTFT of $x[n]$ is, according to Eq. (6.50),

$$X(\Omega) = \frac{1}{T} X_c(\Omega/T), \quad |\Omega| \leq \pi.$$

This spectrum is identical to $X_c(\omega)/T$, frequency-compressed by factor $1/T = 20000$ to ensure that the original bandwidth reduces to a width π (no aliasing).

Using Eq. (6.56), the DT filter output spectrum is

$$Y(\Omega) = X(\Omega) H(\Omega) = X(\Omega) e^{-j5\Omega}.$$

Applying the time-shifting property of Eq. (6.16), we find

$$y[n] = x[n - 5].$$

Both $y[n]$ and $Y(\Omega)$ are depicted in Fig. 6.22e (magnitude response solid, phase response dotted).

To find the continuous-time output $y_c(t)$, we first obtain $Y_c(\omega)$ from Eq. (6.51) as

$$Y_c(\omega) = \begin{cases} TY(\omega T) = X_c(\omega) e^{-j5\omega T} & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}.$$

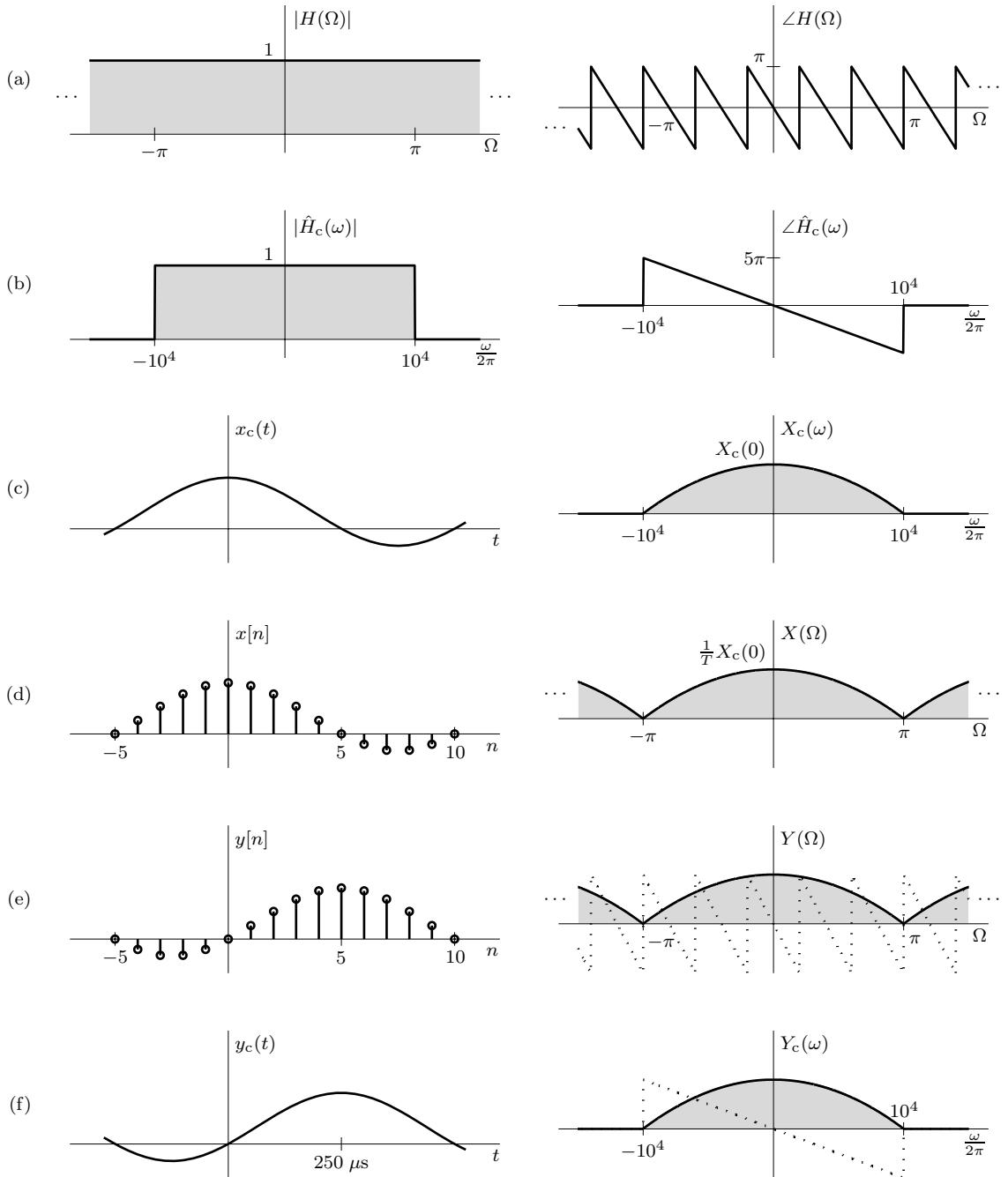


Figure 6.22: Digital realization of an ideal bandlimited analog delay.

Since $X_c(\omega)$ is bandlimited to $|\omega T| \leq \pi$, this reduces to

$$Y_c(\omega) = X_c(\omega) e^{-j5\omega T}.$$

An alternate and perhaps easier way to find $Y_c(\omega)$ is from the fact that the overall system transfer function, as found from Eq. (6.52), is

$$\hat{H}_c(\omega) = \begin{cases} H(\omega T) = e^{-j5\omega T} & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}.$$

Using $Y_c(\omega) = H_c(\omega)X_c(\omega)$, we again obtain $Y_c(\omega) = X_c(\omega)e^{-j5\omega T}$. In either case,

$$y_c(t) = x_c(t - 5T) = x_c(t - \tau), \quad \tau = 250 \mu\text{s}.$$

Both $y_c(t)$ and $Y_c(\omega)$ are shown in Fig. 6.22f (magnitude response solid, phase response dotted).

Example 6.20 

▷ **Example 6.21 (Bandlimited Differentiator)**

Using digital processing of analog signals, as shown in Fig. 6.21a, design an ideal differentiator for bandlimited input signals.

To solve this problem, we need to design an appropriate DT system for Fig. 6.21a. The desired frequency response of an ideal bandlimited differentiator is

$$H_c(\omega) = \begin{cases} j\omega & |\omega T| \leq \pi \\ 0 & \text{otherwise} \end{cases}. \quad (6.57)$$

By selecting a proper value for T , we can accommodate input signals of any bandwidth. Since the frequency range of interest is bandlimited, the DT system in Fig. 6.21a can, in theory, exactly meet the desired response. Using Eq. (6.54), we obtain

$$H(\Omega) = H_c(\Omega/T) = \frac{j\Omega}{T}, \quad |\Omega| \leq \pi. \quad (6.58)$$

Thus,

$$|H(\Omega)| = \left| \frac{\Omega}{T} \right| \quad \text{and} \quad \angle H(\Omega) = \frac{\pi}{2} \text{sgn}(\Omega), \quad |\Omega| \leq \pi.$$

The magnitude and phase responses are shown in Figs. 6.23a and 6.23b, respectively.

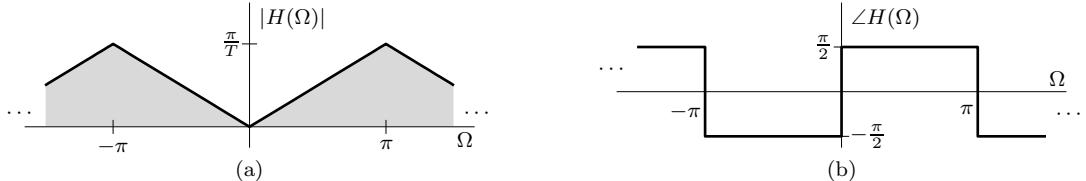


Figure 6.23: Digital realization of an ideal bandlimited differentiator: (a) $|H(\Omega)|$ and (b) $\angle H(\Omega)$.

We can also solve this problem in the time domain using the impulse invariance method of Eq. (6.55). The impulse response $h_c(t)$ of the analog filter in Eq. (6.57) is the inverse Fourier transform of $H_c(\omega)$, given by

$$h_c(t) = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} j\omega e^{j\omega t} d\omega.$$

Following some work, Eq. (6.55) yields

$$h[n] = Th_c(nT) = \frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} j\omega e^{j\omega nT} d\omega = \begin{cases} \frac{\cos(\pi n)}{nT} & n \neq 0 \\ 0 & n = 0 \end{cases}.$$

This result is also the IDTFT of Eq. (6.58), as expected. Notice, however, that this $h[n]$ is noncausal, which makes the system, as designed, impractical to implement. Using some suitable delay n_g , the shift and truncation method of Eq. (6.42) yields a causal (realizable) solution $\hat{h}[n]$ that will only approximate ideal bandlimited differentiator behavior.

Example 6.21 \triangleleft

▷ **Example 6.22 (Lowpass Filter Using Sub-Nyquist Sampling)**

Using Fig. 6.21a and the minimum possible sampling frequency, design an ideal lowpass DT filter with 10-kHz cutoff frequency, assuming the CT input $x_c(t)$ is bandlimited to 40 kHz.

Given our input bandwidth of 40 kHz, Nyquist requires that our system sample at 80 kHz to avoid aliasing. However, our desired DT system lowpass filters the sampled input with a cutoff frequency of 10 kHz, effectively removing all signal content from 10 to 40 kHz. There is really no reason to perfectly preserve the components from 10 to 40 kHz only to see that content removed by the DT filter operation. Rather, let us sample at a lower rate that preserves the desired lower-frequency content (below 10 kHz) but for which some high-frequency distortion (aliasing) may occur. Such distortion is irrelevant since the DT lowpass filter removes those signal components.

To determine a suitable sampling frequency, we notice that we can tolerate folding the highest-frequency content at 40 kHz down to 10 kHz, the upper frequency of our desired signal. This implies a folding frequency of $(40 + 10)/2 = 25$ kHz and a sampling frequency of $F_s = 50$ kHz. Although this sampling frequency is sub-Nyquist for our input signal (50 kHz < 80 kHz), we encounter no detrimental effects since our desired frequency band remains intact, and aliased content is filtered away. An immediate advantage of this approach is that our sub-Nyquist system operates over one-third slower than the Nyquist system. Slower systems consume less power and cost less than their high-speed counterparts.

The desired lowpass filter cutoff frequency is $f_1 = 10$ kHz or $\omega_1 = 20000\pi$. Since $F_s = 1/T = 50$ kHz, $\omega_1 T = 2\pi/5$. The frequency response $H_c(\omega)$ for an ideal (zero delay) lowpass filter of cutoff frequency ω_1 is

$$H_c(\omega) = \Pi\left(\frac{\omega}{2\omega_1}\right).$$

According to Eq. (6.54),

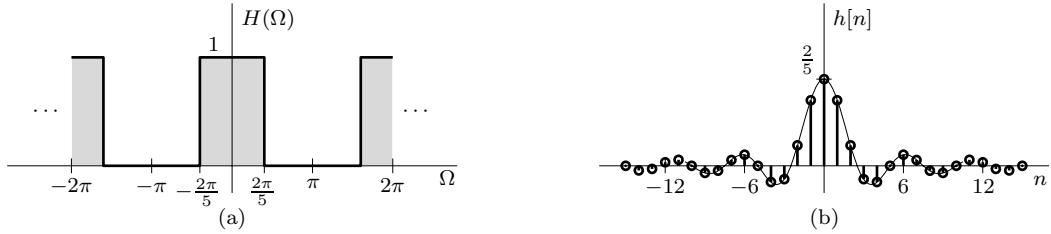
$$H(\Omega) = \Pi\left(\frac{\Omega}{2\omega_1 T}\right) = \Pi\left(\frac{\Omega}{4\pi/5}\right), \quad |\Omega| \leq \pi.$$

This frequency response, shown in Fig. 6.24a, is the frequency-domain description of an ideal lowpass DT filter with cutoff frequency $\Omega_1 = 2\pi/5$. Had we used Nyquist sampling ($F_s = 80$ kHz), the resulting DT filter would have had a smaller cutoff frequency $\Omega_1 = \omega_1/80000 = \pi/4$. As we shall see in Ch. 8, DT filters can become progressively more difficult to implement as their passbands narrow. In this respect, we once again see that our sub-Nyquist solution is superior to the Nyquist solution.

To obtain the time-domain description of our DT system, we use pair 8 of Table 6.2 to obtain

$$h[n] = \frac{2}{5} \text{sinc}(2n/5).$$

This impulse response, shown in Fig. 6.24b, is also obtained by taking the inverse Fourier transform of $H_c(\omega)$ and then applying the impulse-invariance criterion of Eq. (6.55). In any case, $h[n]$ is non-causal and of infinite duration, so a practical realization would require us to appropriately truncate and shift $h[n]$.

Figure 6.24: Digital lowpass filter for sub-Nyquist system: (a) $H(\Omega)$ and (b) $h[n]$.

Example 6.22 ◇

▷ Drill 6.9 (Another Bandlimited Analog Delay)

Discuss what changes occur in Ex. 6.20 if the desired delay is increased to $275 \mu\text{s}$.

◇

6.6 Digital Resampling: A Frequency-Domain Perspective

Sections 4.1.3 and 4.6 briefly discuss the operations of downsampling, decimation, upsampling, and interpolation. To promote an intuitive understanding of these topics, we now investigate these operations from a frequency-domain perspective. Before embarking on this frequency-domain journey, let us quickly review each of these four operations, which Fig. 6.25 depicts in block form.

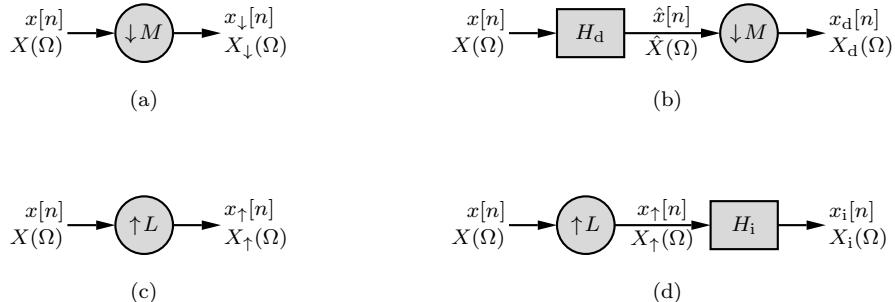


Figure 6.25: Block representations of DT time-scaling operations: (a) downsampling, (b) decimation, (c) upsampling, and (d) interpolation.

Compressing a DT signal $x[n]$ by integer factor M (Fig. 6.25a) is represented mathematically as

$$x_{\downarrow}[n] = x[Mn]. \quad (6.59)$$

When $n = 0$, $x_{\downarrow}[n] = x[0]$; when $n = 1$, $x_{\downarrow}[n] = x[M]$; when $n = 2$, $x_{\downarrow}[n] = x[2M]$; and so on. This means that $x_{\downarrow}[n]$ selects every M th sample of $x[n]$ and deletes the rest of them. If $x[n]$ has sampling frequency F_s , then the downsampled signal $x_{\downarrow}[n]$ has reduced sampling frequency F_s/M . Downsampling by factor M is easily accomplished: retain every M th sample of $x[n]$, and discard the remaining samples to obtain $x_{\downarrow}[n]$.

The downsampling operation may or may not result in loss of information depending on the nature of $x[n]$. For example, if $x[n]$ is obtained by oversampling some continuous-time signal, downsampling $x[n]$ may not cause loss of information. In other cases, aliasing during downsampling causes

a loss or distortion of information. To minimize the adverse effects of aliasing, downsampling is often preceded by a decimation (anti-aliasing) filter H_d . A decimation filter followed by downsampling results in decimation (Fig. 6.25b). The decimation $x_d[n]$ of signal $x[n]$ clearly depends on the scale factor M as well as the nature of the decimation filter H_d .

Expanding a DT signal $x[n]$ by integer factor L (Fig. 6.25c) is represented mathematically as

$$x_{\uparrow}[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - Lk] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{otherwise} \end{cases}. \quad (6.60)$$

Clearly, $x_{\uparrow}[0] = x[0]$, $x_{\uparrow}[L] = x[1]$, $x_{\uparrow}[2L] = x[2]$, $x_{\uparrow}[3L] = x[3]$, and so on. Moreover, $x_{\uparrow}[n] = 0$ for all $n \neq kL$ (k integer). Thus, expansion amounts to interleaving $L - 1$ zeros between successive samples of $x[n]$. In this way, upsampling increases the effective sampling rate by a factor L .

Unlike downsampling, the upsampling operation never results in a loss of information. The inserted zeros, however, serve as a poor representation of the underlying signal. Thus, upsampling is often followed by an interpolation filter H_i , a process referred to simply as *interpolation* (Fig. 6.25d). The interpolation $x_i[n]$ of signal $x[n]$ clearly depends on the scale factor L as well as the nature of the filter H_i . As we shall see, ideal interpolation preserves the spectral shape of the original signal, which is to say that the spectrum $X_i(\Omega)$ should be a frequency-scaled version of $X(\Omega)$.

Why Resample?

In practice, resampling is often necessary. We find one application of resampling in Ch. 3 in connection with D/A conversion: increasing the sampling rate before samples pass through a D/A converter simplifies the design and implementation of the reconstruction filter (see page 201). Resampling is necessary when input and output processes are naturally at different rates. For example, compact disc (CD) and digital audio tape (DAT) sampling rates are 44.1 and 48 kHz, respectively. If we wish to transfer a CD signal to DAT, we need resampling. Resampling is frequently used with portable MP3 players to balance sound quality with song size. In telecommunications, we need conversion between different signal code formats, such as conversion from delta modulation to pulse-code modulation, which operate at different sampling rates. In telecommunication systems, we also receive and transmit different types of signals, such as teletype, facsimile, speech, video, etc., at different rates. Resampling is also used to minimize computations by reducing the sampling rate when the signal bandwidth has been reduced because of lowpass filtering. This is often done with signals that have unused bandwidth. In medical image processing, digital resampling is necessary for image enhancement, image scale change, and image rotation [2].

6.6.1 Using Bandlimited Interpolation to Understand Resampling

To develop a frequency-domain perspective of digital resampling, it proves very useful to use $x_c(t)$, which we take as the bandlimited interpolation of $x[n]$, as an intermediary.[†] Although direct methods exist to derive the desired results, $x_c(t)$ provides a more intuitive understanding of the results.

Recall that the samples of $x_c(t)$ at T -second intervals represent $x[n]$, that is,

$$x[n] = x_c(nT).$$

Although the choice of parameter T is arbitrary in this development, our investigation of resampling is greatly simplified if we choose $T = 1$. Thus, the bandwidth of $x_c(t)$, the bandlimited interpolation of $x[n]$, is $B \leq \frac{2\pi}{2T} = \pi$ rad/s or 0.5 Hz. Since $\Omega = \omega T$, the choice $T = 1$ also allows us to conveniently interchange ω and Ω as circumstances warrant. Using $T = 1$, we now have

$$x[n] = x_c(n).$$

[†]Actually, it is not necessary that $x_c(t)$ be the bandlimited interpolation of $x[n]$. All we need is that $x_c(nT) = x[n]$. The bandlimited interpolation of $x[n]$ is a particularly convenient choice because its spectrum is just a frequency-scaled version of $X(\Omega)$ over the fundamental band.

Figures 6.26a and 6.26c depict $x[n]$ and the corresponding bandlimited interpolation $x_c(t)$ for $T = 1$, respectively. Interestingly, when $T = 1$, we recognize from Eq. (6.44) that

$$X(\Omega) = \sum_{k=-\infty}^{\infty} X_c(\Omega - 2\pi k).$$

Since $x_c(t)$ is the bandlimited interpolation of $x[n]$, this further simplifies (Eq. (6.46)) to

$$X(\Omega) = X_c(\Omega), \quad |\Omega| \leq \pi.$$

Clearly, the choice $T = 1$ forces $X(\Omega)$ to be identical to $X_c(\omega)$ in the fundamental band $|\Omega| = |\omega| \leq \pi$, as shown in Figs. 6.26b and 6.26d. Outside the fundamental band, the 2π -periodic DTFT spectrum $X(\Omega)$ differs from the bandlimited CTFT spectrum $X_c(\omega)$, which is zero for $|\omega| > \pi$. Let us now use our CT bandlimited interpolation of $x[n]$ to investigate both digital downsampling and interpolation.

As per Sec. 1.2.2, time compressing a CT signal $x_c(t)$ by factor M yields $x_c(Mt)$, as shown in Fig. 6.26e for $M = 2$. The signal $x_c(Mt)$, when sampled using the sampling interval $T = 1$, yields the downsampled signal $x_c(Mn) = x[Mn] = x_\downarrow[n]$, as shown in Fig. 6.26g for $M = 2$. Similarly, time expanding $x_c(t)$ by factor L yields $x_c(t/L)$, as shown in Fig. 6.26i for $L = 2$. The signal $x_c(t/L)$, when sampled using the sampling interval $T = 1$, yields the interpolated signal $x_c(n/L) = x_i[n]$, as shown in Fig. 6.26k for $L = 2$.[†] This signal has L times as many samples as those of $x[n]$: every L th sample is a value from $x[n]$, and the remaining samples are interpolated. With these time-domain preliminaries in place, we now turn our attention to the frequency domain.

Finding $X_\downarrow(\Omega)$ and $X_i(\Omega)$ from $X_c(\Omega)$

By setting the sampling interval to $T = 1$, any discrete-time signal and its corresponding bandlimited interpolation have identical spectra in the fundamental band $|\Omega| = |\omega| \leq \pi$. Hence,

$$X(\Omega) = X_c(\Omega), \quad |\Omega| \leq \pi. \quad (6.61)$$

To find the spectrum of $x_\downarrow[n]$, let us consider the spectrum of the closely related signal $x_c(Mt)$. According to the scaling property of Eq. (1.85),

$$x_c(Mt) \iff \frac{1}{M} X_c\left(\frac{\omega}{M}\right). \quad (6.62)$$

The spectrum $X_c(\omega/M)$ is the spectrum $X_c(\omega)$ expanded by factor M . Hence, although the bandwidth of $X_c(\omega)$ is $\leq \pi$, the bandwidth of $x_c(Mt)$ can be greater than π , causing aliasing and folding in $X_\downarrow(\Omega)$. Consequently, downsampling can result in a loss of information. If, however, M is small enough to ensure that the spectrum $X_c(\omega/M)$ does not spill beyond π , the spectral shape is preserved, and downsampling causes no loss of information. Using $X_c(\omega)$ from Fig. 6.26d, we sketch $\frac{1}{M} X_c(\omega/M)$ in Fig. 6.26f for $M = 2$. As shown in Fig. 6.26h, the spectrum $X_\downarrow(\Omega)$ of the sampled signal $x_\downarrow[n] = x_c(Mn)$ is just a 2π -periodic replication of the spectrum $\frac{1}{M} X_c(\omega/M)$.[‡]

In a similar way, we have from Eq. (1.85)

$$x_c(t/L) \iff L X_c(L\omega). \quad (6.63)$$

The spectrum $X_c(L\omega)$ is the spectrum $X_c(\omega)$ compressed by factor L . Because the bandwidth of $X_c(\omega)$ is $B \leq \pi$, the bandwidth of $X_c(L\omega)$ is always below π . Moreover, since $x_c(t)$ is the

[†]Since $x_c(Mt)$ yields the downsampled signal $x_\downarrow[n]$, it is natural to wonder why $x_c(t/L)$ yields the interpolated signal $x_i[n]$ rather than the upsampled signal $x_\uparrow[n]$. The reason is that the ideal lowpass filter implicit in the generation of the bandlimited interpolation $x_c(t)$ also functions as the ideal lowpass filter needed for DT interpolation.

[‡]In Fig. 6.26f, we have shown the spectrum $X_c(\omega/M)$ to be bandlimited to π , although this restriction is not necessary. When the spectrum $X_c(\omega/M)$ is not bandlimited to π , Fig. 6.26h consists of overlapping repetitions of the spectrum $\frac{1}{M} X_c(\Omega/M)$. In such a case, $x_c(Mt)$ is not a bandlimited interpolation of $x_\downarrow[n]$, which indicates that aliasing distortion occurs during the downsampling process.

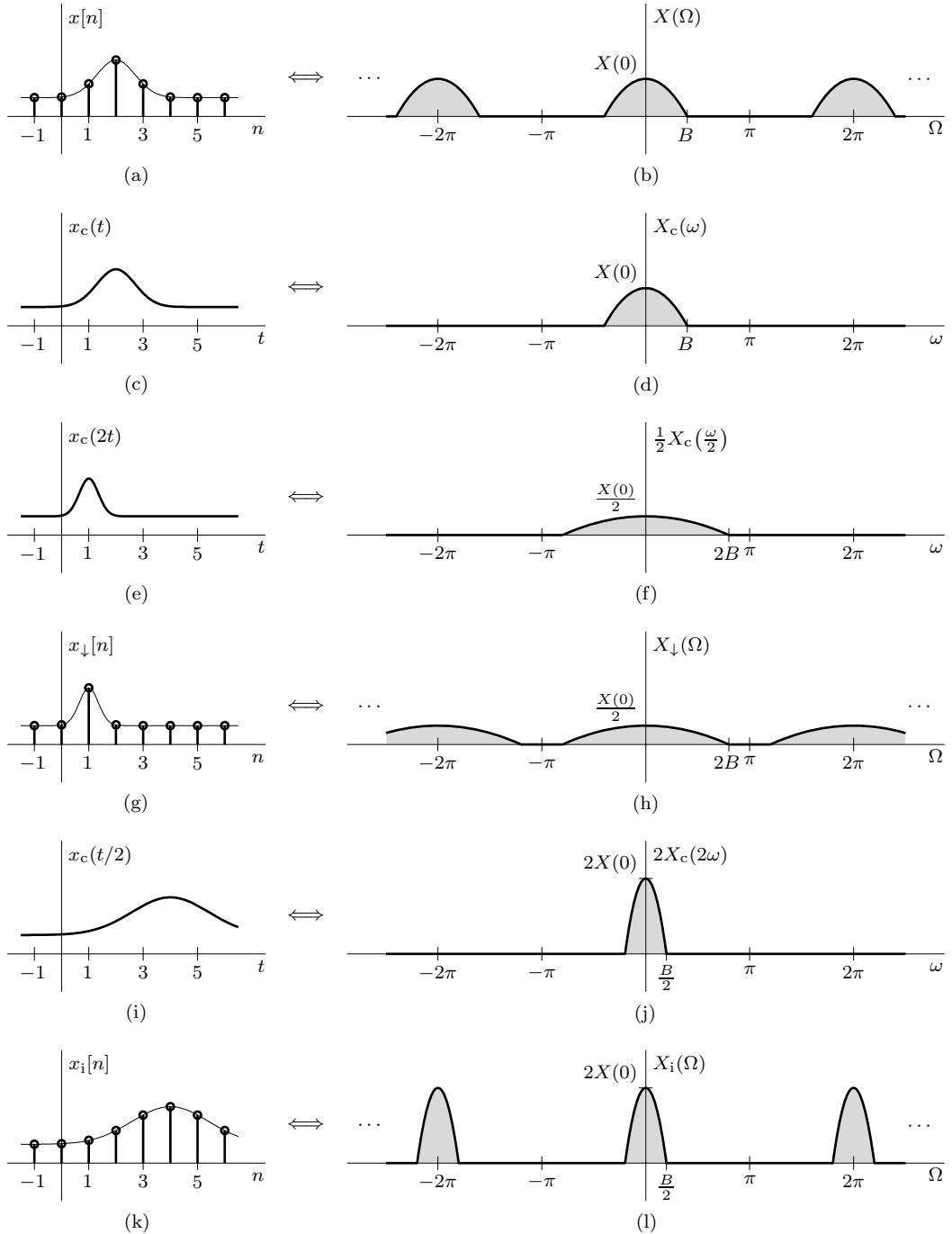


Figure 6.26: Signals $x[n]$, $x_{\downarrow}[n]$, and $x_i[n]$ and their spectra obtained from the corresponding bandlimited interpolations $x_c(t)$, $x_c(2t)$, and $x_c(t/2)$ and their spectra.

(ideal) bandlimited CT interpolation of $x[n]$, the signal $x_i[n] = x_c(n/L)$ is the ideal (factor- L) DT interpolation of $x[n]$. That is, the spectral shape of the interpolated signal is preserved, as required for optimum (ideal) interpolation. Using $X_c(\omega)$ from Fig. 6.26d, we sketch $LX_c(L\omega)$ in Fig. 6.26j

for $L = 2$. As shown in Fig. 6.26l, the spectrum $X_i(\Omega)$ of the sampled signal $x_i[n] = x_c(n/L)$ is just a 2π -periodic replication of the spectrum $LX_c(L\omega)$.

The next problem is to find expressions for the 2π -periodic signals $X_{\downarrow}(\Omega)$ and $X_i(\Omega)$ in Figs. 6.26h and 6.26l. To shift a spectrum by 2π , we replace Ω with $\Omega - 2\pi$. Hence, to produce a 2π -periodic repetition, we replace Ω with $\Omega - 2\pi k$ and sum over all (integer) k . Thus, with an eye on Fig. 6.26 and Eqs. (6.61), (6.62), and (6.63), we express $X(\Omega)$, $X_{\downarrow}(\Omega)$, and $X_i(\Omega)$ in terms of $X_c(\Omega)$ as

$$X(\Omega) = \sum_{k=-\infty}^{\infty} X_c(\Omega - 2\pi k), \quad (6.64)$$

$$X_{\downarrow}(\Omega) = \frac{1}{M} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\Omega - 2\pi k}{M}\right), \quad (6.65)$$

$$\text{and } X_i(\Omega) = L \sum_{k=-\infty}^{\infty} X_c(L[\Omega - 2\pi k]). \quad (6.66)$$

Equations (6.65) and (6.66) represent $X_{\downarrow}(\Omega)$ and $X_i(\Omega)$ in terms of $X_c(\Omega)$. We really want to express $X_{\downarrow}(\Omega)$ and $X_i(\Omega)$ in terms of $X(\Omega)$, not $X_c(\Omega)$. Additionally, we hope to obtain frequency-domain expressions for decimation and upsampling. This is what we shall now do.

6.6.2 Downsampling and Decimation

To express $X_{\downarrow}(\Omega)$ (Fig. 6.26h) in terms of $X(\Omega)$ (Fig. 6.26b), we observe that the spectrum $X_{\downarrow}(\Omega)$ appears somewhat like frequency-expanded version of $X(\Omega)$. The frequency-expanded version of $X(\Omega)$ is $X(\Omega/M)$. Let us closely examine $\frac{1}{M}X(\Omega/M)$ and see how it compares to the spectrum in $X_{\downarrow}(\Omega)$ in Fig. 6.26h. From Eq. (6.64), we find $\frac{1}{M}X(\Omega/M)$ as

$$\frac{1}{M}X\left(\frac{\Omega}{M}\right) = \frac{1}{M} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\Omega}{M} - 2\pi k\right) = \frac{1}{M} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\Omega - 2\pi Mk}{M}\right). \quad (6.67)$$

The right-hand side of this equation is almost identical to $X_{\downarrow}(\Omega)$ in Eq. (6.65), except that the repetition period is $2\pi M$ instead of 2π . The frequency expansion of the spectrum not only expands the spectrum in each cycle but also expands the period by the factor M , as depicted in Fig. 6.27 using $M = 2$ and $X(\Omega)$ from Fig. 6.26b. Can we relate this spectrum to the spectrum $X_{\downarrow}(\Omega)$ shown in Fig. 6.26h? Very simple. In Fig. 6.27, if we just add $\frac{1}{2}X(\Omega/2)$ (shown solid) and a 2π -shifted version of the same spectrum (shown dotted), then we get the Fig. 6.26h spectrum $X_{\downarrow}(\Omega)$.

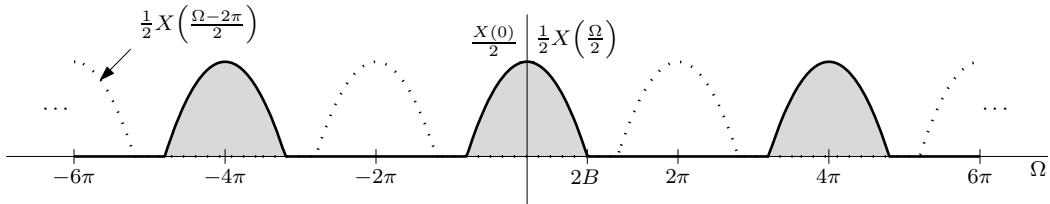


Figure 6.27: Obtaining $X_{\downarrow}(\Omega)$ from $X(\Omega)$ for $M = 2$.

For the $M = 2$ case, we need to add only one shifted version to the original. For a general value of M , the spectrum $\frac{1}{M}X(\Omega/M)$ has period $2\pi M$. To fill in the empty band between cycles, we need

to add $\frac{1}{M}X(\Omega/M)$ shifted by $0, 2\pi, 4\pi, 6\pi, \dots, 2(M-1)\pi$ to obtain the desired $X_{\downarrow}(\Omega)$. That is,

$$\begin{aligned} X_{\downarrow}(\Omega) &= \frac{1}{M} \left\{ X\left(\frac{\Omega}{M}\right) + X\left(\frac{\Omega - 2\pi}{M}\right) + \cdots + X\left(\frac{\Omega - 2(M-1)\pi}{M}\right) \right\} \\ &= \frac{1}{M} \sum_{m=0}^{M-1} X\left(\frac{\Omega - 2\pi m}{M}\right). \end{aligned} \quad (6.68)$$

This is the expression we desire, where $X_{\downarrow}(\Omega)$ is expressed directly in terms of $X(\Omega)$.

Denoting the bandwidth of $x[n]$ as B , then, from Fig. 6.26h, it is clear that the bandwidth of the downsampled signal $x_{\downarrow}[n]$ is MB . To avoid aliasing (spectral overlap) due to downsampling, the bandwidth $MB \leq \pi$ or, equivalently,

$$B \leq \frac{\pi}{M}. \quad (6.69)$$

To avoid loss of data, downsampling of $x[n]$ by factor M should be performed only if $x[n]$ satisfies Eq. (6.69) or has sufficient built-in redundancy.

If, for whatever reason, we have to downsample when Eq. (6.69) is not satisfied, then a decimation (lowpass anti-aliasing) filter should be used to eliminate all spectral components of $x[n]$ beyond frequency π/M to obtain its approximation $\hat{x}[n]$ that is bandlimited to π/M . Referring to Fig. 6.25b, remember that a decimation filter must come *before* the downsampling operation. Letting $\Omega_1 = \pi/M$ in Fig. 6.14a, the frequency response of an ideal lowpass decimation filter is

$$H_d(\Omega) = \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{2\pi/M}\right).$$

A decimation filter eliminates all signal components above the frequency π/M , which saves spectral components below π/M from being corrupted by the folding of components above π/M . Still, when components above π/M are removed, $\hat{x}[n] \neq x[n]$ and $x_d[n] \neq x[Mn]$. That is, decimation is not the same as simple downsampling unless $x[n]$ is bandlimited to π/M rad/sample, which ensures that $\hat{x}[n] = x[n]$. Since $x_{\downarrow}[n] = x[Mn]$, a downampler can eliminate, but never change, its input samples. A decimator, on the other hand, not only eliminates samples but can also change them. Superficially, it might therefore seem that downsampling causes less distortion than decimation. As we shall very soon see, however, this conclusion is entirely false.

Combining Eq. (6.68) with $\hat{X}(\Omega) = X(\Omega)H_d(\Omega)$ and Fig. 6.25, we can express the decimator output in terms of its input as

$$X_d(\Omega) = \frac{1}{M} \sum_{m=0}^{M-1} \hat{X}\left(\frac{\Omega - 2\pi m}{M}\right) = \frac{1}{M} \sum_{m=0}^{M-1} X\left(\frac{\Omega - 2\pi m}{M}\right) H_d\left(\frac{\Omega - 2\pi m}{M}\right). \quad (6.70)$$

Downsampling Foolishness

There is an old adage that it is better to remain silent and be thought a fool than to open your mouth and remove all doubt. A downampler speaks its mind, even when it cannot know the answer, and looks the fool in the process. A decimator, on the other hand, is more guarded in its responses: it answers truthfully when it can but otherwise remains silent. In most cases, it is wiser and safer to use a decimator than a downampler. To illustrate, consider the signal $x[n] = \cos(\pi n)$, which varies between 1 and -1. Downsampling by a factor $M = 2$ yields the constant $x_{\downarrow}[n] = 1$. This certainly seems to be a foolish result, since we expect compression to produce a signal of higher, not lower, frequency. Of course, this odd behavior is a consequence of the aliasing implicit in the downsampling operation. A decimator, on the other hand, whose anti-aliasing filter completely removes $\cos(\pi n)$, outputs $x_d[n] = 0$. Rather than produce a nonsensical output, the decimator is wisely silent in its response.

The frequency-domain perspectives of Eqs. (6.68) and (6.70) provide even more insight into the distortion in downsampling and decimation when the condition of Eq. (6.69) is not satisfied. Figure 6.28a shows $X(\Omega)$, the spectrum of some signal $x[n]$. Because Eq. (6.69) is not satisfied in this case, downsampling results in aliasing, as depicted in Fig. 6.28b for $M = 2$. A decimator, on the other hand, first applies an anti-aliasing filter to produce a signal $\hat{x}[n]$ with bandlimited spectrum $\hat{X}(\Omega)$, shown in Fig. 6.28c. Downsampling $\hat{x}[n]$ produces the decimator output $x_d[n]$. As Fig. 6.28d shows, although $X_d(\Omega)$ is missing the spectral tails of $X(\Omega)$ (causing distortion), the tails do not reappear through aliasing (folding) to cause additional distortion, as is the case in simple downsampling. In general, decimation causes less distortion than downsampling.

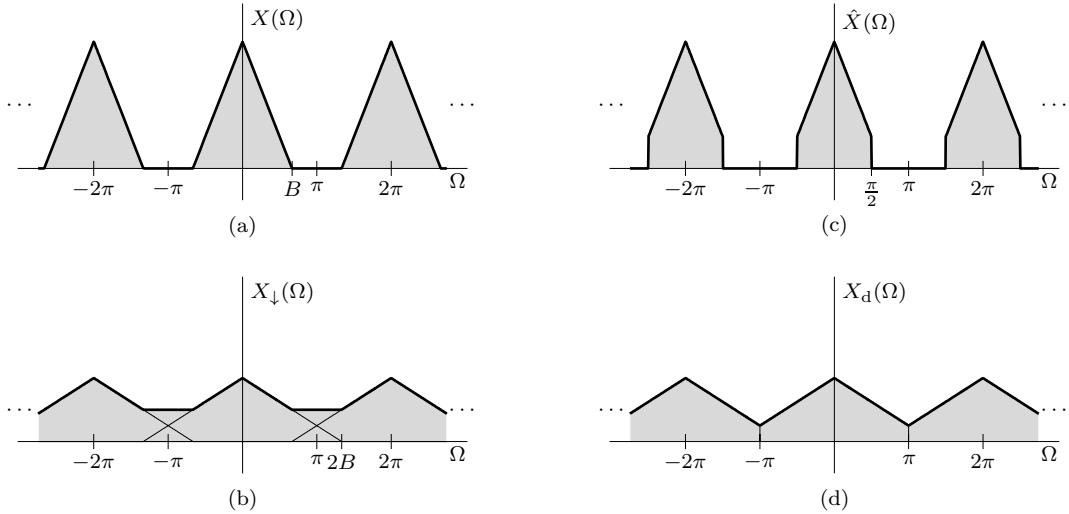


Figure 6.28: Distortion from downsampling and decimation using $M = 2$: (a) $X(\Omega)$, (b) $X_{\downarrow}(\Omega)$, (c) $\hat{X}(\Omega)$, and (d) $X_d(\Omega)$.

▷ Example 6.23 (Downsampling and Decimation)

Figure 6.29a shows the DTFT of a signal $x[n]$. Using $M = 3$, determine the DTFTs of the downsampled signal $x_{\downarrow}[n]$ and the decimated signal $x_d[n]$. Assume that the decimator uses an ideal lowpass filter.

From Eq. (6.68), the spectrum of the $M = 3$ downsampled signal is given as

$$X_{\downarrow}(\Omega) = \frac{1}{3} \sum_{m=0}^2 X\left(\frac{\Omega - 2\pi m}{3}\right).$$

Hence, the spectrum $X_{\downarrow}(\Omega)$ is obtained by multiplying the spectrum $X(\Omega)$ in Fig. 6.29a by $1/3$, frequency expanding it by a factor of 3, and adding shifted replicates to achieve 2π -periodicity, as shown in Fig. 6.29b. Expansion by $M = 3$ triples the bandwidth from $\pi/2$ to $3\pi/2$, which causes significant aliasing distortion because the repeating spectra overlap. Such aliasing is to be expected since the condition of Eq. (6.69) is not satisfied ($B = \pi/2 > \pi/3 = \pi/M$).

In decimation, we first produce an estimate of our signal $\hat{x}[n]$ that is bandlimited to $\pi/3$, the spectrum of which is shown in Fig. 6.29c. This is achieved by passing $x[n]$ through an ideal lowpass filter with a cutoff frequency of $\pi/3$. Once filtered, $\hat{x}[n]$ is downsampled by $M = 3$ to produce the decimator output $x_d[n]$. The decimator output spectrum $X_d(\Omega)$, shown in Fig. 6.29d, produces much less distortion than a simple downsampler (Fig. 6.29b).

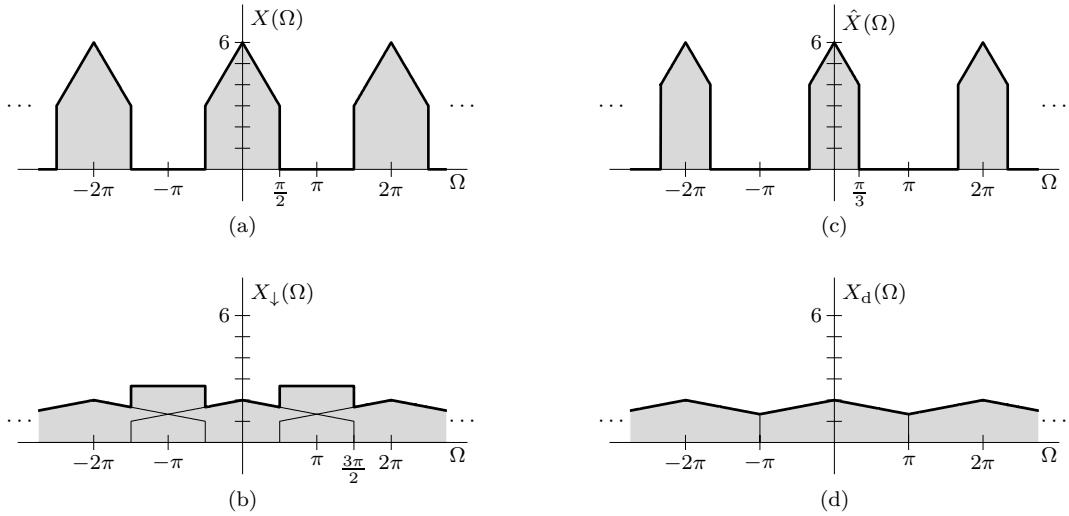


Figure 6.29: Downsampling and decimation using $M = 3$: (a) $X(\Omega)$, (b) $X_{\downarrow}(\Omega)$, (c) $\hat{X}(\Omega)$, and (d) $X_d(\Omega)$.

Example 6.23 ◀

▷ Example 6.24 (Determining $x_{\downarrow}[n]$ and $X_{\downarrow}(\Omega)$ from $x[n]$)

A signal $x[n] = \gamma^n u[n]$, where $|\gamma| < 1$, is downsampled by factor $M = 2$. Determine the resulting downsampled signal $x_{\downarrow}[n] = x[2n]$ and its spectrum $X_{\downarrow}(\Omega)$.

From Table 6.1 or Eq. (6.1), we know that

$$x[n] = \gamma^n u[n] \iff \frac{e^{j\Omega}}{e^{j\Omega} - \gamma} = \frac{1}{1 - \gamma e^{-j\Omega}} = X(\Omega).$$

Also,

$$x_{\downarrow}[n] = x[Mn] = x[2n] = \gamma^{2n} u[2n] = \gamma^{2n} u[n].$$

Although we can determine $X_{\downarrow}(\Omega)$ directly using Table 6.1 or Eq. (6.1), it is instructive to derive the result using Eq. (6.68).

From Eq. (6.68), we have

$$\begin{aligned} X_{\downarrow}(\Omega) &= \frac{1}{2} \left[X\left(\frac{\Omega}{2}\right) + X\left(\frac{\Omega - 2\pi}{2}\right) \right] \\ &= \frac{1}{2} \left[X\left(\frac{\Omega}{2}\right) + X\left(\frac{\Omega}{2} - \pi\right) \right] \\ &= \frac{1}{2(1 - \gamma e^{-j\Omega/2})} + \frac{1}{2(1 + \gamma e^{-j\Omega/2})} \\ &= \frac{1}{1 - \gamma^2 e^{-j\Omega}}. \end{aligned}$$

Example 6.24 ◀

▷ **Example 6.25 (Practical Decimation)**

Using the realizable filter of Fig. 6.16, sketch the $M = 3$ decimation of the signal $x[n] = u[n] - u[n - 20]$.

With MATLAB, solving this problem is nearly effortless. Referring to Fig. 6.16a, we first define the impulse response of our decimation filter.

```
01 ng = 12; wc = pi/3; n = 0:2*ng; hd = wc/pi*sinc(wc*(n-ng)/pi);
```

As shown in Fig. 6.16b, the cutoff frequency of this filter is $\pi/3$, which is an appropriate value for an $M = 3$ decimator.

The decimation filter output $\hat{x}[n]$ is the convolution of the finite-duration impulse response of the decimation filter and the input $x[n]$, which is a duration-20 pulse with unit amplitude. Since both $x[n]$ and $h_d[n]$ are of finite duration, MATLAB's `conv` command is able to compute the convolution. This filter output is then downsampled by $M = 3$ to produce the decimator output $x_d[n]$.

```
02 x = ones(20,1); xhat = conv(x,hd); xd = xhat(1:3:end);
03 subplot(121); stem(0:length(xhat)-1,xhat); subplot(122); stem(0:length(xd)-1,xd);
```

As shown in Fig. 6.30a, the output of the decimation filter is a smoothed (filtered) pulse that has an effective duration slightly greater than the input duration of 20. This pulse, bandlimited to approximately $\pi/3$, is delayed by $n_g = 12$ due to the inherent time delay needed to render the decimation filter causal (practical). The decimator output, shown in Fig. 6.30b, has an effective duration of around 7, which is about one-third the original pulse duration. This is sensible, given the $M = 3$ decimation factor. Unlike simple downsampling, we see that $x_d[n]$ changes the samples of the original input, which were either 0 or 1 in amplitude.

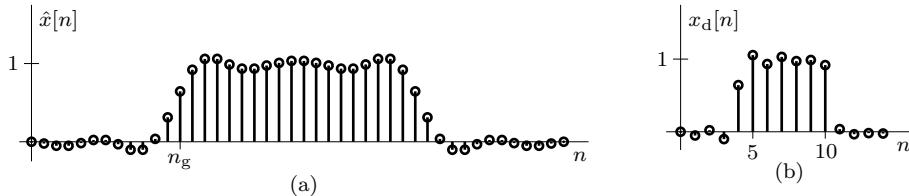


Figure 6.30: Practical $M = 3$ decimation: (a) $\hat{x}[n]$ and (b) $x_d[n]$.

Example 6.25 □

▷ **Drill 6.10 (Downsampling a Sinusoid)**

A signal $x[n] = \cos(\Omega_0 n)$ downsampled by factor 2 is $x_{\downarrow}[n] = x[2n] = \cos(2\Omega_0 n)$. Show this fact in the frequency domain using Eq. (6.68). What happens if $|\Omega_0| > \pi/2$? Hint: $\delta(ax) = \delta(x)/|a|$.

□

6.6.3 Interpolation and Upsampling

Earlier we derived $X_d(\Omega)$ in terms of $X(\Omega)$. We now do the same for $X_i(\Omega)$ by representing it in terms of $X(\Omega)$. From Fig. 6.26l, we observe that the spectrum $X_i(\Omega)$ somewhat appears like $L X(L\Omega)$, which is $L X(\Omega)$ frequency-compressed by factor $L = 2$. Let us see if this is true. Recall

that $X(L\Omega)$ is obtained by replacing Ω with $L\Omega$ in the expression for $X(\Omega)$. Hence, from Eq. (6.64), we find $LX(L\Omega)$ as

$$LX(L\Omega) = L \sum_{k=-\infty}^{\infty} X_c(L\Omega - 2\pi k) = L \sum_{k=-\infty}^{\infty} X_c\left(L\left[\Omega - \frac{2\pi k}{L}\right]\right). \quad (6.71)$$

Using $L = 2$, Fig. 6.31a shows the spectrum $LX(L\Omega)$ as obtained on the right-hand side of Eq. (6.71). This spectrum repeats at intervals $2\pi/L$ instead of 2π , as in Fig. 6.26l. The frequency compression of spectrum $X(\Omega)$ by factor L not only compresses each cycle but also compresses the period by the same factor L .

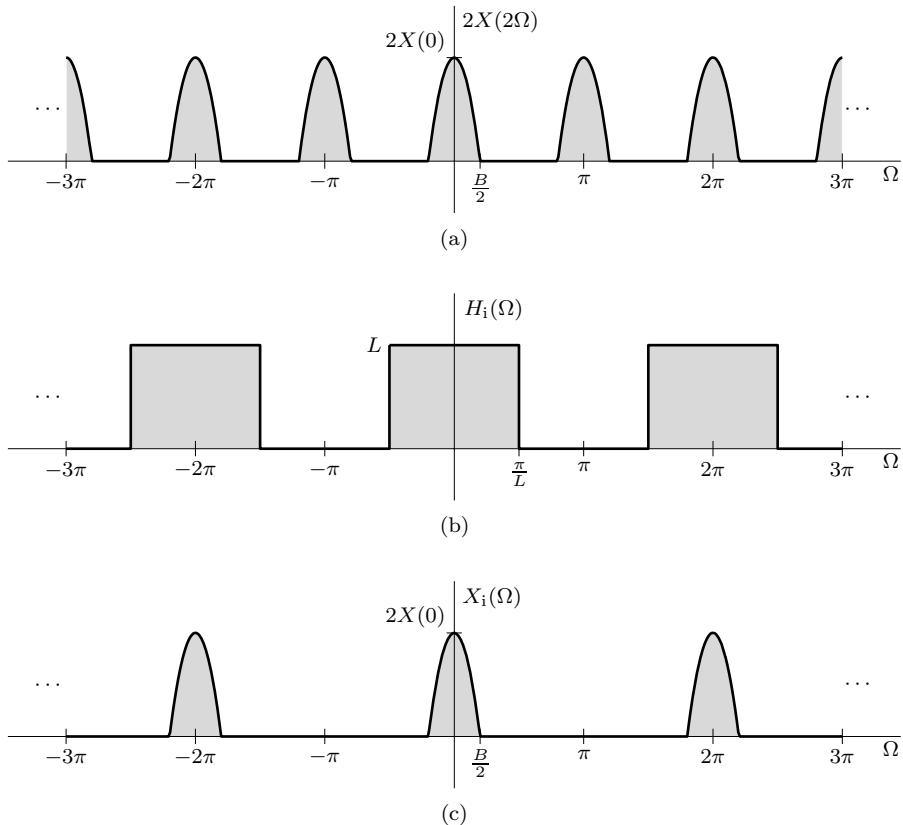


Figure 6.31: Obtaining $X_i(\Omega)$ from $X(\Omega)$ for $L = 2$: (a) $LX(L\Omega)$, (b) $H_i(\Omega)$, and (c) $X_i(\Omega) = X(L\Omega)H_i(\Omega)$.

The spectrum $LX(L\Omega)$ in Eq. (6.71) is almost identical to $X_i(\Omega)$ in Eq. (6.66) except that the repetition period is $2\pi/L$ instead of 2π . In each period 2π , $LX(L\Omega)$ contains L repeating cycles. The additional (unwanted) repeating spectra are called *images* of the input spectrum. There are $L - 1$ images. We can readily obtain the desired spectrum in Fig. 6.26l by passing the spectrum $LX(L\Omega)$ (Fig. 6.31a) through an ideal lowpass filter of bandwidth π/L , which is the same as passing the spectrum $X(L\Omega)$ through an ideal lowpass filter of gain L and bandwidth π/L , as shown in Fig. 6.31b. This filter suppresses the unwanted $L - 1$ images. Thus,

$$X_i(\Omega) = X(L\Omega)H_i(\Omega), \quad (6.72)$$

where $H_i(\Omega)$ is an ideal lowpass filter of gain L and bandwidth π/L rad/sample represented as

$$H_i(\Omega) = L \sum_{k=-\infty}^{\infty} \Pi\left(\frac{\Omega - 2\pi k}{2\pi/L}\right). \quad (6.73)$$

The output of the filter is the desired spectrum $X_i(\Omega)$, depicted in Fig. 6.31c. The interpolating filter in Eq. (6.73), being an ideal filter, is unrealizable. In practice, we must compromise and use some realizable approximation.

Equation (6.72) shows that $x_i[n]$ can be obtained by lowpass filtering the spectrum $X(L\Omega)$. But what is $X(L\Omega)$? How do we generate it? We now show that the DTFT of the upsampled signal $x_{\uparrow}[n]$ is $X(L\Omega)$. Using Eq. (6.60), we obtain

$$X_{\uparrow}(\Omega) = \sum_{n=-\infty}^{\infty} x_{\uparrow}[n]e^{-j\Omega n} = \sum_{n=0, \pm L, \pm 2L, \dots} x[n/L]e^{-j\Omega n}.$$

Letting $k = n/L$ so that k takes on all integer values, we obtain

$$X_{\uparrow}(\Omega) = \sum_{k=-\infty}^{\infty} x[k]e^{-j\Omega Lk} = X(L\Omega). \quad (6.74)$$

In other words, upsampling a signal by factor L results in a simple compression of its spectrum by the same factor L .

Substituting Eq. (6.74) into Eq. (6.72), it follows that

$$X_i(\Omega) = X_{\uparrow}(\Omega)H_i(\Omega). \quad (6.75)$$

This development shows that the interpolated signal $x_i[n]$ is generated by lowpass filtering the upsampled signal $x_{\uparrow}[n]$, exactly as shown in Fig. 6.25d. We emphasize that the interpolating filter H_i is placed *after* the expander. The interpolation filter essentially fills in the gaps (zeros) created during the upsampling operation.

▷ Example 6.26 (An Upsampling and Interpolation Puzzle)

Consider a signal $x[n] = \gamma^{2n}u[n]$, where $|\gamma| < 1$. Using $L = 2$, determine the spectra of the upsampled signal $x_{\uparrow}[n]$ and the ideally interpolated signal $x_i[n]$.

In Ex. 6.24, we found that downsampling the signal $\gamma^n u[n]$ by a factor of 2 results in the signal $\gamma^{2n} u[n]$. Thus, we might expect that upsampling or interpolating the signal $\gamma^{2n} u[n]$ by a factor of 2 results in $\gamma^n u[n]$. Let us see.

We have $x[n] = \gamma^{2n} u[n] = (\gamma^2)^n u[n]$. From Table 6.1, we obtain $X(\Omega) = 1/(1 - \gamma^2 e^{-j\Omega})$. Figure 6.32a shows the magnitude of $X(\Omega)$. For convenience, we shall display only the magnitude spectra in this problem.

According to Eq. (6.74), the expanded signal spectrum for $L = 2$ is

$$X_{\uparrow}(\Omega) = X(2\Omega) = \frac{1}{1 - \gamma^2 e^{-j2\Omega}}.$$

Clearly, $X_{\uparrow}(\Omega)$ is the spectrum $X(\Omega)$ frequency-compressed by a factor of 2. Figure 6.32b shows the magnitude response $|X_{\uparrow}(\Omega)|$, which is just Fig. 6.32a compressed by 2.

To obtain the spectrum $X_i(\Omega)$ for the upsampled signal $x_i[n]$, we pass the spectrum $X_{\uparrow}(\Omega)$ through an ideal lowpass filter with cutoff frequency $\frac{\pi}{L} = \frac{\pi}{2}$ and gain $L = 2$ to obtain

$$X_i(\Omega) = X_{\uparrow}(\Omega)H_i(\Omega) = \frac{2}{1 - \gamma^2 e^{-j2\Omega}} \Pi\left(\frac{\Omega}{\pi}\right), \quad |\Omega| \leq \pi.$$

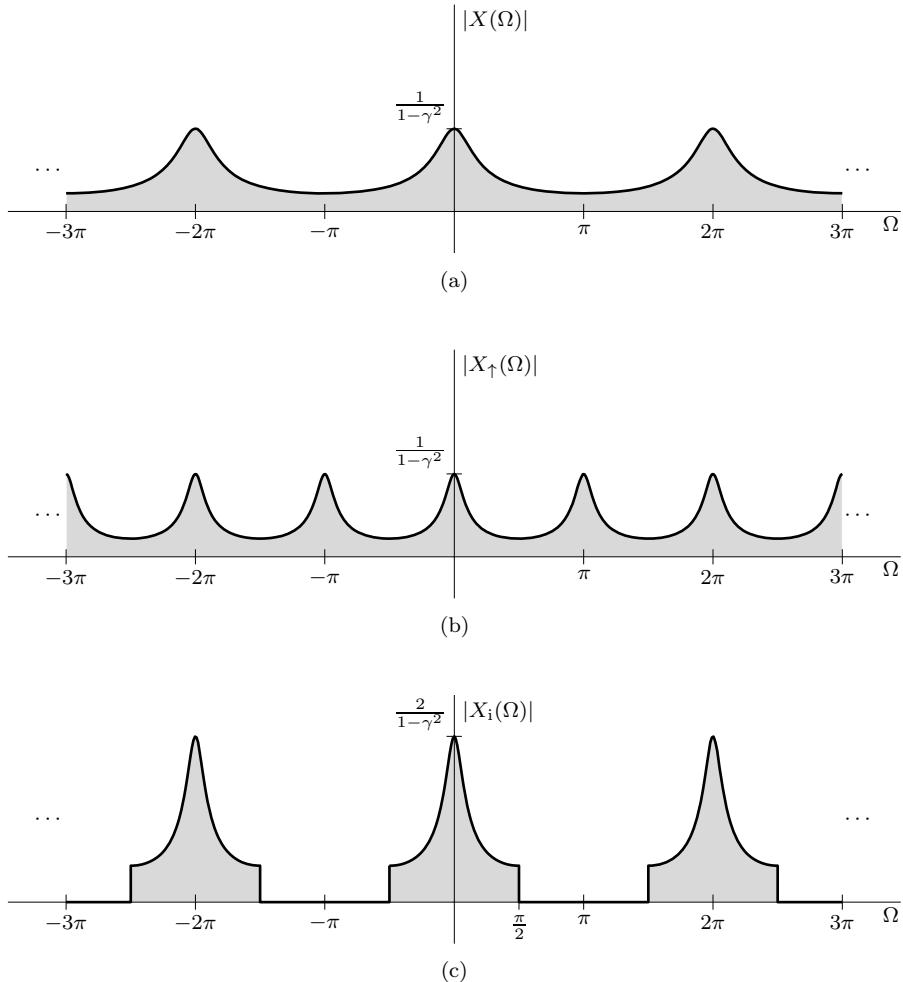


Figure 6.32: Upsampling and interpolation of $x[n] = \gamma^{2n}u[n]$ by factor $L = 2$: (a) $|X(\Omega)|$, (b) $|X_{\uparrow}(\Omega)|$, and (c) $|X_i(\Omega)|$.

The magnitude of the resulting spectrum $X_i(\Omega)$ is shown in Fig. 6.32c.

Both $X_{\uparrow}(\Omega)$ and $X_i(\Omega)$ are quite different from $1/(1-\gamma e^{-j\Omega})$, the spectrum for $\gamma^n u[n]$, shown in Fig. 6.3. Clearly, upsampling or interpolating $\gamma^{2n}u[n]$ does not yield $\gamma^n u[n]$. How could this be? We note that the signal $\gamma^n u[n]$, having a full bandwidth of π , does not satisfy the condition of Eq. (6.69). Hence, in this case, downsampling leads to aliasing, causing loss of data. In contrast, upsampling and ideal interpolation never cause a loss of data. Hence, unless there is no aliasing in downsampling, we should not generally expect that upsampling or interpolation will restore a downsampled signal to its original form. In contrast, a signal upsampled or ideally interpolated by factor L and then downsampled by the same factor always yields the original signal. This is because upsampling and ideal interpolation do not cause loss of data but create redundancies that are eliminated by the subsequent downsampling, thus yielding the original signal. The reader is encouraged to verify that the upsampled or interpolated signals in this example, when downsampled by $M = L = 2$, yield the original signal $\gamma^{2n}u[n]$. These observations confirm the conclusions of Ch. 4 that the order of operations is important when compressing and expanding DT signals.

Example 6.26 ◀

▷ **Drill 6.11 (Interpolating a Sinusoid)**

A signal $x[n] = \cos(\Omega_0 n)$ downsampled by factor $M = 2$ is $x_{\downarrow}[n] = x[2n] = \cos(2\Omega_0 n)$ (see Drill 6.10). Now show that $\cos(2\Omega_0 n)$ interpolated by factor $L = 2$ is $\cos(\Omega_0 n)$. Show this in the frequency domain using Eq. (6.74) and then lowpass filtering the unwanted image spectrum (Eq. (6.75)).

△

6.6.4 Time-Domain Characterizations

While the frequency domain is very intuitive, resampling itself most often occurs using time-domain expressions. For this reason, we must not lose sight of our time-domain perspectives of resampling. Both upsampling and downsampling are easily represented in the time domain, as shown in Eqs. (6.60) and (6.59). Basically, upsampling stuffs $L - 1$ zeros between each sample, and downsampling throws out $M - 1$ out of every M samples. Time-domain characterizations of interpolation and decimation are not quite so simple.

To express $x_i[n]$ directly in terms of $x[n]$, we take the inverse of Eq. (6.75) to obtain

$$x_i[n] = x_{\uparrow}[n] * h_i[n] = \sum_{m=-\infty}^{\infty} x_{\uparrow}[m] h_i[n-m].$$

Using Eq. (6.60), we obtain

$$\begin{aligned} x_i[n] &= \sum_{m=-\infty}^{\infty} \left(\sum_{k=-\infty}^{\infty} x[k] \delta[m - kL] \right) h_i[n-m] \\ &= \sum_{k=-\infty}^{\infty} x[k] \left(\sum_{m=-\infty}^{\infty} h_i[n-m] \delta[m - kL] \right) \\ &= \sum_{k=-\infty}^{\infty} x[k] h_i[n - kL]. \end{aligned}$$

Using entry 8 of Table 6.1, the impulse response of the ideal interpolator of Eq. (6.73) is

$$h_i[n] = \text{sinc}(n/L). \quad (6.76)$$

Using this result, the output of an ideal interpolator thus simplifies to

$$x_i[n] = \sum_{k=-\infty}^{\infty} x[k] \text{sinc}\left(\frac{n}{L} - k\right). \quad (6.77)$$

Following a similar procedure (see Prob. 6.6-12), we find that the impulse response of an ideal decimation filter is

$$h_d[n] = \frac{1}{M} \text{sinc}(n/M), \quad (6.78)$$

and the output of an ideal decimator is

$$x_d[n] = \frac{1}{M} \sum_{k=-\infty}^{\infty} x[k] \text{sinc}\left(n - \frac{k}{M}\right). \quad (6.79)$$

Remember that an interpolator outputs samples $x_i[n]$ at a rate L times faster than its input samples, while a decimator outputs samples $x_d[n]$ at a rate M times slower. Even with this difference, the

similarity between Eqs. (6.77) and (6.79) is striking, although not altogether surprising, given that both rely on lowpass filters that are nearly identical in form. Since both involve ideal lowpass filters, neither Eq. (6.77) nor Eq. (6.79) is in realizable form.

In practice, interpolators and decimators require some realizable approximation of their ideal filters. To provide one example, let us consider *linear interpolation*, which is a popular and realizable approximation of ideal interpolation. This method approximates the main lobe of $h_i[n]$ in Eq. (6.76) with an inscribed triangle and discards all side lobes, as shown for $L = 4$ in Fig. 6.33.

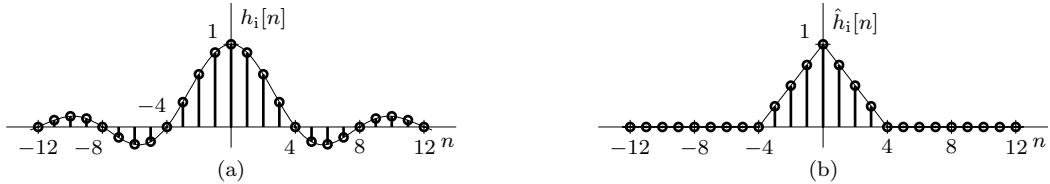


Figure 6.33: $L = 4$ interpolator impulse responses: (a) ideal and (b) linear approximation.

Thus, the impulse response of a linear interpolator is

$$\hat{h}_i[n] = \Lambda\left(\frac{n}{2L}\right). \quad (6.80)$$

For convenience, Eq. (6.80) ignores the delay of $L - 1$ samples needed to render $\hat{h}_i[n]$ causal (realizable). By incorporating such a delay, the only effect is a corresponding delay of $L - 1$ samples in the output.

▷ Example 6.27 (Linear Interpolation)

Using the $L = 4$ linear interpolator of Fig. 6.33b and an input of $x[n] = \sin(2\pi n/7)(u[n] - u[n-7])$,

Figure 6.34a shows the signal $x[n]$, which consists of a single cycle of a sine wave. To upsample by factor $L = 4$, three zeros are inserted between each sample of $x[n]$ to produce $x_{\uparrow}[n]$, as shown in Fig. 6.34b.

As per Eq. (6.80), the impulse response of the linear interpolating filter is given by

$$\hat{h}_i[n] = \Lambda\left(\frac{n}{8}\right).$$

This response is shown in Fig. 6.33b. The interpolator output $x_i[n]$ can be found as the convolution of $x_{\uparrow}[n]$ and $\hat{h}_i[n]$. In this particular example, however, it is simpler to find the output directly by observing that each nonzero sample of $x_{\uparrow}[n]$ is an impulse and hence generates $\Lambda(n/8)$ centered at that sample and with a height equal to that of the sample, as shown in Fig. 6.34c. The interpolator output $x_i[n]$, shown in Fig. 6.34d, is the sum of all these scaled and shifted triangle functions. Clearly, linear interpolation expands $x[n]$ using straight-line approximations between samples.

Since both $x[n]$ and $\hat{h}_i[n]$ have finite duration, the results of Figs. 6.34a, 6.34b, and 6.34d are easily confirmed using MATLAB.

```

01 x = @(n) sin(2*pi*n/7).*((n>=0)&(n<7)).*(mod(n,1)==0); n = -2:8;
02 subplot(311); stem(n,x(n)); xlabel('n'); ylabel('x[n]');
03 xup = @(n) x(n/4); n2 = (n1)*4:n(end)*4;
04 subplot(312); stem(n2,xup(n2)); xlabel('n'); ylabel('x_{\uparrow}[n]');
05 hi = [1 2 3 4 3 2 1]/4; n3 = (-3+n2(1)):3+n2(end));
06 subplot(313); stem(n3,conv(xup(n2),hi)); xlabel('n'); ylabel('x_i[n]');

```

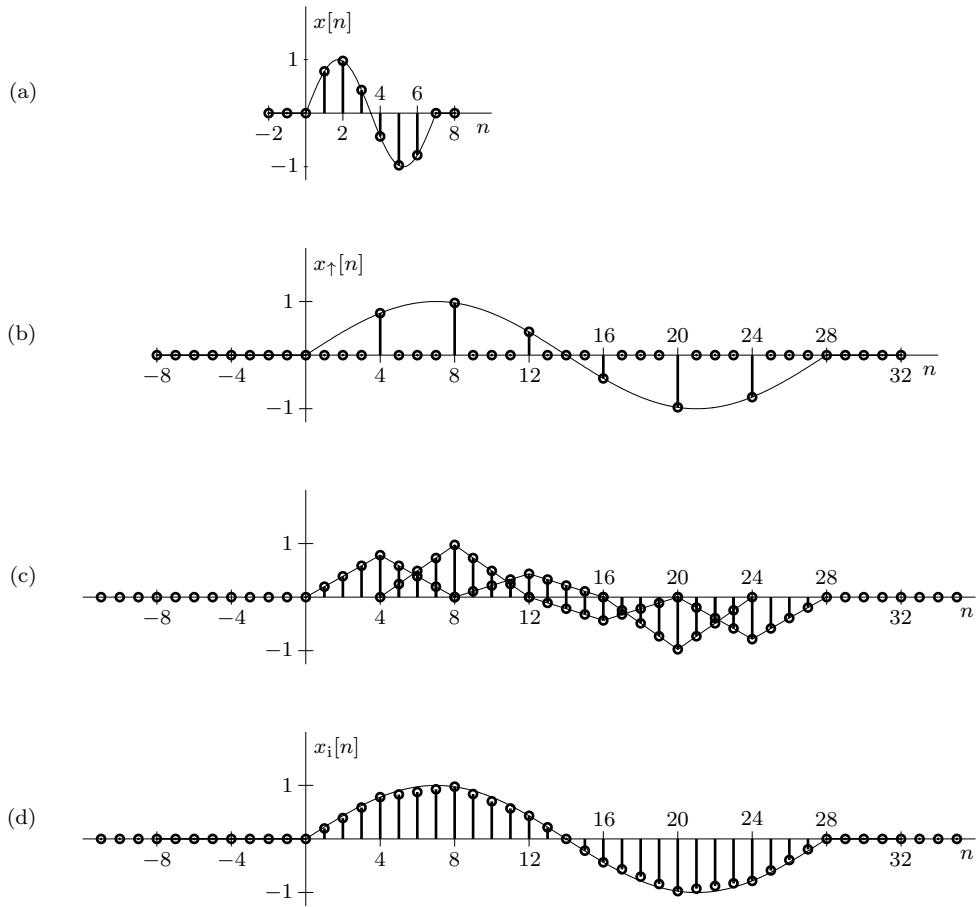


Figure 6.34: $L = 4$ linear interpolation: (a) $x[n]$, (b) $x_{\uparrow}[n]$, (c) constructing $x_i[n]$, and (d) $x_i[n]$.

The term `(mod(n,1)==0)` in line 01 ensures that the upsampling in line 03 correctly inserts zeros for non-integer arguments of $x[n]$. This particular MATLAB code is just one of many possible coding strategies to solve this problem.

Example 6.27 ◀

Analog or Digital Implementation?

Resampling of $x[n]$ can be accomplished by first constructing a bandlimited signal $x_c(t)$ from $x[n]$ using D/A conversion. By time scaling $x_c(t)$, we next obtain $x_c(Mt)$ or $x_c(t/L)$. Lastly, the time-scaled signal is sampled (A/D conversion) to obtain the desired signal $x_{\downarrow}[n]$ or $x_i[n]$. This analog procedure, though feasible, requires costly analog components and introduces spectral distortion due to non-ideal converter characteristics, including quantization effects in A/D converter, which limit the effective dynamic range. Thus, it is more desirable to resample $x[n]$ directly in the digital domain without going through $x_c(t)$ as an intermediary. The only disadvantage of the digital method is that the sampling rate can be changed only by a rational factor; not a serious problem. Moreover, the digital route is much simpler. For downsampling, all we need is to retain every M th sample and discard the remaining samples. For upsampling, we interleave $L - 1$ zeros between each sample. Decimation and interpolation further require the use of digital lowpass filters.

Time-Frequency Duality

Since $x_c(t)$ is constructed to be a bandlimited interpolation of $x[n]$ using $T = 1$, we recognize that

$$X_c(\Omega) = X(\Omega), \quad |\Omega| \leq \pi.$$

Assuming that Eq. (6.69) is satisfied and downsampling does not cause aliasing, it follows that $x_d[n] = x_{\downarrow}[n]$ and, from Eqs. (6.65) and (6.66), that

$$X_d(\Omega) = X_{\downarrow}(\Omega) = \frac{1}{M}X(\Omega/M) \quad \text{and} \quad X_i(\Omega) = LX(L\Omega), \quad |\Omega| \leq \pi. \quad (6.81)$$

Observe the time-frequency duality. Time compression of $x[n]$ by a factor M results in spectral expansion by M , while time expansion of $x[n]$ by a factor L results in spectral compression by factor L . These relations governing time scaling are identical to that for continuous-time signal scaling in Eq. (1.85). However, we should keep in mind that for discrete-time signals, the time-frequency duality applies to spectra in the fundamental band only. It is not the same situation when we try to apply it to the entire frequency range, as seen earlier.[†]

▷ Drill 6.12 (Frequency Response of a Linear Interpolator)

Show that the frequency response of the linear interpolation filter of Eq. (6.80) is

$$\hat{H}_i(\Omega) = \frac{1}{L} \left[\frac{\sin(\Omega L/2)}{\sin(\Omega/2)} \right]^2.$$

Using $L = 4$, sketch this frequency response, and compare it with the frequency response needed for ideal interpolation.

△

▷ Drill 6.13 (Linear Interpolation)

Repeat Ex. 6.27 using the signal $x[n] = \cos(2\pi n/7)(u[n] - u[n - 7])$ rather than $x[n] = \sin(2\pi n/7)(u[n] - u[n - 7])$. Comment on the results.

△

6.6.5 Fractional Sampling Rate Conversion

Section 4.6 discusses cascading upsamplers and downsamplers to achieve fractional changes in sampling rate. Unless the upsampling and downsampling factors L and M are coprime, the order of operations is important. Further, it is typically preferable to perform upsampling prior to downsampling. Upsampling by L followed by downsampling by M changes the overall sampling rate by a fractional amount L/M .

To minimize aliasing and image problems, interpolation and decimation are generally preferable to upsampling and downsampling. Thus, it is most common to realize fractional sampling rate changes through an interpolator/decimator cascade. Since interpolators and decimators utilize upsamplers and downsamplers, order of operation is almost always important. Figure 6.35a shows the most common structure, which places the interpolator before the decimator. This structure resamples input $x[n]$ to produce output $x_r[n]$ at L/M of the original sampling rate F_s .

[†]But in the fundamental sense, this result is valid for all discrete-time frequencies. Ultimately, all discrete-time signals are bandlimited to a band $|\Omega| \leq \pi$, and in the real sense, frequencies beyond π do not exist. Hence, the conclusion applicable to the fundamental band is a general conclusion. But since we admit discrete frequencies higher than π , we have to make concessions for a general result that is valid for all frequencies. This is accomplished by finding more general expressions for $X_d(\Omega)$ and $X_i(\Omega)$ that also satisfy 2π -periodicity.

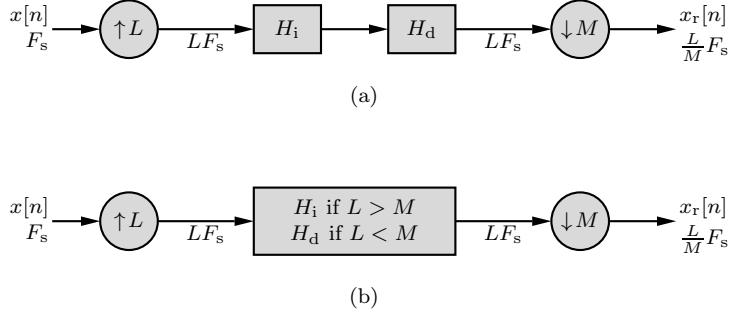


Figure 6.35: Block representations for fractional sampling rate conversion.

The order of operations in Fig. 6.35a (interpolation followed by decimation) is important for several reasons. First, the two lowpass filters in Fig. 6.35a, being in cascade, can be replaced by a single lowpass filter of cutoff frequency π/L or π/M , whichever is lower, as depicted in Fig. 6.35b. Second, the ordering of Fig. 6.35a is necessary to avoid needless loss of data resulting from the decimation filter used to reduce aliasing. For example, suppose that the bandwidth of $x[n]$ is $\pi/3$, and we wish to change the sampling rate by factor 3/5. In this case, we need to decimate the signal by factor 5 and interpolate it by factor 3. If we decimate first, the signal must be passed through a decimation filter of cutoff frequency $\pi/5$. Since the bandwidth of $x[n]$ is $\pi/3$, this filter eliminates the band from $\pi/5$ to $\pi/3$, causing data loss. The resulting interpolated signal has bandwidth $\pi/3$, which is lower than the expected bandwidth of $\frac{M}{L}(\frac{\pi}{3}) = \frac{5\pi}{9}$. On the other hand, if we perform interpolation first, we expand the signal by factor 3, which reduces the bandwidth of the signal from $\pi/3$ to $\pi/9$. This signal next passes through a lowpass filter of bandwidth $\pi/5$, which serves as both the decimation and interpolation filters. Since the filter bandwidth ($\pi/5$) exceeds the signal bandwidth ($\pi/9$), the original signal information remains intact. The final step, which downsamples by 5, expands the spectrum by a factor of 5, which produces an output signal $x_r[n]$ with bandwidth $5\pi/9$, as expected.

6.7 Generalization of the DTFT to the z -Transform

In this chapter, we have seen that LTID systems can be analyzed using the DTFT. The DTFT approach, however, has significant limitations.

1. Existence of the DTFT is guaranteed only for absolutely summable signals. The DTFT does not exist for exponentially growing signals. This means that DTFT analysis applies only for a limited class of inputs.
2. The DTFT method can be applied only to asymptotically stable systems. It cannot be used for unstable or even marginally stable systems.

These are serious limitations in LTID system analysis. Actually, it is the first limitation that is also the cause of the second limitation. Because the DTFT is incapable of handling growing signals, it is incapable of handling unstable or marginally stable systems.[†] Our goal is therefore to extend the concept of the DTFT so that it can handle exponentially growing signals.

What causes this limitation so that the DTFT is incapable of handling exponentially growing signals? Recall that in the DTFT, we are synthesizing an arbitrary signal $x[n]$ using sinusoids or complex exponentials of the form $e^{j\Omega n}$. These signals oscillate but do not grow (or decay) with time. Thus, they are incapable of synthesizing exponentially growing signals no matter how many

[†]Recall that the output of an unstable system grows exponentially, as does the output of a marginally stable system in response to inputs that contain unit-circle modes of the system.

such components we add. Our hope, therefore, lies in trying to synthesize $x[n]$ using exponentially growing sinusoids or exponentials. This goal can be accomplished by generalizing the frequency variable $j\Omega$ to $\sigma + j\Omega$, that is, by using exponentials of the form $e^{(\sigma+j\Omega)n}$ instead of exponentials $e^{j\Omega n}$. The procedure is almost identical to that used in extending the Fourier transform to the Laplace transform.

Let us define $\hat{X}(j\Omega) = X(\Omega)$. Hence, the DTFT analysis and synthesis equations are

$$\hat{X}(j\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n} \quad \text{and} \quad x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{X}(j\Omega) e^{j\Omega n} d\Omega.$$

Now, the DTFT of $x[n] e^{-\sigma n}$, where σ is real, is given as

$$\begin{aligned} \text{DTFT}\{x[n] e^{-\sigma n}\} &= \sum_{n=-\infty}^{\infty} x[n] e^{-\sigma n} e^{-j\Omega n} \\ &= \sum_{n=-\infty}^{\infty} x[n] e^{-(\sigma+j\Omega)n} \\ &= \hat{X}(\sigma + j\Omega). \end{aligned}$$

Clearly, the inverse DTFT of $\hat{X}(\sigma + j\Omega)$ is $x[n] e^{-\sigma n}$. Therefore,

$$x[n] e^{-\sigma n} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{X}(\sigma + j\Omega) e^{j\Omega n} d\Omega.$$

Multiplying both sides of this equation by $e^{\sigma n}$ yields

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{X}(\sigma + j\Omega) e^{(\sigma+j\Omega)n} d\Omega.$$

Let us define a new variable z as

$$z = e^{\sigma+j\Omega}$$

so that

$$\ln(z) = \sigma + j\Omega \quad \text{and} \quad \frac{1}{z} dz = j d\Omega.$$

Because $z = e^{\sigma+j\Omega}$ is complex, we can express it as $z = re^{j\Omega}$, where $r = e^\sigma$. In the complex plane, z lies on a circle of radius $r = e^\sigma$, and as Ω varies from $-\pi$ to π , z circumambulates along this circle, completing exactly one rotation in a counterclockwise direction, as illustrated in Fig. 6.36. Much like s , we interpret variable z as complex frequency.

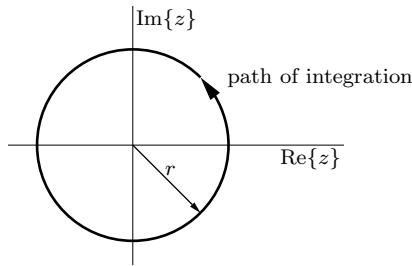


Figure 6.36: Contour of integration for the z -transform.

Thus,

$$\hat{X}(\ln[z]) = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \quad \text{and} \quad x[n] = \frac{1}{2\pi j} \oint \hat{X}(\ln[z]) z^{n-1} dz, \quad (6.82)$$

where the integral \oint indicates a contour integral in the complex plane around a circle of radius r in counterclockwise direction.

Equation (6.82) is the DTFT extension we desire. The expressions are, however, in a clumsy form. For the sake of convenience, we make another notational change by observing that $\hat{X}(\ln[z])$ is a function of z . Let us denote it by a simpler notation $X(z)$.[†] Using this notation, we obtain

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \quad (6.83)$$

and

$$x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz. \quad (6.84)$$

These are the analysis and synthesis equations of the (bilateral) z -transform, which is detailed in the next chapter. The synthesis equation (Eq. (6.84)) expresses $x[n]$ as a sum of exponentials of the form $z^n = e^{(\sigma+j\Omega)n} = r^n e^{j\Omega n}$ over a continuum of Ω . By selecting a proper value for r (or σ), we can make the exponential grow (or decay) at any exponential rate we desire, thereby accommodating all manner of signals, including exponentially growing ones.

If we let $\sigma = 0$, we have $z = e^{j\Omega}$, and

$$X(z)|_{z=e^{j\Omega}} = \hat{X}(\ln[z])|_{z=e^{j\Omega}} = \hat{X}(j\Omega) = X(\Omega).$$

Thus, the familiar DTFT is just a special case of the z -transform $X(z)$ obtained by letting $z = e^{j\Omega}$ and assuming that the sum on the right-hand side of the analysis equation (Eq. (6.83)) converges when $z = e^{j\Omega}$. In other words, we can obtain the DTFT from the z -transform if the region of convergence (ROC) for $X(z)$ includes the unit circle.

6.8 Summary

This chapter deals with the frequency-domain analysis of discrete-time signals and systems. The discrete-time Fourier transform (DTFT) is very similar to its continuous-time counterpart, and in fact, it is possible to derive a DTFT pair from a corresponding CTFT pair. Further, most of the properties of DTFT spectra are similar to their continuous-time counterparts. Thus, many frequency-domain analysis concepts apply to both CT and DT signals. Unlike CTFT spectra, which are aperiodic, all DTFT spectra are 2π -periodic and completely defined using the fundamental band $|\Omega| \leq \pi$. This difference in frequency-domain periodicity leads to several subtle but important differences between the CTFT and the DTFT.

If $H(\Omega)$ is the DTFT of an LTID system's impulse response $h[n]$, then $|H(\Omega)|$ is the magnitude response and $\angle H(\Omega)$ is the phase response of the system. Moreover, if $X(\Omega)$ and $Y(\Omega)$ are the DTFTs of the input $x[n]$ and the corresponding output $y[n]$, then $Y(\Omega) = H(\Omega)X(\Omega)$. Therefore, the output spectrum is the product of the input spectrum and the system's frequency response. Like continuous-time systems, distortionless transmission conditions for discrete-time systems require constancy of magnitude response and linearity of phase response. Phase linearity corresponds to simple delay. Even with delay, however, ideal filters, being noncausal, are realizable only approximately through suitable impulse-response truncation.

A continuous-time signal $x_c(t)$ is said to be a bandlimited interpolation of $x[n]$ if $x_c(nT) = x[n]$ and its bandwidth is $\leq 1/2T$ Hz. Although the parameter T is arbitrary and can be selected to suit the needs of the application, $x[n]$ are the Nyquist samples of $x_c(t)$ if T is chosen so that the

[†]Once again, we find ourselves using the notation $X(\cdot)$ to denote several distinct things: the CTFT, the Laplace transform, the DTFT, and now the z -transform. This is the price we must pay to avoid overly cluttered notation. Fortunately, the desired form is almost always easily discerned by inspection of the function's argument: ω for the CTFT, s for the Laplace transform, Ω for the DTFT, and z for the z -transform.

bandwidth of $x_c(t)$ equals $1/2T$ Hz. In any case, the relationship between $x[n]$ and the bandlimited interpolation $x_c(t)$ is useful in several applications, such as finding the continuous-time Fourier transform using discrete-time techniques, processing continuous-time signals using discrete-time systems, digital resampling, and so on.

Digital resampling produces a change in sampling rate and is useful or necessary in many applications. Decimation and downsampling reduce the sampling rate, while interpolation and upsampling increase it. Downsampling a signal $x[n]$ by factor M retains every M th sample and deletes the remaining samples, a process that expands the signal's fundamental-band spectrum and may cause a loss of information through aliasing. Such a loss is avoided only if B , the radian bandwidth of $x[n]$, is smaller than π/M . If this condition is not satisfied, we should use a decimator, which precedes downsampling with a DT lowpass filter to eliminate frequency folding and reduce distortion during downsampling. Upsampling a signal $x[n]$ by factor L involves interleaving $L - 1$ zeros between successive samples, a process that compresses the signal's fundamental-band spectrum and produces $L - 1$ images in that band. Interpolation removes these images by following upsampling with a DT lowpass filter. Fractional sampling rate conversion is normally achieved through a cascade of interpolation followed by decimation.

The discrete-time Fourier transform is guaranteed only for absolutely summable signals. Moreover, the DTFT can be used only in the analysis of asymptotically stable systems. These limitations can be overcome by generalizing the frequency variable from ω to $\sigma + j\Omega$. This allows us to use exponentially growing and decaying sinusoids and leads to the z -transform. The z -transform overcomes the limitations of the DTFT and can be used to analyze stable and unstable systems with even exponentially growing input signals. The relationship of the DTFT to the z -transform is similar to that of the Fourier transform to the Laplace transform. Whereas the z -transform is superior to the DTFT in the analysis of LTID systems, the DTFT is preferable in most signal analysis.

References

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
2. Lyons, R. G., *Understanding Digital Signal Processing*, 2nd Ed., Prentice-Hall, Englewood Cliffs, NJ, 2004.

Problems

6.1-1 Show that for a real $x[n]$, Eq. (6.2) can be expressed as

$$x[n] = \frac{1}{\pi} \int_0^\pi |X(\Omega)| \cos[\Omega n + \angle X(\Omega)] d\Omega.$$

This is the trigonometric form of the DTFT.

6.1-2 As shown in Eq. (4.25), a signal $x[n]$ can be expressed as a sum of its even and odd components as

$$x[n] = x_e[n] + x_o[n].$$

(a) If $x[n] \Leftrightarrow X(\Omega)$, show for real $x[n]$ that

$$x_e[n] \Leftrightarrow \operatorname{Re}\{X(\Omega)\}$$

and

$$x_o[n] \Leftrightarrow j \operatorname{Im}\{X(\Omega)\}.$$

(b) Verify the results of part (a) by finding the DTFT of the even and odd components of the signal $(0.8)^n u[n]$.

6.1-3 Using the definition of Eq. (6.1), determine the DTFTs of the signals shown in Fig. P6.1-3.

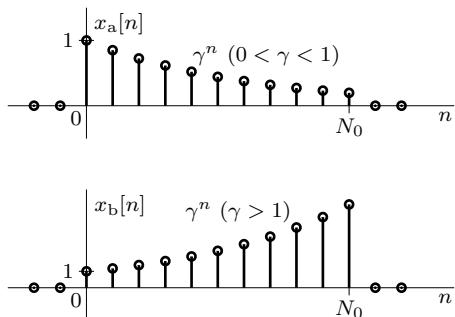


Figure P6.1-3

6.1-4 For the following signals, assume $|\gamma| < 1$ and find the DTFT directly using Eq. (6.1). For each case, plot the signal and its magnitude and phase spectra (when needed, use $\gamma = 0.8$).

(a) $x_a[n] = \delta[n]$

(b) $x_b[n] = \delta[n - k]$

(c) $x_c[n] = \gamma^n u[n - 1]$

(d) $x_d[n] = \gamma^n u[n + 1]$

(e) $x_e[n] = (-\gamma)^n u[n]$

(f) $x_f[n] = \gamma^{|n|}$

6.1-5 Using the definition of Eq. (6.1), determine the DTFTs of the signals shown in Fig. P6.1-5.

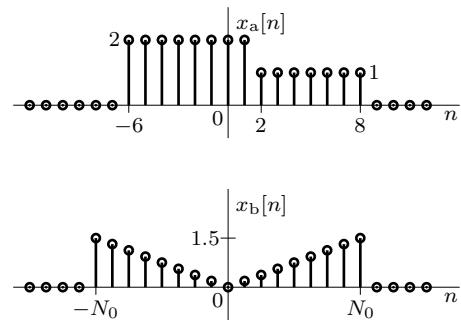


Figure P6.1-5

6.1-6 Using the definition of Eq. (6.2), determine the IDTFTs of the (real) spectra shown in Fig. P6.1-6.

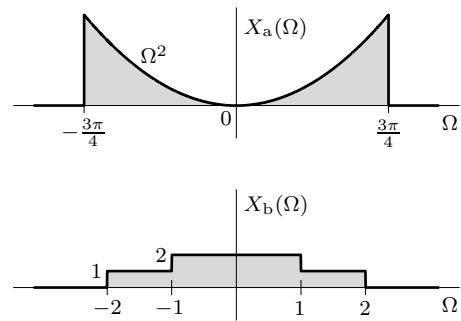


Figure P6.1-6

6.1-7 Using Eq. (6.2), find the IDTFTs for the following spectra, described over the fundamental band $|\Omega| \leq \pi$:

(a) $X_a(\Omega) = e^{jk\Omega}$, integer k

(b) $X_b(\Omega) = \cos(k\Omega)$, integer k

(c) $X_c(\Omega) = \cos^2(\Omega/2)$

(d) $X_d(\Omega) = \Lambda\left(\frac{\Omega}{2B}\right)$

(e) $X_e(\Omega) = 2\pi\delta(\Omega - \Omega_0)$

(f) $X_f(\Omega) = \pi [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)]$

Hint: Use the fact that in the sense of *generalized functions*,

$$\int_{-\infty}^{\infty} e^{jxy} dx = 2\pi\delta(y).$$

- 6.1-8** Using the definition of Eq. (6.2), determine the IDTFTs of the (real) spectra shown in Fig. P6.1-8.

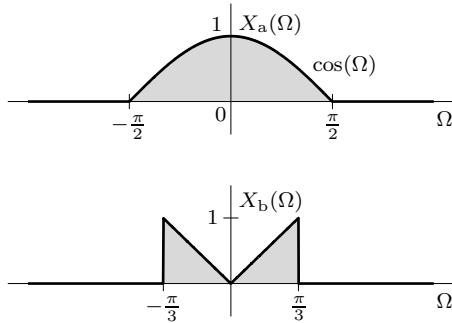


Figure P6.1-8

- 6.1-9** Determine the DTFTs of the signals shown in Fig. P6.1-9.

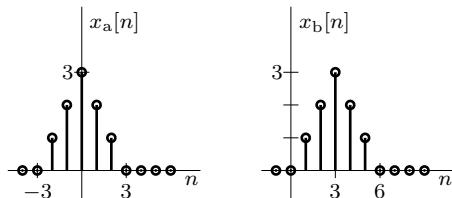


Figure P6.1-9

- 6.1-10** Derive pair 4 from pairs 2 and 3 in Table 6.1.

- 6.1-11** Derive pairs 9 and 14 in Table 6.1 using the technique of Ex. 6.5.

- 6.1-12** (a) Can you derive pairs 2, 3, 4, 5, 6, 7, 11, 15, and 16 in Table 6.1 using the technique used in Ex. 6.5? Explain your answer with reason(s).

- (b) In Ex. 6.5, we derived pair 8 of Table 6.1 for the case $0 < B \leq \pi$. Derive the DTFT of $\frac{B}{\pi}\text{sinc}(Bn/\pi)$ for $\pi < B \leq 2\pi$. Sketch this DTFT.

- 6.1-13** The DTFT of a certain signal $x[n]$ is described over the fundamental band by $\Pi\left(\frac{\Omega - \Omega_0}{\pi}\right)$. Using Eq. (6.2), show that the signal $x[n] = 0.5 \text{sinc}(n/2) e^{j\Omega_0 n}$ if $\Omega_0 \leq \pi/2$.

- 6.1-14** Find the inverse DTFT of $X(\Omega)$ shown in Fig. P6.1-14, assuming $\Omega_1 \leq \pi$. Compare this DTFT pair with that shown in Fig. 6.15. How do the frequency-domain signals $X(\Omega)$ and $H(\Omega)$ compare? How do the time-domain signals $x[n]$ and $h[n]$ compare?

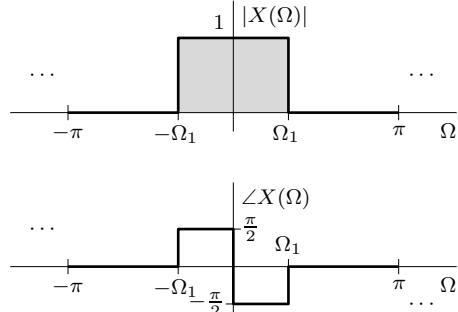


Figure P6.1-14

- 6.1-15** Determine whether the following frequency-domain signals are valid DTFTs. Justify your answer.

- (a) $X_a(\Omega) = \Omega + \pi$
- (b) $X_b(\Omega) = j + \pi$
- (c) $X_c(\Omega) = \sin(10\Omega)$
- (d) $X_d(\Omega) = \sin(\Omega/10)$
- (e) $X_e(\Omega) = \delta(\Omega)$

- 6.2-1** Using only pairs 2 and 5 of Table 6.1 and the time-shifting property of Eq. (6.16), find the DTFTs of the following signals, assuming that m and k are integers and $|\gamma| < 1$:

- (a) $x_a[n] = u[n] - u[n - 9]$
- (b) $x_b[n] = \gamma^{n-m} u[n - m]$
- (c) $x_c[n] = a\gamma^{n-3} (u[n] - u[n - 10])$
- (d) $x_d[n] = \gamma^{n-m} u[n]$
- (e) $x_e[n] = \gamma^n u[n - m]$
- (f) $x_f[n] = (n - m)\gamma^{n-m} u[n - m]$
- (g) $x_g[n] = n^2 \gamma^n u[n]$
- (h) $x_h[n] = (n - k)\gamma^{2n} u[n - m]$
- (i) $x_i[n] = (n - m)\gamma^n u[n]$
- (j) $x_j[n] = n\gamma^{n-k} u[n - m]$

6.2-2 Consider the triangular pulse $x[n]$, which is shown in Fig. P6.2-2.

- (a) Show that the DTFT of $x[n]$ is represented as

$$X(\Omega) = e^{j\Omega} + 2e^{j2\Omega} + 3e^{j3\Omega} + 4e^{j4\Omega}.$$

- (b) Show that the DTFT of the $x[n]$ can be expressed in an alternate form as

$$X(\Omega) = \frac{4e^{j6\Omega} - 5e^{j5\Omega} + e^{j\Omega}}{(e^{j\Omega} - 1)^2}.$$

- (c) Using $X(\Omega)$ from part (a) or part (b) and the time-reversal and time-shifting properties, find the DTFTs of the signals $x_a[n]$, $x_b[n]$, $x_c[n]$, and $x_d[n]$, all of which are shown in Fig. P6.2-2.

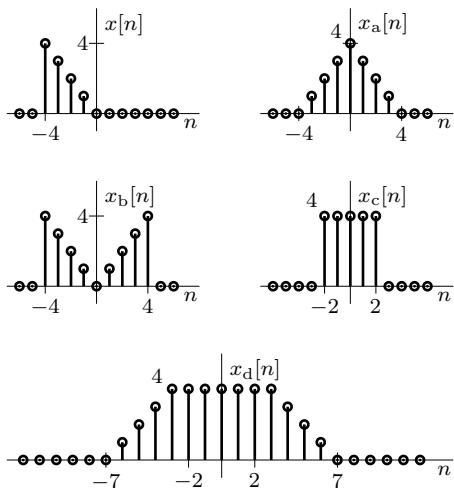


Figure P6.2-2

6.2-3 Using only pair 2 from Table 6.1 and properties of the DTFT, find the DTFTs of the following signals, assuming that $|\gamma| < 1$ and $\Omega_0 < \pi$:

- (a) $x_a = (n + 1)\gamma^n u[n]$
- (b) $x_b = n^2 \gamma^n u[n]$
- (c) $x_c = (n - k)\gamma^{2n} u[n - m]$
- (d) $x_d = \gamma^n \cos(\Omega_0 n) u[n]$
- (e) $x_e = \gamma^n \sin(\Omega_0 n) u[n]$

6.2-4 Using Table 6.1 and the time-shifting property, find the DTFTs of the signals shown in Fig. P6.2-4.

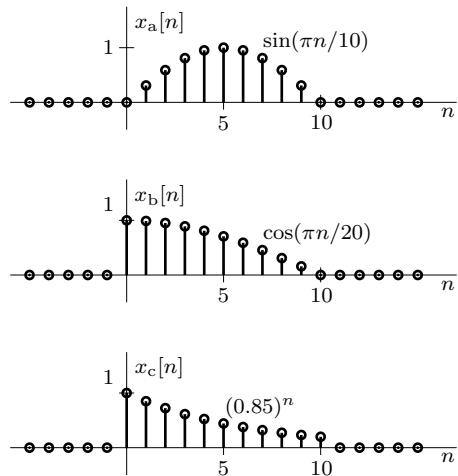


Figure P6.2-4

6.2-5 For the signal $x[n] = 0.5 \operatorname{sinc}^2(n/4)$, find and sketch its DTFT. This signal modulates a carrier $\cos(\Omega_c n)$. Find and sketch the DTFT of the modulated signal $x[n] \cos(\Omega_c n)$ if Ω_c is (a) $\pi/2$, (b) $3\pi/4$, and (c) π .

6.2-6 Using time-shifting property, show that

$$x[n + k] - x[n - k] \iff 2jX(\Omega) \sin(k\Omega).$$

Using this result, find the DTFT of the signal shown in Fig. P6.2-6.

6.2-7 Using the time-shifting property, show that

$$x[n + k] + x[n - k] \iff 2X(\Omega) \cos(k\Omega).$$

Using this result, find the DTFTs of the signals shown in Fig. P6.2-7.

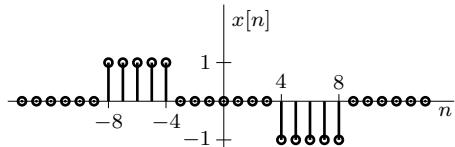


Figure P6.2-6

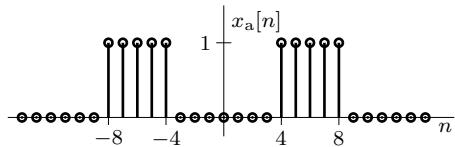


Figure P6.2-7

- 6.2-8** Suppose that $X(\Omega)$ is a spectrum bandlimited to $\pi/2$ rad/sample. Find the spectrum of the signal defined by

$$y[n] = \begin{cases} x[n] & n \text{ even} \\ 0 & n \text{ odd} \end{cases}.$$

Using some hypothetical (bandlimited) spectrum $X(\Omega)$, sketch the spectrum $Y(\Omega)$.

- 6.2-9** Derive the DTFT of $\sin(2\Omega_0 n)$ using the frequency-convolution property and pairs 13 and 14 in Table 6.1 or 6.2. Consider two cases: $0 < \Omega_0 < \pi/2$ and $\pi/2 < \Omega_0 < \pi$.

- 6.2-10** Using pair 1 in Table 6.1 and suitable properties of the DTFT, derive pairs 6, 7, 10, 12, 13, and 14.

- 6.2-11** Using pair 11 in Table 6.1 and suitable properties of the DTFT, derive pairs 10, 12, 13, 14, 15, and 16.

- 6.2-12** Show that periodic convolution $\frac{1}{2\pi} Y(\Omega) \circledast X(\Omega) = X(\Omega)$ if

$$Y(\Omega) = \frac{\sin(5\Omega/2)}{\sin(\Omega/2)} e^{-j2\Omega}$$

and

$$X(\Omega) = \sum_{n=0}^4 c_n e^{-jn\Omega},$$

where c_n are arbitrary constants.

- 6.2-13** From pair 12 of Table 6.1, we obtain $e^{j(\Omega_0/2)n} \iff 2\pi\delta(\Omega - \frac{\Omega_0}{2})$ over the fundamental band. Use this result and the frequency-convolution property to find the DTFT of $e^{j\Omega_0 n}$. To simplify your derivation, assume that $\Omega_0 < \pi/2$.

- 6.2-14** Using only pair 2 in Table 6.1 and the convolution and time-shifting properties, find the inverse DTFT of

$$X(\Omega) = e^{j\Omega} / (e^{j\Omega} - \gamma)^2.$$

- 6.2-15** Let $0 < \Omega_1 < \pi$ and $0 < \Omega_2 < \pi/2$. From the definition and properties of the DTFT, show that

- (a) $\sum_{n=-\infty}^{\infty} \text{sinc}(\Omega_1 n / \pi) = \frac{\pi}{\Omega_1}$
- (b) $\sum_{n=-\infty}^{\infty} (-1)^n \text{sinc}(\Omega_1 n) = 0$
- (c) $\sum_{n=-\infty}^{\infty} \text{sinc}^2(\Omega_2 n / \pi) = \frac{\pi}{\Omega_2}$
- (d) $\sum_{n=-\infty}^{\infty} (-1)^n \text{sinc}^2(\Omega_2 n) = 0$
- (e) $\sum_{n=-\infty}^{\infty} |\text{sinc}(\Omega_2 n)|^4 = 2\pi/3\Omega_2$
- (f) $\int_{-\pi}^{\pi} \frac{\sin(M\Omega/2)}{\sin(\Omega/2)} d\Omega = 2\pi, M \text{ odd}$

- 6.3-1** Using the DTFT method, find the zero-state response $y[n]$ of a causal system with frequency response

$$H(\Omega) = \frac{e^{j\Omega} - 0.5}{(e^{j\Omega} + 0.5)(e^{j\Omega} - 1)}$$

and input

$$x[n] = 3^{-(n+1)} u[n].$$

- 6.3-2** Repeat Prob. 6.3-1 using frequency response

$$H(\Omega) = \frac{e^{j\Omega} + 0.32}{e^{j2\Omega} + e^{j\Omega} + 0.16}$$

and input

$$x[n] = u[n].$$

- 6.3-3** Repeat Prob. 6.3-1 using frequency response

$$H(\Omega) = \frac{e^{j\Omega}}{e^{j\Omega} - 0.5}$$

and input

$$x[n] = (0.8)^n u[n] + 2(2)^n u[-n - 1].$$

- 6.3-4** Over the fundamental band, an LTID system's frequency response is

$$H(\Omega) = \Pi\left(\frac{\Omega}{\pi}\right) e^{-j2\Omega}.$$

Find the system output $y[n]$ for the following inputs:

- (a) $x_a[n] = \text{sinc}(n/2)$
- (b) $x_b[n] = \text{sinc}(n)$
- (c) $x_c[n] = \text{sinc}(9n/4)$
- (d) $x_d[n] = \text{sinc}^2(n/4)$
- (e) $x_e[n] = \text{sinc}^2(n/2)$

- 6.3-5** Find and sketch the magnitude and phase responses for an accumulator system described as

$$y[n] - y[n - 1] = x[n].$$

Determine the responses of this system to the following input sinusoids:

- (a) $x_a[n] = \cos(0.1n)$
- (b) $x_b[n] = \sin(\pi n/2)$
- (c) $x_c(t) = \cos(10^6\pi t)$ sampled at rate $F_s = 1.25$ MHz

- 6.3-6** Find and sketch the magnitude and phase responses of a (noncausal) 5-point moving-average system specified by the equation

$$y[n] = \frac{1}{5} \sum_{k=-2}^2 x[n - k].$$

How can this system be made causal? Plot the magnitude and phase responses of the causal system, and comment on any differences that result.

- 6.3-7** As seen in Ex. 4.9, an analog differentiator can be realized using a backward difference system specified by

$$y[n] = \frac{1}{T}x[n] - \frac{1}{T}x[n - 1].$$

Find and sketch the magnitude and phase responses of this system. Further, find the responses of this system to the following input sinusoids:

- (a) $x_a[n] = \cos(0.1n)$

- (b) $x_b[n] = \sin(\pi n/6)$

- (c) $x_c(t) = \cos(10^6\pi t)$ sampled at rate $F_s = 2$ MHz

- 6.3-8** Repeat Prob. 6.3-7 for an analog integrator that is realized using the trapezoidal approximation (see Ex. 4.10)

$$y[n] - y[n - 1] = \frac{T}{2} (x[n] + x[n - 1]).$$

- 6.3-9** Determine and sketch the magnitude and phase responses for an LTID system specified by the equation

$$y[n + 1] - 0.5y[n] = x[n + 1] + 0.8x[n].$$

Find the system output $y[n]$ for the input $x[n] = \cos(0.5n - \frac{\pi}{3})$.

- 6.3-10** The input-output relationships of two systems, H_1 and H_2 , are described by

$$H_1 : y[n] = -0.9y[n - 1] + x[n]$$

and

$$H_2 : y[n] = 0.9y[n - 1] + x[n].$$

- (a) For each system, find and sketch the magnitude and phase responses. What types of filters (highpass, lowpass, etc.) are H_1 and H_2 ?
- (b) Find the responses of these filters to the sinusoids $x_a[n] = \cos(0.01\pi n)$ and $x_b[n] = \cos(0.99\pi n)$. In general, show that the gain of filter H_1 at Ω_0 is the same as the gain of filter H_2 at $\pi - \Omega_0$; that is, show that $|H_1(\Omega_0)| = |H_2(\pi - \Omega_0)|$.

- 6.3-11** The input-output relationship of a certain filter is described by

$$y[n] - ry[n - 1] = x[n] - \frac{1}{r}x[n - 1],$$

where $|r| < 1$.

- (a) Find and sketch the magnitude and phase response of this system, and state the type (highpass, lowpass, etc.) of this filter.

- (b) Find the responses of this filter to the sinusoids $x_a[n] = \cos(0.01\pi n)$, $x_b[n] = \cos(0.5\pi n)$, and $x_c[n] = \cos(\pi n)$.

6.3-12 An LTID system impulse response is $h[n] = (0.5)^n u[n]$. Find the system output $y[n]$ if the input is the bandpass signal $x[n] = v[n] \cos(3\pi/4)$, where $v[n]$ is a narrowband signal of bandwidth 0.02π . Use the criterion that magnitude-response variations within $\pm 5\%$ and time-delay (group delay) variations within $\pm 3\%$ are considered distortionless.

6.3-13 (a) If $y[n] = (-1)^n x[n]$, then show that

$$Y(\Omega) = X(\Omega \pm \pi).$$

Is there any merit to the claim that this is a spectral inversion system? Explain.

- (b) Sketch the signals $(0.8)^n u[n]$ and $(-0.8)^n u[n]$. Figures 6.3a and 6.3b show the magnitude and phase spectra of $(0.8)^n u[n]$. From these spectra, sketch the magnitude and phase spectra of $(-\gamma)^n u[n]$.
- (c) If $h_{lp}[n]$ is the impulse response of an ideal lowpass filter with frequency response as shown in Fig. 6.14a, sketch the frequency response of a filter whose impulse response is $h[n] = (-1)^n h_{lp}[n]$. What type of filter is this?

6.3-14 A filter with $h_0[n] \Leftrightarrow H_0(\Omega)$ is modified as shown in Fig. P6.3-14. Express the impulse response $h[n]$ and frequency response $H(\Omega)$ of the overall system in terms of $h_0[n]$ and $H_0(\Omega)$, respectively.

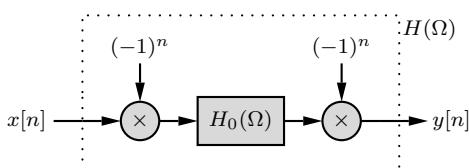


Figure P6.3-14

6.3-15 Consider the signals $x[n]$ and $y[n] = [1 + (-1)^n]x[n]$, where $x[n]$ is assumed to be bandlimited to $|\Omega| \leq \pi/2$. For the sake of illustration, you may take $x[n] = \text{sinc}(n/6)$.

- (a) Sketch the signals $x[n]$ and $y[n]$.
- (b) Sketch the spectra $X(\Omega)$ and $Y(\Omega)$.

(c) Is it possible to recover $x[n]$ from $y[n]$? How?

6.3-16 (a) Consider a lowpass filter specified by a difference equation in the form of Eq. (6.33) with $a_0 = 1$. Show that if coefficients a_k ($k = 1, 2, \dots, K$) are replaced by coefficients $(-1)^k a_k$ and all coefficients b_l ($l = 0, 1, 2, \dots, K$) are replaced by coefficients $(-1)^l b_l$, then the resulting difference equation represents a highpass filter.

(b) Determine the type of filter (highpass, lowpass, etc.) specified by the difference equations

$$y[n] - 0.8y[n-1] = x[n]$$

and

$$y[n] + 0.8y[n-1] = x[n].$$

6.3-17 A general bandpass signal can be expressed as

$$\begin{aligned} x_{bp}[n] &= x_c[n] \cos(\Omega_c n) + x_s[n] \sin(\Omega_c n) \\ &= x[n] \cos(\Omega_c n + \theta[n]), \end{aligned}$$

where $x[n] = \sqrt{x_c^2[n] + x_s^2[n]}$ is the envelope, and $\theta[n] = -\tan^{-1} \frac{x_s[n]}{x_c[n]}$ is the (time-varying) phase of the bandpass signal. Generally, both $x_c[n]$ and $x_s[n]$ are lowpass signals with bandwidth well below Ω_c . Hence, both $x[n]$ and $\theta[n]$ are also lowpass signals. Show that when this signal is passed through a GLP system whose frequency response is shown in Fig. 6.13, the output is also a bandpass signal given by

$$\begin{aligned} y_{bp}[n] &= |a|x[n - n_g] \times \\ &\quad \cos(\Omega_c(n - n_g) + \phi_0 + \theta[n - n_g]). \end{aligned}$$

Note that since the envelope of the signal is merely delayed, the signal transmission is distortionless.

6.3-18 Let $h[n]$ designate the impulse response of the *quadrature system* shown in Fig. P6.3-18. The impulse response of the LTID subsystem $H_0(\Omega)$ is $h_0[n]$.

- (a) Show that the system $h[n]$ is an LTID system, and express $h[n]$ in terms of the impulse response $h_0[n]$. Hint: Find the system response to input $\delta[n - k]$.

- (b) If $H_0(\Omega)$ is an ideal lowpass filter of bandwidth B and is given by $\Pi(\Omega/2B)$, where $\Omega_c + B \leq \pi$, then show that the quadrature system $H(\Omega)$ represents a bandpass filter of bandwidth $2B$ centered at Ω_c .

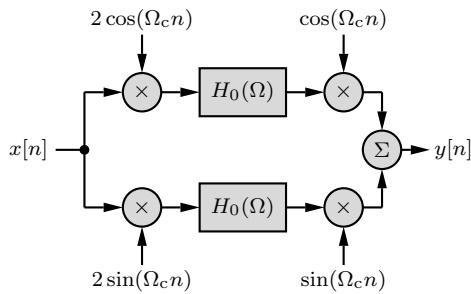


Figure P6.3-18

- 6.3-19** Verify that the MATLAB code in Ex. 6.16 correctly computes the system frequency response $H(\Omega)$. Why does this code require the coefficient vectors A and B to have equal length?

- 6.4-1** Show that the power of a signal $x[n]$ and the power of its bandlimited interpolation $x_c(t)$ (defined in Eq. (6.45)) are identical. Hint: Use the fact that certain sinc functions are orthogonal; that is,

$$\int_{-\infty}^{\infty} \text{sinc}(t-m)\text{sinc}(t-n) dt = \begin{cases} 0 & m \neq n \\ 1 & m = n \end{cases}.$$

- 6.4-2** A bandlimited CT signal $x_c(t)$ cannot also be timelimited, which is to say that a CT signal cannot have finite duration in both the time and frequency domains. Does the same situation hold for DT signals? Hint: Consider sampling the bandlimited CT signal $x_c(t) = \text{sinc}(t/T)$.

- 6.4-3** Determine the DTFT of $x[n] = \sin(\Omega_0 n)$ from the CTFT.

- 6.4-4** The signal $x[n] = u[n] - u[n-10]$ is passed through a system with frequency response given as

$$H(\Omega) = e^{-j5\Omega/2}, \quad |\Omega| \leq \pi.$$

Determine the output $y[n]$ of this system.

- 6.4-5** Repeat Prob. 6.4-6 if the input signal is instead $x[n] = \cos(\pi n/10) + \sin(3\pi n/10)$.

- 6.4-6** A CT signal $x(t)$, bandlimited to 20 kHz, is sampled at 40 kHz to produce

$$x[n] = \begin{cases} 1 & n = -4, -2, 0, 2, 4 \\ -2 & n = -3, -1, 1, 3 \\ 0 & \text{otherwise} \end{cases}.$$

Determine the CTFT $X(\omega)$.

- 6.4-7** Repeat Prob. 6.4-6 if the signal $x(t)$ is instead sampled at 50 kHz.

- 6.4-8** Repeat Ex. 6.18 using a sampling interval of $T = 0.02$.

- 6.5-1** A system that processes analog signals digitally (as in Fig. 6.21a) is designed to act as an ideal lowpass filter with 20-kHz cutoff frequency. Assuming the input is bandlimited to 60 kHz and no anti-aliasing filter is used at the input, find the smallest sampling frequency that allows this system to function properly as a 20-kHz lowpass filter.

Designating the system sampling interval as T , consider two inputs $x_1(t) = u(t) - u(t-T/2)$ and $x_2(t) = x_1(t-T/4)$. Sketch signals and the corresponding spectra at (a) the input of the C/D converter, (b) the input of $H(\Omega)$, (c) the output of $H(\Omega)$, and (d) the output of the D/C converter. Because the input $x_2(t)$ is just a delayed version of signal $x_1(t)$, is the system response to $x_2(t)$ just a delayed version of the response to $x_1(t)$? Explain your answer with reasons.

- 6.5-2** For the differentiator system found in Ex. 6.21, the input is $x_c(t) = \text{sinc}(Bt/\pi)$, where $B = 10000\pi$, and the sampling interval is $T = 10^{-4}$. Sketch the signals and the corresponding spectra at (a) the input of the C/D converter, (b) the input of $H(\Omega)$, (c) the output of $H(\Omega)$, and (d) the output of the D/C converter. Is the system doing what is expected of it? Explain.

- 6.5-3** For the differentiator system found in Ex. 6.21, the input is $x_c(t) = \cos(2000\pi t)$, and the sampling rate is 4 kHz. Sketch the signals and the corresponding spectra at (a) the input of the C/D converter, (b) the input of $H(\Omega)$, (c) the output of $H(\Omega)$, and (d) the output of the D/C converter.

Is the system doing what is expected of it? Explain.

6.5-4 Repeat Prob. 6.5-3 if the sampling rate is 2 kHz. Comment.

6.5-5 Repeat Prob. 6.5-3 if the sampling rate is 1 kHz. Comment.

6.5-6 An input to the ideal differentiator in Ex. 6.21 is $x_c(t) = \Pi(t/\tau)$, where $\tau = 25 \mu\text{s}$, and the sampling interval is $T = 50 \mu\text{s}$.

- (a) If an anti-aliasing filter is not used at the input, find the output $y_c(t)$. Is this output the derivative of the input $x_c(t)$? Explain.
- (b) If an anti-aliasing filter of bandwidth 10 kHz is used at the input, find the output spectrum $Y_c(\Omega)$. Show that the output is the derivative of the input $x_c(t)$ lowpass filtered with the anti-aliasing filter.

6.5-7 Consider the lowpass filter designed in Ex. 6.22 operating at a sampling frequency of $F_s = 50$ kHz. For an input signal $x_c(t) = \text{sinc}(100000t)$, find the spectra of signals at (a) the input of the C/D converter, (b) the input of $H(\Omega)$, (c) the output of $H(\Omega)$, and (d) the output of the D/C converter. Does the output represent the signal $\text{sinc}(100000t)$ lowpass filtered with cutoff frequency 10 kHz? Comment.

6.5-8 We have shown in Ch. 2 that the response of an LTIC system with frequency response $H_c(\omega)$ to an everlasting exponential $e^{j\omega t}$ is given by $H_c(\omega)e^{j\omega t}$. Likewise, Eq. (6.31) shows that the response of an LTID system with frequency response $H(\Omega)$ to an everlasting exponential $e^{j\Omega n}$ is given by $H(\Omega)e^{j\Omega n}$. Using these facts and assuming that the input frequency $\omega \leq \pi/T$, derive Eq. (6.52) (or Eq. (6.53)).

6.6-1 Show that downsampling and upsampling are linear but not time-invariant operations.

6.6-2 The following signals are downsampled by factor M . Find the maximum value of M so that the downsampling operation causes no loss of information.

(a) $x_a[n] = \text{sinc}(n/8)$

(b) $x_b[n] = \delta[n]$

(c) $x_c[n] = \delta[n - 8]$

(d) $x_d[n] = \cos\left(\frac{\pi n}{3}\right)$

(e) $x_e[n] = (0.8)^n u[n]$

(f) $x_f[n] = 1$

Recall that downsampling can be considered lossless if a signal $x[n]$ can be recovered from the downsampled signal $x[Mn]$ through ideal interpolation by factor M .

6.6-3 The cascade system of Fig. P6.6-3 consists of an upsampler followed by a down-sampler, both of factor L . Show that this system is an identity system, which is to say that the output is identical to the input ($y[n] = x[n]$). Prove this in the time domain and also in the frequency domain. Consider a typical signal $x[n]$ and its spectrum $X(\Omega)$. Sketch the signals and the corresponding spectra at the output of the expander and the output of the compressor.

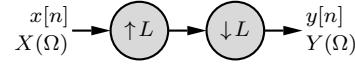


Figure P6.6-3

6.6-4 Show that the two systems in Fig. P6.6-4 are equivalent.

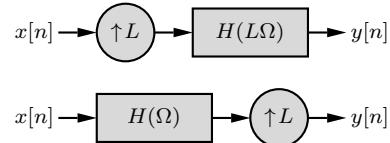


Figure P6.6-4

6.6-5 Show that the two systems in Fig. P6.6-5 are equivalent.

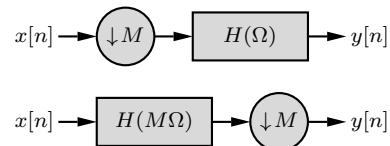


Figure P6.6-5

6.6-6 The cascade system of Fig. P6.6-6 consists of an ideal interpolator followed by a down-sampler, both of factor L . Show that this system is an identity system, which is to say that the output is identical to the input ($y[n] = x[n]$). Prove this in the time domain and also in the frequency domain. Consider a typical signal $x[n]$ and its spectrum $X(\Omega)$. Sketch the signals and the corresponding spectra at the output of the expander, the output of the interpolating filter $H_i(\Omega)$, and the output of the compressor.

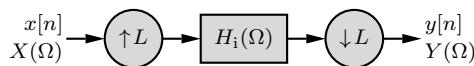


Figure P6.6-6

6.6-7 Find the frequency response of the system illustrated in Fig. P6.6-7.

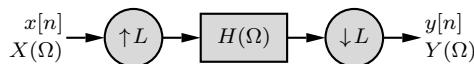


Figure P6.6-7

6.6-8 The cascade system of Fig. P6.6-8 consists of a down sampler followed by an upsampler, both of factor M . Relate the output $y[n]$ to the input $x[n]$, and relate $Y(\Omega)$ to $X(\Omega)$. Consider a typical signal $x[n]$ and its spectrum $X(\Omega)$. Sketch signals and the corresponding spectra at the output of the compressor and the output of the expander. Is the system an identity system? Explain.

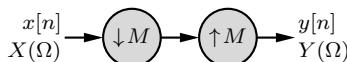


Figure P6.6-8

6.6-9 A system consists of a sampling-rate compressor of factor M followed by an ideal interpolator of factor M . Relate the output spectrum $Y(\Omega)$ of this system to the input spectrum $X(\Omega)$. Under what condition will this be an identity system? Consider a typical signal spectrum $X(\Omega)$. Sketch spectra at the output of the compressor and the output of the interpolator.

6.6-10 Find the output $y[n]$ for the system shown in Fig. P6.6-10. Do the problem in the time domain and the frequency domain.

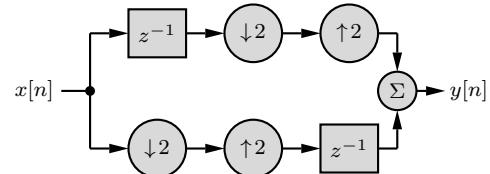


Figure P6.6-10

6.6-11 Equation (6.80) describes the impulse response for a linear interpolation filter. An even simpler method is zero-order hold interpolation, which is characterized by the impulse response

$$\hat{h}_i[n] = u[n] - u[n - L].$$

This is a moving-average filter.

- (a) Using this interpolation filter and $L = 4$, determine and sketch the output $x_i[n]$ of an interpolator (upsampler and filter cascade) in response to the input $x[n] = \sin(2\pi n/7)(u[n] - u[n - 7])$. Compare this with the output found in Ex. 6.27.
- (b) Find the frequency response of the zero-order hold interpolation filter. Compare this response with the ideal as well as the frequency response of a linear interpolation filter, as found in Drill 6.12.
- (c) Show that a cascade of two zero-order hold interpolation filters is identical, within a shift and gain factor, to a linear interpolation filter.

6.6-12 Show that the impulse response of an ideal decimation filter is

$$h_d[n] = \frac{1}{M} \text{sinc}(n/M).$$

Use this result to prove Eq. (6.79).

- (a) For the signal $x[n] = \frac{3}{16} \text{sinc}(3n/16)$, find the $M = 2$ downsampled signal $x_{\downarrow}[n]$. What is the maximum factor M that still permits lossless (no aliasing) downsampling?

- (b) Find the spectrum $X_{\downarrow}(\Omega)$ from $x_{\downarrow}[n]$ found in part (a). Verify the result by finding $X_{\downarrow}(\Omega)$ using Eq. (6.68).
- (c) The downsampled signal $y[n] = x_{\downarrow}[n]$, found in part (a), is ideally interpolated by factor $L = 2$. Find the DTFT of the interpolated signal $y_i[n]$. Verify this result by finding $Y_i(\Omega)$ using Eq. (6.72). Is $y_i[n] = x[n]$? Why or why not?

- 6.6-14** (a) For the signal $x[n] = \frac{3}{4}\text{sinc}(3n/4)$, the $M = 2$ downsampled signal $x_{\downarrow}[n] = \frac{3}{4}\text{sinc}(1.5n)$. How do you interpret this signal? Find an alternate (but equivalent) description of this signal using components that are bandlimited to $\Omega \leq \pi$. Using this alternate description, find and sketch the spectrum $X_{\downarrow}(\Omega)$ of the downsampled signal.
- (b) Use Eq. (6.68) to verify the expression for $X_{\downarrow}(\Omega)$ found in part (a).
- (c) The downsampled signal $y[n] = x_{\downarrow}[n]$, found in part (a), is ideally interpolated by factor $L = 2$. Sketch the spectrum of the interpolated signal $y_i[n]$. Is the interpolated signal equal to $x[n]$? Explain.

- 6.6-15** (a) For the signal $x[n] = n\gamma^n u[n]$, find the $M = 3$ downsampled signal $x_{\downarrow}[n]$. By direct computation, determine the spectrum $X_{\downarrow}(\Omega)$ of the downsampled signal.
- (b) Use Eq. (6.68) to verify the expression for $X_{\downarrow}(\Omega)$ found in part (a).
- (c) The downsampled signal $y[n] = x_{\downarrow}[n]$, found in part (a), is ideally interpolated by factor $L = 3$. Sketch the spectrum of the interpolated signal $y_i[n]$. Is $y_i[n] = x[n]$? Why or why not?

- 6.6-16** (a) For the signal $x[n] = n\gamma^n u[n]$, find the $L = 3$ interpolated signal $x_i[n]$. Determine and sketch the spectrum $X_i(\Omega)$ of the interpolated signal. Assume an ideal interpolation filter.
- (b) The interpolated signal $y[n] = x_i[n]$, found in part (a), is downsampled by

factor $M = 3$. Determine and sketch the spectrum of the downsampled signal $y_{\downarrow}[n]$. Is $y_{\downarrow}[n] = x[n]$? Why or why not?

- 6.6-17** (a) A signal $x_a[n] = 0.3\text{sinc}(0.3n)$ is passed through the discrete-time system shown in Fig. P6.6-17a, where $H_a(\Omega)$ is an ideal lowpass filter of bandwidth $\pi/3$ rad/sample and unit gain. Find the output spectrum $Y_a(\Omega)$.
- (b) A signal $x_b[n] = 0.3\text{sinc}(0.3n)$ is passed through the discrete-time system shown in Fig. P6.6-17b, where $H_b(\Omega)$ is an ideal lowpass filter of bandwidth 0.9π rad/sample and unit gain. Find the output spectrum $Y_b(\Omega)$.
- (c) A signal $x_c[n] = 0.3\text{sinc}(0.3n)$ is passed through the discrete-time system shown in Fig. P6.6-17c, where $H_c(\Omega)$ is an ideal lowpass filter of bandwidth 0.2π rad/sample and unit gain. Find the output spectrum $Y_c(\Omega)$.

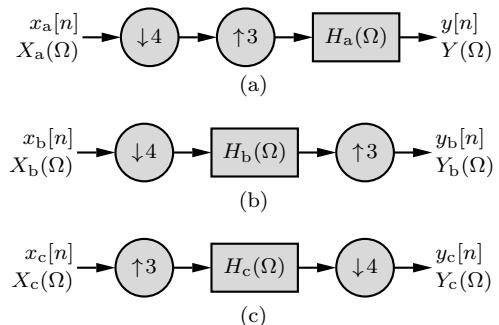


Figure P6.6-17

- 6.6-18** For the system shown in Fig. P6.6-18, $x[n] = 0.3\text{sinc}(0.3n)$ and $T_2 = T_1/\sqrt{2}$. Find the output $y[n]$, assuming ideal D/C and C/D converters.

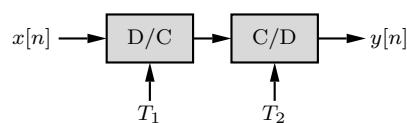


Figure P6.6-18

6.6-19 (a) Show that, in general, the operations of upsampling and downsampling do not commute.

(b) Show that upsampling and downsampling do commute when upsampling factor L and downsampling factor M are coprime, that is, have a greatest common divisor of 1.

6.6-20 A signal $x[n] = \cos(\pi n/6)$ is decimated by factor 3 using the same system of Ex. 6.25. Sketch the decimator output $x_d[n]$.

Chapter 7

Discrete-Time System Analysis Using the z -Transform

The counterpart of the Laplace transform for discrete-time systems is the z -transform. The Laplace transform converts integro-differential equations into algebraic equations. In the same way, the z -transform changes difference equations into algebraic equations, thereby simplifying the analysis of discrete-time systems. The z -transform method of analysis of discrete-time systems parallels the Laplace transform method of analysis of continuous-time systems, with some minor differences. In fact, we shall see in Sec. 7.8 that *the z -transform is the Laplace transform in disguise*.

The behavior of discrete-time systems is, with some differences, similar to that of continuous-time systems. As shown in Sec. 5.5.4, the frequency-domain analysis of discrete-time systems is based on the fact that the response of a linear time-invariant discrete-time (LTID) system to an everlasting exponential z^n is the same exponential within a multiplicative constant, given by $H(z)z^n$. More generally, when an input $x[n]$ is a sum of (everlasting) exponentials of the form z^n , the system response to $x[n]$ is easily found as a sum of the system's responses to all these exponential components. The tool that allows us to represent an arbitrary input $x[n]$ as a sum of everlasting exponentials (complex frequencies) of the form z^n is the z -transform.

7.1 The z -Transform

Much like the Laplace transform, the z -transform comes in two flavors: bilateral and unilateral. We begin with the bilateral case and then treat the unilateral case second.

7.1.1 The Bilateral z -Transform

Just as the Laplace transform can be derived from the Fourier transform by generalizing the frequency variable from $j\omega$ to $\sigma + j\omega$, the z -transform can be obtained from the discrete-time Fourier transform by generalizing the frequency variable from $j\Omega$ to $\sigma + j\Omega$. In this way, as shown in Sec. 6.7, the bilateral z -transform and its inverse are given as

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \quad (7.1)$$

and[†]

$$x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz. \quad (7.2)$$

[†]In Eq. (7.2), the contour of integration must be chosen in the ROC of $X(z)$.

Here, $z = e^{\sigma+j\Omega}$ is the complex frequency variable, and \oint indicates integration in a counterclockwise direction around a closed path in the complex plane (see Fig. 6.36). As in the case of the Laplace transform, we need not worry about this integral at this point because the inverse z -transforms of many signals of engineering interest are readily found using z -transform tables. The direct and inverse z -transforms can be expressed symbolically as

$$X(z) = \mathcal{Z}\{x[n]\} \quad \text{and} \quad x[n] = \mathcal{Z}^{-1}\{X(z)\}$$

or simply as

$$x[n] \xleftrightarrow{\mathcal{Z}} X(z).$$

Further, note that

$$\mathcal{Z}^{-1}\{\mathcal{Z}\{x[n]\}\} = x[n] \quad \text{and} \quad \mathcal{Z}\{\mathcal{Z}^{-1}\{X(z)\}\} = X(z).$$

To foster greater intuition, it is important to keep in mind the stories that Eqs. (7.1) and (7.2) tell us about the signal $x[n]$. In short, Eq. (7.2) suggests that a signal $x[n]$ can be constructed (synthesized) from a sum (integral) of everlasting complex exponentials z^n , each weighted by some complex scale factor $X(z)$.[†] Equation (7.1) compares (analyzes) the signal $x[n]$ with the exponentials z^n to determine the scale factors $X(z)$. Using this perspective, Eqs. (7.1) and (7.2) are often referred as the *analysis* and *synthesis* equations of the bilateral z -transform. Similar to the Laplace transform, the limits of summation are from $-\infty$ to ∞ for the bilateral case and from 0 to ∞ for the unilateral case, to be discussed later.

The Region of Convergence

The sum in Eq. (7.1) defining the direct z -transform $X(z)$ may not converge (exist) for all values of z . The values of z (the region in the complex plane) for which the sum in Eq. (7.1) converges (or exists) is called the *region of existence* or, more commonly, the *region of convergence* (ROC) for $X(z)$. This concept will become clear in the following example.

► **Example 7.1 (Bilateral z -Transform of a Causal Signal)**

Find the z -transform and the corresponding ROC for the causal signal $x[n] = \gamma^n u[n]$.

By definition

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}.$$

The signal $x[n] = \gamma^n u[n]$ is a causal exponential, as depicted in Fig. 7.1a for the case $\gamma = 0.85$. Since $u[n] = 1$ for all $n \geq 0$,

$$X(z) = \sum_{n=0}^{\infty} \left(\frac{\gamma}{z}\right)^n = 1 + \left(\frac{\gamma}{z}\right) + \left(\frac{\gamma}{z}\right)^2 + \left(\frac{\gamma}{z}\right)^3 + \dots \quad (7.3)$$

It is helpful to remember the following well-known geometric progression and its sum:

$$1 + r + r^2 + r^3 + \dots = \frac{1}{1 - r} \quad \text{if } |r| < 1. \quad (7.4)$$

Applying Eq. (7.4) to Eq. (7.3) yields

$$X(z) = \frac{1}{1 - \frac{\gamma}{z}} = \frac{z}{z - \gamma}, \quad \text{ROC: } \left|\frac{\gamma}{z}\right| < 1 \quad \text{or} \quad |z| > |\gamma|.$$

[†]More precisely, the scale factor is actually $\frac{1}{2\pi j z} X(z)$.

Observe that $X(z)$ exists only for $|z| > |\gamma|$. For $|z| < |\gamma|$, the sum in Eq. (7.3) does not converge but rather goes to infinity. Therefore, the ROC of $X(z)$ is the region in the z -plane outside the circle of radius $|\gamma|$ centered at the origin, as shown shaded in Fig. 7.1b. Since $z = \gamma$ is the sole pole of $X(z)$, it is not particularly surprising that $|z| = |\gamma|$ forms the boundary for the region of convergence. By their very nature, poles cause nonconvergent behavior. In fact, *poles are never found within the region of convergence*.

In Ex. 7.2, we show that the z -transform of the signal $-\gamma^n u[-n - 1]$ is $z/(z - \gamma)$ just like $X(z)$, although the ROC is $|z| < |\gamma|$ rather than $|z| > |\gamma|$. Clearly, it is not possible to determine the original time-domain function unless the ROC is specified. It is necessary to specify the ROC to avoid ambiguity and to ensure unique z -transform pairs.

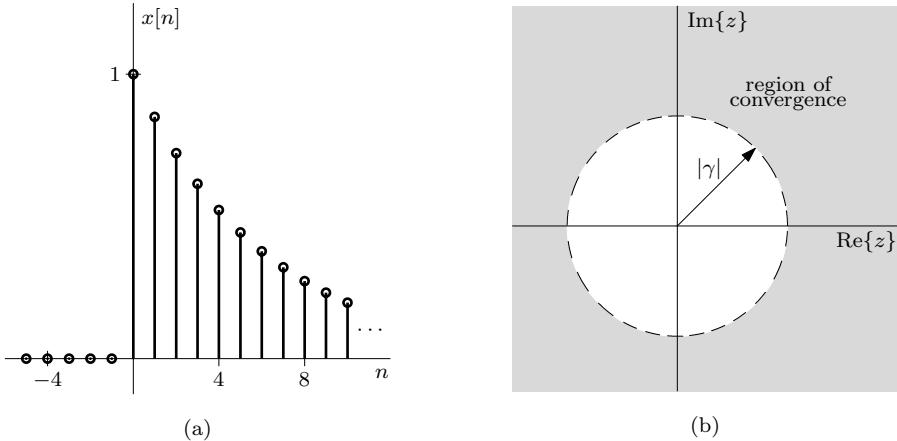


Figure 7.1: (a) $x[n] = \gamma^n u[n]$ and (b) the region of convergence $|z| > |\gamma|$ for $X(z)$.

Example 7.1 ◀

▷ Example 7.2 (Bilateral z -Transform of an Anti-Causal Signal)

Find the z -transform and the corresponding ROC for the anti-causal signal $y[n] = -\gamma^n u[-n - 1]$.

The signal $y[n] = -\gamma^n u[-n - 1]$ is an anti-causal exponential, as depicted in Fig. 7.2a for the case $\gamma = 0.85$. Since $u[-n - 1] = 1$ for all $n \leq -1$, Eq. (7.1) yields

$$\begin{aligned} Y(z) &= \sum_{n=-\infty}^{-1} -\left(\frac{\gamma}{z}\right)^n \\ &= -\left(\frac{z}{\gamma}\right) - \left(\frac{z}{\gamma}\right)^2 - \left(\frac{z}{\gamma}\right)^3 - \dots \\ &= 1 - \left[1 + \left(\frac{z}{\gamma}\right) + \left(\frac{z}{\gamma}\right)^2 + \left(\frac{z}{\gamma}\right)^3 + \dots\right]. \end{aligned} \quad (7.5)$$

Applying Eq. (7.4) to Eq. (7.5) yields

$$Y(z) = 1 - \frac{1}{1 - \frac{z}{\gamma}} = \frac{z}{z - \gamma}, \quad \text{ROC: } \left|\frac{z}{\gamma}\right| < 1 \quad \text{or} \quad |z| < |\gamma|.$$

Comparing $Y(z)$ to $X(z)$ from Ex. 7.1, we see that the z -transforms of $\gamma^n u[n]$ and $-\gamma^n u[-n - 1]$ both equal $z/(z - \gamma)$. The regions of convergence, however, are different. In the former case, $X(z)$

converges for $|z| > |\gamma|$; in the latter, $Y(z)$ converges for $|z| < |\gamma|$ (see Fig. 7.2b). To be complete and allow a unique inverse transform, the z -transform must specify the ROC.

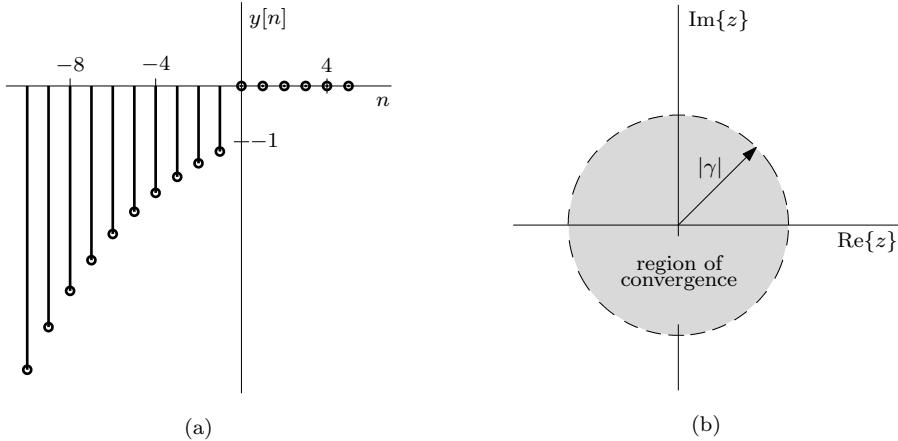


Figure 7.2: (a) $y[n] = -\gamma^n u[-n - 1]$ and (b) the region of convergence $|z| < |\gamma|$ for $Y(z)$.

Example 7.2 ◀

Lock and Key

Without its key, even the finest lock is useless. Similarly, a bilateral z -transform $X(z)$ is useless without its region of convergence. The ROC is the key that allows us to unlock $X(z)$ and find its corresponding signal $x(t)$. To be useful, every bilateral z -transform $X(z)$ must include an appropriate region of convergence.

Existence of the Bilateral z -Transform

To exist, the z -transform must possess a (nonempty) region of convergence. To understand the conditions required to ensure a region of convergence, let us first consider a causal signal $x[n] = x[n]u[n]$, such as found in Ex. 7.1. By definition,

$$X(z) = \sum_{n=0}^{\infty} x[n]z^{-n} = \sum_{n=0}^{\infty} \frac{x[n]}{z^n}.$$

In this case, the existence of the z -transform is guaranteed if, for some $|z|$,

$$|X(z)| \leq \sum_{n=0}^{\infty} \frac{|x[n]|}{|z|^n} < \infty.$$

Any signal $x[n]$ that grows no faster than an exponential signal r^n , for some real and positive r , satisfies this condition. Thus, if

$$|x[n]| \leq r^n \quad \text{for some real and positive } r,$$

then

$$|X(z)| \leq \sum_{n=0}^{\infty} \left(\frac{r}{|z|} \right)^n = \frac{1}{1 - \frac{r}{|z|}} < \infty \quad \text{for } |z| > r.$$

That is, $X(z)$ exists since we are guaranteed a region of convergence that includes $|z| > r$.

Similar arguments hold for anti-causal and other infinite-duration signals. The important point is that for the z -transform to exist in these cases, a signal cannot grow at a rate faster than some exponential signal r^n . Almost all practical signals satisfy this requirement and are therefore z -transformable. Some signal models (e.g., γ^{n^2}) grow faster than any choice of exponential signal r^n and are therefore not z -transformable. Fortunately, such signals are of little practical or theoretical interest. Still, as we shall next see, even such signals are z -transformable when restricted to having finite duration (length), which is generally the case in practice.

Region of Convergence for Finite-Length Sequences

A finite-length sequence $x[n]$ is a sequence that is zero outside some interval $N_1 \leq n \leq N_2$, where both N_1 and N_2 are finite integers. For a finite-length sequence with bounded amplitude, the ROC is the entire z -plane, except possibly $z = 0$ or ∞ . In other words, the z -transform of such a sequence converges for any value of z except possibly $z = 0$ or ∞ . If $x[n]$ is nonzero for some negative n , then $|x[n]|/|z^n| \rightarrow \infty$ for $z = \infty$. In contrast, if $x[n]$ is nonzero for some positive n , then $|x[n]|/|z^n| \rightarrow \infty$ for $z = 0$. Except these two values of z , $|x[n]|/|z^n|$ is finite for any value of z .

Linearity of the Bilateral z -Transform

Although Sec. 7.3 covers the properties of the z -transform in detail, we shall need to make use of one property, the linearity property, to complete our discussion of the region of convergence. Like the Laplace transform, the z -transform is a linear operator. Thus, if

$$x[n] \xrightleftharpoons{z} X(z) \text{ with ROC } R_x \quad \text{and} \quad y[n] \xrightleftharpoons{z} Y(z) \text{ with ROC } R_y,$$

then

$$ax[n] + by[n] \xrightleftharpoons{z} aX(z) + bY(z) \text{ with ROC at least } R_x \cap R_y. \quad (7.6)$$

The proof is trivial and follows from the definition of the z -transform. This result can be extended to finite sums.

Further Observations of the Region of Convergence

Extending the linearity property, if $x[n] = \sum_{k=1}^K x_k[n]$, then the ROC for $X(z)$ is (at least) the intersection of the ROCs (region common to all ROCs) for the transforms $X_1(z), X_2(z), \dots, X_K(z)$. This result leads to the conclusion similar to that for the Laplace transform that the ROC for a causal sequence extends outward from the origin-centered circle that lies on the largest finite-magnitude pole of $X(z)$. The ROC for an anti-causal sequence extends inward from the origin-centered circle that lies on the smallest nonzero-magnitude pole of $X(z)$.

These conclusions are readily generalized to right- and left-sided sequences. A right-sided sequence, which is zero for $n < N_1$ for some finite integer N_1 , can be expressed as a sum of a causal sequence and a finite-length sequence. Thus, the ROC for a right-sided sequence begins outside the origin-centered circle that lies on the largest finite-magnitude pole of $X(z)$ and extends to, but possibly excludes, $z = \infty$. Only if $x[n]$ is nonzero for some negative n is $z = \infty$ excluded.

Similarly, a left-sided sequence, which is zero for $n > N_2$ for some finite integer N_2 , can be expressed as a sum of an anti-causal sequence and a finite-length sequence. The ROC for a left-sided sequence begins inside the origin-centered circle that lies on the smallest nonzero-magnitude pole of $X(z)$ and extends to, but possibly excludes, $z = 0$. Only if $x[n]$ is nonzero for some positive n is $z = 0$ excluded.

A two-sided sequence that is of infinite duration in both directions can be expressed as a sum of a right-sided sequence and a left-sided sequence. If there is overlap between the individual ROCs of these two pieces, the ROC of the two-sided sequence will take an annular shape. Even if the

z -transforms of two component pieces exist, the z -transform of the combined two-sided sequence does not exist if there is no overlap between the individual ROCs.

▷ **Example 7.3 (Bilateral z -Transform of a Two-Sided Signal)**

Determine the z -transform of

$$w[n] = \underbrace{(0.9)^n u[n]}_{x[n]} + \underbrace{(1.2)^n u[-n-1]}_{y[n]}.$$

What is the z -transform of $w[n]$ if $x[n]$ is changed to $2^n u[n]$?

Using the results of Exs 7.1 and 7.2, we have

$$X(z) = \frac{z}{z - 0.9} \text{ with ROC } |z| > 0.9$$

and

$$Y(z) = \frac{-z}{z - 1.2} \text{ with ROC } |z| < 1.2.$$

The common region where both $X(z)$ and $Y(z)$ converge is $0.9 < |z| < 1.2$. Hence,

$$\begin{aligned} W(z) &= X(z) + Y(z) \\ &= \frac{z}{z - 0.9} + \frac{-z}{z - 1.2} \\ &= \frac{-0.3z}{(z - 0.9)(z - 1.2)}, \quad \text{ROC: } 0.9 < |z| < 1.2. \end{aligned}$$

Figure 7.3 depicts the sequence $w[n]$ and the annular ROC of $W(z)$.

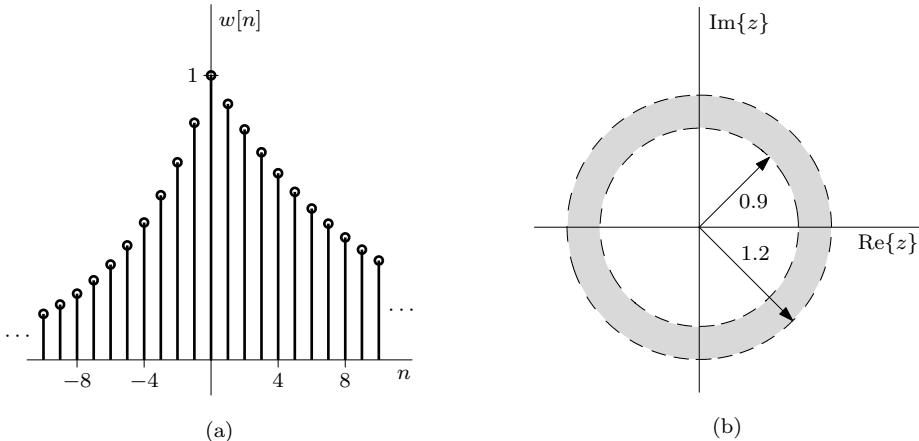


Figure 7.3: (a) $w[n] = (0.9)^n u[n] + (1.2)^n u[-n-1]$ and (b) the ROC $0.9 < |z| < 1.2$ for $W(z)$.

When $x[n] = 2^n u[n]$ rather than $(0.9)^n u[n]$, the ROC of $X(z)$ becomes $|z| > 2$. In this case, $R_x \cap R_y = \emptyset$, and the z -transform of $w[n]$ does not exist.

Example 7.3 ◀

7.1.2 The Unilateral z -Transform

Whether it is discrete-time or continuous-time, we most often deal with causal signals in practice. When we observe the restriction of causality, the z -transform is called the *unilateral z -transform* and is given as

$$X(z) = \sum_{n=0}^{\infty} x[n] z^{-n}. \quad (7.7)$$

The only difference between Eq. (7.7) and bilateral case of Eq. (7.1) is the lower limit of summation. Quite simply, the unilateral z -transform is just a special case of the bilateral z -transform. Although the type of z -transform is normally easy to infer, to help avoid ambiguity, the unilateral z -transform is sometimes denoted symbolically as

$$X(z) = \mathcal{Z}_u \{x[n]\}$$

or simply as

$$x[n] \xrightarrow{\mathcal{Z}_u} X(z).$$

Since the unilateral z -transform is only used with causal signals, its region of convergence can be inferred and thus need not be specified. The inverse of the unilateral z -transform does not differ from that of the bilateral case and is thus represented by Eq. (7.2). As in the case of the unilateral Laplace transform, there are two primary reasons to use the unilateral z -transform. First, the unilateral z -transform can be more convenient than the bilateral case, particularly since there is no need to specify any ROCs. Second, the unilateral transform better facilitates the analysis of linear constant-coefficient difference equations that possess nonzero initial conditions. This will become more clear in Sec. 7.3, which covers the properties of the z -transform.

▷ Example 7.4 (Unilateral z -Transforms)

Using the definition of Eq. (7.7), determine the unilateral z -transforms of (a) $x_a[n] = \delta[n]$, (b) $x_b[n] = u[n]$, (c) $x_c[n] = \cos(\beta n)u[n]$, and (d) $x_d[n] = u[n] - u[n - 5]$.

From Eq. (7.7),

$$X(z) = \sum_{n=0}^{\infty} x[n] z^{-n} = x[0] + \frac{x[1]}{z} + \frac{x[2]}{z^2} + \frac{x[3]}{z^3} + \dots. \quad (7.8)$$

(a)

For $x_a[n] = \delta[n]$, $x_a[0] = 1$ and $x_a[2] = x_a[3] = x_a[4] = \dots = 0$. Therefore,

$$x_a[n] = \delta[n] \xrightarrow{\mathcal{Z}_u} X_a(z) = 1, \quad \text{ROC: all } z.$$

(b)

For $x_b[n] = u[n]$, $x_b[0] = x_b[1] = x_b[2] = \dots = 1$. Therefore,

$$X_b(z) = 1 + \frac{1}{z} + \frac{1}{z^2} + \frac{1}{z^3} + \dots.$$

Applying Eq. (7.4), it follows that

$$X_b(z) = \frac{1}{1 - \frac{1}{z}} = \frac{z}{z - 1}, \quad \text{ROC: } \left| \frac{1}{z} \right| < 1 \quad \text{or} \quad |z| > 1.$$

(c)

Recall that $\cos(\beta n) = (e^{j\beta n} + e^{-j\beta n})/2$. Moreover, as demonstrated in Ex. (7.1),

$$e^{\pm j\beta n} u[n] \xrightarrow{\mathcal{Z}} \frac{z}{z - e^{\pm j\beta}}, \quad \text{ROC: } |z| > |e^{\pm j\beta}| = 1.$$

Therefore,

$$\begin{aligned} X_c(z) &= \frac{1}{2} \left(\frac{z}{z - e^{j\beta}} + \frac{z}{z - e^{-j\beta}} \right) \\ &= \frac{z[z - \cos(\beta)]}{z^2 - 2\cos(\beta)z + 1}, \quad \text{ROC: } |z| > 1. \end{aligned}$$

(d)

Here, $x_d[0] = x_d[1] = x_d[2] = x_d[3] = x_d[4] = 1$ and $x_d[5] = x_d[6] = \dots = 0$. Therefore,

$$\begin{aligned} X_d(z) &= 1 + \frac{1}{z} + \frac{1}{z^2} + \frac{1}{z^3} + \frac{1}{z^4} \\ &= \frac{z^4 + z^3 + z^2 + z + 1}{z^4}, \quad \text{ROC: all } z \neq 0. \end{aligned}$$

We can also express this result in a more compact form by utilizing the geometric progression sum. Using entry 1 of Table 5.1 on page 290, we obtain

$$X_d(z) = \frac{\left(\frac{1}{z}\right)^0 - \left(\frac{1}{z}\right)^5}{1 - \frac{1}{z}} = \frac{z}{z-1}(1 - z^{-5}), \quad \text{ROC: all } z \neq 0.$$

Example 7.4 \triangleleft

Table 7.1 gives a short table of selected bilateral z -transform pairs, including their ROCs. Entries 1–11, which involve causal signals $x[n]$, double as a table of unilateral transforms, in which case specification of the ROC is unnecessary. Entry 12 is either causal or anti-causal depending on k . Entries 13–15 involve anti-causal signals and therefore do not possess a unilateral z -transform. Most practical applications involve only causal signals and can therefore be treated with the unilateral z -transform, entries 1–11 (no ROC needed), and entry 12 with $k > 0$.

▷ Drill 7.1 (Unilateral z -Transforms)

- (a) Determine the z -transform of the signal $x_a[n] = u[n - 4] - u[n - 10]$ shown in Fig. 7.4.
- (b) Use linearity and pair 10 of Table 7.1 to determine the z transform of

$$x_b[n] = 20.65(\sqrt{2})^n \cos\left(\frac{\pi}{4}n - 1.415\right) u[n].$$

\triangleleft

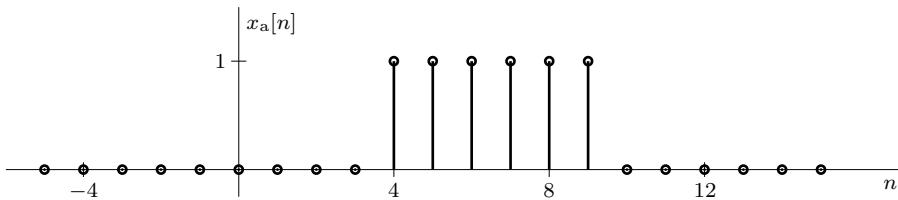


Figure 7.4: Finite-length signal $x_a[n] = u[n - 4] - u[n - 10]$.

Relationship between $h[n]$ and $H(z)$

If $h[n]$ is the unit impulse response for an LTID system, then the system transfer function is defined by Eq. (5.31) as

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n}. \quad (7.9)$$

$x[n]$	$X(z)$	ROC
1. $\delta[n]$	1	All z
2. $u[n]$	$\frac{z}{z-1}$	$ z > 1$
3. $\gamma^n u[n]$	$\frac{z}{z-\gamma}$	$ z > \gamma $
4. $\gamma^{n-1} u[n-1]$	$\frac{1}{z-\gamma}$	$ z > \gamma $
5. $n\gamma^n u[n]$	$\frac{\gamma z}{(z-\gamma)^2}$	$ z > \gamma $
6. $n^2 \gamma^n u[n]$	$\frac{\gamma z(z+\gamma)}{(z-\gamma)^3}$	$ z > \gamma $
7. $\frac{n!}{(n-m)!m!} \gamma^{n-m} u[n]$	$\frac{z}{(z-\gamma)^{m+1}}$	$ z > \gamma $
8. $ \gamma ^n \cos(\beta n) u[n]$	$\frac{z[z- \gamma \cos(\beta)]}{z^2-2 \gamma \cos(\beta)z+ \gamma ^2}$	$ z > \gamma $
9. $ \gamma ^n \sin(\beta n) u[n]$	$\frac{z \gamma \sin(\beta)}{z^2-2 \gamma \cos(\beta)z+ \gamma ^2}$	$ z > \gamma $
10. $ \gamma ^n \cos(\beta n + \theta) u[n]$	$\frac{z[z \cos(\theta)- \gamma \cos(\beta-\theta)]}{z^2-2 \gamma \cos(\beta)z+ \gamma ^2}$ $= \frac{(0.5e^{j\theta})z}{z- \gamma e^{j\beta}} + \frac{(0.5e^{-j\theta})z}{z- \gamma e^{-j\beta}}$	$ z > \gamma $
11. $r \gamma ^n \cos(\beta n + \theta) u[n]$	$\frac{z(az+b)}{z^2+2cz+ \gamma ^2}$ $r = \sqrt{\frac{a^2 \gamma ^2+b^2-2abc}{ \gamma ^2-c^2}}$ $\beta = \cos^{-1}\left(\frac{-c}{ \gamma }\right)$ $\theta = \tan^{-1}\left(\frac{ac-b}{a\sqrt{ \gamma ^2-c^2}}\right)$	$ z > \gamma $
12. $\delta[n-k]$	z^{-k}	$ z > 0 \quad k > 0$ $ z < \infty \quad k < 0$
13. $-u[-n-1]$	$\frac{z}{z-1}$	$ z < 1$
14. $-\gamma^n u[-n-1]$	$\frac{z}{z-\gamma}$	$ z < \gamma $
15. $-n\gamma^n u[-n-1]$	$\frac{z\gamma}{(z-\gamma)^2}$	$ z < \gamma $

Table 7.1: Selected bilateral z -transform pairs.

For causal systems, the limits on the sum are from $n = 0$ to ∞ . Comparing this equation to Eq. (7.1) shows that the transfer function $H(z)$ is the z -transform of the impulse response $h[n]$ of an LTID system; that is,

$$h[n] \xleftrightarrow{z} H(z) \text{ with ROC } R_h. \quad (7.10)$$

This important result relates the time-domain specification $h[n]$ of a system to the transform-domain specification $H(z)$. The result is parallel to that for LTIC systems.

Connection between the DTFT and the z -Transform

Since we use the same notation $X(\cdot)$ for both the z -transform and the DTFT, in this subsection we shall subscript all z -transforms with a “ z ” to help avoid confusion between the two. Thus, the

DTFT of signal $x[n]$ is denoted $X(\Omega)$, and following Eq. (7.1), the z -transform is written as

$$X_z(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}.$$

Setting $z = e^{j\Omega}$ in this equation yields

$$X_z(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}. \quad (7.11)$$

As we can see by referring to Eq. (6.1), the right-hand-side sum of Eq. (7.11) defines $X(\Omega)$, the DTFT of $x[n]$. Does this mean that the DTFT can be obtained from the corresponding z -transform by setting $z = e^{j\Omega}$? In other words, is it true that $X_z(e^{j\Omega}) = X(\Omega)$? In most cases, the answer is yes. For example, when $x[n] = \gamma^n u[n]$, its z -transform is $z/(z - \gamma)$, and $X_z(e^{j\Omega}) = e^{j\Omega}/(e^{j\Omega} - \gamma)$, which is equal to $X(\Omega)$ (assuming $|\gamma| < 1$). However, for the unit step function $u[n]$, the z -transform is $z/(z - 1)$, and $X_z(e^{j\Omega}) = e^{j\Omega}/(e^{j\Omega} - 1)$. As seen from pair 11 of Table 6.1 (page 340), this is not equal to the DTFT of $u[n]$.

We obtain $X_z(e^{j\Omega})$ by setting $z = e^{j\Omega}$ in Eq. (7.11). This implies that the sum on the right-hand side of Eq. (7.11) converges for $z = e^{j\Omega}$, which means the unit circle (characterized by $z = e^{j\Omega}$) lies in the ROC for $X_z(z)$. Hence, setting $z = e^{j\Omega}$ in $X_z(z)$ yields the DTFT $X(\Omega)$ only when the ROC for $X(z)$ includes the unit circle. This condition holds for all $x[n]$ that are absolutely summable. If the ROC of $X_z(z)$ excludes the unit circle, $X_z(e^{j\Omega}) \neq X(\Omega)$. This applies to all non-decaying $x[n]$. The reason for this peculiar behavior has something to do with the nature of convergence of the z -transform and the DTFT.[†]

The connection between the z -transform and the DTFT also applies to system functions $H_z(e^{j\Omega})$ and $H(\Omega)$. Recall that for a BIBO or asymptotically stable LTID system, the impulse response $h[n]$ is absolutely summable. Hence, for a BIBO stable system, the DTFT of $h[n]$ is identical to the system's transfer function $H_z(z)$ with $z = e^{j\Omega}$, which is to say that

$$H_z(e^{j\Omega}) = H(\Omega) \text{ for BIBO or asymptotically stable systems.}$$

7.2 The Inverse z -Transform

The ROC, either explicitly stated as in the case of the bilateral z -transform or possibly inferred as in the case of the unilateral z -transform, is required to determine $x[n]$ from $X(z)$. For example, the integral in Eq. (7.2) is a contour integral whose path of integration is counterclockwise along a closed path that lies within the ROC.[‡] Without an ROC, it is not possible to locate a suitable path of integration to evaluate Eq. (7.2).

As it requires a background in complex variable theory, we often avoid the integration of Eq. (7.2) and determine inverse z -transforms through the use of tables, such as Table 7.1. Take, for example, $X(z) = z/(z - 1)$. If the ROC is $|z| > 1$, then the inverse is $x[n] = u[n]$ (entry 2). If, however, the ROC is $|z| < 1$, then the inverse is $x[n] = -u[-n - 1]$ (entry 13). Without an ROC, however, it is impossible to know whether the inverse is $u[n]$ or $-u[-n - 1]$. The ROC ensures that $x[n]$ and $X(z)$ form a unique transform pair.

[†]To further explain this point, consider the unit step function $u[n]$ and its transforms. Both the z -transform and the DTFT synthesize $x[n]$ using everlasting exponentials of the form z^n . For the z -transform, the value of z has the freedom to be anywhere outside the unit circle ($|z| > 1$), but it is restricted to the unit circle ($z = e^{j\Omega}$) in the case of the DTFT. This restriction necessitates a more complex DTFT spectrum (extra terms) to synthesize $u[n]$ than does the relatively simple z -transform spectrum $X(z) = z/(z - 1)$. In contrast, when $x[n]$ is absolutely summable, the region of convergence for the z -transform includes the unit circle, and both transforms can synthesize $x[n]$ by using z along the unit circle. This leads to $X_z(e^{j\Omega}) = X(\Omega)$.

[‡]Typically, the path is chosen as a circle that is centered at the origin. However, a closed path of arbitrary shape, as long as it is counterclockwise and fully contained within the ROC, yields the same result, namely, $x[n]$.

Many of the transforms $X(z)$ of practical interest are rational functions (ratios of polynomials in z), which can be expressed as a sum of partial fractions, whose inverse transforms can be readily found using even short tables of transform pairs. Basic methods for computing partial fraction expansions, such as the method of clearing fractions and the method of residues, are covered in most introductory signals texts, such as [1], and are not covered in detail here. Rather, we demonstrate the approach through a series of examples.

▷ **Example 7.5 (Inverse Unilateral z -Transform by Partial Fraction Expansion)**

Using partial fraction expansions and Table 7.1, determine the inverse unilateral z -transforms of

$$(a) \quad X_a(z) = \frac{8z-19}{(z-2)(z-3)} \quad (b) \quad X_b(z) = \frac{z(2z^2-11z+12)}{(z-1)(z-2)^3} \quad (c) \quad X_c(z) = \frac{2z(3z+17)}{(z-1)(z^2-6z+25)}$$

(a) Simple Poles

Although hand calculations are easy in this case, MATLAB provides a convenient mechanism to expand $X(z)$ into partial fractions.

```
01 [r,p,k] = residue([8 -19],poly([2 3]))
r = 5   3
p = 3   2
k = []
```

Here, the `poly` command converts the roots at $z = 2$ and $z = 3$ into the expanded polynomial coefficient vector required by the `residue` command. Vector `r` provides the residues (fraction coefficients), vector `p` provides the roots, and `k` provides the direct terms, if any. Thus, the partial fraction expansion of $X_a(z)$ is

$$X_a(z) = \frac{3}{z-2} + \frac{5}{z-3}.$$

Since $X(z)$ is a unilateral z -transform, the ROC is implied as outside the circle $|z| > 3$. From pair 4 of Table 7.1, we obtain

$$x_a[n] = [3(2)^{n-1} + 5(3)^{n-1}] u[n-1]. \quad (7.12)$$

Notice that when we expand a rational (unilateral) transform $X(z)$ into partial fractions directly, we shall always obtain an answer that is multiplied by $u[n-1]$. Such a form is rather unnatural as well as inconvenient. We prefer a form that contains $u[n]$ rather than $u[n-1]$. A glance at Table 7.1 shows that the z -transform of every signal that is multiplied by $u[n]$ has a factor z in the numerator. This observation suggests that we expand $X(z)$ into *modified partial fractions*, where each term has a factor z in the numerator. This goal can be accomplished by expanding $X(z)/z$ into partial fractions and then multiplying both sides by z . To demonstrate, we first use MATLAB to expand $X_a(z)/z = \frac{8z-19}{z(z-2)(z-3)}$.

```
02 format rat; [r,p,k] = residue([8 -19],poly([0 2 3]))
r = 5/3  3/2  -19/6
p = 3   2   0
k = []
```

Thus,

$$\frac{X_a(z)}{z} = \frac{(-19/6)}{z} + \frac{(3/2)}{z-2} + \frac{(5/3)}{z-3}.$$

Multiplying both sides by z yields the modified partial fraction expansion of

$$X_a(z) = -\frac{19}{6} + \frac{3}{2} \left(\frac{z}{z-2} \right) + \frac{5}{3} \left(\frac{z}{z-3} \right).$$

From pairs 1 and 3 in Table 7.1, it follows that

$$x_a[n] = -\frac{19}{6} \delta[n] + \left[\frac{3}{2}(2)^n + \frac{5}{3}(3)^n \right] u[n].$$

The reader can verify that this answer is equivalent to that in Eq. (7.12) by computing $x[n]$ in both cases for $n = 0, 1, 2, 3, \dots$ and then comparing the results.

(b) Repeated Poles

To provide some review, we compute the modified partial fraction expansion by hand and then later verify the result using MATLAB. In this case,

$$X_b(z) = \frac{z(2z^2 - 11z + 12)}{(z-1)(z-2)^3}$$

and

$$\frac{X_b(z)}{z} = \frac{2z^2 - 11z + 12}{(z-1)(z-2)^3} = \frac{k}{z-1} + \frac{a_0}{(z-2)^3} + \frac{a_1}{(z-2)^2} + \frac{a_2}{(z-2)}.$$

Using the Heaviside “cover-up” method, we obtain

$$k = \left. \frac{2z^2 - 11z + 12}{(z-1)(z-2)^3} \right|_{z=1} = -3$$

and

$$a_0 = \left. \frac{2z^2 - 11z + 12}{(z-1)(z-2)^3} \right|_{z=2} = -2.$$

Therefore,

$$\frac{X_b(z)}{z} = \frac{-3}{z-1} - \frac{2}{(z-2)^3} + \frac{a_1}{(z-2)^2} + \frac{a_2}{(z-2)}. \quad (7.13)$$

We can determine a_1 and a_2 by the method of residues, clearing fractions, or we may employ any of various short cuts. For example, to determine a_2 , we multiply both sides of Eq. (7.13) by z and let $z \rightarrow \infty$. This yields

$$0 = -3 - 0 + 0 + a_2 \implies a_2 = 3.$$

This result leaves only one unknown, a_1 , which is readily determined by letting z take any convenient value, say, $z = 0$, on both sides of Eq. (7.13). This step yields

$$\frac{12}{8} = 3 + \frac{1}{4} + \frac{a_1}{4} - \frac{3}{2} \implies a_1 = -1.$$

MATLAB confirms these results with substantially less effort.

```
03 [r,p,k] = residue([2 -11 12],poly([1 2 2 2]))
r = 3  -1  -2  -3
p = 2  2  2  1
k = []
```

Taking the results together,

$$\frac{X_b(z)}{z} = \frac{-3}{z-1} - \frac{2}{(z-2)^3} - \frac{1}{(z-2)^2} + \frac{3}{z-2}$$

and

$$X_b(z) = -3\frac{z}{z-1} - 2\frac{z}{(z-2)^3} - \frac{z}{(z-2)^2} + 3\frac{z}{z-2}.$$

Recognizing that the result must be causal, pairs 3 and 7 of Table 7.1 yield

$$\begin{aligned} x_b[n] &= \left[-3 - 2\frac{n(n-1)}{8}(2)^n - \frac{n}{2}(2)^n + 3(2)^n \right] u[n] \\ &= - \left[3 + \frac{1}{4}(n^2 + n - 12)2^n \right] u[n]. \end{aligned}$$

(c) Complex Poles

To begin, we first determine the unknown roots of $X_c(z)$.

```
04 format short; roots([1 -6 25])
ans = 3.0000+4.0000i 3.0000-4.0000i
```

Thus,

$$X_c(z) = \frac{2z(3z + 17)}{(z - 1)(z^2 - 6z + 25)} = \frac{2z(3z + 17)}{(z - 1)(z - 3 - j4)(z - 3 + j4)}.$$

The poles of $X_c(z)$ are 1, $3 + j4$, and $3 - j4$. Whenever there are complex-conjugate poles, the problem can be worked out in two ways. In the first method, we expand $X_c(z)$ into (modified) first-order partial fractions. In the second method, rather than obtain one factor corresponding to each complex-conjugate pole, we obtain quadratic factors corresponding to each pair of complex-conjugate poles. These two approaches are considered in turn.

Method of First-Order Factors

Note that

$$\frac{X_c(z)}{z} = \frac{2(3z + 17)}{(z - 1)(z - 3 - j4)(z - 3 + j4)} = \frac{k_1}{z - 1} + \frac{k_2}{z - 3 - j4} + \frac{k_2^*}{z - 3 + j4}.$$

Next, we use MATLAB to determine the partial fraction expansion of $X_c(z)/z$. It is likewise possible to perform the expansion by hand using the Heaviside “cover-up” method or other techniques.

```
05 [r,p,k] = residue(2*[3 17],poly([1 3+1j*4 3-1j*4]))
r = -1.0000-1.2500i -1.0000+1.2500i 2.0000
p = 3.0000+4.0000i 3.0000-4.0000i 1.0000
k = []
```

Using these results,

$$\frac{X_c(z)}{z} = \frac{2}{z - 1} + \frac{-1 - j1.25}{z - 3 - j4} + \frac{-1 + j1.25}{z - 3 + j4}$$

and

$$X_c(z) = \frac{2z}{z - 1} + \frac{(-1 - j1.25)z}{z - 3 - j4} + \frac{(-1 + j1.25)z}{z - 3 + j4}.$$

Since the result must be causal, the inverse transform of the first term on the right-hand side is $2u[n]$. The inverse transform of the remaining two terms (complex-conjugate poles) can be obtained by scaling pair 10 (Table 7.1) by $r = 2| - 1 - j1.25 | = 3.2016$ and identifying $|\gamma| = |3 + j4| = 5$, $\beta = \angle(3 + j4) = 0.9273$, and $\theta = \angle(-1 - j1.25) = -2.2455$. Therefore,

$$x_c[n] = [2 + 3.2016(5)^n \cos(0.9273n - 2.2455)] u[n]. \quad (7.14)$$

Method of Quadratic Factors

In this method, complex-conjugate poles are kept together as

$$\frac{X_c(z)}{z} = \frac{2(3z + 17)}{(z - 1)(z^2 - 6z + 25)} = \frac{2}{z - 1} + \frac{Az + B}{z^2 - 6z + 25}.$$

MATLAB does not provide a built-in function to find the coefficients for quadratic factors, so the coefficients A and B need to be determined by hand. Multiplying both sides by z and letting $z \rightarrow \infty$, we find

$$0 = 2 + A \implies A = -2$$

and

$$\frac{X_c(z)}{z} = \frac{2(3z + 17)}{(z - 1)(z^2 - 6z + 25)} = \frac{2}{z - 1} + \frac{-2z + B}{z^2 - 6z + 25}.$$

To find B , we let z take any convenient value, say, $z = 0$. This step yields

$$\frac{-34}{25} = -2 + \frac{B}{25} \implies B = 16.$$

Therefore,

$$\frac{X_c(z)}{z} = \frac{2}{z-1} + \frac{-2z+16}{z^2-6z+25}$$

and

$$X_c(z) = \frac{2z}{z-1} + \frac{z(-2z+16)}{z^2-6z+25}.$$

From pair 11 of Table 7.1, where $a = -2$, $b = 16$, $c = -3$, and $|\gamma| = 5$, we compute

$$r = \sqrt{\frac{100+256-192}{25-9}} = 3.2016, \quad \beta = \cos^{-1}\left(\frac{3}{5}\right) = 0.9273, \quad \text{and} \quad \theta = \tan^{-1}\left(\frac{-10}{-8}\right) = -2.2455.$$

As expected, these values produce a result that is identical to Eq. (7.14),

$$x_c[n] = [2 + 3.2016(5)^n \cos(0.9273n - 2.2455)] u[n].$$

Example 7.5 □

▷ Example 7.6 (Inverse Bilateral z -Transform by Partial Fraction Expansion)

Using partial fraction expansions and Table 7.1, determine the inverse bilateral z -transform of

$$X(z) = \frac{-z(z+0.4)}{(z-0.8)(z-2)}$$

if the ROC is (a) $|z| > 2$, (b) $|z| < 0.8$, and (c) $0.8 < |z| < 2$.

Using MATLAB, we first expand $X(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([-1 -0.4],poly([0.8 2]))
r = -2 1
p = 2 0.8
k = []
```

Thus,

$$\frac{X(z)}{z} = \frac{-(z+0.4)}{(z-0.8)(z-2)} = \frac{1}{z-0.8} - \frac{2}{z-2}$$

and

$$X(z) = \frac{z}{z-0.8} - 2\frac{z}{z-2}.$$

This single expansion is used to invert $X(z)$ regardless of the ROC.

(a) Causal Components

Since the ROC is $|z| > 2$, both terms correspond to causal sequences. Using pair 3 of Table 7.1, the inverse transform is thus

$$x_a[n] = [(0.8)^n - 2(2)^n] u[n].$$

This sequence is shown in Fig. 7.5a.

(b) Anti-Causal Components

In this case, $|z| < 0.8$, which is less than the magnitudes of both poles. Hence, both terms correspond to anti-causal sequences, and pair 14 of Table 7.1 yields

$$x_b[n] = [-(0.8)^n + 2(2)^n] u[-n - 1].$$

This sequence is shown in Fig. 7.5b.

(c) Causal and Anti-Causal Components

In this case, $0.8 < |z| < 2$. The part of $X(z)$ corresponding to the pole at 0.8 is a causal sequence, and the part corresponding to the pole at 2 is an anti-causal sequence. Using pairs 3 and 14 of Table 7.1, the inverse transform is thus

$$x_c[n] = (0.8)^n u[n] + 2(2)^n u[-n - 1].$$

This sequence is shown in Fig. 7.5c.

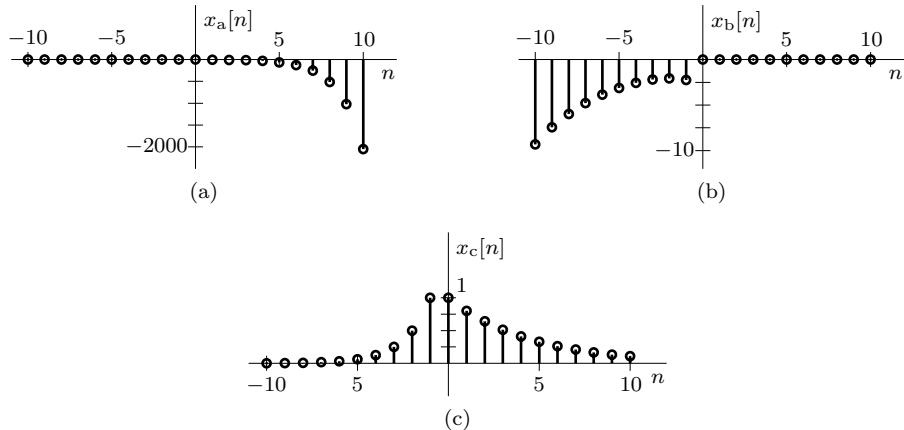


Figure 7.5: Inverse transforms of $X(z)$ given ROCs: (a) $|z| > 2$, (b) $|z| < 0.8$, and (c) $0.8 < |z| < 2$.

Example 7.6 \triangleleft
▷ Drill 7.2 (Inverse Unilateral z -Transform by Partial Fraction Expansion)

Find the inverse unilateral z -transforms of

$$\begin{array}{ll} \text{(a)} & X_a(z) = \frac{z(2z-1)}{(z-1)(z+0.5)} \\ \text{(c)} & X_c(z) = \frac{9}{(z+2)(z-0.5)^2} \end{array} \quad \begin{array}{ll} \text{(b)} & X_b(z) = \frac{1}{(z-1)(z+0.5)} \\ \text{(d)} & X_d(z) = \frac{5z(z-1)}{z^2-1.6z+0.8} \end{array}$$

 \triangleleft

▷ **Drill 7.3 (Impulse Response by Inverse z -Transform)**

By using Eq. (5.33) and taking the inverse z -transform of $H(z)$, find the impulse response $h[n]$ of the following causal LTID systems from Drill 5.4:

- (a) $y[n+2] - 5y[n+1] + 6y[n] = 8x[n+1] - 19x[n]$
- (b) $y[n+2] - 4y[n+1] + 4y[n] = 2x[n+2] - 2x[n+1]$
- (c) $y[n] = x[n] - 2x[n-1]$

△

▷ **Drill 7.4 (Inverse Bilateral z -Transform by Partial Fraction Expansion)**

Find the inverse bilateral z -transform of

$$X(z) = \frac{z}{z^2 + \frac{5}{6}z + \frac{1}{6}}, \quad \text{ROC: } \frac{1}{3} < |z| < \frac{1}{2}.$$

△

7.2.1 Inverse z -Transform by Power Series Expansion

Although most transforms of interest are rational functions and thus efficiently inverted using partial fraction expansion techniques, it is also possible to invert the transforms of causal and anti-causal sequences using power series expansions. This technique is particularly useful in cases where the transform $X(z)$ is not a rational function or if only the first few terms of $x[n]$ are needed.

Let us consider a causal signal $x[n] = x[n]u[n]$ first. By definition,

$$\begin{aligned} X(z) &= \sum_{n=0}^{\infty} x[n]z^{-n} \\ &= x[0] + \frac{x[1]}{z} + \frac{x[2]}{z^2} + \frac{x[3]}{z^3} + \dots \\ &= x[0]z^0 + x[1]z^{-1} + x[2]z^{-2} + x[3]z^{-3} + \dots \end{aligned}$$

This result is a power series in z^{-1} . Therefore, if we can expand the transform $X(z)$ of a causal signal $x[n]$ into a power series in z^{-1} , the coefficients of this power series can be identified as $x[0], x[1], x[2], x[3]$, and so on. In many cases, a suitable expansion is obtained using a Taylor or Maclaurin series of $X(z)$. Further, a rational $X(z)$ can be expanded into a power series in z^{-1} through polynomial long division. For example, consider a causal signal with transform given by

$$X(z) = \frac{z^2(7z-2)}{(z-0.2)(z-0.5)(z-1)} = \frac{7z^3 - 2z^2}{z^3 - 1.7z^2 + 0.8z - 0.1}.$$

To obtain a series expansion in powers of z^{-1} , we divide the numerator by the denominator as

$$\begin{array}{r} 7 + 9.9z^{-1} + 11.23z^{-2} + 11.87z^{-3} + \dots \\ \hline z^3 - 1.7z^2 + 0.8z - 0.1 \left(\begin{array}{r} 7z^3 - 2z^2 \\ 7z^3 - 11.9z^2 + 5.60z - 0.7 \\ \hline 9.9z^2 - 5.60z + 0.7 \\ 9.9z^2 - 16.83z + 7.92 - 0.99z^{-1} \\ \hline 11.23z - 7.22 + 0.99z^{-1} \\ 11.23z - 19.09 + 8.98z^{-1} + \dots \\ \hline 11.87 - 7.99z^{-1} + \dots \\ \vdots \end{array} \right) \end{array} .$$

Thus,

$$X(z) = \frac{z^2(7z - 2)}{(z - 0.2)(z - 0.5)(z - 1)} = 7 + 9.9z^{-1} + 11.23z^{-2} + 11.87z^{-3} + \dots$$

and

$$x[0] = 7, x[1] = 9.9, x[2] = 11.23, x[3] = 11.87, \dots$$

Although this procedure yields $x[n]$ directly, it suffers from the disadvantage that it generally does not produce a simple closed-form solution.

The treatment of an anti-causal signal $x[n] = x[n]u[-n - 1]$ is similar. By definition,

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{-1} x[n]z^{-n} \\ &= x[-1]z + x[-2]z^2 + x[-3]z^3 + x[-4]z^4 + \dots \end{aligned}$$

This result is a power series in z . Therefore, if we can expand the transform $X(z)$ of an anti-causal signal $x[n]$ into a power series in z , the coefficients of this power series can be identified as $x[-1]$, $x[-2]$, $x[-3]$, $x[-4]$, and so on. When $X(z)$ is rational, we can find the inverse z -transform by dividing the numerator polynomial by the denominator polynomial, both in ascending powers of z , to obtain a polynomial in ascending powers of z . Thus, to find the inverse transform of $z/(z - 0.5)$ (when the ROC is $|z| < 0.5$), we divide z by $-0.5 + z$ to obtain $-2z - 4z^2 - 8z^3 - \dots$. Hence, $x[-1] = -2$, $x[-2] = -4$, $x[-3] = -8$, and so on.

▷ Example 7.7 (Inverse z -Transform by Power Series Expansion)

Using a power series expansion, determine the anti-causal signal $x[n]$ that has z -transform

$$X(z) = ze^z, \quad \text{ROC: } |z| < \infty.$$

Recall that the Taylor series of function $f(x)$ using expansion point a is given as

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x - a)^k, \quad (7.15)$$

where $f^{(k)}(a)$ denotes the k th derivative of $f(x)$ evaluated at $x = a$. A Maclaurin series is a special case of a Taylor series with $a = 0$.

To expand $X(z)$, we first compute the Maclaurin series of e^z . Since all derivatives of e^z evaluated at $z = 0$ are 1, the Maclaurin series is

$$e^z = \sum_{k=0}^{\infty} \frac{z^k}{k!} = 1 + \frac{z}{1!} + \frac{z^2}{2!} + \frac{z^3}{3!} + \dots$$

Thus,

$$X(z) = ze^z = \sum_{k=0}^{\infty} \frac{z^{k+1}}{k!} = z + \frac{z^2}{1!} + \frac{z^3}{2!} + \frac{z^4}{3!} + \dots$$

Using pair 12 of Table 7.1, the (anti-causal) time-domain signal is thus

$$x[n] = \sum_{k=0}^{\infty} \frac{\delta[n+k+1]}{k!} = \delta[n+1] + \frac{\delta[n+2]}{1!} + \frac{\delta[n+3]}{2!} + \frac{\delta[n+4]}{3!} + \dots$$

As shown in Fig. 7.6, the rapid decay of $x[n]$ makes it effectively timelimited, which helps explain why the ROC of $X(z)$ includes the entire z -plane except $|z| = \infty$.

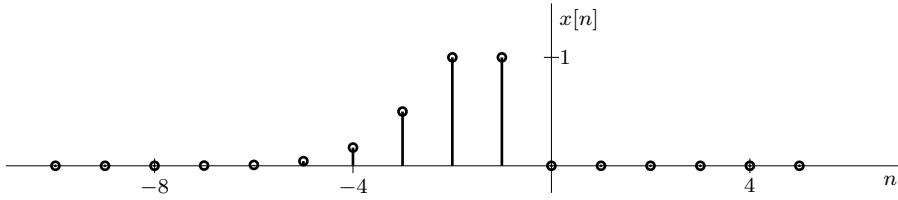


Figure 7.6: Anti-causal signal $x[n]$ obtained from Maclaurin series expansion of $X(z) = ze^z$.

Example 7.7 \triangleleft

▷ **Drill 7.5 (Inverse z -Transform by Power Series Expansion)**

Using polynomial long division, determine the inverse z -transform of $X(z) = z/(z - 0.5)$ if the ROC is (a) $|z| > 0.5$ and (b) $|z| < 0.5$.

\triangleleft

7.3 Properties of the z -Transform

A variety of properties are useful in the derivation and computation of many z -transforms and in the solution of linear difference equations with constant coefficients. Here we consider a selection of the most important properties of both the bilateral and unilateral z -transforms.

In our discussion, the variable n appearing in signals, such as $x[n]$, $y[n]$, etc., may or may not stand for time. However, in most applications of our interest, n is proportional to time. For this reason, we shall loosely refer to variable n as time.

7.3.1 Linearity Property

As shown in Sec. 7.1.1 with Eq. (7.6), the bilateral z -transform is a linear operator. This result also holds for the unilateral z -transform, although the ROC need not be specified. Thus, if

$$x[n] \xrightarrow{\mathcal{Z}_u} X(z) \quad \text{and} \quad y[n] \xrightarrow{\mathcal{Z}_u} Y(z),$$

then

$$ax[n] + by[n] \xrightarrow{\mathcal{Z}_u} aX(z) + bY(z). \quad (7.16)$$

The proof is again trivial and follows directly from the definition of the unilateral z -transform.

7.3.2 Complex-Conjugation Property

The complex-conjugation property is identical in form to the complex-conjugation property of the Laplace transform, and it is the same for both the bilateral and unilateral z -transforms. In particular,

$$\text{if } x[n] \xrightarrow{\mathcal{Z}} X(z) \text{ with ROC } R_x, \text{ then } x^*[n] \xrightarrow{\mathcal{Z}} X^*(z^*) \text{ with ROC } R_x. \quad (7.17)$$

The proof of this property is straightforward,

$$\mathcal{Z}\{x^*[n]\} = \left[\left(\sum_{n=-\infty}^{\infty} x^*[n]z^{-n} \right)^* \right]^* = \left[\sum_{n=-\infty}^{\infty} x[n](z^*)^{-n} \right]^* = X^*(z^*).$$

7.3.3 Time Scaling and the Time-Reversal Property

Much as we saw with the DTFT in Sec. 6.2.4, it is tricky to represent a general time-scaling property for the z -transform. The primary reason for this difficulty is that a DT time-scaling operation almost always alters the fundamental nature of the time-domain signal. It follows that the transform of such a radically altered signal will significantly differ from the original signal's transform. Two exceptions, however, are the cases of simple upsampling and time reversal.

If a signal $x[n]$ is upsampled by factor L (meaning that $L - 1$ zeros are inserted between each sample of $x[n]$), then the z -transform of the upsampled signal $x_{\uparrow}[n]$ can be represented as a simple function of $X(z)$, the z -transform of $x[n]$. Specifically,

$$\text{if } x[n] \xleftrightarrow{Z} X(z), \text{ then } x_{\uparrow}[n] \xleftrightarrow{Z} X(z^L). \quad (7.18)$$

This result applies to both the unilateral and bilateral cases of the z -transform, although the ROC in the bilateral case must be modified by changing z to z^L . We have already derived this result for the DTFT (see Eq. (6.74)), and the Prob. 7.3-2 proof of Eq. (7.18) is similar.

For the bilateral z -transform, the time-reversal property states that[†]

$$\text{if } x[n] \xleftrightarrow{Z} X(z) \text{ with ROC } R_x, \text{ then } x[-n] \xleftrightarrow{Z} X(1/z) \text{ with ROC } 1/R_x. \quad (7.19)$$

The inverted region of convergence $1/R_x$ is obtained from R_x by replacing z with $1/z$. For example, if R_x is $|\gamma_1| < |z| < |\gamma_2|$, then $1/R_x$ is given by $|\gamma_1| < |1/z| < |\gamma_2|$ or, more simply, by $1/|\gamma_2| < |z| < 1/|\gamma_1|$. To prove this property, note that

$$\mathcal{Z}\{x[-n]\} = \sum_{n=-\infty}^{\infty} x[-n]z^{-n}.$$

Applying the change of variable $k = -n$ yields

$$\mathcal{Z}\{x[-n]\} = \sum_{k=-\infty}^{\infty} x[k]z^k = \sum_{k=-\infty}^{\infty} x[k](1/z)^{-k} = X(1/z).$$

▷ Drill 7.6 (Time-Reversal Property)

Using the time-reversal property and pair 2 in Table 7.1, show that

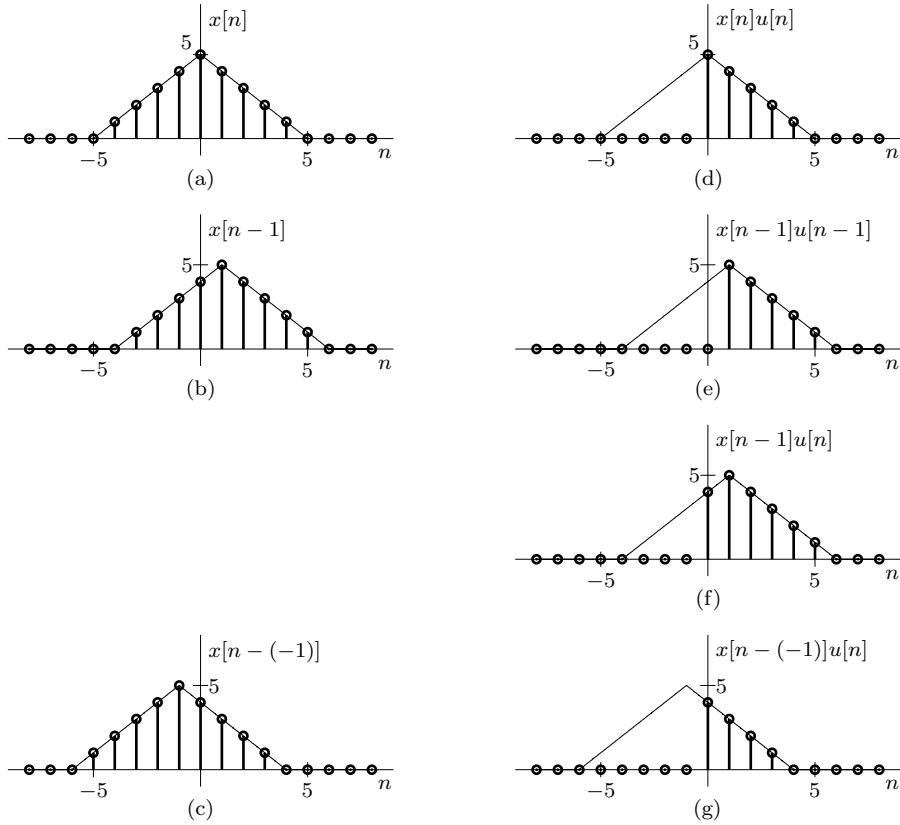
$$u[-n] \xleftrightarrow{Z} \frac{-1}{z-1} \text{ with ROC } |z| < 1.$$

△

7.3.4 Time-Shifting Property

In the following discussion of the time-shifting property, we deal with various forms of shifting, such as $x[n - m]$, $x[n - m]u[n - m]$, and $x[n - m]u[n]$. Unless we physically understand the meaning of such shifts, an understanding of the shifting property remains mechanical rather than intuitive or heuristic. For this reason, Fig. 7.7 illustrates these various shifts for a hypothetical signal $x[n]$ and $m = \pm 1$.

[†]We only need to consider the bilateral case of the z -transform since time reversal changes a causal signal into a noncausal signal, which is unsuited to the unilateral z -transform.

Figure 7.7: A signal $x[n]$ and various time-shifted versions.

Bilateral z -Transform Time-Shifting Property

When using the bilateral z -transform, a shift to the right (Fig. 7.7b) or a shift to the left (Fig. 7.7c) relocates, but does not change, the original signal (Fig. 7.7a). Thus, we intuitively expect the transform of a time-shifted signal to closely resemble the transform of the original signal. In fact, this is precisely the case. For the bilateral z -transform, the time-shifting property states that

$$\text{if } x[n] \xleftrightarrow{z} X(z) \text{ with ROC } R_x, \text{ then } x[n-m] \xleftrightarrow{z} z^{-m}X(z) \text{ with ROC almost } R_x. \quad (7.20)$$

The new ROC is identical to R_x except for the possible addition or deletion of $z = 0$ or $|z| = \infty$ caused by the factor z^{-m} . Using the intermediate change of variable $k = n - m$, the proof of this property is given as

$$\mathcal{Z}\{x[n-m]\} = \sum_{n=-\infty}^{\infty} x[n-m]z^{-n} = \sum_{k=-\infty}^{\infty} x[k]z^{-(k+m)} = z^{-m}X(z).$$

Unilateral z -Transform Time-Shifting Properties

Since the unilateral z -transform essentially multiplies a signal by $u[n]$, the time-shifting property is more complicated than for the bilateral case. Since different forms result, we treat right and left shifts separately.

Right Shift (Delay)

If

$$x[n]u[n] \xrightarrow{\mathcal{Z}_u} X(z),$$

then

$$x[n-1]u[n-1] \xrightarrow{\mathcal{Z}_u} z^{-1}X(z).$$

This result is similar to that for the bilateral z -transform since the signal $x[n-1]u[n-1]$ (Fig. 7.7e) relocates, but does not change, the original signal $x[n]u[n]$ (Fig. 7.7d). More generally,

$$x[n-m]u[n-m] \xrightarrow{\mathcal{Z}_u} z^{-m}X(z), \quad (m > 0). \quad (7.21)$$

A more interesting case occurs when we look at $x[n-1]u[n]$, which has

$$x[n-1]u[n] \xrightarrow{\mathcal{Z}_u} z^{-1}X(z) + x[-1].$$

In this case, $x[n-1]u[n]$ (Fig. 7.7f) is just $x[n-1]u[n-1]$ (Fig. 7.7e) plus an extra term $x[-1]\delta[n]$. It is this extra term that produces the extra term $x[-1]$ in the transform. Thought of another way, $x[n-1]u[n]$ (Fig. 7.7f) not only relocates but also changes the original signal $x[n]u[n]$ (Fig. 7.7d). Repeating this step yields

$$\begin{aligned} x[n-2]u[n] &\xrightarrow{\mathcal{Z}_u} z^{-1}(z^{-1}X(z) + x[-1]) + x[-2] \\ &= z^{-2}X(z) + z^{-1}x[-1] + x[-2]. \end{aligned}$$

In general,

$$x[n-m]u[n] \xrightarrow{\mathcal{Z}_u} z^{-m}X(z) + z^{-m} \sum_{n=1}^m x[-n]z^n, \quad (m > 0). \quad (7.22)$$

Notice that a right shift produces the same number of extra terms in both the time domain and the transform domain.

To prove Eq. (7.21), note that

$$\mathcal{Z}_u\{x[n-m]u[n-m]\} = \sum_{n=0}^{\infty} x[n-m]u[n-m]z^{-n}.$$

Since $m > 0$ and $x[n-m]u[n-m] = 0$ for $n < m$, the limits of summation can be taken from $n = m$ to ∞ . Therefore,

$$\mathcal{Z}_u\{x[n-m]u[n-m]\} = \sum_{n=m}^{\infty} x[n-m]z^{-n}.$$

Applying the change of variable $k = n - m$ yields the desired result of

$$\mathcal{Z}_u\{x[n-m]u[n-m]\} = \sum_{k=0}^{\infty} x[k]z^{-(k+m)} = z^{-m}X(z), \quad (m > 0).$$

To prove Eq. (7.22), we have

$$\mathcal{Z}_u\{x[n-m]u[n]\} = \sum_{n=0}^{\infty} x[n-m]z^{-n}, \quad (m > 0).$$

Using the change of variable $k = n - m$ yields

$$\begin{aligned} \mathcal{Z}_u\{x[n-m]u[n]\} &= \sum_{k=-m}^{\infty} x[k]z^{-(k+m)} \\ &= z^{-m} \left(\sum_{k=0}^{\infty} x[k]z^{-k} + \sum_{k=-m}^{-1} x[k]z^{-k} \right). \end{aligned}$$

Recognizing the first sum as $X(z)$ and letting $n = -k$ in the second sum, Eq. (7.22) directly follows.

Left Shift (Advance)

Following a left shift, the unilateral z -transform chops off any terms that move before $n = 0$. Consequently, the general (bilateral) result of Eq. (7.20) must be modified to account for these lost terms. For example, if

$$x[n]u[n] \xleftrightarrow{\mathcal{Z}_u} X(z),$$

then

$$x[n - (-1)]u[n] \xleftrightarrow{\mathcal{Z}_u} zX(z) - zx[0].$$

Here, $x[n - (-1)]u[n]$ (Fig. 7.7g) is a left shift of the original signal $x[n]u[n]$ (Fig. 7.7d) minus the term $x[0]\delta[n - 1]$. This lost term produces the term $-zx[0]$ in the transform. Repeating this step yields

$$x[n - (-2)]u[n] \xleftrightarrow{\mathcal{Z}_u} z(zX(z) - zx[0]) - zx[1] = z^2X(z) - z^2x[0] - zx[1].$$

In general,

$$x[n - m]u[n] \xleftrightarrow{\mathcal{Z}_u} z^{-m}X(z) - z^{-m} \sum_{n=0}^{-m-1} x[n]z^{-n}, \quad (m < 0). \quad (7.23)$$

To prove Eq. (7.23), we have

$$\mathcal{Z}_u \{x[n - m]u[n]\} = \sum_{n=0}^{\infty} x[n - m]z^{-n}, \quad (m < 0).$$

Using the change of variable $k = n - m$ yields

$$\begin{aligned} \mathcal{Z}_u \{x[n - m]u[n]\} &= \sum_{k=-m}^{\infty} x[k]z^{-(k+m)} \\ &= z^{-m} \left(\sum_{k=0}^{\infty} x[k]z^{-k} - \sum_{k=0}^{-m-1} x[k]z^{-k} \right). \end{aligned}$$

Recognizing the first sum as $X(z)$ and replacing the second sum variable k with n , Eq. (7.23) directly follows.

► Example 7.8 (Using the Time-Shifting Property)

Find the z -transform of the signal $x[n]$ depicted in Fig. 7.8.

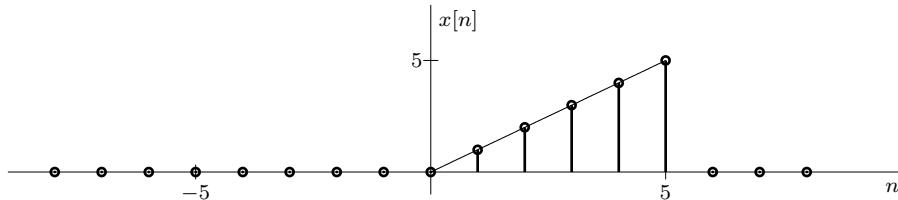


Figure 7.8: Signal $x[n] = n(u[n] - u[n - 6])$ for Ex. 7.8.

The signal $x[n]$ can be expressed as a product of n and the gate pulse $u[n] - u[n - 6]$. Therefore,

$$x[n] = n(u[n] - u[n - 6]) = nu[n] - nu[n - 6].$$

We cannot find the z -transform of $nu[n-6]$ directly by using the right-shifting property of Eq. (7.21). Thus, we express $x[n]$ in terms of $(n-6)u[n-6]$ as

$$x[n] = nu[n] - \{(n-6)u[n-6] + 6u[n-6]\}.$$

We can now find the z -transform of the bracketed term by using the right-shifting property of Eq. (7.21). Because $u[n] \xleftrightarrow{z_u} \frac{z}{z-1}$,

$$u[n-6] \xleftrightarrow{z_u} \frac{1}{z^6} \frac{z}{z-1} = \frac{1}{z^5(z-1)}.$$

Also, because $nu[n] \xleftrightarrow{z_u} \frac{z}{(z-1)^2}$,

$$(n-6)u[n-6] \xleftrightarrow{z_u} \frac{1}{z^6} \frac{z}{(z-1)^2} = \frac{1}{z^5(z-1)^2}.$$

Therefore,

$$X(z) = \frac{z}{(z-1)^2} - \frac{1}{z^5(z-1)^2} - \frac{6}{z^5(z-1)} = \frac{z^6 - 6z + 5}{z^5(z-1)^2}.$$

Example 7.8 \triangleleft

▷ **Drill 7.7 (Using the Time-Shifting Property)**

Using only the fact that $u[n] \xleftrightarrow{z_u} \frac{z}{z-1}$ and the right-shifting property of Eq. (7.21), find the z -transform of the signal $x_a[n]$ in Fig. 7.4.

\triangleleft

7.3.5 z -Domain Scaling Property

Scaling in the z -domain is equivalent to multiplying a time-domain signal by an exponential. More formally, if

$$x[n] \xleftrightarrow{z} X(z) \text{ with ROC } R_x,$$

then

$$\gamma^n x[n] \xleftrightarrow{z} X\left(\frac{z}{\gamma}\right) \text{ with ROC } |\gamma|R_x. \quad (7.24)$$

To prove Eq. (7.24), we note that

$$\mathcal{Z}\{\gamma^n x[n]\} = \sum_{n=-\infty}^{\infty} \gamma^n x[n] z^{-n} = \sum_{n=-\infty}^{\infty} x[n] \left(\frac{z}{\gamma}\right)^{-n} = X\left(\frac{z}{\gamma}\right).$$

If a point z is in the ROC R_x of $X(z)$, then the point $|\gamma|z$ is in the ROC of $X(z/\gamma)$. Put another way, scaling the z variable predictably requires the ROC to be scaled by the same amount.

▷ **Drill 7.8 (Using the z -Domain Scaling Property)**

Using Eq. (7.24), derive pairs 3 and 14 in Table 7.1 from pairs 2 and 13, respectively.

\triangleleft

7.3.6 z -Domain Differentiation Property

If

$$x[n] \xleftrightarrow{Z} X(z) \text{ with ROC } R_x,$$

then

$$nx[n] \xleftrightarrow{Z} -z \frac{d}{dz} X(z) \text{ with ROC } R_x. \quad (7.25)$$

Thus, multiplication by n in the time domain produces differentiation in the z -domain.

To prove Eq. (7.25), we first note that

$$\frac{d}{dz} X(z) = \frac{d}{dz} \sum_{n=-\infty}^{\infty} x[n] z^{-n} = \sum_{n=-\infty}^{\infty} -nx[n] z^{-n-1}.$$

Multiplying both sides by $-z$ yields

$$-z \frac{d}{dz} X(z) = -z \sum_{n=-\infty}^{\infty} -nx[n] z^{-n-1} = \sum_{n=-\infty}^{\infty} nx[n] z^{-n} = \mathcal{Z}\{nx[n]\}.$$

▷ **Drill 7.9 (Using the z -Domain Differentiation Property)**

Using Eq. (7.25), derive pairs 5 and 6 in Table 7.1 from pair 3. What is the z -transform of $x[n] = n^3 u[n]$? □

7.3.7 Time-Domain Convolution Property

The time-domain convolution property states that if[†]

$$x[n] \xleftrightarrow{Z} X(z) \text{ with ROC } R_x \quad \text{and} \quad y[n] \xleftrightarrow{Z} Y(z) \text{ with ROC } R_y,$$

then

$$x[n] * y[n] \xleftrightarrow{Z} X(z)Y(z) \text{ with ROC at least } R_x \cap R_y. \quad (7.26)$$

By moving to the transform domain, the simple operation of multiplication replaces the cumbersome and often difficult time-domain operation of convolution.

To prove Eq. (7.26), we begin with

$$\mathcal{Z}\{x[n] * y[n]\} = \mathcal{Z}\left\{\sum_{m=-\infty}^{\infty} x[m]y[n-m]\right\} = \sum_{n=-\infty}^{\infty} \left(\sum_{m=-\infty}^{\infty} x[m]y[n-m]\right) z^{-n}.$$

Interchanging the order of summation produces

$$\mathcal{Z}\{x[n] * y[n]\} = \sum_{m=-\infty}^{\infty} x[m] \sum_{n=-\infty}^{\infty} y[n-m] z^{-n}.$$

[†]There is also the frequency-domain convolution property, which states that

$$x_1[n]x_2[n] \xleftrightarrow{Z} \frac{1}{2\pi j} \oint X_1(u)X_2\left(\frac{z}{u}\right) u^{-1} du.$$

This property requires an understanding of complex contour integration and is outside the scope of this book.

Finally, the change of variable $k = n - m$ yields

$$\begin{aligned}\mathcal{Z}\{x[n] * y[n]\} &= \sum_{m=-\infty}^{\infty} x[m] \sum_{k=-\infty}^{\infty} y[k] z^{-(k+m)} \\ &= \sum_{m=-\infty}^{\infty} x[m] z^{-m} \sum_{k=-\infty}^{\infty} y[k] z^{-k} = X(z)Y(z).\end{aligned}$$

Since the transforms $X(z)$ and $Y(z)$ must both simultaneously exist, the ROC of $X(z)Y(z)$ generally equals $R_x \cap R_y$, although certain pole cancellations may increase the ROC. Table 7.2 summarizes the most important properties of the z -transform.

Bilateral z -Transform	Unilateral z -Transform
Synthesis: $x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz$ Analysis: $X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$, ROC: R_x Linearity: $ax[n] + by[n] \xrightleftharpoons{z} aX(z) + bY(z)$, ROC: At least $R_x \cap R_y$ Complex Conjugation: $x^*[n] \xrightleftharpoons{z} X^*(z^*)$, ROC: R_x Time Reversal: $x[-n] \xrightleftharpoons{z} X(1/z)$, ROC: $1/R_x$ Time Shifting: $x[n-m] \xrightleftharpoons{z} z^{-m} X(z)$, ROC: Almost R_x z-Domain Scaling: $\gamma^n x[n] \xrightleftharpoons{z} X(z/\gamma)$, ROC: $ \gamma R_x$ z-Domain Differentiation: $nx[n] \xrightleftharpoons{z} -z \frac{d}{dz} X(z)$, ROC: R_x Time Convolution: $x[n] * y[n] \xrightleftharpoons{z} X(z)Y(z)$, ROC: At least $R_x \cap R_y$	Synthesis: $x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz$ Analysis: $X(z) = \sum_{n=0}^{\infty} x[n] z^{-n}$ Linearity: $ax[n] + by[n] \xrightleftharpoons{z} aX(z) + bY(z)$ Complex Conjugation: $x^*[n] \xrightleftharpoons{z} X^*(z^*)$ Time Reversal: Time Shifting: If $m > 0$: $x[n-m]u[n-m] \xrightleftharpoons{z} z^{-m} X(z)$ (general case given below) z-Domain Scaling: $\gamma^n x[n] \xrightleftharpoons{z} X(z/\gamma)$ z-Domain Differentiation: $nx[n] \xrightleftharpoons{z} -z \frac{d}{dz} X(z)$ Time Convolution: $x[n] * y[n] \xrightleftharpoons{z} X(z)Y(z)$
Unilateral z-Transform Time Shifting, General Case If $m > 0$: $x[n-m]u[n] \xrightleftharpoons{z} z^{-m} X(z) + z^{-m} \sum_{n=1}^m x[-n]z^n$ If $m < 0$: $x[n-m]u[n] \xrightleftharpoons{z} z^{-m} X(z) - z^{-m} \sum_{n=0}^{-m-1} x[n]z^{-n}$	

Table 7.2: Properties of the bilateral and unilateral z -transforms.

LTID System Zero-State Response

It is interesting to apply the time-convolution property to the LTID zero-state response equation $y[n] = x[n] * h[n]$. In Eq. (7.10), we have shown that $h[n] \xrightleftharpoons{z} H(z)$ with ROC R_h . As long as $X(z)$

and $H(z)$ exist, then it follows from Eq. (7.26) that

$$Y(z) = X(z)H(z) \text{ with ROC at least } R_x \cap R_h. \quad (7.27)$$

Further, as long as $Y(z)$ exists, which is to say it has a non-empty ROC, then the zero-state response $y[n]$ is given as

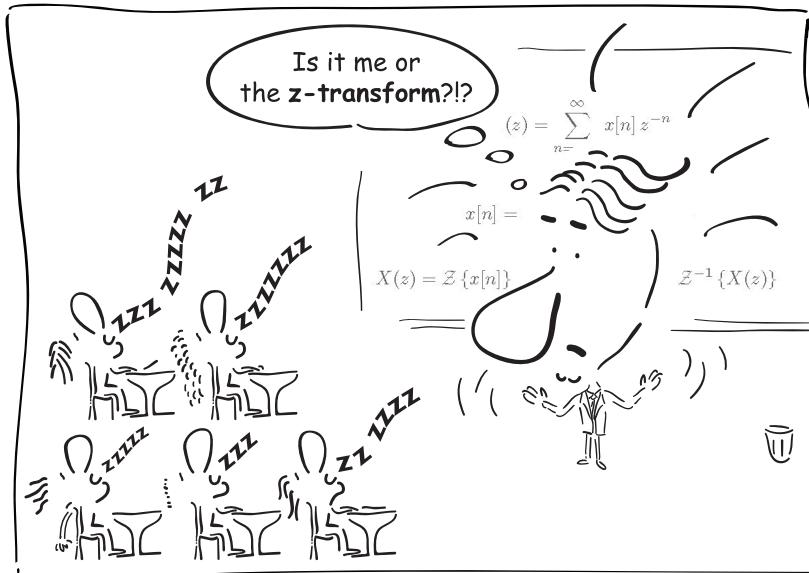
$$y[n] = \mathcal{Z}^{-1}\{X(z)H(z)\}. \quad (7.28)$$

Often, it is easier to compute $y[n]$ using Eq. (7.28) rather than the time-domain convolution operation $x[n] * h[n]$. Section 7.4 develops these ideas further and provides several examples.

▷ Drill 7.10 (Using the Convolution Property)

Using the (time-domain) convolution property of Eq. (7.26) and appropriate pairs in Table 7.1, show that $u[n] * u[n - 1] = nu[n]$.

△



Another property of the z -transform?

7.3.8 Initial and Final Value Theorems

While generally not as useful as the properties already discussed, the initial value and final value theorems provide simple and quick means to relate the initial and final values of a causal signal $x[n]$ to its transform $X(z)$. These relations can serve as a partial check on the correctness of a computed z -transform.

Restricting our attention to a causal signal $x[n]$, the initial value theorem states that

$$x[0] = \lim_{z \rightarrow \infty} X(z). \quad (7.29)$$

This result follows immediately from Eq. (7.8).

Further, provided that $(z - 1)X(z)$ has no poles outside the unit circle, the final value theorem states that

$$\lim_{N \rightarrow \infty} x[N] = \lim_{z \rightarrow 1} (z - 1)X(z). \quad (7.30)$$

To show the final value theorem, note that

$$x[n] - x[n-1] \xrightarrow{Z_u} \left\{ 1 - \frac{1}{z} \right\} X(z) = \frac{(z-1)X(z)}{z}$$

and

$$(z-1)X(z) = \lim_{N \rightarrow \infty} \sum_{n=0}^N (x[n] - x[n-1]) z^{-n+1}.$$

Taking the limit as $z \rightarrow 1$ yields

$$\begin{aligned} \lim_{z \rightarrow 1} (z-1)X(z) &= \lim_{z \rightarrow 1} \left(\lim_{N \rightarrow \infty} \sum_{n=0}^N (x[n] - x[n-1]) z^{-n+1} \right) \\ &= \lim_{N \rightarrow \infty} \sum_{n=0}^N (x[n] - x[n-1]) \\ &= \lim_{N \rightarrow \infty} \{ (x[0] + x[1] + x[2] + \dots + x[N]) - (x[-1] + x[0] + \dots + x[N-1]) \} \\ &= \lim_{N \rightarrow \infty} x[N]. \end{aligned}$$

7.4 z -Transform Solution of Linear Difference Equations

The time-shifting property sets the stage for solving linear difference equations with constant coefficients. As in the case of the Laplace transform with differential equations, the z -transform converts difference equations into algebraic equations that are readily solved to find the solution in the z -domain. Taking the inverse z -transform of the z -domain solution yields the desired time-domain solution. When initial conditions are present, the unilateral z -transform is generally the appropriate analysis tool. When only the zero-state response is required, either the bilateral or unilateral z -transform may be appropriate. The following examples demonstrate these cases.

▷ Example 7.9 (z -Transform Solution of Linear Difference Equations)

Given input $x[n] = (0.5)^n u[n]$ and initial conditions $y[-1] = \frac{11}{6}$ and $y[-2] = \frac{37}{36}$, use the unilateral z -transform to solve the second-order ($K = 2$) constant-coefficient linear difference equation

$$y[n+2] - 5y[n+1] + 6y[n] = 3x[n+1] + 5x[n]. \quad (7.31)$$

As we shall see, difference equations can be solved by using the right-shifting property or the left-shifting property. Because the system of Eq. (7.31) is in advance form, the use of the left-shifting property may seem appropriate for its solution. Unfortunately, as seen from Eq. (7.23), the left-shifting property requires a knowledge of the auxiliary conditions $y[0], y[1], \dots, y[K-1]$ rather than of the initial conditions $y[-1], y[-2], \dots, y[-K]$, which are generally given. This difficulty can be overcome by expressing Eq. (7.31) in delay form (obtained by replacing n with $n-2$) and then using the right-shifting property.[†] Expressed in delay form, Eq. (7.31) is

$$y[n] - 5y[n-1] + 6y[n-2] = 3x[n-1] + 5x[n-2]. \quad (7.32)$$

We now use the right-shifting property to take the (unilateral) z -transform of this equation. But before proceeding, we must be clear about the meaning here of a term such as $y[n-1]$. Does it mean $y[n-1]u[n-1]$ or $y[n-1]u[n]$? Since a unilateral z -transform is being used, multiplication

[†]Another approach is to find $y[0], y[1], y[2], \dots, y[K-1]$ from $y[-1], y[-2], \dots, y[-K]$ iteratively, as in Sec. 5.1, and then apply the left-shifting property.

by the unit step $u[n]$ is implied. Thought of another way, any equation with initial conditions has some time reference (usually $n = 0$), and every term is referenced from this instant. Hence, $y[n - 1]$ means $y[n - 1]u[n]$. Remember also that although we are considering the solution for $n \geq 0$, $y[n]$ is present even before $n = 0$ (in the form of initial conditions).

Applied to the output terms, the right-shifting property of Eq. (7.22) yields

$$\begin{aligned} y[n]u[n] &\xrightleftharpoons{\mathcal{Z}_u} Y(z), \\ y[n - 1]u[n] &\xrightleftharpoons{\mathcal{Z}_u} z^{-1}Y(z) + y[-1] = z^{-1}Y(z) + \frac{11}{6}, \\ \text{and } y[n - 2]u[n] &\xrightleftharpoons{\mathcal{Z}_u} z^{-2}Y(z) + z^{-1}y[-1] + y[-2] = z^{-2}Y(z) + \frac{11}{6}z^{-1} + \frac{37}{36}. \end{aligned}$$

Since our input is causal, we see that $x[-1] = x[-2] = \dots = 0$ and

$$x[n - m]u[n] = x[n - m]u[n - m] \xrightleftharpoons{\mathcal{Z}_u} z^{-m}X(z), \quad (m > 0).$$

Thus, using pair 3 of Table 7.1,

$$\begin{aligned} x[n]u[n] &\xrightleftharpoons{\mathcal{Z}_u} X(z) = \frac{z}{z - 0.5}, \\ x[n - 1]u[n] &\xrightleftharpoons{\mathcal{Z}_u} z^{-1}X(z) = \frac{1}{z - 0.5}, \\ \text{and } x[n - 2]u[n] &\xrightleftharpoons{\mathcal{Z}_u} z^{-2}X(z) = \frac{1}{z(z - 0.5)}. \end{aligned}$$

Using these results to take the z -transform of Eq. (7.32) yields

$$Y(z) - 5 \left(z^{-1}Y(z) + \frac{11}{6} \right) + 6 \left(z^{-2}Y(z) + z^{-1}\frac{11}{6} + \frac{37}{36} \right) = \frac{3}{z - 0.5} + \frac{5}{z(z - 0.5)},$$

and then

$$(1 - 5z^{-1} + 6z^{-2}) Y(z) - (3 - 11z^{-1}) = \frac{3}{z - 0.5} + \frac{5}{z(z - 0.5)}, \quad (7.33)$$

from which we obtain

$$(z^2 - 5z + 6) Y(z) = \frac{z(3z^2 - 9.5z + 10.5)}{(z - 0.5)}$$

and

$$\frac{Y(z)}{z} = \frac{(3z^2 - 9.5z + 10.5)}{(z - 0.5)(z^2 - 5z + 6)}.$$

Next, we expand this result using MATLAB. As shown in Sec. 5.5.2 (page 299), the DT convolution `conv([1 -.5], [1 -5 6])` computes the coefficients of the expansion $(z - 0.5)(z^2 - 5z + 6) = z^3 - 5.5z^2 + 8.5z - 3$, which is the format required by the `residue` command.

```
01 format rat; [r,p,k] = residue([3 -9.5 10.5],conv([1 -.5],[1 -5 6]))
r = 18/5   -7/3   26/15
p = 3    2    1/2
k = []
```

Thus,

$$\frac{Y(z)}{z} = \frac{(26/15)}{z - 0.5} - \frac{(7/3)}{z - 2} + \frac{(18/5)}{z - 3}$$

and

$$Y(z) = \frac{26}{15} \left(\frac{z}{z - 0.5} \right) - \frac{7}{3} \left(\frac{z}{z - 2} \right) + \frac{18}{5} \left(\frac{z}{z - 3} \right).$$

Using Table 7.1, the final result is

$$y[n] = \left[\frac{26}{15}(0.5)^n - \frac{7}{3}(2)^n + \frac{18}{5}(3)^n \right] u[n]. \quad (7.34)$$

This example demonstrates the ease with which linear difference equations with constant coefficients can be solved by z -transform. This method is general; it can be used to solve a single difference equation or a set of simultaneous difference equations of any order as long as the equations are linear with constant coefficients.

Example 7.9 ◇

Comment on Initial Conditions

Sometimes auxiliary conditions $y[0], y[1], \dots, y[K-1]$ (instead of initial conditions $y[-1], y[-2], \dots, y[-n]$) are given to solve a difference equation. In this case, the equation can be solved by expressing it in the advance-operator form and then using the left-shifting property. Drill 7.12 explores this case.

▷ Drill 7.11 (z -Transform Solution of Linear Difference Equations)

Given input $x[n] = u[n]$ and initial conditions $y[-1] = 2$ and $y[-2] = 0$, use the unilateral z -transform to solve the second-order ($K = 2$) constant-coefficient linear difference equation

$$y[n+2] - \frac{5}{6}y[n+1] + \frac{1}{6}y[n] = 5x[n+1] - x[n].$$

◇

▷ Drill 7.12 (z -Transform Solution Using Initial Conditions $y[0], y[1], \dots, y[K-1]$)

Given input $x[n] = u[n]$ and initial conditions $y[0] = 1$ and $y[1] = 2$, use the unilateral z -transform to solve the second-order ($K = 2$) constant-coefficient linear difference equation

$$y[n] + 3y[n-1] + 2y[n-2] = x[n-1] + 3x[n-2].$$

◇

Zero-Input and Zero-State Components

In Ex. 7.9 we found the total solution of the difference equation. It is relatively easy to separate the solution into zero-input and zero-state components. All we have to do is separate the response into terms arising from the input and terms arising from initial conditions. For example, we can separate the response in Eq. (7.33) as

$$(1 - 5z^{-1} + 6z^{-2}) Y(z) - \underbrace{(3 - 11z^{-1})}_{\text{IC terms}} = \underbrace{\frac{3}{z - 0.5}}_{\text{input terms}} + \underbrace{\frac{5}{z(z - 0.5)}}_{\text{input terms}}.$$

Therefore,

$$(1 - 5z^{-1} + 6z^{-2}) Y(z) = \underbrace{(3 - 11z^{-1})}_{\text{IC terms}} + \underbrace{\frac{(3z + 5)}{z(z - 0.5)}}_{\text{input terms}}.$$

Multiplying both sides by z^2 yields

$$(z^2 - 5z + 6) Y(z) = \underbrace{z(3z - 11)}_{\text{IC terms}} + \underbrace{\frac{z(3z + 5)}{(z - 0.5)}}_{\text{input terms}}.$$

Thus,

$$Y(z) = \underbrace{\frac{z(3z - 11)}{z^2 - 5z + 6}}_{\text{ZIR}} + \underbrace{\frac{z(3z + 5)}{(z - 0.5)(z^2 - 5z + 6)}}_{\text{ZSR}}.$$

We expand the right-hand side into modified partial fractions to yield

$$Y(z) = \underbrace{5 \left(\frac{z}{z-2} \right) - 2 \left(\frac{z}{z-3} \right)}_{\text{ZIR}} + \underbrace{\frac{26}{15} \left(\frac{z}{z-0.5} \right) - \frac{22}{3} \left(\frac{z}{z-2} \right) + \frac{28}{5} \left(\frac{z}{z-3} \right)}_{\text{ZSR}}.$$

Inverting, we obtain

$$\begin{aligned} y[n] &= \underbrace{[5(2)^n - 2(3)^n] u[n]}_{\text{ZIR}} + \underbrace{\left[\frac{26}{15}(0.5)^n - \frac{22}{3}(2)^n + \frac{28}{5}(3)^n \right] u[n]}_{\text{ZSR}} \\ &= \left[\frac{26}{15}(0.5)^n - \frac{7}{3}(2)^n + \frac{18}{5}(3)^n \right] u[n], \end{aligned}$$

which agrees with the result in Eq. (7.34).

▷ **Drill 7.13 (Finding the ZIR and ZSR by z -Transform)**

Given input $x[n] = u[n]$ and initial conditions $y[-1] = 2$ and $y[-2] = 0$, use the unilateral z -transform to determine the ZIR and ZSR of

$$y[n+2] - \frac{5}{6}y[n+1] + \frac{1}{6}y[n] = 5x[n+1] - x[n].$$

▫

7.4.1 Zero-State Response of LTID Systems: The Transfer Function

Consider a K th-order LTID system specified by the difference equation

$$\begin{aligned} y[n+K] + a_1y[n+(K-1)] + \cdots + a_{K-1}y[n+1] + a_Ky[n] &= \\ b_0x[n+K] + b_1x[n+(K-1)] + \cdots + b_{K-1}x[n+1] + b_Kx[n]. \end{aligned} \tag{7.35}$$

Recall from Sec. 5.2 that Eq. (7.35) is expressed in operator notation as

$$A(E) \{y[n]\} = B(E) \{x[n]\},$$

where $A(E)$ and $B(E)$ are the K th-order polynomial operators

$$\begin{aligned} A(E) &= E^K + a_1E^{K-1} + \cdots + a_{K-1}E + a_K \\ \text{and } B(E) &= b_0E^K + b_1E^{K-1} + \cdots + b_{K-1}E + b_K. \end{aligned}$$

We now derive a general expression for the zero-state response, that is, the system response to input $x[n]$ when all initial conditions, such as $y[-1] = y[-2] = \cdots = y[-K] = 0$, are zero (zero

state). Without initial conditions to contend with, we proceed using the bilateral z -transform and its simplified shifting property.

Expressed in delay form, Eq. (7.35) is

$$\begin{aligned} y[n] + a_1 y[n-1] + \cdots + a_{K-1} y[n-(K-1)] + a_K y[n-K] = \\ b_0 x[n] + b_1 x[n-1] + \cdots + b_{K-1} x[n-(K-1)] + b_K x[n-K]. \end{aligned} \quad (7.36)$$

Using the shifting property of Eq. (7.20), we note that[†]

$$y[n-k] \xrightarrow{\mathcal{Z}} z^{-k} Y(z) \quad \text{and} \quad x[n-k] \xrightarrow{\mathcal{Z}} z^{-k} X(z).$$

The z -transform of Eq. (7.36) is consequently given by

$$\begin{aligned} \left(1 + a_1 z^{-1} + \cdots + a_{K-1} z^{-(K-1)} + a_K z^{-K}\right) Y(z) = \\ \left(b_0 + b_1 z^{-1} + \cdots + b_{K-1} z^{-(K-1)} + b_K z^{-K}\right) X(z). \end{aligned}$$

Multiplication of both sides by z^K yields

$$(z^K + a_1 z^{K-1} + \cdots + a_{K-1} z + a_K) Y(z) = (b_0 z^K + b_1 z^{K-1} + \cdots + b_{K-1} z + b_K) X(z).$$

Therefore,

$$Y(z) = \left(\frac{b_0 z^K + b_1 z^{K-1} + \cdots + b_{K-1} z + b_K}{z^K + a_1 z^{K-1} + \cdots + a_{K-1} z + a_K} \right) X(z) = \frac{B(z)}{A(z)} X(z). \quad (7.37)$$

We have shown in Eq. (7.27) that $Y(z) = X(z)H(z)$. Hence, it follows that

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 z^K + b_1 z^{K-1} + \cdots + b_{K-1} z + b_K}{z^K + a_1 z^{K-1} + \cdots + a_{K-1} z + a_K}. \quad (7.38)$$

As in the case of LTIC systems, this result leads to an alternative definition of the LTID system transfer function as a ratio of (zero-state) output $Y(z)$ to input $X(z)$,

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\mathcal{Z}\{\text{zero-state response}\}}{\mathcal{Z}\{\text{input}\}}. \quad (7.39)$$

Because $Y(z)$, the z -transform of the zero-state response $y[n]$, is the product of $X(z)$ and $H(z)$, we can represent an LTID system in the transform domain by a block diagram, as illustrated in Fig. 7.9.

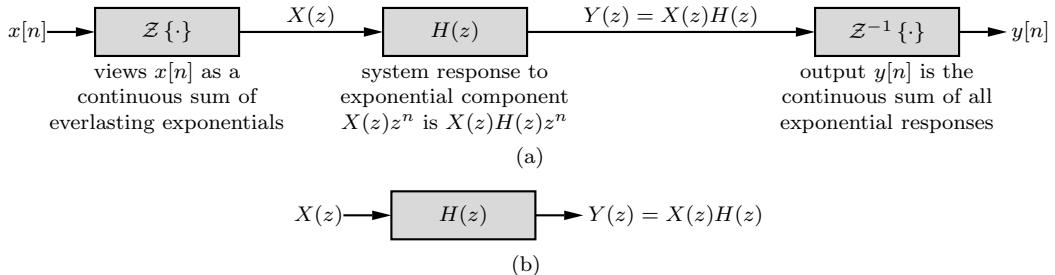


Figure 7.9: Transform-domain solution and representation of an LTID system.

Just as with continuous-time systems, we form transform-domain representations of discrete-time systems by representing all signals by their z -transforms and all system components (or elements) by

[†]To avoid unnecessary distraction, we do not include ROCs throughout most of this discussion.

their transfer functions. The result $Y(z) = X(z)H(z)$ greatly facilitates derivation of the system's zero-state response to a given input. We shall demonstrate these concepts with several examples.

▷ **Example 7.10 (Transfer Function of a Time-Shifting System)**

Show that the transfer function of a system that time shifts its input by k units is $H(z) = z^{-k}$.

As depicted in Fig. 7.10, if the input to the time-shifting system is $x[n]$, then its output is given by $y[n] = x[n - k]$. Using Eq. (7.20), the z -transform of this input-output equation is

$$Y(z) = z^{-k}X(z).$$

From Eq. (7.27), we know that $Y(z) = H(z)X(z)$, so it follows that the transfer function of a time-shifting system is

$$H(z) = z^{-k}. \quad (7.40)$$

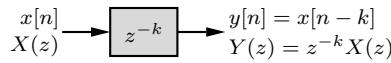


Figure 7.10: Block representation of a general time-shifting system.

Example 7.10 ◀

▷ **Example 7.11 (ZSR by z -Transform for Causal Input)**

Given input $x[n] = (-2)^{-n}u[n]$, use the z -transform to determine the zero-state response $y[n]$ of a causal LTID system described by the difference equation

$$y[n+2] + y[n+1] + 0.16y[n] = x[n+1] + 0.32x[n] \quad (7.41)$$

or

$$(E^2 + E + 0.16)\{y[n]\} = (E + 0.32)\{x[n]\}.$$

Since the system and input are both causal, the output $y[n]$ is also causal, and we are content to use the unilateral z -transform for our analysis. Using Eq. (7.41), we find that

$$H(z) = \frac{B(z)}{A(z)} = \frac{z + 0.32}{z^2 + z + 0.16}.$$

For the input $x[n] = (-2)^{-n}u[n] = (-0.5)^n u[n]$,

$$X(z) = \frac{z}{z + 0.5}$$

and

$$Y(z) = X(z)H(z) = \frac{z(z + 0.32)}{(z^2 + z + 0.16)(z + 0.5)}.$$

Using MATLAB, the modified partial fractions are determined by expanding $Y(z)/z$.

```

01 format rat; [r,p,k] = residue([1 0.32],conv([1 1 0.16],[1 0.5]))
r = -8/3 2 2/3
p = -4/5 -1/2 -1/5
k = []

```

Therefore,

$$\frac{Y(z)}{z} = \frac{2/3}{z+0.2} + \frac{2}{z+0.5} - \frac{8/3}{z+0.8}$$

and

$$Y(z) = \frac{2}{3} \left(\frac{z}{z+0.2} \right) - \frac{8}{3} \left(\frac{z}{z+0.8} \right) + 2 \left(\frac{z}{z+0.5} \right).$$

Inverting, we obtain

$$y[n] = \left[\frac{2}{3}(-0.2)^n - \frac{8}{3}(-0.8)^n + 2(-0.5)^n \right] u[n].$$

Example 7.11 ◀

▷ Example 7.12 (ZSR by z -Transform for Two-Sided Input)

Given two-sided input $x[n] = (0.8)^n u[n] + 2(2)^n u[-n-1]$, use the z -transform to determine the zero-state response $y[n]$ of a causal LTID system described by the transfer function

$$H(z) = \frac{z}{z-0.5}. \quad (7.42)$$

In this case, the input is composed of a causal component $(0.8)^n u[n]$ and an anti-causal component $2(2)^n u[-n-1]$. Taking the z -transform of these pieces yields

$$\mathcal{Z}\{(0.8)^n u[n]\} = \frac{z}{z-0.8} \text{ with ROC } |z| > 0.8$$

and

$$\mathcal{Z}\{2(2)^n u[-n-1]\} = -\frac{2z}{z-2} \text{ with ROC } |z| < 2.$$

Combining these results, the z -transform of the input is

$$X(z) = \frac{-z(z+0.4)}{(z-0.8)(z-2)}, \quad \text{ROC: } 0.8 < |z| < 2.$$

Since the system is causal, the ROC of $H(z)$ is $|z| > 0.5$. The ROC of $X(z)$ is $0.8 < |z| < 2$. The common region of convergence for $X(z)$ and $H(z)$ is $0.8 < |z| < 2$. Therefore,

$$Y(z) = X(z)H(z) = \frac{-z^2(z+0.4)}{(z-0.5)(z-0.8)(z-2)}, \quad \text{ROC: } 0.8 < |z| < 2.$$

Using MATLAB, the modified partial fractions are determined by expanding $Y(z)/z$.

```
01 format rat; [r,p,k] = residue([-1 -0.4 0],poly([0.5,0.8,2]))
r = -8/3 8/3 -1
p = 2 4/5 1/2
k = []
```

Therefore,

$$\frac{Y(z)}{z} = \frac{-1}{z-0.5} + \frac{8/3}{z-0.8} - \frac{8/3}{z-2} \text{ with ROC } 0.8 < |z| < 2,$$

and

$$Y(z) = -\left(\frac{z}{z-0.5} \right) + \frac{8}{3} \left(\frac{z}{z-0.8} \right) - \frac{8}{3} \left(\frac{z}{z-2} \right) \text{ with ROC } 0.8 < |z| < 2.$$

Since the ROC extends outward from the pole at 0.8, the poles at 0.5 and 0.8 correspond to causal sequences. The ROC extends inward from the pole at 2. Hence, the pole at 2 corresponds to an anti-causal sequence. Therefore,

$$y[n] = \left[-(0.5)^n + \frac{8}{3}(0.8)^n \right] u[n] + \frac{8}{3}(2)^n u[-n-1].$$

Example 7.12 ◀

▷ **Example 7.13 (ZSR by z -Transform for a Non-Transformable Input)**

For the system in Ex. 7.12, find the zero-state response to the input

$$x[n] = \underbrace{(0.8)^n u[n]}_{x_1[n]} + \underbrace{(0.6)^n u[-n-1]}_{x_2[n]}.$$

The z -transforms of the causal and anti-causal components $x_1[n]$ and $x_2[n]$ are

$$X_1(z) = \frac{z}{z-0.8} \text{ with ROC } |z| > 0.8$$

and

$$X_2(z) = -\frac{z}{z-0.6} \text{ with ROC } |z| < 0.6.$$

Observe that a common ROC for $X_1(z)$ and $X_2(z)$ does not exist. Therefore, $X(z)$ does not exist. In such cases, we can apply the superposition principle and find $y_1[n]$ and $y_2[n]$, the system responses to $x_1[n]$ and $x_2[n]$, separately. The desired response $y[n]$ is the sum of $y_1[n]$ and $y_2[n]$. Now,

$$H(z) = \frac{z}{z-0.5} \text{ with ROC } |z| > 0.5,$$

$$Y_1(z) = X_1(z)H(z) = \frac{z^2}{(z-0.5)(z-0.8)} \text{ with ROC } |z| > 0.8,$$

$$\text{and } Y_2(z) = X_2(z)H(z) = \frac{-z^2}{(z-0.5)(z-0.6)} \text{ with ROC } 0.5 < |z| < 0.6.$$

Expanding $Y_1(z)$ and $Y_2(z)$ into modified partial fractions yields

$$Y_1(z) = -\frac{5}{3} \left(\frac{z}{z-0.5} \right) + \frac{8}{3} \left(\frac{z}{z-0.8} \right) \text{ with ROC } |z| > 0.8$$

$$\text{and } Y_2(z) = 5 \left(\frac{z}{z-0.5} \right) - 6 \left(\frac{z}{z-0.6} \right) \text{ with ROC } 0.5 < |z| < 0.6.$$

Therefore,

$$y_1[n] = \left[-\frac{5}{3}(0.5)^n + \frac{8}{3}(0.8)^n \right] u[n]$$

$$\text{and } y_2[n] = 5(0.5)^n u[n] + 6(0.6)^n u[-n-1].$$

Due to linearity, $y[n] = y_1[n] + y_2[n]$, and

$$y[n] = \left[\frac{10}{3}(0.5)^n + \frac{8}{3}(0.8)^n \right] u[n] + 6(0.6)^n u[-n-1].$$

Example 7.13 ◀

▷ **Drill 7.14 (Working with a System Transfer Function)**

Consider a causal DT system that is both controllable and observable with transfer function

$$H(z) = \frac{z - 0.5}{(z + 0.5)(z - 1)}.$$

- (a) Find the zero-state response $y[n]$ to input $x[n] = 3^{-(n+1)}u[n]$.
- (b) Write the difference equation that relates the system output $y[n]$ to input $x[n]$.

△

▷ **Drill 7.15 (ZSR by z -Transform for a Two-Sided Input)**

Consider a discrete-time system with transfer function $H(z) = z/(z - 0.5)$ and input signal

$$x[n] = \left(\frac{1}{4}\right)^n u[n] + 5(3)^n u[-n - 1].$$

- (a) Find the zero-state response $y_a[n]$ if the system is causal.
- (b) Find the zero-state response $y_b[n]$ if the system is anti-causal.

△

System Stability and the Transfer Function $H(z)$

Equation (7.38) shows that the denominator of $H(z)$ is the characteristic polynomial $A(z)$. Does this mean that the poles (denominator roots) of $H(z)$ are the characteristic roots of the system? This may or may not be the case because if $A(z)$ and $B(z)$ in Eq. (7.38) have any common factors, then they cancel out, and the effective denominator of $H(z)$ is not necessarily equal to $A(z)$. Recall also that the system transfer function $H(z)$, like $h[n]$, is defined in terms of measurements at the external terminals. Consequently, $H(z)$ and $h[n]$ are both external descriptions of the system. In contrast, the characteristic polynomial $A(z)$ is an internal description. Clearly, we can determine only external (BIBO) stability from $H(z)$ alone. If the ROC of $H(z)$ includes the unit circle, then all the terms in $h[n]$ are decaying exponentials, $h[n]$ is absolutely summable, and the system is BIBO stable. Otherwise, the system is BIBO unstable. For causal systems, BIBO stability requires that all poles of $H(z)$ are within the unit circle.

Asymptotic stability, which offers a more complete picture than BIBO stability, requires knowledge of the system's internal structure. The characteristic polynomial $A(z)$ captures the internal structure of a system, and the characteristic roots of a system, which are found by solving $A(z) = 0$, provide a convenient mechanism to determine asymptotic stability. Further, if $A(z)$ and $B(z)$ do not have common factors, then the denominator of $H(z)$ is identical to $A(z)$, and the poles of $H(z)$ are the characteristic roots of the system; only in this case can we determine internal stability using the poles of $H(z)$.[†] As presented in Sec. 5.7, the internal stability criterion for causal systems is stated in terms of characteristic roots as

1. A causal LTID system is asymptotically stable if and only if all the characteristic roots are inside the unit circle. The roots may be simple or repeated.
2. A causal LTID system is marginally stable if and only if there are no roots outside the unit circle and there are non-repeated roots on the unit circle.

[†]Based on $H(z)$ alone, where $A(z)$ and $B(z)$ are not explicitly known, asymptotic stability cannot be determined since it is impossible to determine if common factors exist between $A(z)$ and $B(z)$.

3. A causal LTID system is unstable if and only if at least one root is outside the unit circle, there are repeated roots on the unit circle, or both.

For systems that do not have common factors between $A(z)$ and $B(z)$, these conditions generalize to

1. An LTID system is asymptotically stable if and only if the ROC of $H(z)$ includes the unit circle.
2. An LTID system is marginally stable if and only if the ROC of $H(z)$ contacts (but does not include) the unit circle and there are non-repeated roots on the unit circle.
3. An LTID system is unstable if and only if the ROC of $H(z)$ does not include or contact the unit circle, there are repeated roots on the unit circle, or both.

This generalization allows us to assess internal stability for most systems, including noncausal systems.

▷ **Drill 7.16 (Assessing System Stability)**

- (a) Show that an accumulator, whose impulse response $h_a[n] = u[n]$, is marginally stable but BIBO unstable.
- (b) Assuming that there are no pole-zero cancellations, determine the stability of a system with transfer function

$$H_b(z) = \frac{z+1}{(z+0.5)(z-1)(z-2)}, \quad \text{ROC: } 0.5 < |z| < 1.$$

△

7.5 Block Diagrams and System Realization

Large systems may consist of an enormous number of components or elements, and analyzing such systems all at once can be next to impossible. Generally, it is more convenient to represent systems using suitably interconnected subsystems, each of which can be readily analyzed.

As shown in Figs. 7.11a and 7.11b, subsystems may be connected in two elementary ways: parallel and cascade. When two transfer functions appear in parallel, the overall transfer function is given by $H_1(z) + H_2(z)$, the sum of the two transfer functions. The proof is trivial, and the result can be extended to any number of systems in parallel. Similarly, when two transfer functions $H_1(z)$ and $H_2(z)$ appear in cascade, the transfer function of the overall system is the product of the two transfer functions $H_1(z)H_2(z)$. This conclusion follows from the fact that in Fig. 7.11b,

$$\frac{Y(z)}{X(z)} = \frac{W(z)}{X(z)} \frac{Y(z)}{W(z)} = H_1(z)H_2(z).$$

This result can be extended to any number of transfer functions in cascade.

A Word of Caution

For block representations such as those found in Fig. 7.11, there is an implicit assumption that there is no loading between subsystems. Basically, this means that the behavior (transfer function) of any subsystem must be unaffected by any connections to it. For example, the transfer function $H_1(z)$ in Fig. 7.11 should not change whether or not $H_2(z)$ is connected (and vice versa). Only when this assumption is true is the equivalent transfer function of a cascade or parallel connection given as a simple product or sum. If loading does occur between blocks, the equivalent transfer function is not so easily obtained. Fortunately, most common DT system implementations meet this requirement.

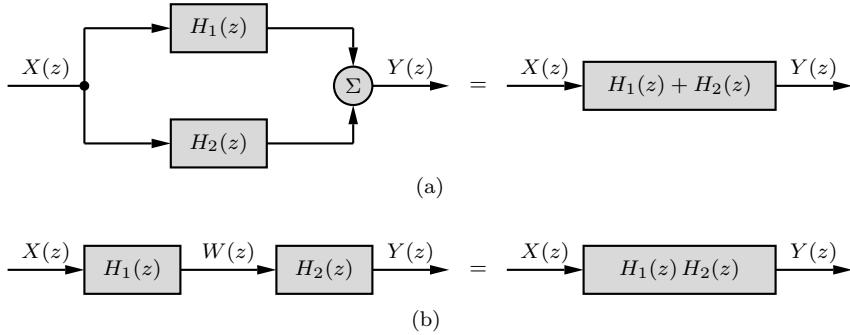


Figure 7.11: Elementary connections and their equivalents: (a) parallel and (b) cascade.

▷ **Example 7.14 (Feedback System Transfer Function)**

Verify the transfer function equivalent of the feedback system shown in Fig. 7.12.

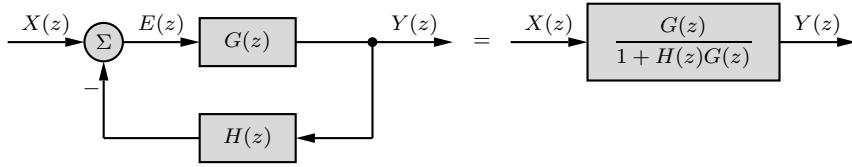


Figure 7.12: Feedback system and its equivalent.

When the output is fed back to the input, as shown in Fig. 7.12, the overall transfer function $Y(z)/X(z)$ can be computed as follows. The inputs to the adder are $X(z)$ and $-Y(z)H(z)$. Therefore, $E(z)$, the output of the adder, is

$$E(z) = X(z) - Y(z)H(z).$$

Since $Y(z) = E(z)G(z)$, it follows that

$$Y(z) = [X(z) - Y(z)H(z)]G(z).$$

Rearranging,

$$Y(z)[1 + H(z)G(z)] = X(z)G(z)$$

and

$$\frac{Y(z)}{X(z)} = \frac{G(z)}{1 + H(z)G(z)}. \quad (7.43)$$

This the transfer function equivalent of the feedback system shown in Fig. 7.12.

Example 7.14 ◀

Inverse Systems

When connected in cascade, a system $H(z)$ and its inverse $H_i(z)$ form an *identity system*, where the final output exactly equals the original input. For this to occur, the transfer function of the composite system must be unity, which is to say that $H(z)H_i(z) = 1$. Using this simple fact, we see

that the transfer function of an inverse system is simply the reciprocal of the transfer function of the original system,

$$H_i(z) = \frac{1}{H(z)}. \quad (7.44)$$

For example, an accumulator, whose transfer function is $H(z) = z/(z-1)$, and a backward difference system, whose transfer function is $H_i(z) = (z-1)/z$, are inverses of each other.

▷ Drill 7.17 (Inverse Systems)

Consider a causal and stable LTID system with transfer function

$$H(z) = \frac{z+2}{z-0.5}.$$

Determine the transfer function $H_i(z)$ of the inverse system. Explain why the inverse system $H_i(z)$ is not well behaved like $H(z)$.

△

7.5.1 Direct Form Realizations

We now develop a systematic method for realization (or simulation) of an arbitrary K th-order transfer function. Consider an LTID system with a transfer function

$$H(z) = \frac{b_0 z^L + b_1 z^{L-1} + \cdots + b_{L-1} z + b_L}{z^K + a_1 z^{K-1} + \cdots + a_{K-1} z + a_K}.$$

For causal systems, we must have $L \leq K$. With this restriction, the most general case is $K = L$, with transfer function given by Eq. (7.38). Multiplying the numerator and denominator by z^{-K} , the transfer function is expressed in terms of z^{-1} as

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{K-1} z^{-(K-1)} + b_K z^{-K}}{1 + a_1 z^{-1} + \cdots + a_{K-1} z^{-(K-1)} + a_K z^{-K}}. \quad (7.45)$$

A transfer function of this form can be realized by using adders, scalar multipliers, and delays.

Since realization is basically a synthesis problem, there is no unique way of realizing a system. A given transfer function can be realized in many different ways. To begin, we present the two forms of *direct realization*, so called because it is possible to go directly from the transfer function (or the corresponding difference equation) to the block diagram with no further manipulations. The coefficients in the difference equations (or the transfer function) appear as multiplier coefficients in these realizations. Later, we shall study some indirect forms of realization. Further, each of the direct and indirect forms can be employed in both parallel and cascade structures.

Rather than immediately realize the general system described by Eq. (7.45), we begin with the second-order case and then extend the results to the K th-order case. Second-order systems are particularly important to study since high-order designs are most often implemented using combinations of second-order systems. In the second-order case, the transfer function is given as

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}. \quad (7.46)$$

Keep in mind that we express $H(z)$ in terms of z^{-1} to produce a practical realization that uses time-delay elements rather than time-advance elements. Next, we express this transfer function as

$$H(z) = \underbrace{(b_0 + b_1 z^{-1} + b_2 z^{-2})}_{H_1(z)} \underbrace{\left(\frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}} \right)}_{H_2(z)}. \quad (7.47)$$

We can realize $H(z)$ as a cascade of transfer function $H_1(z)$ followed by $H_2(z)$, as depicted in Fig. 7.13a, where the output of $H_1(z)$ is denoted by $W(z)$. From Eq. (7.47), the transfer function $H_1(z)$ implements the *zeros* of the system, whereas $H_2(z)$ implements the *poles* of the system. Because of the commutative property of LTI system transfer functions in cascade, we can also realize $H(z)$ as a cascade of $H_2(z)$ followed by $H_1(z)$, as illustrated in Fig. 7.13b, where the (intermediate) output of $H_2(z)$ is denoted by $V(z)$.

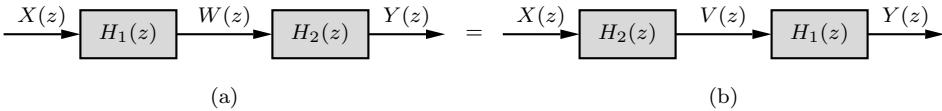


Figure 7.13: Realizations of a transfer function in two steps.

Direct Form I

We now develop a realization of $H(z)$ based on Fig. 7.13a. In this case, the output of $H_1(z)$ is given by $W(z) = H_1(z)X(z)$. Hence,

$$W(z) = (b_0 + b_1 z^{-1} + b_2 z^{-2}) X(z). \quad (7.48)$$

Further, the output of $H_2(z)$ in Fig. 7.13a is given by $Y(z) = H_2(z)W(z)$. Hence,

$$(1 + a_1 z^{-1} + a_2 z^{-2}) Y(z) = W(z). \quad (7.49)$$

We shall first realize Eq. (7.48), which shows that the output $W(z)$ can be synthesized by adding the input $b_0 X(z)$ to $b_1 z^{-1} X(z)$ and $b_2 z^{-2} X(z)$. Because the transfer function of a unit delay is z^{-1} , the signals $z^{-1} X(z)$ and $z^{-2} X(z)$ represent successive delays of the input $x[n]$. The left-half section of Fig. 7.14, which represents a realization of $H_1(z)$, shows how $W(z)$ can be synthesized from $X(z)$ using adder, scalar multiplier, and delay elements.

To complete the picture, we next realize $H_2(z)$. First, we rearrange Eq. (7.49) as

$$Y(z) = W(z) - (a_1 z^{-1} + a_2 z^{-2}) Y(z). \quad (7.50)$$

Hence, to obtain $Y(z)$, we subtract $a_1 z^{-1} Y(z)$ and $a_2 z^{-2} Y(z)$ from $W(z)$. We have already obtained $W(z)$ as the output of $H_1(z)$. To obtain signals $z^{-1} Y(z)$ and $z^{-2} Y(z)$, we assume that we already have the desired output $Y(z)$. Successive delay of $y[n]$ (i.e., successive multiplication of $Y(z)$ by z^{-1}) yields the needed signals.[†] The right-half section of Fig. 7.14, which represents a realization of $H_2(z)$, shows how we synthesize the final output $Y(z)$ from $W(z)$ according to Eq. (7.50).

Figure 7.14 depicts the so-called *direct form I* (DFI) realization of a second-order LTID system. In DFI, system zeros precede system poles, and the number of delay elements is double the order of the system. The number of delay elements is an important consideration because each delay block in a particular realization corresponds to physical memory. Thus, the second-order DFI structure of Fig. 7.14 requires four memory locations. In our next realization, direct form II, we shall realize the exact same system using half the number of delay elements, thereby reducing our memory requirements by half.

[†]It may seem odd that we first assume the existence of the output $y[n]$, delay it successively, and then in turn generate $y[n]$ from $w[n]$ and the successive delays of signal $y[n]$. This procedure poses a dilemma similar to “Which came first, the chicken or egg?” Such is the nature of recursion. The problem is satisfactorily resolved by writing the expression for $Y(z)$ at the output of the top adder of $H_2(z)$ in Fig. 7.14 and verifying that this expression is indeed the same as Eq. (7.50).

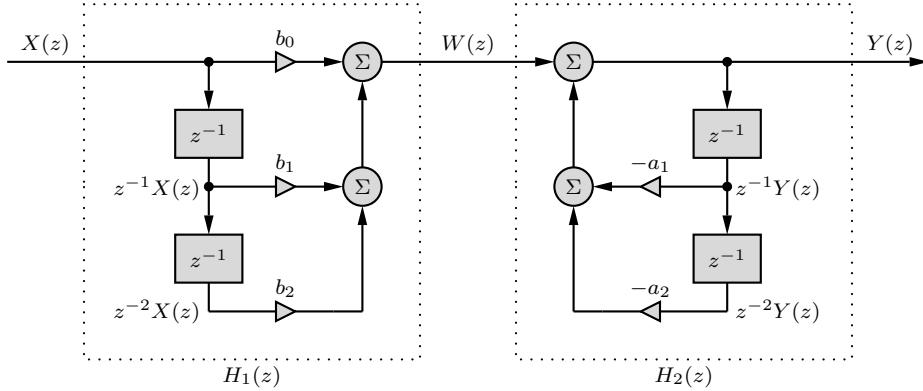


Figure 7.14: Direct form I realization of a second-order LTID system.

Direct Form II

Direct form I implements $H_1(z)$ followed by $H_2(z)$, as shown in Fig. 7.13a. We can also realize $H(z)$ as shown in Fig. 7.13b, where $H_2(z)$ precedes $H_1(z)$. This approach leads to the *direct form II* realization. To illustrate the second-order case, Fig. 7.15 interchanges $H_1(z)$ and $H_2(z)$ from the DFI case of Fig. 7.14. Instead of $H_1(z)$, it is now $H_2(z)$ that produces the intermediate variable, which we denote as $V(z)$. It is straightforward to show that equations relating $X(z)$, $V(z)$, and $Y(z)$ in Fig. 7.15 are

$$V(z) = X(z) - (a_1 z^{-1} + a_2 z^{-2}) V(z)$$

and

$$Y(z) = (b_0 + b_1 z^{-1} + b_2 z^{-2}) V(z)$$

so that, as desired,

$$\frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = H(z).$$

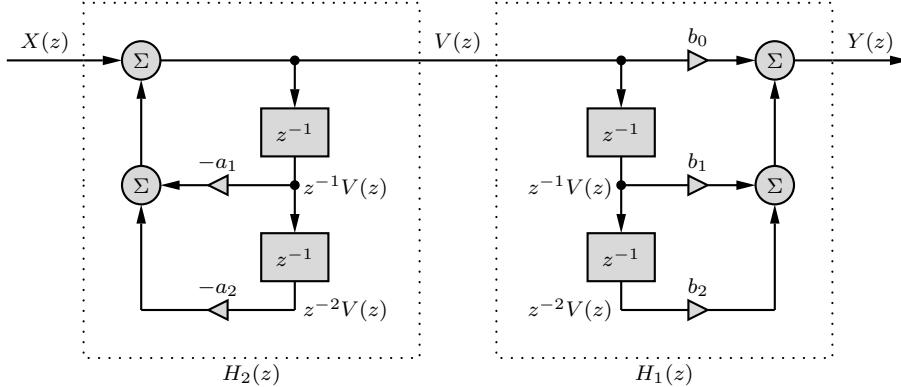


Figure 7.15: Direct form II realization with redundant delays.

An interesting observation in Fig. 7.15 is that the input signal to both chains of delays is $V(z)$. Clearly, the outputs of the delays in the left-side chain are identical to the corresponding outputs of the right-side delay chain, thus making one chain redundant. We can eliminate this chain and obtain the required signals from a single delay chain, as shown in Fig. 7.16. This modification halves the number of delays and, thus, is more efficient in hardware utilization than either Fig. 7.14 or Fig. 7.15. This is the *direct form II* (DFII) realization.

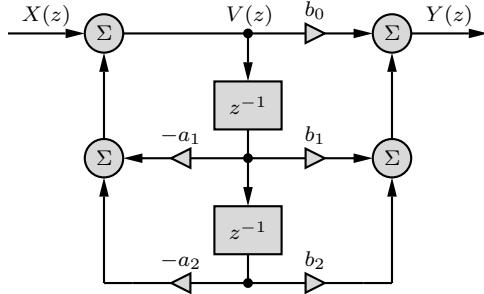
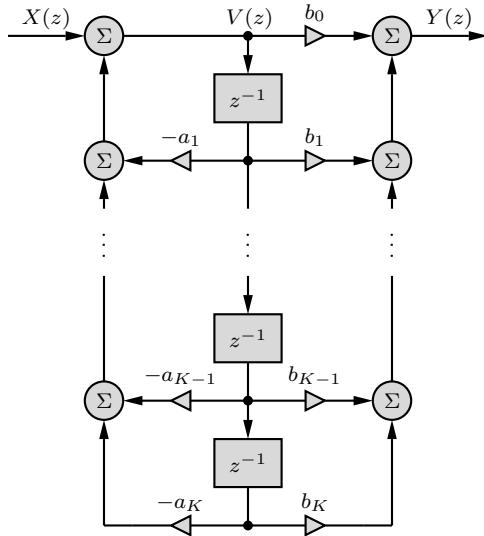


Figure 7.16: Direct form II realization of a second-order LTID system.

It is a simple task to generalize the DFI and DFII realizations for an arbitrary K th-order system: simply extend the delay-scale-sum chain to match the order of the system. The general DFII structure, for example, is shown in Fig. 7.17, and the K th-order DFI structure is shown in Fig. 4.32 (page 255).

Figure 7.17: Direct form II realization of a K th-order LTID system.

A K th-order difference equation has the property that its implementation requires a minimum of K delays. A realization is *canonic* if the number of delays used in the realization is equal to the order of the transfer function realized. Thus, a canonical realization has no redundant delays. The DFII form in Fig. 7.17 is a canonical realization and is also called the *canonical direct form*. Note that DFI realizations are non-canonical.

Although DFII is canonical and DFI is not, DFII is not automatically preferred over DFI. Both realizations result in the same transfer function, but they generally behave differently from the viewpoint of sensitivity to parameter variations. These and other factors need to be considered when choosing a particular system realization. For instance, DFI orders zeros before poles, a strategy that tends to reduce the likelihood of certain overflow errors. Would it not be nice to find a realization that is canonical and also orders zeros before poles? As we shall see next, the transposed direct form II realization achieves both these goals.

7.5.2 Transposed Realizations

The direct form II structure in Fig. 7.15 lends itself readily to elimination of half the delays, which results in the canonical direct form in Fig. 7.16. The direct form I structure in Fig. 7.14 does not permit such straightaway elimination. With a little reorganization of Fig. 7.14, however, we can eliminate the unnecessary delays and produce another canonical form.

We start with Fig. 7.14, which first realizes $H_1(z)$ followed by $H_2(z)$ with intermediate output $W(z)$. In this arrangement, the delay blocks precede the scalar multipliers b_k and $-a_k$. Let us reverse these sequences and order scalar multiplication before delay, as shown in Fig. 7.18. This slight change does not alter the overall transfer function, but it allows for the elimination of half the delay elements.

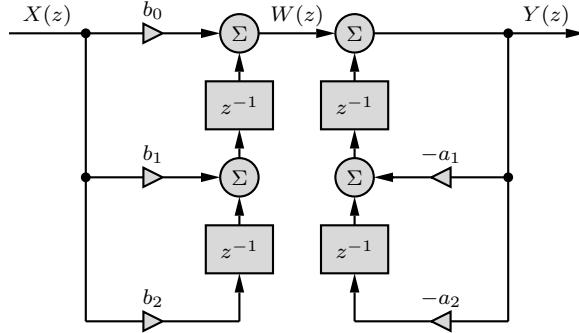


Figure 7.18: DFI with multipliers ordered before delays.

Extending this structure to a K th-order system and removing the redundant delay chain produce the canonical form shown in Fig. 7.19. This realization is known as the *transposed canonical direct form* or the *transposed direct form II* (TDFII) realization because it can be obtained from the direct form II structure by a simple transpose operation, explained next. Due to its desirable properties, TDFII is among the most popular structures to realize LTID systems.

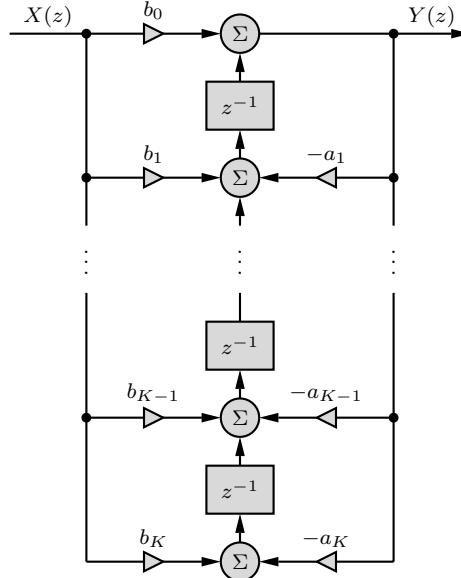


Figure 7.19: Transposed direct form II realization of a K th-order LTID system.

Transpose Operation

We can obtain the transpose of a block diagram by the following operations:

1. Reverse the directions of all paths.
2. Replace summing nodes with pick-off nodes and pick-off nodes with summing nodes.
3. Interchange the input $x[n]$ and the output $y[n]$.

The transpose operation does not affect the transfer function of a system, meaning that a system and its transpose have identical transfer functions. This observation applies to any realization, direct or not, canonic or not. The reader is encouraged to show that Fig. 7.19 is the transpose of the DFII form in Fig. 7.17, and vice versa. Also observe that the transpose of the transpose results in the original form, which is to say that the transpose operation is its own inverse.

▷ Example 7.15 (Canonical System Realizations)

Find the DFII and TDFII realizations of an LTID system with transfer function

$$H(z) = \frac{2z - 3}{4z^2 - 1}.$$

To begin, we normalize coefficient a_0 to express $H(z)$ in standard form,

$$H(z) = \frac{\frac{1}{2}z - \frac{3}{4}}{z^2 - \frac{1}{4}}.$$

From this form, we recognize the feedback and feedforward coefficients as

$$a_1 = 0 \text{ and } a_2 = -\frac{1}{4} \quad \text{and} \quad b_0 = 0, b_1 = \frac{1}{2}, \text{ and } b_2 = -\frac{3}{4}.$$

To obtain the DFII and TDFII realizations, it is simply a matter of substituting these coefficient values into the structures of Figs. 7.17 and 7.19, respectively. Figure 7.20 shows the results. Notice that because some coefficients are zero, certain paths, summing nodes, and multiplier elements are not needed.

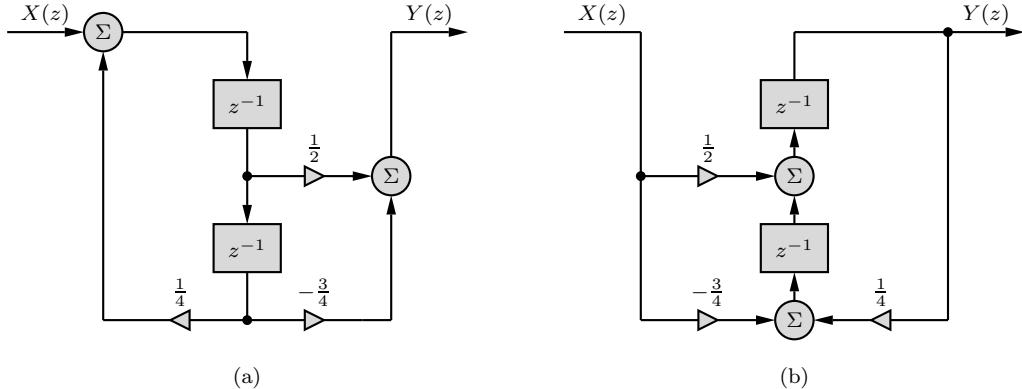


Figure 7.20: Canonical realizations of $H(z) = \frac{2z-3}{4z^2-1}$: (a) DFII and (b) TDFII.

Example 7.15 ◀

▷ **Drill 7.18 (Canonical System Realizations)**

Find the DFII and TDFII realizations of an LTID system with transfer function

$$H(z) = \frac{z}{16z^2 - 24z + 18}.$$

△

▷ **Drill 7.19 (The Unpopular Transposed Direct Form I)**

Apply the transpose operation to Fig. 7.14 to obtain the transposed direct form I (TDFI) realization of a second-order system. Provide at least two reasons why TDFI is not a popular realization choice.

△

7.5.3 Cascade and Parallel Realizations

Usually, a K th-order transfer function $H(z)$ can be expressed as a product or a sum of K first-order transfer functions. Accordingly, we can also realize $H(z)$ as a cascade (series) or parallel form of these K first-order transfer functions. Consider, for instance, the transfer function of Ex. 7.15,

$$H(z) = \frac{2z - 3}{4z^2 - 1}.$$

As one option, we can express $H(z)$ as

$$H(z) = \frac{\frac{1}{2}z - \frac{3}{4}}{(z + \frac{1}{2})(z - \frac{1}{2})} = \underbrace{\left(\frac{\frac{1}{2}}{z + \frac{1}{2}} \right)}_{H_1(z)} \underbrace{\left(\frac{z - \frac{3}{2}}{z - \frac{1}{2}} \right)}_{H_2(z)}. \quad (7.51)$$

Even in this simple case, there are an infinite number of ways to represent $H(z)$ as a cascade of two first-order systems. For example, we could also choose

$$H(z) = \left(\frac{1}{z - \frac{1}{2}} \right) \left(\frac{\frac{1}{2}z - \frac{3}{4}}{z + \frac{1}{2}} \right).$$

Later, we discuss strategies to choose effective combinations. We can also express $H(z)$ as a sum of partial fractions as

$$H(z) = \frac{\frac{1}{2}z - \frac{3}{4}}{(z + \frac{1}{2})(z - \frac{1}{2})} = \underbrace{\frac{1}{z + \frac{1}{2}}}_{H_3(z)} + \underbrace{\frac{-\frac{1}{2}}{z - \frac{1}{2}}}_{H_4(z)}. \quad (7.52)$$

Equation (7.51) lets us realize $H(z)$ as a cascade of $H_1(z)$ and $H_2(z)$, as shown in Fig. 7.21a, whereas Eq. (7.52) leads to a parallel realization of $H_3(z)$ and $H_4(z)$, as depicted in Fig. 7.21b. Each of the first-order transfer functions in Fig. 7.21 can be realized by using any form desired. For example, we can realize $H_1(z)$ and $H_2(z)$ as DFII, $H_3(z)$ as DFI, and $H_4(z)$ as TDFII, or we can use any other combination imaginable.

In the preceding example of cascade and parallel realizations, we separated $H(z)$ into first-order factors. For higher orders, we can separate $H(z)$ into factors, not all of which are necessarily of the first order. For example, we can realize a third-order transfer function $H(z)$ as a cascade or parallel combination of a first-order factor and a second-order factor. As we shall see next, second-order factors are particularly convenient for systems that possess complex roots.

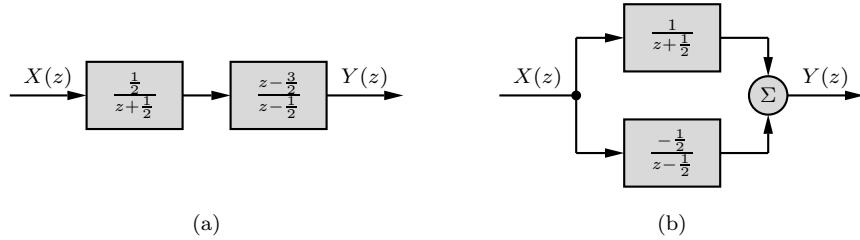


Figure 7.21: Realization of $H(z) = \frac{2z^2 - 3}{4z^2 - 1}$: (a) cascade form and (b) parallel form.

Realization of Complex-Conjugate Roots

Consider a system with transfer function given as

$$H(z) = \frac{z^3 + z}{16z^3 - 28z^2 + 20z - 6}.$$

To realize this system, we might separate $H(z)$ into a product or sum of first-order factors as

$$\begin{aligned} H(z) &= \left(\frac{\frac{1}{16}z}{z - \frac{3}{4}} \right) \left(\frac{z + j}{z - (\frac{1}{2} + j\frac{1}{2})} \right) \left(\frac{z - j}{z - (\frac{1}{2} - j\frac{1}{2})} \right) \\ &= \frac{\frac{5}{16}z}{z - \frac{3}{4}} + \frac{-\frac{1}{8}z}{z - (\frac{1}{2} + j\frac{1}{2})} + \frac{-\frac{1}{8}z}{z - (\frac{1}{2} - j\frac{1}{2})}. \end{aligned} \quad (7.53)$$

Written in this way as a product or a sum, the resulting realization is either a cascade or a parallel combination of three first-order subsystems. Although digital systems are flexible enough to accommodate such realizations, the presence of complex-valued scale factors greatly complicates the implementation. Even though the overall system is real (meaning that real inputs always generate real outputs), these implementations require complex signal paths, complex adders, and complex multipliers. This, in turn, can double the memory requirements and computational burden of the implementation. It is generally better practice to combine complex-conjugate roots and thereby eliminate the need for complex-valued signals and operations. In the present case, we can express $H(z)$ as

$$H(z) = \left(\frac{\frac{1}{16}z}{z - \frac{3}{4}} \right) \left(\frac{z^2 + 1}{z^2 - z + \frac{1}{2}} \right) = \frac{\frac{5}{16}z}{z - \frac{3}{4}} + \frac{-\frac{1}{4}z^2 + \frac{1}{8}z}{z^2 - z + \frac{1}{2}}.$$

The resulting cascade and parallel realizations involve only real signals and operations.

Realization of Repeated Poles

When repeated poles occur, the procedure for a cascade realization is exactly the same as discussed earlier. For a parallel realization, however, the procedure requires a special precaution, as explained in the following example.

▷ Example 7.16 (Parallel Realization of Repeated Poles)

Determine a parallel realization of

$$H(z) = \frac{\frac{3}{16}(z^2 + 1)}{(z + \frac{1}{2})(z - \frac{1}{2})^2}.$$

First, we expand $H(z)$ into partial fractions using the `residue` command in MATLAB.

```

01 format rat; [r,p,k] = residue(3/16*[1 0 1],poly([-1/2 1/2 1/2]))
r = -3/64 15/64 15/64
p = 1/2 1/2 -1/2
k = []

```

Thus,

$$H(z) = \frac{-\frac{3}{64}}{z - \frac{1}{2}} + \frac{\frac{15}{64}}{(z - \frac{1}{2})^2} + \frac{\frac{15}{64}}{z + \frac{1}{2}}.$$

This third-order transfer function should require no more than three delays. Realizing each of the three partial fractions separately, however, requires four delays because one of the terms is second order. This difficulty can be avoided by sharing the $\frac{-\frac{3}{64}}{z - \frac{1}{2}}$ term with two paths, as shown in Fig. 7.22. Each of the three first-order transfer functions in Fig. 7.22 may now be realized using whatever desired form, such as DFII or TDFII.

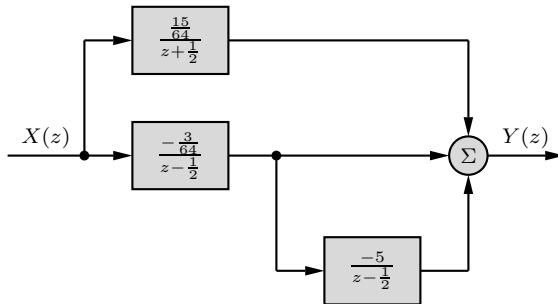


Figure 7.22: Parallel realization of $H(z) = \frac{\frac{3}{16}(z^2+1)}{(z+\frac{1}{2})(z-\frac{1}{2})^2}$.

Example 7.16 ◀

Realization of FIR Filters

So far we have been quite general in our development of realization techniques. They can be applied to IIR or FIR filters. For FIR filters, a_0 is normalized to unity, and the remaining coefficients $a_k = 0$ for all $k \neq 0$. Hence, FIR filters can be readily implemented using the schemes developed so far by eliminating all branches with a_k coefficients. The implication of the fact that $a_k = 0$ is that all the poles of an FIR filter are at $z = 0$.

▷ Example 7.17 (FIR Filter Realization)

Using direct, transposed direct, and cascade forms, realize the FIR filter with transfer function

$$H(z) = \frac{\frac{1}{2}z^3 + \frac{1}{4}z^2 + \frac{1}{8}z + \frac{1}{16}}{z^3} = \frac{1}{2} + \frac{1}{4}z^{-1} + \frac{1}{8}z^{-2} + \frac{1}{16}z^{-3}.$$

From the transfer function $H(z)$, we recognize that $a_k = 0$ for all $k \neq 0$ and that $b_0 = \frac{1}{2}$, $b_1 = \frac{1}{4}$, $b_2 = \frac{1}{8}$, and $b_3 = \frac{1}{16}$. Substituting these values into either of the two direct form structures (DFI or DFII) produces the same direct form realization, as shown in Fig. 7.23a. This figure emphasizes that an FIR filter is basically a tapped delay line, which is why this structure is also known as a *tapped delay line* or *transversal filter*. Figure 7.23b shows the corresponding transposed implementation, which is nearly identical to the direct form.

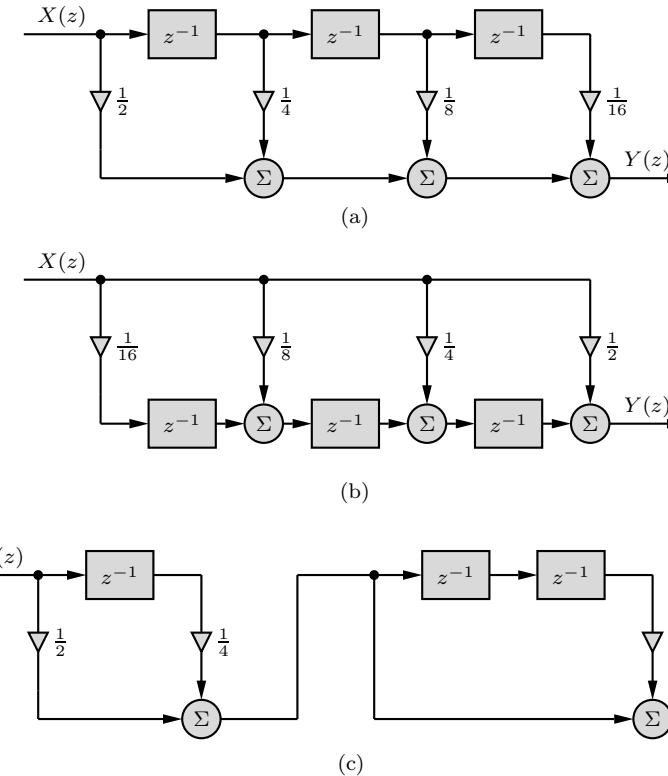


Figure 7.23: Realizations of $H(z) = \frac{1}{2} + \frac{1}{4}z^{-1} + \frac{1}{8}z^{-2} + \frac{1}{16}z^{-3}$: (a) direct, (b) transposed, and (c) cascade.

To obtain a cascade form of this filter, we first express the transfer function as

$$H(z) = \left(\frac{\frac{1}{2}z + \frac{1}{4}}{z} \right) \left(\frac{z^2 + \frac{1}{4}}{z^2} \right).$$

We use a second-order factor rather than two first-order factors to accommodate the system's complex-conjugate roots without using complex multipliers, complex adders, or complex signal paths. Figure 7.23c shows the resulting realization, where the two cascaded subsystems are each implemented in direct form. For this particular system $H(z)$, notice that the cascade form offers a computational advantage over the direct or transposed direct forms: the required number of multiplies and additions are each reduced by one. Although such savings are not guaranteed in general, it can be worthwhile to seek them out.

Example 7.17 \triangleleft

▷ Drill 7.20 (A Non-Realizable Transfer Function)

Explain why there is no practical way to realize the LTID system that has the transfer function

$$H(z) = \frac{z^2 + \frac{1}{4}}{z}.$$

\triangleleft

Do All Realizations Lead to the Same Performance?

For a given transfer function, we have presented here several possible different realizations such as DFI and the canonic form DFII. There are also cascade and parallel versions, and there are many possible groupings of the factors in the numerator and the denominator of $H(z)$, each leading to different realizations. We can also use various combinations of these forms in implementing different subsections of a system. Moreover, the transpose of each version doubles the number. However, this discussion by no means exhausts all the possibilities. By transforming variables, there are limitless possible realizations of the same transfer function.

Theoretically, all these realizations are equivalent; that is, they lead to the same transfer function. This, however, is true only when we implement them with infinite precision. In practice, finite word-length restrictions cause each realization to behave differently in terms of sensitivity to parameter variation, frequency response distortion error, and so on. These effects are serious for higher-order transfer functions, which require a correspondingly higher number of delay elements. The finite word-length errors that plague these implementations include coefficient quantization, overflow errors, and round-off errors. From a practical viewpoint, parallel and cascade forms using low-order filters minimize finite word-length effects. Parallel and certain cascade forms are numerically less sensitive than the canonic direct form to small parameter variations in the system. In a canonic direct form structure with large K , a small change in a filter coefficient due to parameter quantization results in a large change in the location of the poles and the zeros of the system. Qualitatively, this difference can be explained by the fact that in a direct form (or its transpose), all the coefficients interact with each other, and a change in any coefficient will be magnified through its repeated influence from feedback and feedforward connections. In a parallel realization, in contrast, a change in a coefficient will affect only a localized segment; the case with a cascade realization is similar. For this reason, the most popular technique for minimizing finite word-length effects is to design filters using cascade or parallel forms employing low-order filters. In practice, high-order filters are most commonly realized using a cascade of multiple second-order sections, which are not only easier to design but are also less susceptible to coefficient quantization and round-off errors. Their implementations allow easier data word scaling to reduce the potential of various overflow effects. Further, a cascaded system using second-order sections usually requires fewer multiplications for a given filter frequency response [2].

There are several ways to pair the poles and zeros of a K th-order $H(z)$ into a cascade of second-order sections and several ways to order the resulting sections. Quantization errors will be different for each of the combinations. Although several published papers provide guidelines for predicting and minimizing finite word-length errors, it is advisable to perform computer simulation of the filter design. This way, one can vary filter hardware characteristics, such as coefficient word lengths, accumulator register sizes, sequence of cascaded sections, and input signal sets. Such an approach is both reliable and economical [2].

A Magical Art?

Given the enormous options for realizing even relatively low-order designs, it may seem that one needs to be a magician to properly realize a digital filter. While there is certainly an art to digital filter realization, one need not devote a lifetime of study to the tomes of digital filters in order to realize good designs. In Ch. 8, we present a simple procedure for pole-zero pairing and section ordering that generally produces agreeable system behavior.

7.6 Frequency Response of Discrete-Time Systems

In Sec. 5.5.4, we showed that an LTID system response to an (everlasting) exponential z^n is also an (everlasting) exponential $H(z)z^n$. This result is valid only for those values of z for which $H(z)$, as defined in Eq. (7.9), exists (converges). As usual, we represent this input-output relationship by a

directed arrow notation as

$$z^n \implies H(z)z^n.$$

Setting $z = e^{\pm j\Omega}$ in this relationship yields

$$e^{j\Omega n} \implies H(e^{j\Omega})e^{j\Omega n} \quad \text{and} \quad e^{-j\Omega n} \implies H(e^{-j\Omega})e^{-j\Omega n}. \quad (7.54)$$

For real systems, addition of these two equations yields

$$2 \cos(\Omega n) \implies H(e^{j\Omega})e^{j\Omega n} + H(e^{-j\Omega})e^{-j\Omega n} = 2\operatorname{Re}\{H(e^{j\Omega})e^{j\Omega n}\}. \quad (7.55)$$

Expressing $H(e^{j\Omega})$ in polar form, we see that

$$H(e^{j\Omega}) = |H(e^{j\Omega})|e^{j\angle H(e^{j\Omega})}. \quad (7.56)$$

Thus, Eq. (7.55) can be expressed as

$$\cos(\Omega n) \implies |H(e^{j\Omega})| \cos[\Omega n + \angle H(e^{j\Omega})]$$

or, more generally, as

$$\cos(\Omega n + \theta) \implies |H(e^{j\Omega})| \cos[\Omega n + \theta + \angle H(e^{j\Omega})]. \quad (7.57)$$

This result is valid only for BIBO stable or asymptotically stable systems since only in these cases does the region of convergence for $H(z)$ include $z = e^{j\Omega}$. For BIBO unstable systems, which include marginally stable and asymptotically unstable systems, the ROC for $H(z)$ does not include the unit circle where $z = e^{j\Omega}$. In other words, $H(e^{j\Omega})$ is meaningless if $z = e^{j\Omega}$ is not in the ROC of $H(z)$.

Equation (7.57) demonstrates an important idea: *the response of a real and asymptotically or BIBO stable LTID system to a sinusoidal input is a sinusoid of identical frequency, modified only in gain and phase.* Only exponentials of the form z^n , of which sinusoids are a special case, have this amazing and guaranteed ability to pass through an LTID system undistorted in their underlying shape. The magnitude of the output sinusoid is $|H(e^{j\Omega})|$ times the input magnitude, and a plot of $|H(e^{j\Omega})|$ versus Ω is the magnitude response of the discrete-time system. Similarly, $\angle H(e^{j\Omega})$ tells how the system modifies or shifts the phase of an input sinusoid, and a plot of $\angle H(e^{j\Omega})$ versus Ω is the phase response of the discrete-time system. As shown by Eq. (7.56), $H(e^{j\Omega})$ incorporates both the magnitude response and the phase response and therefore is called the *frequency response* of the system.

Steady-State Response to Causal Sinusoidal Inputs

As in the case of continuous-time systems, we can show that the response of an LTID system to a causal sinusoidal input $\cos(\Omega n + \theta)u[n]$ is the output in Eq. (7.57) plus a natural component consisting of the characteristic modes (see Prob. 7.6-1). For a stable system, all the modes decay exponentially, and only the sinusoidal component of Eq. (7.57) persists. For this reason, this component is called the sinusoidal *steady-state* response of the system. Thus, $y_{ss}[n]$, the steady-state response of a system to a causal sinusoidal input $\cos(\Omega n + \theta)u[n]$, is

$$y_{ss}[n] = |H(e^{j\Omega})| \cos[\Omega n + \theta + \angle H(e^{j\Omega})] u[n].$$

System Response to Sampled Continuous-Time Sinusoids

So far we have considered the response of a discrete-time system to a discrete-time sinusoid $\cos(\Omega n)$ (or exponential $e^{j\Omega n}$). In practice, the input may be sampled from a continuous-time sinusoid $\cos(\omega t)$ (or exponential $e^{j\omega t}$). When a sinusoid $\cos(\omega t)$ is sampled with sampling interval T , the resulting

signal is a discrete-time sinusoid $\cos(\omega nT)$, obtained by setting $t = nT$ in $\cos(\omega t)$. Therefore, all the results developed in this section apply if we substitute ωT for Ω , that is,

$$\Omega = \omega T. \quad (7.58)$$

From Eq. (7.58), we see that DT frequency Ω is just CT frequency ω scaled by the sampling interval T . We have seen this relationship several times before in Ch. 4 (see Eq. (4.15)) as well as Ch. 6.

▷ **Example 7.18 (Determining and Using Frequency Response)**

Determine the frequency response $H(e^{j\Omega})$ of the system specified by the equation

$$y[n+1] - 0.8y[n] = x[n+1].$$

Using $H(e^{j\Omega})$, determine the system responses to the inputs (a) $x_a[n] = 1^n = 1$, (b) $x_b[n] = \cos(\frac{\pi}{6}n - 0.2)$, and (c) $x_c(t) = \cos(1500t)$ sampled using sampling interval $T = 0.001$.

In advance-operator form, the system equation is expressed as

$$(E - 0.8) \{y[n]\} = E \{x[n]\}.$$

Therefore, the transfer function of the system is

$$H(z) = \frac{z}{z - 0.8} = \frac{1}{1 - 0.8z^{-1}}.$$

Letting $z = e^{j\Omega}$, the frequency response is

$$H(e^{j\Omega}) = \frac{1}{1 - 0.8e^{-j\Omega}} = \frac{1}{1 - 0.8 \cos(\Omega) + j0.8 \sin(\Omega)}, \quad (7.59)$$

and the magnitude and phase responses are thus

$$|H(e^{j\Omega})| = \frac{1}{\sqrt{1.64 - 1.6 \cos(\Omega)}} \quad \text{and} \quad \angle H(e^{j\Omega}) = -\tan^{-1} \left(\frac{0.8 \sin(\Omega)}{1 - 0.8 \cos(\Omega)} \right). \quad (7.60)$$

MATLAB easily computes and graphs the magnitude and phase responses, which are shown in Figs. 7.24a and 7.24b, respectively.

```
01 Omega = linspace(-2*pi,2*pi,500); H = 1./(1-0.8*exp(-1j*Omega));
02 subplot(121); plot(Omega,abs(H)); subplot(122); plot(Omega,angle(H));
```

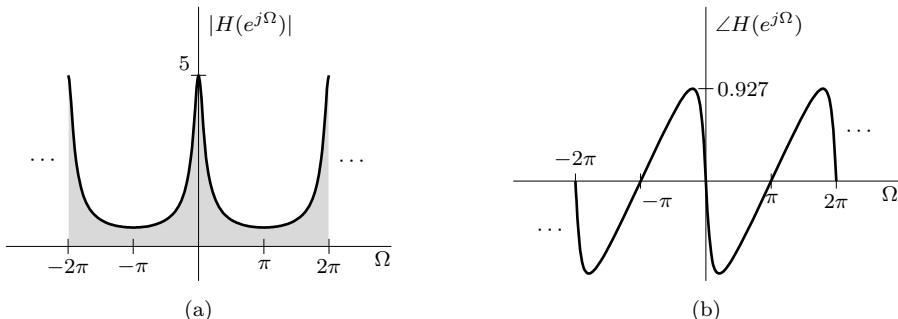


Figure 7.24: Frequency response of $y[n+1] - 0.8y[n] = x[n+1]$: (a) $|H(e^{j\Omega})|$ and (b) $\angle H(e^{j\Omega})$.

We now use $|H(e^{j\Omega})|$ and $\angle H(e^{j\Omega})$ to compute the system responses to the given inputs.

(a) Since $1^n = (e^{j\Omega})^n$ with $\Omega = 0$, the response to $x_a[n] = 1^n = 1$ is found using $H(e^{j0})$. From Eq. (7.59), we obtain

$$H(e^{j0}) = \frac{1}{\sqrt{1.64 - 1.6 \cos(0)}} = \frac{1}{\sqrt{0.04}} = 5 = 5\angle 0.$$

Therefore,

$$|H(e^{j0})| = 5 \quad \text{and} \quad \angle H(e^{j0}) = 0.$$

These values also can be read directly from Figs. 7.24a and 7.24b, respectively. Using Eq. (7.54), the system response to input $x_a[n] = 1^n = 1$ is therefore

$$y_a[n] = 5(1^n) = 5.$$

(b) The frequency of $x_b[n] = \cos(\frac{\pi}{6}n - 0.2)$ is $\Omega = \frac{\pi}{6}$. Using Eq. (7.60),

$$|H(e^{j\pi/6})| = \frac{1}{\sqrt{1.64 - 1.6 \cos(\frac{\pi}{6})}} = 1.9828$$

and

$$\angle H(e^{j\pi/6}) = -\tan^{-1} \left(\frac{0.8 \sin(\frac{\pi}{6})}{1 - 0.8 \cos(\frac{\pi}{6})} \right) = -0.9159.$$

Figures 7.24a and 7.24b confirm these calculations. Using Eq. (7.57), the system response to input $x_b[n] = \cos(\frac{\pi}{6}n - 0.2)$ is therefore

$$y_b[n] = 1.9828 \cos \left(\frac{\pi}{6}n - 0.2 - 0.9159 \right) = 1.9828 \cos \left(\frac{\pi}{6}n - 1.1159 \right).$$

(c) The sinusoid $x_c = \cos(1500t)$ sampled every T seconds ($t = nT$) results in the discrete-time sinusoid

$$x_c[n] = \cos(1500nT).$$

For $T = 0.001$, the DT system input is therefore

$$x_c[n] = \cos(1.5n).$$

In this case, $\Omega = 1.5$. According to Eq. (7.60),

$$|H(e^{j1.5})| = \frac{1}{\sqrt{1.64 - 1.6 \cos(1.5)}} = 0.8093$$

and

$$\angle H(e^{j1.5}) = -\tan^{-1} \left(\frac{0.8 \sin(1.5)}{1 - 0.8 \cos(1.5)} \right) = -0.7021.$$

Again, Figs. 7.24a and 7.24b confirm these calculations. Using Eq. (7.57), the system response to input $x_c[n] = \cos(1.5n)$ is therefore

$$y_c[n] = 0.8093 \cos(1.5n - 0.7021).$$

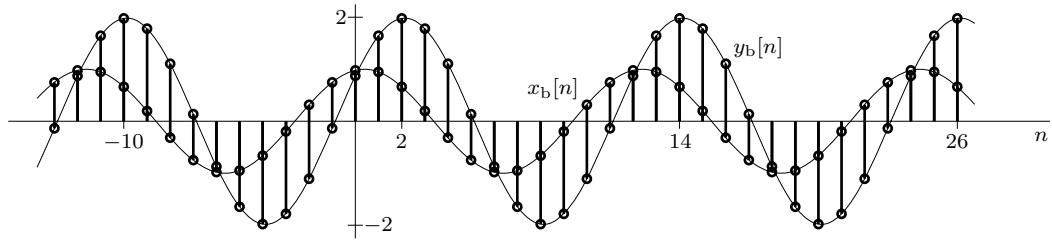


Figure 7.25: Real LTID systems modify sinusoidal inputs only in gain and phase.

The Essence of Frequency Response

As these examples emphasize, the response of a real LTID system to an everlasting sinusoid (or exponential) is the same sinusoid (or exponential), modified only in gain and phase. This is the essence of the frequency-response concept. Stated another way, the underlying shape of an everlasting exponential input does not change as it passes through a stable LTID system. Figure 7.25 illustrates this idea graphically for the input $x_b[n] = \cos(\frac{\pi}{6}n - 0.2)$ and its corresponding output $y_b[n] = 1.9828 \cos(\frac{\pi}{6}n - 1.1159)$.

Example 7.18 \triangleleft

Discrete in Time, Not in Frequency

There is sometimes a misconception that a discrete-time system should be discrete in its frequency response as well. As the magnitude and phase response plots of Fig. 7.24 indicate, *the frequency response of a discrete-time system is a continuous (rather than discrete) function of frequency Ω* . There is no contradiction here. This behavior is merely reflects the fact that a discrete-time sinusoid is unrestricted in its choice of frequency, and therefore, the system response also exists for every value of Ω .

The Periodic Nature of DT System Frequency Response

Figure 7.24 shows that for the system in Ex. 7.18, the frequency response $H(e^{j\Omega})$ is a 2π -periodic function of Ω . This observation, which holds true for DT systems in general, should not surprise us. In Sec. 4.2, we showed that discrete-time sinusoids separated by values of Ω in integral multiples of 2π are identical. Therefore, the system response to such sinusoids (or exponentials) is also identical. This fact also follows from the very structure of the frequency response $H(e^{j\Omega n})$. Its argument $e^{j\Omega n}$ is a periodic function of Ω with period 2π . This fact automatically renders $H(e^{j\Omega n})$ periodic. Due to this 2π -periodic nature, it is only necessary to plot the frequency response of discrete-time systems over a 2π range such as $-\pi \leq \Omega < \pi$ or $0 \leq \Omega < 2\pi$.

▷ Drill 7.21 (Determining and Using Frequency Response)

Determine and plot the magnitude response $|H(e^{j\Omega})|$ and phase response $\angle H(e^{j\Omega})$ of the system specified by the equation

$$y[n+1] - 0.5y[n] = x[n].$$

Determine the system response to the sinusoidal input $x(t) = \cos(1000t - \frac{\pi}{3})$ sampled every $T = 0.5$ ms.

\triangleleft

▷ **Drill 7.22 (Frequency Response of Unit Delay System)**

Determine the frequency response of a unit-delay system

$$y[n] = x[n - 1].$$

How does this system affect the magnitude and phase of an input sinusoid?



7.6.1 Frequency-Response from Pole-Zero Locations

The frequency response of a system is determined by the pole and zero locations of the transfer function $H(z)$. It is possible to determine quickly the magnitude and phase responses and to obtain physical insight into the filter characteristics of a discrete-time system by using a graphical technique. The general K th-order transfer function $H(z)$ in Eq. (7.38) can be expressed in factored form as

$$H(z) = b_0 \frac{(z - z_1)(z - z_2) \cdots (z - z_K)}{(z - p_1)(z - p_2) \cdots (z - p_K)} = b_0 \frac{\prod_{l=1}^K (z - z_l)}{\prod_{k=1}^K (z - p_k)}. \quad (7.61)$$

To see how Eq. (7.61) helps us determine a system's frequency response, we must develop a graphical understanding of terms in the form $(z - \gamma)$. As shown in Fig. 7.26a, the directed line segment from γ to z in the complex plane represents the complex number $(z - \gamma)$. The length of this segment is $|z - \gamma|$, and its angle with the horizontal axis is $\angle(z - \gamma)$.

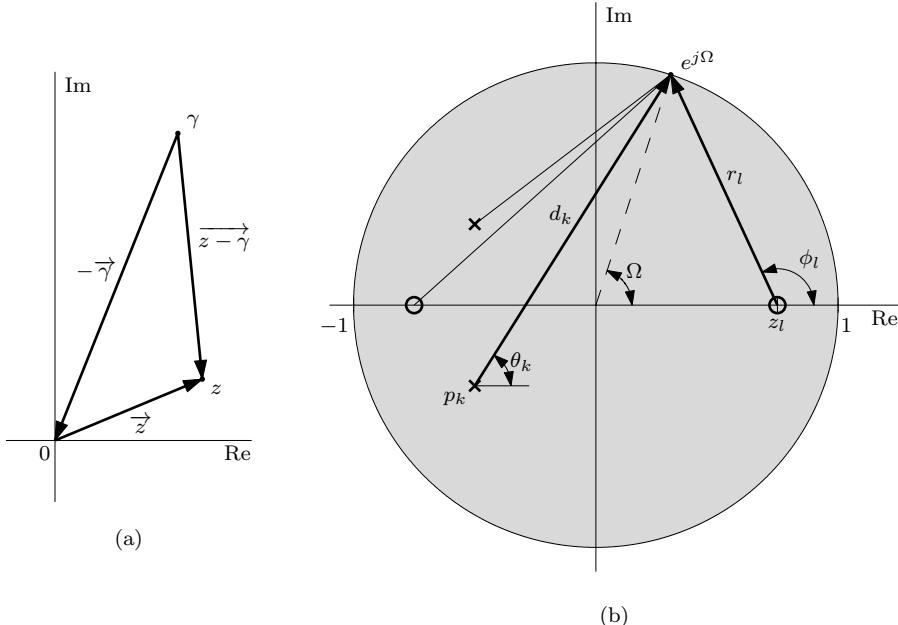


Figure 7.26: Vector representations of (a) complex numbers and (b) factors of $H(z)$.

To compute the frequency response $H(e^{j\Omega})$, we evaluate $H(z)$ at $z = e^{j\Omega}$. For $z = e^{j\Omega}$, $|z| = 1$ and $\angle z = \Omega$ so that $z = e^{j\Omega}$ represents a point on the unit circle at an angle Ω with the horizontal. We now connect all zeros (z_1, z_2, \dots, z_K) and all poles (p_1, p_2, \dots, p_K) to the point $e^{j\Omega}$, as indicated in Fig. 7.26b. Let r_1, r_2, \dots, r_K be the lengths and $\phi_1, \phi_2, \dots, \phi_K$ be the angles, respectively, of

the straight lines connecting z_1, z_2, \dots, z_K to the point $e^{j\Omega}$. Similarly, let d_1, d_2, \dots, d_K be the lengths and $\theta_1, \theta_2, \dots, \theta_K$ be the angles, respectively, of the lines connecting p_1, p_2, \dots, p_K to $e^{j\Omega}$. From Eq. (7.61),

$$\begin{aligned} H(e^{j\Omega}) &= H(z)|_{z=e^{j\Omega}} = b_0 \frac{(r_1 e^{j\phi_1})(r_2 e^{j\phi_2}) \cdots (r_K e^{j\phi_K})}{(d_1 e^{j\theta_1})(d_2 e^{j\theta_2}) \cdots (d_K e^{j\theta_K})} = b_0 \frac{\prod_{l=1}^K r_l e^{j\phi_l}}{\prod_{k=1}^K d_k e^{j\theta_k}} \\ &= b_0 \frac{r_1 r_2 \cdots r_K}{d_1 d_2 \cdots d_K} e^{j[(\phi_1 + \phi_2 + \cdots + \phi_K) - (\theta_1 + \theta_2 + \cdots + \theta_K)]}. \end{aligned}$$

The magnitude response is therefore

$$\begin{aligned} |H(e^{j\Omega})| &= |b_0| \frac{r_1 r_2 \cdots r_K}{d_1 d_2 \cdots d_K} = |b_0| \frac{\prod_{l=1}^K r_l}{\prod_{k=1}^K d_k} \\ &= |b_0| \frac{\text{product of the distances of zeros to } e^{j\Omega}}{\text{product of the distances of poles to } e^{j\Omega}}, \end{aligned} \quad (7.62)$$

and the phase response is

$$\begin{aligned} \angle H(e^{j\Omega}) &= \angle b_0 + (\phi_1 + \phi_2 + \cdots + \phi_K) - (\theta_1 + \theta_2 + \cdots + \theta_K) = \angle b_0 + \sum_{l=1}^K \phi_l - \sum_{k=1}^K \theta_k \\ &= \angle b_0 + \text{sum of zero angles to } e^{j\Omega} - \text{sum of pole angles to } e^{j\Omega}. \end{aligned} \quad (7.63)$$

In this manner, we can compute the frequency response $H(e^{j\Omega})$ for any value of Ω . To obtain a complete picture of a system's frequency response, we repeat the calculations of Eqs. (7.62) and (7.63) for all values of Ω over a suitable 2π interval, such as $-\pi$ to π .

Controlling Gain by Placement of Poles and Zeros

With minor differences, the nature of the influence of pole and zero locations on frequency response is similar to that observed in continuous-time systems. In place of the imaginary axis for continuous-time systems, we have the unit circle in the discrete-time case. The nearer a pole (or zero) is to the point $e^{j\Omega}$, the more influence that pole (or zero) yields on the magnitude response at the frequency Ω because the length of the vector joining that pole (or zero) to the point $e^{j\Omega}$ is small. The proximity of a pole (or zero) has a similar effect on the phase response.

From Eq. (7.62), it is clear that to enhance the magnitude response at a frequency Ω , we should place a pole close to the point $e^{j\Omega}$. Of course, the closest we can place a pole is on the unit circle at the point representing Ω . Although this leads to maximum (infinite) gain, it also renders the system marginally stable (BIBO unstable) and should be avoided.[†] By similar logic, to suppress the magnitude response at a frequency Ω , we should place a zero close to the point $e^{j\Omega}$. Unlike poles, there is no particular disadvantage to placing zeros on (or outside) the unit circle. In fact, total suppression of signal transmission at a given frequency can only be achieved by placing a zero at the corresponding point on the unit circle. This is the underlying principle of a notch (bandstop) filter. Repeating poles or zeros further enhances their influence. Notice that a pole at a point has the opposite (reciprocal) effect of a zero at that point. Placing a zero close to a pole tends to cancel the effect of that pole on the frequency response (and vice versa).

Placing a pole or a zero at the origin does not influence the magnitude response because the length of the vector connecting the origin to any point on the unit circle is unity. However, a pole (or zero) at the origin adds angle $-\Omega$ (or Ω) to the phase response $\angle H(e^{j\Omega})$. As seen in Drill 7.22, a linear phase response corresponds to a pure time delay (or time advance) of T seconds. Therefore, a pole (or zero) at the origin causes a time delay (or advance) of T seconds in the response.

[†]Additionally, the closer a pole is to the unit circle, the more sensitive is the system gain to parameter variations.

For a stable system, all the poles must be located inside the unit circle. The zeros may lie anywhere. Also, for a physically realizable system, $H(z)$ must be a proper fraction; that is, $K \geq L$. If, to achieve a certain magnitude response, we require $L > K$, we can still make the system realizable by placing a sufficient number of poles at the origin. This will not change the magnitude response, but it will increase the time delay of the response.

Let us consider these ideas by investigating the examples shown in Fig. 7.27. Keep in mind that although it is possible to effectively design low-order filters using a graphical placement of poles and zeros, high-order designs are basically impossible using this ad hoc method. Chapter 8 introduces systematic and formal methods to design quality filters of arbitrary order.

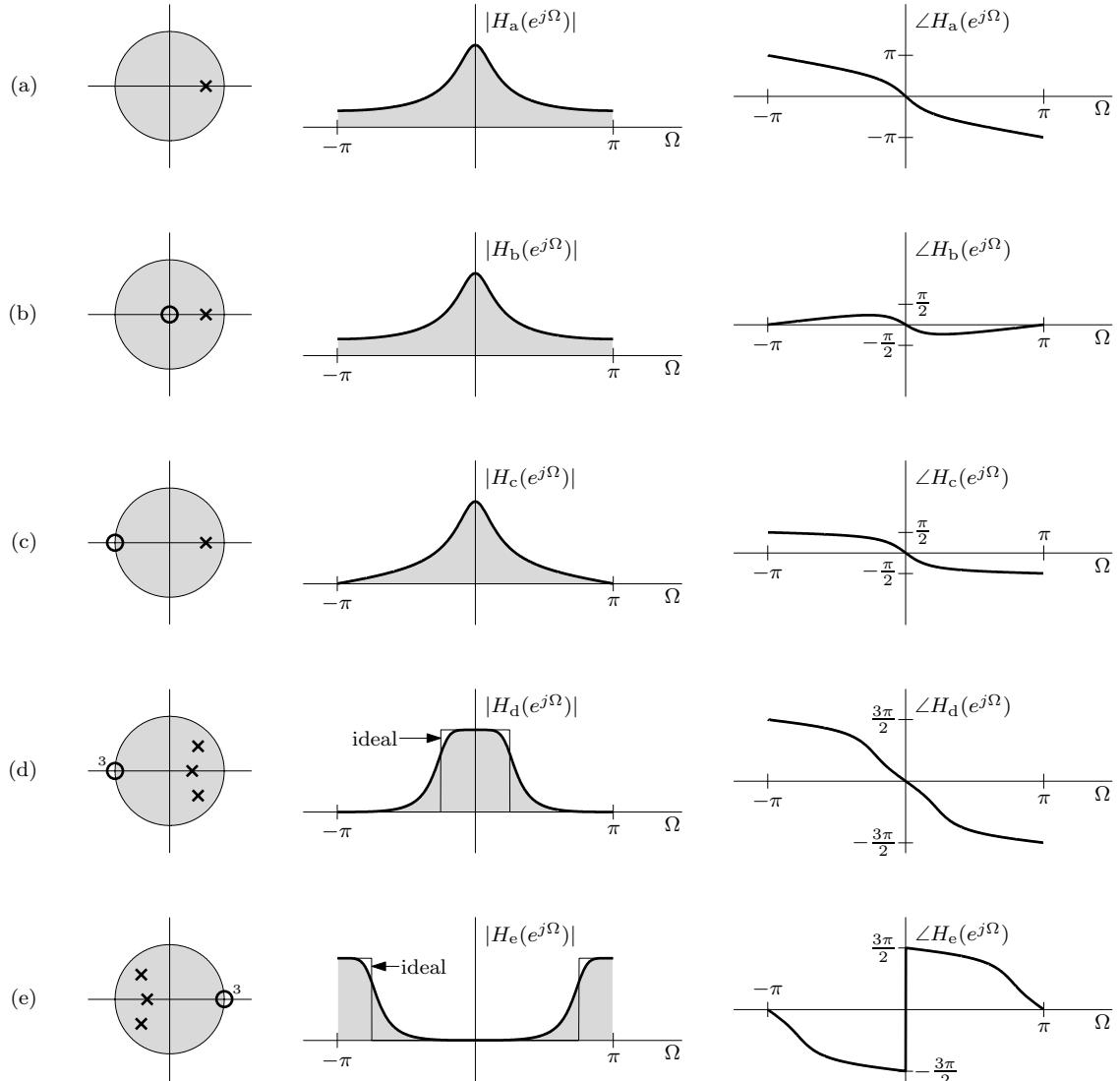


Figure 7.27: Various pole-zero configurations with corresponding magnitude and phase responses.

Lowpass Filters

A lowpass filter generally has a maximum gain at or near $\Omega = 0$, which corresponds to point $e^{j0} = 1$ on the unit circle. Clearly, placing a pole inside the unit circle near the point $z = 1$ encourages a lowpass response. Figure 7.27a shows a simple case of a single pole at $z = \frac{2}{3}$ and the corresponding magnitude and phase responses. For small values of Ω , the point $e^{j\Omega}$ (a point on the unit circle at an angle Ω) is close to the pole, and consequently, the gain is high. As Ω increases, the distance of the point $e^{j\Omega}$ from the pole increases. Consequently, the gain decreases, resulting in a lowpass characteristic.

Placing a zero at the origin does not change the magnitude response, but it does modify the phase response, as illustrated in Fig. 7.27b. Placing a zero at $z = -1$, however, changes both the magnitude and phase responses (Fig. 7.27c). The point $z = -1$ corresponds to frequency $\Omega = \pi$ ($z = e^{j\Omega} = e^{j\pi} = -1$). Consequently, the magnitude response now becomes more attenuated at higher frequencies, with a zero gain at $\Omega = \pi$.

We can better approach ideal lowpass characteristics by using more poles clustered near $z = 1$ (but within the unit circle). Ideal lowpass behavior requires enhanced gain at every frequency in the band ($0 \leq |\Omega| \leq \Omega_c$). Conceptually, this can be achieved by placing a continuous wall of poles (requiring an infinite number of poles) in proximity to this band. Practical designs, however, are restricted to a finite number of poles. Figure 7.27d shows an example third-order lowpass filter with three poles near $z = 1$ and a third-order zero at $z = -1$ with corresponding magnitude and phase responses. Increasing the number of poles improves the approximation but at the expense of increasing filter complexity.

Highpass Filters

A highpass filter has a small gain at lower frequencies and a high gain at higher frequencies. Such a characteristic can be realized by placing a pole or poles near $z = -1$ because we want the gain at $\Omega = \pi$ to be the highest. Placing a zero at $z = 1$ further suppresses the gain at lower frequencies. Figure 7.27e shows a pole-zero configuration for a third-order highpass filter with corresponding magnitude and phase responses. Notice that the pole-zero plot of this highpass filter is just a reflection of the pole-zero plot of the lowpass filter of Fig. 7.27d. This is no coincidence since the region of highest digital frequency ($e^{j\pi} = -1$) is just a reflection of the region of lowest digital frequency ($e^{j0} = 1$). In fact, any lowpass digital filter can be transformed into a highpass filter (and vice versa) by reflecting all the poles and zeros of the system.

Bandpass and Bandstop Filters

In the following two examples, we shall realize continuous-time (analog) bandpass and bandstop filters using discrete-time systems and suitable interface devices (C/D and D/C), as shown in Fig. 7.28 or, more completely, in Fig. 4.2. The C/D device samples the continuous-time input $x(t)$ to yield a discrete-time signal $x[n]$, which serves as the input to the DT system $H(z)$. The output $y[n]$ of $H(z)$ is then converted to a continuous-time signal $y(t)$ by a D/C device. As we learned from Eq. (7.58), a continuous-time sinusoid of frequency ω , when sampled, results in a discrete-time sinusoid of frequency $\Omega = \omega T$. We use this relationship to ensure that the DT frequency responses map to the desired CT frequencies.

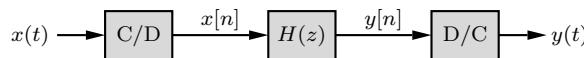


Figure 7.28: Processing a continuous-time signal by a discrete-time system.

▷ **Example 7.19 (Bandpass Filter)**

Following the structure of Fig. 7.28, use trial and error to design a tuned (bandpass) analog filter with zero transmission at 0 Hz and also at the highest frequency $f_{\max} = 500$ Hz. The resonant frequency is required to be 125 Hz.

Because $f_{\max} = 500$ Hz, the sampling theorem requires that $T < \frac{1}{2f_{\max}} = \frac{1}{1000}$ s. Let us select the sampling interval as $T = 10^{-3}$.[†] Recall that the analog frequencies $\omega = 2\pi f$ correspond to digital frequencies $\Omega = \omega T = 2\pi f T$. Hence, analog frequencies $f = 0$ and 500 Hz correspond to $\Omega = 0$ and π , respectively. Since the system gain is required to be zero at these frequencies, we need to place zeros at $z = e^{j\Omega}$ for $\Omega = 0$ and $\Omega = \pi$. Hence, there must be zeros at $z = e^{j0} = 1$ and $z = e^{j\pi} = -1$. Moreover, we need enhanced response at the resonant frequency $f = 125$ Hz, which corresponds to $\Omega = \pi/4$. We therefore place a pole in the vicinity of $z = e^{j\Omega} = e^{j\pi/4}$. Because this is a complex pole, we also need its conjugate near $e^{-j\pi/4}$, as indicated in Fig. 7.29a. Let us choose these two poles as

$$p_1 = |\gamma|e^{j\pi/4} \quad \text{and} \quad p_2 = |\gamma|e^{-j\pi/4},$$

where $|\gamma| < 1$ for stability. The closer the value of $|\gamma|$ is to the unit circle, the more sharply peaked is the response around $f = 125$ Hz.

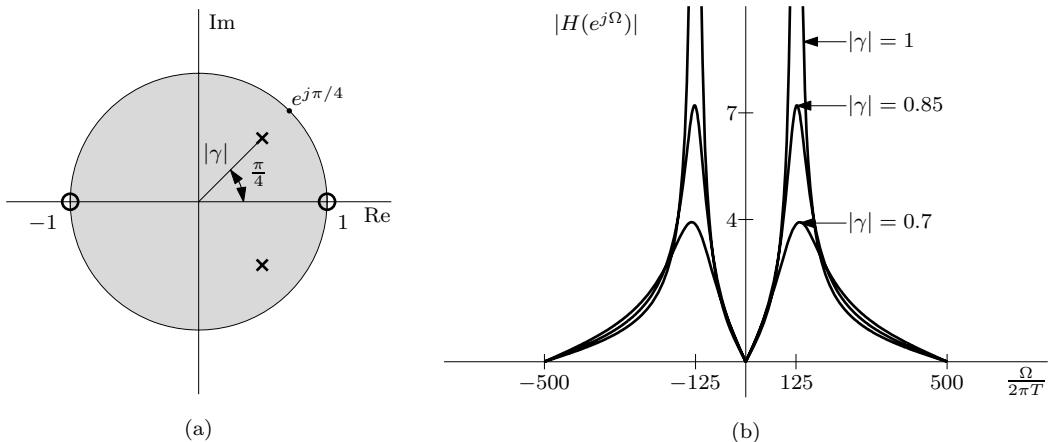


Figure 7.29: Second-order bandpass filter with resonant frequency at $\Omega = \pi/4$.

Taking the two zeros and two poles together, the system function is

$$H(z) = K \frac{(z-1)(z+1)}{(z-|\gamma|e^{j\pi/4})(z-|\gamma|e^{-j\pi/4})} = K \frac{z^2 - 1}{z^2 - \sqrt{2}|\gamma|z + |\gamma|^2}.$$

For convenience, we shall choose $K = 1$. Thus, the frequency response is given by

$$H(e^{j\Omega}) = H(z)|_{z=e^{j\Omega}} = \frac{e^{j2\Omega} - 1}{e^{j2\Omega} - \sqrt{2}|\gamma|e^{j\Omega} + |\gamma|^2}.$$

To demonstrate filter behavior, we use MATLAB to compute and plot the system magnitude response for $|\gamma| = 0.7$, 0.85, and 1.

[†]Strictly speaking, we need $T < 0.001$. However, if the input does not contain a finite-amplitude component of 500 Hz, $T = 0.001$ avoids aliasing. In practice, almost all practical signals with $f_{\max} = 500$ Hz satisfy this condition. Either way, it does not matter in this case since the system has 0 gain at 500 Hz.

```

01 H = @(z,gamma) (z.^2-1)./(z.^2-sqrt(2)*abs(gamma)*z+(abs(gamma)).^2);
02 T = 10^(-3); Omega = linspace(-pi,pi,1001); f = Omega/(2*pi*T); z = exp(j*Omega);
03 plot(f,abs(H(z,0.7)),f,abs(H(z,0.85)),f,abs(H(z,1))); axis([-500 500 -1 10]);

```

As shown in Fig. 7.29b, while we compute $|H(e^{j\Omega})|$ over $(-\pi \leq \Omega \leq \pi)$, we plot the response against $\Omega/2\pi T$ so as to view frequencies in the more natural units of hertz. As expected, the gain is zero at $f = 0$ and at 500 Hz ($\Omega = 0$ and π). The gain peaks at about 125 Hz ($\Omega = \pi/4$). The resonance (peaking) becomes pronounced as $|\gamma|$ approaches 1. Figure 7.30 shows a canonic realization of $H(z)$.

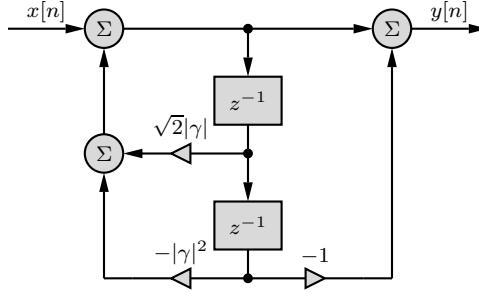


Figure 7.30: DFII realization of second-order BPF with resonant frequency at $\Omega = \pi/4$.

Example 7.19 ▶

▷ Example 7.20 (Bandstop Filter)

Following the structure of Fig. 7.28, design a second-order notch filter with zero transmission at 250 Hz and a sharp recovery of gain to unity on both sides of 250 Hz. The highest significant frequency to be processed is $f_{\max} = 400$ Hz.

In this case, $T < \frac{1}{2f_{\max}} = 1.25 \times 10^{-3}$. Let us choose $T = 10^{-3}$. For the frequency 250 Hz, $\Omega = 2\pi(250)T = \pi/2$. Thus, the frequency 250 Hz is represented by the point $e^{j\Omega} = e^{j\pi/2} = j$ on the unit circle, as depicted in Fig. 7.31a. Since we need zero transmission at this frequency, we must place a zero at both $z = e^{j\pi/2} = j$ and its conjugate $z = e^{-j\pi/2} = -j$. We also require a sharp recovery of gain on both sides of frequency 250 Hz. To accomplish this goal, we place two poles close to the two zeros in order to cancel out the effect of the two zeros as we move away from the point j (corresponding to frequency 250 Hz). For this reason, let us use poles at $\pm j|\gamma|$ with $|\gamma| < 1$ for stability. The closer the poles are to zeros (the closer $|\gamma|$ is to 1), the faster is the gain recovery on either side of 250 Hz. The resulting transfer function is

$$H(z) = K \frac{(z-j)(z+j)}{(z-j|\gamma|)(z+j|\gamma|)} = K \frac{z^2 + 1}{z^2 + |\gamma|^2}.$$

The dc gain of this filter is found setting $z = e^{j0} = 1$ as

$$H(1) = K \frac{2}{1 + |\gamma|^2}.$$

Because we require a dc gain of unity, we must select $K = \frac{1+|\gamma|^2}{2}$. The transfer function is therefore

$$H(z) = \left(\frac{1 + |\gamma|^2}{2} \right) \frac{z^2 + 1}{z^2 + |\gamma|^2}.$$

To demonstrate filter behavior, we again use MATLAB to compute and plot the system magnitude response, this time for $|\gamma| = 0.3, 0.7, and 0.95 .$

```
01 H = @(z,gamma) (1+(abs(gamma)).^2)/2*(z.^2+1)./(z.^2+(abs(gamma)).^2);
02 T = 10^(-3); Omega = linspace(-pi,pi,1001); f = Omega/(2*pi*T); z = exp(1j*Omega);
03 plot(f,abs(H(z,0.30)),f,abs(H(z,0.7)),f,abs(H(z,0.95))); axis([-500 500 -1 10]);
```

As shown in Fig. 7.31b, the filter's notch is correctly located at $f = 250$ Hz ($\Omega = \pi/2$). As $|\gamma|$ approaches 1, the gain more rapidly recovers to unity. Figure 7.31c shows a canonic realization of $H(z)$.

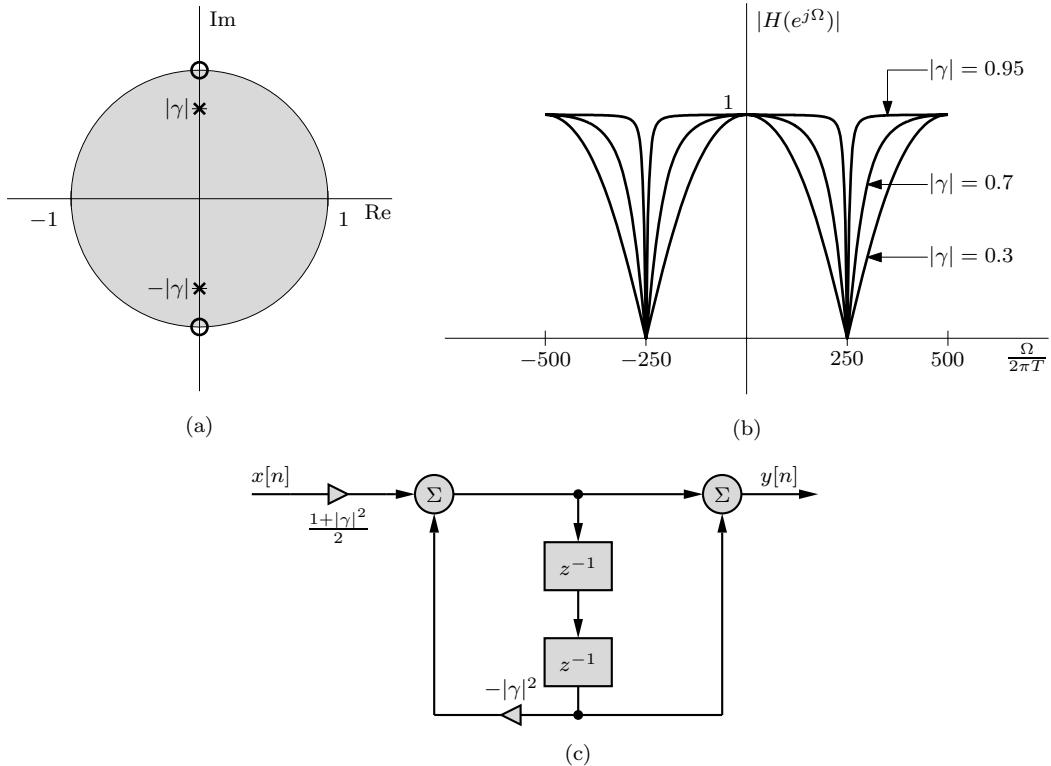


Figure 7.31: Second-order bandstop filter with notch frequency at $\Omega = \pi/2$.

Example 7.20 □

▷ Drill 7.23 (Complex Notch Filter)

Using a graphical argument, show that a filter with transfer function

$$H(z) = \frac{z - j}{2z}$$

acts as a complex notch filter. Accurately plot the magnitude and phase responses of this filter. Determine the filter output $y[n]$ in response to the real input $x[n] = \cos(\pi n/2)$.

□

7.7 Finite Word-Length Effects

Although most of our development of DT signals and systems assumes that signals are represented with infinite precision, practical systems necessarily use some form of quantization and represent numbers using finite word lengths. Finite word-length effects have the potential to severely and negatively impact system behavior. To begin to understand how, consider a simple digital wire represented as

$$y[n] = x[n].$$

As discussed in Sec. 3.6, analog-to-digital conversion quantizes the input $x[n]$ to a finite word length. Digital-to-analog conversion requires the same of output $y[n]$. Thus, even without any digital processing, quantization causes irreversible (and nonlinear) distortion of both input and output.

The situation only worsens when DT processing is put into play. Let us extend our example to a general first-order DT system represented as

$$y[n] = b_0x[n] + b_1x[n - 1] - a_1y[n - 1].$$

We know that finite word lengths render $x[n]$ and $y[n]$ as imperfect versions of the system input and output. Similarly, finite word lengths render the coefficients b_0 , b_1 , and a_1 as imperfect versions of the original DT system parameters. Consequently, coefficient quantization causes the system's pole and zero to move from their desired locations and degrades the system's frequency response. Later in this section we explore both of these adverse effects in greater detail.

Unfortunately, the problems of finite word lengths extend even further. The product $a_1y[n - 1]$, for example, requires a word size that is the sum of the word sizes of a_1 and $y[n - 1]$. The recursive nature of the system thus suggests an ever-increasing word size, which is impractical. Therefore, the computed products $b_0x[n]$, $b_1x[n - 1]$, and $a_1y[n - 1]$ are further quantized, which introduces additional errors into the system realization. Furthermore, summing the terms $b_0x[n]$, $b_1x[n - 1]$, and $-a_1y[n - 1]$ can cause overflow when assigned to the fixed word size of $y[n]$. Dealing with the potential for overflow can be a real conundrum. While overflow can be avoided by scaling the input smaller, doing so undesirably reduces the effective word length (resolution) of the input.

An Imperfect World

When it comes to real-world digital signal processing, it is an imperfect world. As we have seen, finite word lengths produce imperfect systems that imperfectly operate on imperfect data. Furthermore, the nonlinear nature of the quantization process makes these problems difficult, if not impossible, to analyze or quantify. Still, such imperfections need not imply poor system performance. On the contrary, digital signal processing systems are, when properly designed and realized, capable of reliable and accurate operation. There are many available options that help minimize the adverse effects of finite word lengths.

As discussed in Sec. 7.5.3, different system realizations possess different numerical properties. Some realizations better cope with finite word-length effects. For example, cascade and parallel realizations comprised of low-order sections tend to perform better than single-section direct form realizations. Additionally, word lengths can be increased, usually at modest cost, to meet accuracy demands. Floating-point representations, which are generally less sensitive to finite word-length effects, can be adopted over fixed-point representations. Performance can be sometimes improved by over-designing systems or utilizing alternate design methodologies. Simply understanding the nature of finite word-length effects allows for more competent design. To this end, let us next turn our attention to the effects of finite word lengths on poles, zeros, and frequency response.

7.7.1 Finite Word-Length Effects on Poles and Zeros

To demonstrate how finite word lengths can impact a system's poles and zeros, let us focus our attention on a second-order system with complex-conjugate poles at $z = re^{\pm j\theta} = r \cos(\theta) \pm jr \sin(\theta)$.

The corresponding transfer function is

$$H(z) = \frac{1}{(1 - re^{j\theta}z^{-1})(1 - re^{-j\theta}z^{-1})} = \frac{1}{1 - 2r \cos(\theta)z^{-1} + r^2 z^{-2}}.$$

Comparing this expression with Eq. (7.46), we see that $b_0 = 1$, $b_1 = 0$, $b_2 = 0$, $a_1 = -2r \cos(\theta)$, and $a_2 = r^2$. Figure 7.32 shows the system realized in direct form.

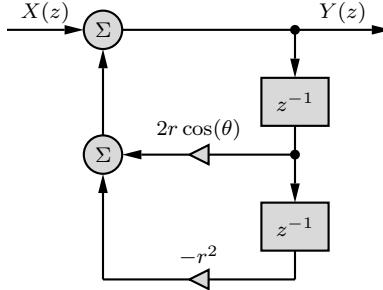


Figure 7.32: Direct form realization of complex-conjugate poles $re^{\pm j\theta}$.

Now, when the system of Fig. 7.32 is implemented in hardware, the coefficients $-a_1 = 2r \cos(\theta)$ and $-a_2 = -r^2$ must be quantized. Let us assume that this quantization follows common conventions and produces B -bit two's-complement signed numbers that cover the desired ranges needed to produce a stable system ($-2 < a_1 < 2$ and $0 \leq a_2 < 1$). Since $r \cos(\theta)$ is the real part of the pole location, quantization of $-a_1 = 2r \cos(\theta)$ demands that the system poles must fall on a series of vertical lines. Similarly, since r is the distance of the pole from the origin, quantization of $-a_2 = -r^2$ forces the system poles to fall on a series of concentric circles. Taken together, quantization of $-a_1$ and $-a_2$ forces the system poles to fall on the intersection of these vertical lines and concentric circles. Figure 7.33 illustrates the idea and plots the stable pole locations that are possible for the direct form realization of Fig. 7.32 using 4-bit and 6-bit quantization.[†]

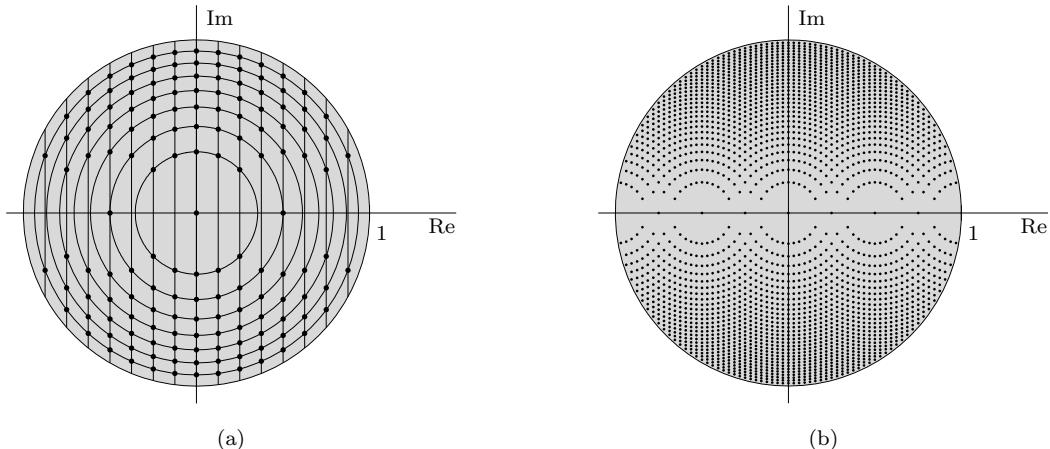


Figure 7.33: Pole locations for the direct form system of Fig. 7.32: (a) 4-bit and (b) 6-bit coefficients.

There are several interesting items to note from Fig. 7.33. To begin, we observe that quantization prevents the free placement of system poles in the z -plane. For a particular word length B , there

[†]The system of Fig. 7.32 can also realize a set of real-valued poles that are not complex-conjugate pairs. For clarity of discussion, Fig. 7.33 does not display these non-conjugate poles.

are only a finite number of fixed pole locations available. Thus, it is virtually certain that the actual (quantized) system poles will differ from the desired design locations. Furthermore, notice that the possible pole locations are not uniformly distributed within the unit circle. In particular, there are relatively few pole locations found along the real axis. This means that the direct form realization of Fig. 7.32 will have difficulty implementing narrow-band lowpass and highpass filters, which tend to have concentrations of poles near $z = 1$ and $z = -1$, respectively. Increasing the number of bits B tends to reduce, but does not eliminate, the problems associated with quantization.

There exist other ways besides increasing the number of quantization bits to improve system performance. We previously noted that cascade and parallel realizations comprised of low-order sections tend to perform better than single-section direct form realizations. Additionally, specialized structures can offer improved performance as well. Consider, for example, the coupled form realization of complex-conjugate poles $re^{\pm j\theta}$ that is shown in Fig. 7.34 [3].

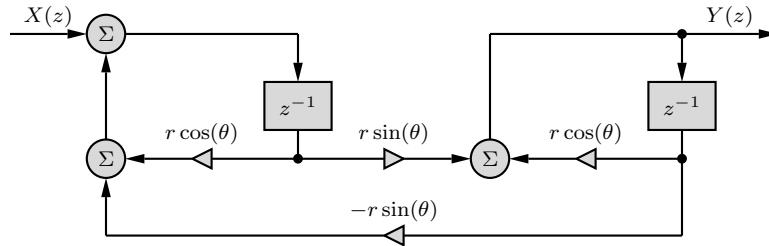


Figure 7.34: Coupled form realization of complex-conjugate poles $re^{\pm j\theta}$.

When the system of Fig. 7.34 is implemented in hardware, the coefficients $r \cos(\theta)$ and $r \sin(\theta)$ must be quantized. Again, let us assume this quantization follows common conventions and produces B -bit two's-complement signed numbers that cover the desired ranges needed to produce a stable system ($-1 < r \cos(\theta) < 1$ and $-1 < r \sin(\theta) < 1$). Since the coefficients $r \cos(\theta)$ and $r \sin(\theta)$ are themselves the real and imaginary parts of the pole location, respectively, their quantization requires that the system poles fall on intersection of a series of vertical and horizontal lines. To illustrate, Fig. 7.35 plots the stable pole locations that are possible for the coupled form realization of Fig. 7.34 using 4-bit and 6-bit quantization.

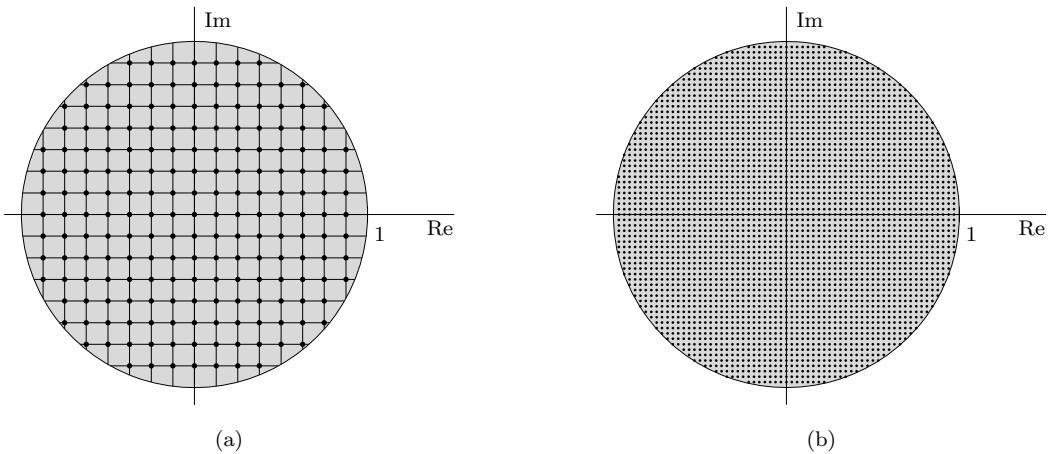


Figure 7.35: Pole locations for the coupled form system of Fig. 7.34: (a) 4-bit and (b) 6-bit coefficients.

From Fig. 7.35, we see that the pole locations of the coupled form realization (Fig. 7.34) are uniformly distributed inside the unit circle. Unlike the direct form realization pole locations shown

in Fig. 7.33, there are no gaps along the real axis. Thus, a coupled form realization may be more effective than a direct form realization, particularly in situations where short word lengths are necessary or system poles are found concentrated along the real axis.

It is worth pointing out, however, that although the pole locations of the coupled form may appear generally better than those of the direct form, there is no guarantee that a coupled form realization will be better than a direct form (or other) realization. It is possible that the poles of a particular system just happen to better match the available pole locations of a direct form realization. Further, comparing Fig. 7.32 and Fig. 7.34, we see that the coupled form requires twice as many multiply operations as does the direct form, which makes its implementation more costly. The point of our discussion is not to endorse one realization over another but rather to demonstrate the general nature of finite word-length effects. It is the responsibility of the DSP engineer to select an appropriate realization for a given design and to verify, generally through computer simulation, proper system operation.

7.7.2 Finite Word-Length Effects on Frequency Response

As shown in Sec. 7.6.1, frequency response depends directly on the poles and zeros of a system. When coefficient quantization alters a system's poles and zeros, the system's frequency response also changes, usually for the worse. Since we commonly use frequency response to describe and assess system behavior, it is generally more useful and natural to investigate finite word-length effects using frequency response rather than pole and zero information. Let us demonstrate with an example.

▷ **Example 7.21 (Finite Word-Length Effects on Frequency Response)**

A digital Chebyshev lowpass filter, which has all its zeros at $z = -1$, can be implemented as a cascade of second-order direct form sections, each with transfer function of the form

$$H(z) = \frac{b_0(1 + 2z^{-1} + z^{-2})}{1 + a_1z^{-1} + a_2z^{-2}}.$$

Assuming that the system operates at a sampling frequency of $F_s = \frac{100}{\pi}$, investigate the effects of 12-, 10-, 8-, and 6-bit coefficient quantization on the magnitude response of the 6th-order Chebyshev lowpass filter described by

Section	b_0	a_1	a_2
1	0.03273793724	-1.81915463768	0.95522952077
2	0.01799913516	-1.80572713311	0.88054034430
3	0.00399530100	-1.82139792759	0.83800435313

To begin, let us first consider the system without finite word-length effects. Figure 7.36 plots the system magnitude response using the original (unquantized) coefficients. The filter exhibits 1 dB of passband ripple to the passband frequency $\omega_p = 12$ rad/s and achieves at least 20 dB of stopband attenuation beyond the stopband frequency $\omega_s = 15$ rad/s. Since the passband and stopband frequencies are both well below the folding frequency of 100 rad/s, the filter is relatively narrowband, and all system poles are concentrated near $z = 1$. As a result, we expect direct form realizations of the system to be somewhat more susceptible than normal to finite word-length effects.

Next, we quantize the filter coefficients. For this particular filter, coefficient quantization does not affect the system zeros since they are all located at $z = -1$. Instead, coefficient quantization affects the system poles and the overall system gain. Designating the number of quantization bits as B , each coefficient needs to be represented by an integer in the range -2^{B-1} to $2^{B-1} - 1$. The idea is to choose the integer within this range that, when scaled by an appropriate power of two, is closest to the original coefficient value. The power-of-two scaling, which is easily accomplished on nearly any



Figure 7.36: Magnitude response of a 6th-order Chebyshev lowpass filter with unquantized coefficients.

hardware platform, can vary as needed between stages and between the a and b coefficients. To help simplify programming, however, the same power-of-two scale factor is often used within coefficient groups. We follow this strategy in this example, which is why any section's coefficient pair a_1 and a_2 share the same power-of-two scale factor. Table 7.3 presents the quantized coefficients using 12-, 10-, 8-, and 6-bit word lengths.

Quantization	Section	b_0	a_1	a_2
12-bit	1	1073×2^{-15}	-1863×2^{-10}	978×2^{-10}
	2	1180×2^{-16}	-1849×2^{-10}	902×2^{-10}
	3	1047×2^{-18}	-1865×2^{-10}	858×2^{-10}
10-bit	1	268×2^{-13}	-466×2^{-8}	245×2^{-8}
	2	295×2^{-14}	-462×2^{-8}	225×2^{-8}
	3	262×2^{-16}	-466×2^{-8}	215×2^{-8}
8-bit	1	67×2^{-11}	-116×2^{-6}	61×2^{-6}
	2	74×2^{-12}	-116×2^{-6}	56×2^{-6}
	3	65×2^{-14}	-117×2^{-6}	54×2^{-6}
6-bit	1	17×2^{-9}	-29×2^{-4}	15×2^{-4}
	2	18×2^{-10}	-29×2^{-4}	14×2^{-4}
	3	16×2^{-12}	-29×2^{-4}	13×2^{-4}

Table 7.3: Quantized coefficients for a 6th-order Chebyshev lowpass filter.

With a set of quantized coefficients in hand, it is a simple matter to compute and plot the corresponding magnitude response. Since the system zeros are in each case unaffected by quantization, we know that the filter stopband behavior will remain intact. Degradation in filter performance will primarily occur in the passband, which is where we shall concentrate our attention.

In Fig. 7.37a, we see that 12-bit coefficient quantization results in a magnitude response (heavy solid line) that matches almost exactly the ideal magnitude response (shaded curve). When the coefficient word length is reduced to 10 bits, we begin to see some degradation in frequency response, as shown in Fig. 7.37b. At this level of quantization, the passband is no longer equiripple, and passband ripple exceeds the original 1-dB specification. Still, the magnitude response remains relatively intact, and the filter, while not exactly meeting design specifications, should operate reliably.

The situation changes when the coefficient word length is reduced to an 8-bit level. As seen in Fig. 7.37c, the magnitude response exhibits further deterioration from the ideal. Although the system still appears lowpass with an essentially correct cutoff frequency, the system gain exceeds a value of 1 for greater than half the passband. Thus, in addition to not meeting design specifications, this filter is likely to encounter overflow or saturation errors during operation.

The situation becomes dire with 6-bit coefficient quantization. In this case, the magnitude

response is unbounded at dc, as shown in Fig. 7.37d. Rather than two complex-conjugate poles contained within the unit circle, the quantized coefficients a_1 and a_2 of the third section of this filter produce real poles at $z = 0.8125$ and $z = 1$. The pole at $z = 1$ is particularly problematic and is the reason the magnitude response becomes unbounded. This system, which is now only marginally stable, is nearly guaranteed not to operate properly. Problem 7.7-2 revisits this problem using a single-section implementation rather than a cascade of second-order sections.

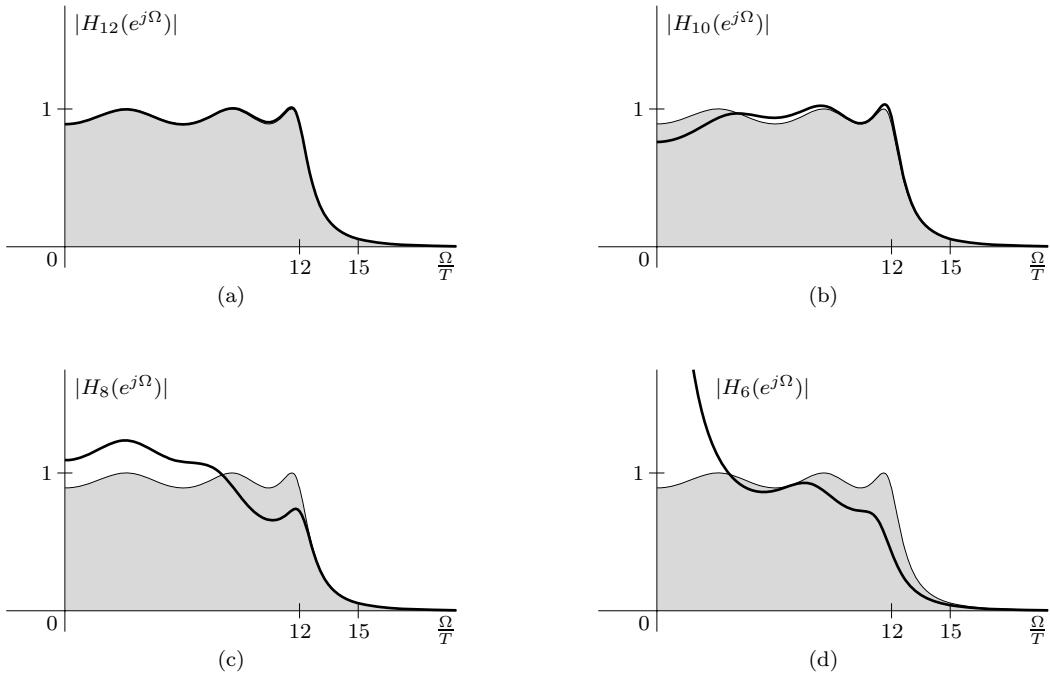


Figure 7.37: Magnitude responses of a 6th-order Chebyshev lowpass filter using (a) 12-, (b) 10-, (c) 8-, and (d) 6-bit coefficients.

Example 7.21 ◀

7.8 Connection between the Laplace and z -Transforms

We now show that discrete-time systems also can be analyzed using the Laplace transform. In fact, we shall see that *the z-transform is the Laplace transform in disguise* and that discrete-time systems can be analyzed as if they were continuous-time systems.

So far we have considered discrete-time signals as sequences of numbers and not as an electrical signal (voltage or current). Similarly, we considered a discrete-time system as a mechanism that processes a sequence of numbers (input) to yield another sequence of numbers (output). The system is built by using delays, adders, and multipliers that operate on sequences of numbers. A digital computer is a perfect example: every signal is a sequence of numbers, and the signal processing involves delaying the number sequences, along with addition and multiplication.

Now suppose that we have a discrete-time system with transfer function $H(z)$ and input $x[n]$. Consider a continuous-time signal $x(t)$ such that its n th sample value is $x[n]$, as shown in Fig. 7.38.[†]

[†]We can construct such $x(t)$ from the sample values $x[n]$, as explained in Ch. 3.

Let the impulse-sampled signal be $x_{\delta}(t)$, consisting of impulses spaced T seconds apart with the n th impulse of strength $x[n]$. Thus, similar to Eq. (3.3),

$$x_{\delta}(t) = \sum_{n=-\infty}^{\infty} x[n]\delta(t - nT).$$

Figure 7.38 shows $x[n]$ and the corresponding $x_{\delta}(t)$.

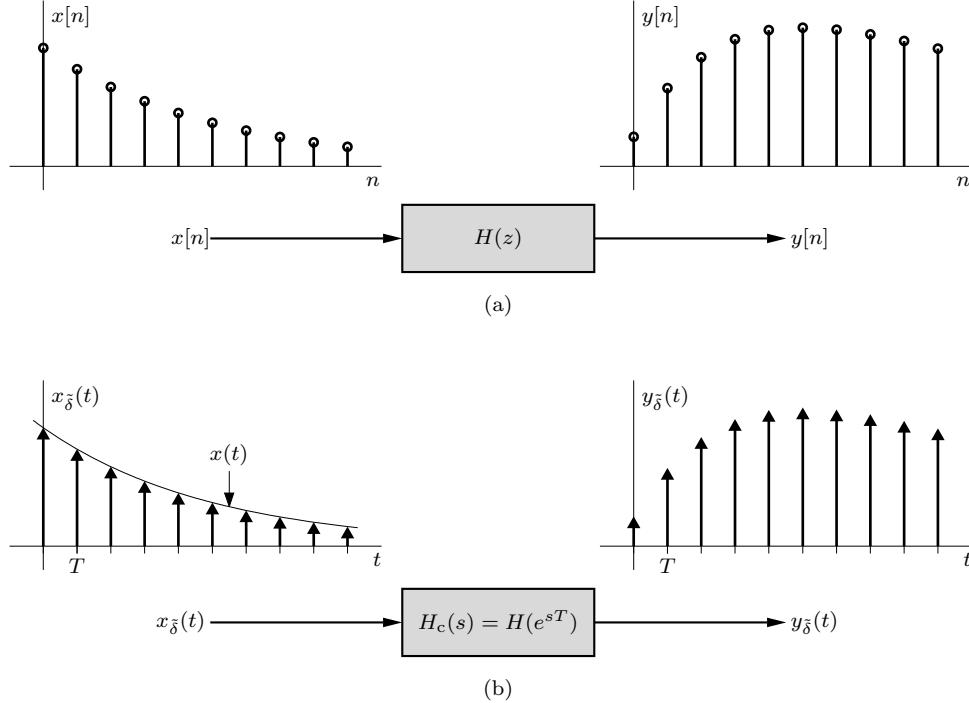


Figure 7.38: Connection between the Laplace and z -transforms.

The signal $x[n]$ is applied as the input of a discrete-time system with transfer function $H(z)$, which we assume is comprised of the typical delays, adders, and scalar multipliers. Hence, processing $x[n]$ through $H(z)$ amounts to operating on the sequence $x[n]$ using delays, adders, and scalar multipliers. Suppose for $x_{\delta}(t)$ that we perform operations identical to those performed on the samples of $x[n]$ by $H(z)$. For this purpose, we need to use a continuous-time system with transfer function $H_c(s)$ that is identical in structure to the discrete-time system $H(z)$ except that the delays in $H(z)$ are replaced by elements that delay continuous-time signals. There is no other difference between realizations of $H(z)$ and $H_c(s)$. If a continuous-time impulse $\delta(t)$ is applied to such a delay of T seconds, the output will be $\delta(t - T)$. The continuous-time transfer function of such a delay is e^{-sT} (see Eq. (1.48) on page 31). Hence, the delay elements with transfer function z^{-1} in the realization of $H(z)$ will be replaced by delay elements with transfer function e^{-sT} in the realization of the corresponding $H_c(s)$. This is the same as z being replaced by e^{sT} . Therefore, $H_c(s) = H(e^{sT})$. Let us now apply $x[n]$ to the input of $H(z)$ and apply $x_{\delta}(t)$ at the input of $H(e^{sT})$. The operations performed by the discrete-time system $H(z)$ on $x[n]$ (Fig. 7.38a) are also performed by the corresponding continuous-time system $H(e^{sT})$ on the impulse sequence $x_{\delta}(t)$ (Fig. 7.38b). The delaying of a sequence in $H(z)$ amounts to delaying an impulse train in $H(e^{sT})$. Add and multiply operations are the same in both cases. In other words, a one-to-one correspondence of the two systems is preserved in every aspect.

Therefore, if $y[n]$ is the output of the discrete-time system in Fig. 7.38a, then $y_{\tilde{\delta}}(t)$, the output of the continuous-time system in Fig. 7.38b, would be a sequence of impulses whose n th impulse strength is $y[n]$. Thus,

$$y_{\tilde{\delta}}(t) = \sum_{n=-\infty}^{\infty} y[n]\delta(t - nT).$$

The system in Fig. 7.38b, being a continuous-time system, can be analyzed via the Laplace transform. If

$$x_{\tilde{\delta}}(t) \xleftrightarrow{\mathcal{L}} X_{\tilde{\delta}}(s) \quad \text{and} \quad y_{\tilde{\delta}}(t) \xleftrightarrow{\mathcal{L}} Y_{\tilde{\delta}}(s),$$

then

$$Y_{\tilde{\delta}}(s) = H(e^{sT})X_{\tilde{\delta}}(s). \quad (7.64)$$

Also,

$$X_{\tilde{\delta}}(s) = \mathcal{L} \left\{ \sum_{n=-\infty}^{\infty} x[n]\delta(t - nT) \right\} \quad \text{and} \quad Y_{\tilde{\delta}}(s) = \mathcal{L} \left\{ \sum_{n=-\infty}^{\infty} y[n]\delta(t - nT) \right\}.$$

Now, because the Laplace transform of $\delta(t - nT)$ is e^{-snT} ,

$$X_{\tilde{\delta}}(s) = \sum_{n=-\infty}^{\infty} x[n]e^{-snT} \quad \text{and} \quad Y_{\tilde{\delta}}(s) = \sum_{n=-\infty}^{\infty} y[n]e^{-snT}. \quad (7.65)$$

Substitution of Eq. (7.65) into Eq. (7.64) yields

$$\sum_{n=-\infty}^{\infty} y[n]e^{-snT} = H(e^{sT}) \left(\sum_{n=-\infty}^{\infty} x[n]e^{-snT} \right).$$

By introducing a new variable $z = e^{sT}$, this equation can be expressed as

$$\sum_{n=-\infty}^{\infty} y[n]z^{-n} = H(z) \sum_{n=-\infty}^{\infty} x[n]z^{-n}$$

or

$$Y(z) = H(z)X(z),$$

where

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n} \quad \text{and} \quad Y(z) = \sum_{n=-\infty}^{\infty} y[n]z^{-n}.$$

It is clear from this discussion that the z -transform can be considered as the Laplace transform with a change of variable $z = e^{sT}$ or $s = \ln(z)/T$. Note that the transformation $z = e^{sT}$ maps the imaginary axis in the s -plane ($s = j\omega$) into a unit circle in the z -plane ($z = e^{sT} = e^{j\omega T}$, or $|z| = 1$). The LHP and RHP in the s -plane map into the inside and the outside, respectively, of the unit circle in the z -plane. A more comprehensive discussion of this mapping is found in Ch. 4 beginning on page 222.

7.9 Summary

In this chapter, we discuss the analysis of linear time-invariant discrete-time (LTID) systems by means of the z -transform. The z -transform changes the difference equations of LTID systems into algebraic equations. Therefore, solving these difference equations reduces to solving algebraic equations. Furthermore, the z -transform provides valuable insight into system realization and behavior.

When it exists (converges), the z -transform represents a signal as a sum of everlasting exponentials of the form z^n , where the variable z can be viewed as generalized frequency. Thus, the

z -transform provides a frequency-domain view of a signal. Much like the Laplace transform, there are both unilateral and bilateral z -transforms. The unilateral z -transform provides a convenient mechanism to solve systems with initial conditions, but it is restricted to causal signals and systems. The bilateral z -transform accommodates noncausal signals and systems but does not handle initial conditions in such a convenient manner as does the unilateral z -transform. To ensure a unique transform pair, the bilateral z -transform must include a region of convergence; the region of convergence is implied for the unilateral case since causal signals are assumed.

The z -transform can be obtained by direct calculation or by using tables. The inverse z -transform, on the other hand, involves complex contour integration and is therefore normally computed using partial fraction expansions and tables. In some cases, power series expansions can also be used. A variety of z -transform properties are useful in computing the z -transform and its inverse, as well as giving insight into the nature of signals and systems.

By using the z -transform and its properties, constant-coefficient linear difference equations are solved algebraically. To this end, the transfer function is particularly useful. The transfer function $H(z)$ of an LTID system is equal to the ratio of the z -transform of the output to the z -transform of the input when all initial conditions are zero. Therefore, if $X(z)$ is the z -transform of the input $x[n]$ and $Y(z)$ is the z -transform of the corresponding zero-state output $y[n]$, then $Y(z) = H(z)X(z)$. For a system specified by the difference equation $A(E)\{y[n]\} = B(E)\{x[n]\}$, the transfer function $H(z) = B(z)/A(z)$. It follows that the BIBO stability of a system can be determined from the poles of $H(z)$. Moreover, $H(z)$ is the z -transform of the system impulse response $h[n]$, and the transfer function provides the system response to an everlasting exponential z^n as $H(z)z^n$.

Many LTID systems, such as those described by constant-coefficient linear difference equations, can be realized by adders, scalar multipliers, and time delays. A given transfer function can be synthesized in many different ways. Canonic, transposed canonic, cascade, and parallel forms of realization are discussed. The realization procedure is identical to that for continuous-time systems with $1/s$ (integrator) replaced by $1/z$ (unit delay).

For BIBO stable DT systems, the frequency response is obtained from the system transfer function $H(z)$ by letting $z = e^{j\Omega}$. Frequency response provides a system's filtering characteristics. The magnitude and phase of an input sinusoid of frequency Ω are changed according to the magnitude response $|H(e^{j\Omega})|$ and the phase response $\angle H(e^{j\Omega})$, respectively. Magnitude and phase response plots are easily obtained from a system's pole and zero locations. By proper placement of poles and zeros, it is possible to design basic lowpass, highpass, bandpass, and bandstop filters.

In Sec. 7.8, we showed that discrete-time systems can be analyzed by the Laplace transform as if they were continuous-time systems. In fact, we showed that the z -transform is the Laplace transform with a change in variable. Thus, CT and DT systems are inherently connected, and understanding one necessarily helps in the understanding of the other.

References

1. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
2. Lyons, R. G., *Understanding Digital Signal Processing*, Addison-Wesley, Reading, MA, 1996.
3. Rader, C. M., and Gold, B., "Effects of Parameter Quantization on the Poles of a Digital Filter," *Proc. of the IEEE*, Vol. 55, May 1967, pp. 688–689.

Problems

7.1-1 Using the definition, find the bilateral z -transforms, including ROCs, of

- (a) $x_a[n] = (0.8)^n u[n] + 2^n u[-n - 1]$
- (b) $x_b[n] = 2^n u[n] - 3^n u[-n - 1]$
- (c) $x_c[n] = (0.8)^n u[n] + (0.9)^n u[-n - 1]$
- (d) $x_d[n] = [(0.8)^n + 3(0.4)^n] u[-n - 1]$
- (e) $x_e[n] = [(0.8)^n + 3(0.4)^n] u[n]$
- (f) $x_f[n] = (0.8)^n u[n] + 3(0.4)^n u[-n - 1]$
- (g) $x_g[n] = (0.5)^{|n|}$
- (h) $x_h[n] = n u[-n - 1]$
- (i) $x_i[n] = n(u[n + 2] - u[n - 3])$

7.1-2 Determine the z -transform of

$$x[n] = \sum_{k=0}^{\infty} k\delta(n - 2k + 1).$$

7.1-3 Using the definition, find the unilateral z -transforms of

- (a) $x_a[n] = u[n - m]$, $m > 0$
- (b) $x_b[n] = \gamma^n \sin(\pi n) u[n]$
- (c) $x_c[n] = \gamma^n \cos(\pi n) u[n]$
- (d) $x_d[n] = \gamma^n \sin\left(\frac{\pi n}{2}\right) u[n]$
- (e) $x_e[n] = \gamma^n \cos\left(\frac{\pi n}{2}\right) u[n]$
- (f) $x_f[n] = \sum_{k=0}^{\infty} 2^{2k} \delta[n - 2k]$
- (g) $x_g[n] = \gamma^{n-1} u[n - 1]$
- (h) $x_h[n] = n \gamma^n u[n]$
- (i) $x_i[n] = n u[n]$
- (j) $x_j[n] = \frac{\gamma^n}{n!} u[n]$
- (k) $x_k[n] = (2^{n-1} - (-2)^{n-1}) u[n]$
- (l) $x_l[n] = \frac{(\ln \alpha)^n}{n!} u[n]$

7.1-4 Using only Table 7.1 and the fact that $\delta[n - m] \xleftrightarrow{z^{-1}} z^{-m}$, determine the z -transforms of

- (a) $x_a[n] = u[n] - u[n - 2]$
- (b) $x_b[n] = \gamma^{n-2} u[n - 2]$
- (c) $x_c[n] = 2^{n+1} u[n - 1] + e^{n-1} u[n]$
- (d) $x_d[n] = [2^{-n} \cos\left(\frac{\pi}{3}n\right)] u[n - 1]$
- (e) $x_e[n] = n \gamma^n u[n - 1]$

(f) $x_f[n] = n(n - 1)(n - 2)2^{n-3} u[n - m]$,
 $m = 0, 1, 2, 3$

(g) $x_g[n] = n(-1)^n u[n]$

(h) $x_h[n] = \gamma^n u[-n]$

7.2-1 Find the inverse bilateral z -transform of

$$X(z) = \frac{(e^{-2} - 2)z}{(z - e^{-2})(z - 2)}$$

when the ROC is

- (a) $|z| > 2$
- (b) $e^{-2} < |z| < 2$
- (c) $|z| < e^{-2}$

7.2-2 Find all possible inverse bilateral z -transforms of

- (a) $X_a(z) = \frac{z(z+1)}{(z-0.5)(z+2)}$
- (b) $X_b(z) = \frac{(z+1)(z-1)}{(z^2+1)}$
- (c) $X_c(z) = \frac{z(z+1)}{(z-0.5)^2(z+2)}$

7.2-3 Find the inverse unilateral z -transforms of

- (a) $X_a(z) = \frac{z(z-4)}{z^2-5z+6}$
- (b) $X_b(z) = \frac{z-4}{z^2-5z+6}$
- (c) $X_c(z) = \frac{(e^{-2}-2)z}{(z-e^{-2})(z-2)}$
- (d) $X_d(z) = \frac{(z-1)^2}{z^3}$
- (e) $X_e(z) = \frac{z(2z+3)}{(z-1)(z^2-5z+6)}$
- (f) $X_f(z) = \frac{z(-5z+22)}{(z+1)(z-2)^2}$
- (g) $X_g(z) = \frac{z(1.4z+0.08)}{(z-0.2)(z-0.8)^2}$
- (h) $X_h(z) = \frac{z(z-2)}{z^2-z+1}$
- (i) $X_i(z) = \frac{2z^2-0.3z+0.25}{z^2+0.6z+0.25}$
- (j) $X_j(z) = \frac{2z(3z-23)}{(z-1)(z^2-6z+25)}$
- (k) $X_k(z) = \frac{z(3.83z+11.34)}{(z-2)(z^2-5z+25)}$
- (l) $X_l(z) = \frac{z^2(-2z^2+8z-7)}{(z-1)(z-2)^3}$

7.2-4 Expanding $X(z)$ as a power series in z^{-1} , find the first three terms of $x[n]$ if

$$X(z) = \frac{2z^3 + 13z^2 + z}{z^3 + 7z^2 + 2z + 1}.$$

- 7.2-5** Extend the procedure used in Prob. 7.2-4 to find the first four terms of $x[n]$ if

$$X(z) = \frac{2z^4 + 16z^3 + 17z^2 + 3z}{z^3 + 7z^2 + 2z + 1}.$$

- 7.2-6** Let

$$X(z) = \frac{\gamma z}{(z - \gamma)^2}.$$

Using a power series expansion, determine $x[n]$ if the ROC of $X(z)$ is

- (a) $|z| > |\gamma|$
- (b) $|z| < |\gamma|$

- 7.2-7** Let the transform $X(z)$ of a causal signal $x[n]$ be a rational function in z with numerator order L , denominator order K , and $L \leq K$.

- (a) How does $K - L$ relate to the time-domain signal $x[n]$?
- (b) Without actually finding the z -transform, determine $K - L$ for $X(z)$ that corresponds to $x[n] = \gamma^n u[n - 4]$.

- 7.3-1** Let signals $x[n]$ and $y[n]$ possess z -transforms $X(z)$ and $Y(z)$. Explain why the transform of $x[n] + y[n]$ is guaranteed to equal $X(z) + Y(z)$ in the unilateral case but not necessarily in the bilateral case.

- 7.3-2** Prove Eq. (7.18), which relates the z -transform of an upsampled signal $x_{\uparrow}[n]$ to the z -transform of the original signal $x[n]$.

- 7.3-3** For the signal shown in Fig. P7.3-3, show that

$$X(z) = \frac{1 - z^{-m}}{1 - z^{-1}}, \quad R_x: \text{all } z.$$

Find your answer by two methods: (a) using the definition in Eq. (7.1) and (b) using Table 7.1 and an appropriate z -transform property.

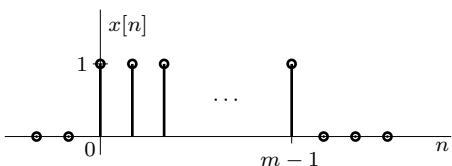


Figure P7.3-3

- 7.3-4** Find the z -transform of the signal illustrated in Fig. P7.3-4. Solve this problem in two ways: (a) as in Ex. 7.4d and (b) as in Ex. 7.8. Verify that the two answers are equivalent.

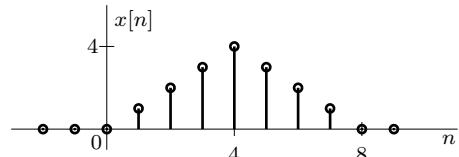


Figure P7.3-4

- 7.3-5** Using only unilateral z -transform properties and the fact that $\gamma^n u[n] \xleftrightarrow{Z_u} \frac{z}{z - \gamma}$, find the z -transforms of

- (a) $x_a[n] = n^2 u[n]$
- (b) $x_b[n] = n^2 \gamma^n u[n]$
- (c) $x_c[n] = n^3 u[n]$
- (d) $x_d[n] = a^n (u[n] - u[n - m]), m > 0$.
- (e) $x_e[n] = n e^{-2n} u[n - m], m > 0$
- (f) $x_f[n] = (n - 2)(0.5)^{n-3} u[n - 4]$

- 7.3-6** Using only pair 1 in Table 7.1 and appropriate properties of the z -transform, derive iteratively pairs 2 through 6. In other words, first derive pair 2. Then use pair 2 (and pair 1, if needed) to derive pair 3, and so on.

- 7.3-7** Using pair 2 in Table 7.1 and appropriate properties of the z -transform, determine the z -transform of $x[n] = n^3 u[n]$.

- 7.3-8** Using pairs 1 and 9 in Table 7.1 and appropriate properties of the z -transform, determine the z -transform of $x[n] = \cos(\pi n/4) u[n]$.

- 7.3-9** Apply the time-reversal property to pair 4 to show that $\beta^n u[-n - 1] \xleftrightarrow{Z_u} -z/(z - \beta)$ with ROC given by $|z| < |\beta|$.

- 7.3-10** Let $x[n] \xleftrightarrow{Z_u} X(z)$.

- (a) Show that $(-1)^n x[n] \xleftrightarrow{Z_u} X(-z)$.
- (b) Using the result of part (a), show that $(-\gamma)^n u[n] \xleftrightarrow{Z_u} z/(z + \gamma)$.

- (c) Using the result of part (b), find the z -transform of $(2^{n-1} - (-2)^{n-1}) u[n]$.
 (d) Using the result of part (b), find the z -transform of $\gamma^n \cos(\pi n) u[n]$.

7.3-11 Let $x[n] \xrightarrow{\mathcal{Z}_u} X(z)$.

- (a) Show that

$$\sum_{k=0}^n x[k] \xrightarrow{\mathcal{Z}_u} \frac{z}{z-1} X(z).$$

- (b) Using the result of part (a), derive pair 2 from pair 1 in Table 7.1.

7.4-1 To pay off a loan of M dollars in N payments using a fixed monthly payment of P dollars, use z -transform methods to show that

$$P = \frac{rM}{1 - (1+r)^{-N}},$$

where r is the monthly interest rate.

7.4-2 An LTID system is described as

$$y[n+1] + 2y[n] = x[n+1],$$

with $y[0] = 1$ and input $x[n] = e^{-(n-1)} u[n]$.

- (a) Determine the total response $y[n]$.
 (b) Determine the zero-input and zero-state components of the response.

7.4-3 An LTID system is described as

$$2y[n+2] - 3y[n+1] + y[n] = 4x[n+2] - 3x[n+1],$$

with $y[-1] = 0$, $y[-2] = 1$, and input $x[n] = (4)^{-n} u[n]$.

- (a) Determine the total response $y[n]$.
 (b) Determine the zero-input and zero-state components of the response.
 (c) Determine the transient and steady-state components of the response.

7.4-4 Solve Prob. 7.4-3 if instead of initial conditions $y[-1]$ and $y[-2]$ you are given the auxiliary conditions $y[0] = \frac{3}{2}$ and $y[1] = \frac{35}{4}$.

7.4-5 Using $y[-1] = 2$, $y[-2] = 3$, and $x[n] = (3)^n u[n]$, solve

$$y[n+2] - 3y[n+1] + 2y[n] = x[n+1].$$

7.4-6 An LTID system is described as

$$4y[n+2] + 4y[n+1] + y[n] = x[n+1],$$

with $y[-1] = 0$, $y[-2] = 1$, and input $x[n] = u[n]$.

- (a) Determine the total response $y[n]$.
 (b) Determine the zero-input and zero-state components of the response.
 (c) Determine the transient and steady-state components of the response.

7.4-7 Using $y[-1] = 1$, $y[-2] = 0$, and $x[n] = u[n]$, solve

$$y[n+2] - 2y[n+1] + 2y[n] = x[n].$$

7.4-8 Using $y[0] = 0$, $y[1] = 1$, and $x[n] = e^n u[n]$, solve

$$y[n] + 2y[n-1] + 2y[n-2] = x[n-1] + 2x[n-2].$$

7.4-9 Find the following sums:

- (a) $s_a[n] = \sum_{k=0}^n k$
 (b) $s_b[n] = \sum_{k=0}^n k^2$
 (c) $s_c[n] = \sum_{k=0}^n k^3$
 (d) $s_d[n] = \sum_{k=0}^n k a^k$, $a \neq 1$

Hint: Consider a system whose output is $s[n]$, the desired sum. Examine the relationship between $s[n]$ and $s[n-1]$. Note also that $s[0] = 0$.

7.4-10 A causal LTID system has transfer function

$$H(z) = \frac{z}{(z+0.2)(z-0.8)}.$$

- (a) Is this system stable? Explain.
 (b) Find the zero-state response to input $x[n] = e^{(n+1)} u[n]$.
 (c) Write the difference equation relating the output $y[n]$ to input $x[n]$.
 (d) Determine the system's unit impulse response $h[n]$.

7.4-11 Repeat Prob. 7.4-10 if $x[n] = u[n]$ and

$$H(z) = \frac{2z+3}{(z-2)(z-3)}.$$

7.4-12 Repeat Prob. 7.4-10 if

$$H(z) = \frac{6(5z - 1)}{6z^2 - 5z + 1},$$

and the input is

- (a) $x_a = (4)^{-n}u[n]$
- (b) $x_b = (4)^{-(n-2)}u[n-2]$
- (c) $x_c = (4)^{-(n-2)}u[n]$
- (d) $x_d = (4)^{-n}u[n-2]$

7.4-13 Repeat Prob. 7.4-10 if $x[n] = u[n]$ and

$$H(z) = \frac{2z - 1}{z^2 - 1.6z + 0.8}.$$

7.4-14 Consider a system with transfer function

$$H(z) = \frac{z}{(z + 0.2)(z - 0.8)} \quad |z| > 0.8.$$

Determine the zero-state responses to the inputs

- (a) $x_a[n] = e^n u[n]$
- (b) $x_b[n] = 2^n u[-n - 1]$
- (c) $x_c[n] = e^n u[n] + 2^n u[-n - 1]$

7.4-15 For the system in Prob. 7.4-14, determine the zero-state response if the input is

$$x[n] = 2^n u[n] + u[-n - 1].$$

7.4-16 For the system in Prob. 7.4-14, determine the zero-state response if the input is

$$x[n] = e^{-2n} u[-n - 1].$$

7.4-17 Find the transfer functions corresponding to the systems specified by the difference equations in

- (a) Prob. 7.4-2
- (b) Prob. 7.4-3
- (c) Prob. 7.4-6
- (d) Prob. 7.4-8

7.4-18 Find the unit impulse responses of the systems described by the following equations:

$$(a) \quad y[n] + 3y[n - 1] + 2y[n - 2] = x[n] + 3x[n - 1] + 3x[n - 2]$$

$$(b) \quad y[n + 2] + 2y[n + 1] + y[n] = 2x[n + 2] - x[n + 1]$$

$$(c) \quad y[n] - y[n - 1] + 0.5y[n - 2] = x[n] + 2x[n - 1]$$

7.5-1 (a) Show the canonic direct form, a parallel realization, and a series realization of

$$H(z) = \frac{z(3z - 1.8)}{z^2 - z + 0.16}.$$

(b) Find the transpose of the realizations obtained in part (a).

7.5-2 Repeat Prob. 7.5-1 for

$$H(z) = \frac{5z + 2.2}{z^2 + z + 0.16}$$

7.5-3 Repeat Prob. 7.5-1 for

$$H(z) = \frac{3.8z - 1.1}{(z - 0.2)(z^2 - 0.6z + 0.25)}.$$

7.5-4 Repeat Prob. 7.5-1 for

$$H(z) = \frac{z(1.6z - 1.8)}{(z - 0.2)(z^2 + z + 0.5)}.$$

7.5-5 Repeat Prob. 7.5-1 for

$$H(z) = \frac{z(2z^2 + 1.3z + 0.96)}{(z + 0.5)(z - 0.4)^2}.$$

7.5-6 Realize a system with transfer function

$$H(z) = \frac{2z^4 + z^3 + 0.8z^2 + 2z + 8}{z^4}.$$

7.5-7 Realize a system with transfer function

$$H(z) = \sum_{n=0}^6 nz^{-n}.$$

7.5-8 This problem demonstrates the enormous number of ways of implementing even a relatively low-order transfer function. Consider a second-order transfer function with two real zeros and two real poles. Discuss various ways of realizing such a transfer function. Consider canonic direct, cascade, parallel, and the corresponding transposed forms. Note also that interchanging cascaded sections yields different realizations.

7.5-9 Consider an LTID system with transfer function

$$H(z) = \frac{z - 0.5}{z + 0.5}.$$

- (a) Determine the DFII realization of this system.
- (b) Determine the TDFII realization of this system.
- (c) Show that this system can be realized as a parallel combination of an identity system and $\frac{-1}{z+0.5}$.

7.6-1 For an asymptotically stable LTID system, show that the steady-state response to input $e^{j\Omega n}u[n]$ is $H(e^{j\Omega})e^{j\Omega n}u[n]$. The steady-state response is that part of the response which does not decay with time and persists forever.

7.6-2 Determine and plot the magnitude and phase responses of the digital filters depicted in Fig. P7.6-2. State the type (high-pass, lowpass, etc.) of each filter.

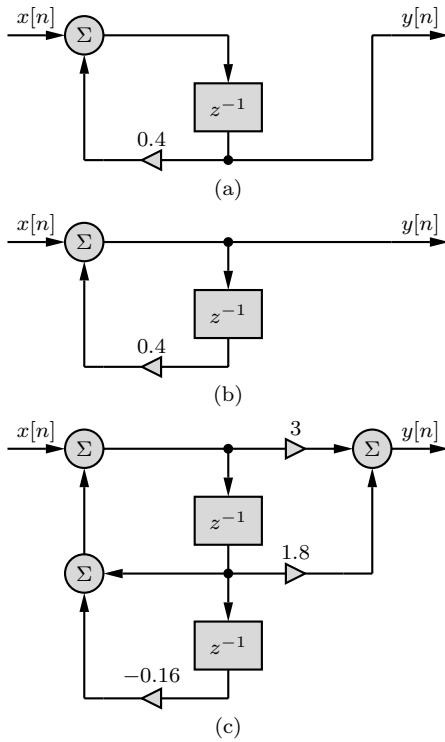


Figure P7.6-2

7.6-3 Determine the frequency response of the 5-point moving-average system given by

$$y[n] = \frac{1}{5} \sum_{k=0}^4 x[n-k].$$

7.6-4 Sketch the magnitude responses of the four filters depicted by the pole-zero plots shown in Fig. P7.6-4.

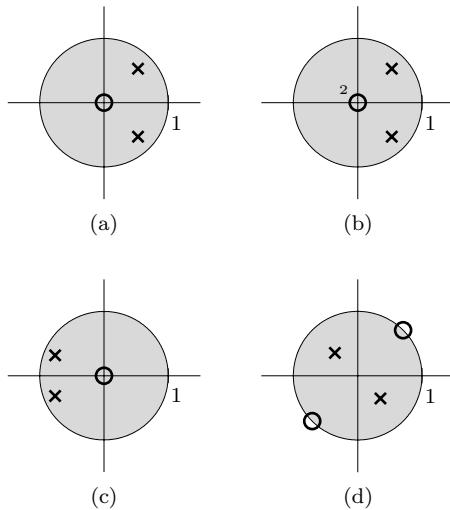


Figure P7.6-4

7.6-5 Let the LTID system shown in Fig. P7.6-5 have $a = \frac{1}{2}$.

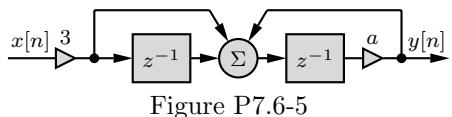


Figure P7.6-5

- (a) Determine the standard delay form difference equation description of this system.
- (b) Sketch the pole/zero plot for this system. Is this system primarily LP, HP, BP, or BS?
- (c) Determine the impulse response $h[n]$ of this system.
- (d) Is this system stable? Explain.
- (e) Is this system causal? Explain.
- (f) Determine the output of this system in response to $x[n] = 1 + \cos(\pi n)$.

7.6-6 Repeat Prob. 7.6-5 using $a = 2$.

7.6-7 The input-output relationships of two filters are described by

$$y[n] = -0.9y[n-1] + x[n]$$

and

$$y[n] = 0.9y[n-1] + x[n].$$

- (a) Determine the transfer function and frequency response of each filter.
- (b) For each filter, sketch the magnitude response and state the type (highpass, lowpass, etc.).
- (c) Find the response of each of these filters to a sinusoid $x[n] = \cos(\Omega_0 n)$ for $\Omega_0 = 0.01\pi$ and 0.99π . In general, show that the gain (magnitude response) of the first filter at frequency Ω_0 is the same as the gain of the second filter at frequency $\pi - \Omega_0$.

7.6-8 Determine and plot the magnitude and phase responses of the digital filters depicted in Fig. P7.6-8. Confirm that these are *linear phase filters*.

Hint: Express $H(e^{j\Omega})$ as $e^{-j2.5\Omega}$ times a real function of Ω .

7.6-9 The two linear phase filters of Prob. 7.6-8 possess symmetry in their scalar multipliers. Exploit this symmetry to realize each filter using half the number of multiplies found in Fig. P7.6-8.

7.6-10 Consider an LTID system specified by the equation

$$y[n+1] - 0.5y[n] = x[n+1] + 0.8x[n].$$

- (a) Determine and sketch the magnitude and the phase responses.
- (b) Find the system response $y[n]$ to the input $x[n] = \cos(0.5n - \pi/3)$.

7.6-11 A digital filter has a sampling interval T and sampling frequency $F_s = 1/T$.

- (a) If $T = 50 \mu s$, determine the highest frequency that can be processed by this filter without aliasing.

(b) If the highest frequency to be processed is 50 kHz, determine the minimum value of the sampling frequency F_s that can be used.

7.6-12 A continuous-time lowpass filter is realized using a discrete-time system, as in Fig. 7.28. When the discrete-time system operates with sampling interval $T = 50 \mu s$, the system has a cutoff frequency of 10 kHz. If the sampling interval is changed to $T = 25 \mu s$, determine the new cutoff frequency of the filter. Repeat the problem if T is changed to $200 \mu s$.

7.6-13 Consider a causal digital filter with transfer function

$$H(z) = K \frac{z+1}{z-\gamma},$$

where K and γ are both real.

- (a) Draw an efficient realization of this filter.
- (b) Sketch the magnitude response of this filter, assuming $|a| < 1$.
- (c) The magnitude response of this low-pass filter is maximum at $\Omega = 0$. The 3-dB bandwidth is the frequency where the magnitude response drops $1/\sqrt{2}$ times its maximum value. Determine the 3-dB bandwidth of this filter when $\gamma = 0.2$.

7.6-14 Design a (real) digital notch filter that rejects 5000-Hz content completely and sharply recovers on either side of 5000 Hz to a gain of unity. The highest frequency to be processed is 20 kHz ($f_{\max} = 20000$ Hz). Provide a canonic realization of the filter, and determine the filter's magnitude response.

Hint: See Ex. 7.20. The zeros should be at $e^{\pm j\omega T}$ for ω corresponding to 5000 Hz, and the poles are at $|\gamma|e^{\pm j\omega T}$ with $|\gamma| < 1$. Leave your answer in terms of $|\gamma|$.

7.6-15 Consider a first-order LTID system with a pole at $z = \gamma$ and a zero at $z = \frac{1}{\gamma^*}$, where $|\gamma| \leq 1$.

- (a) Show that this system is an allpass filter. That is, show that the magnitude

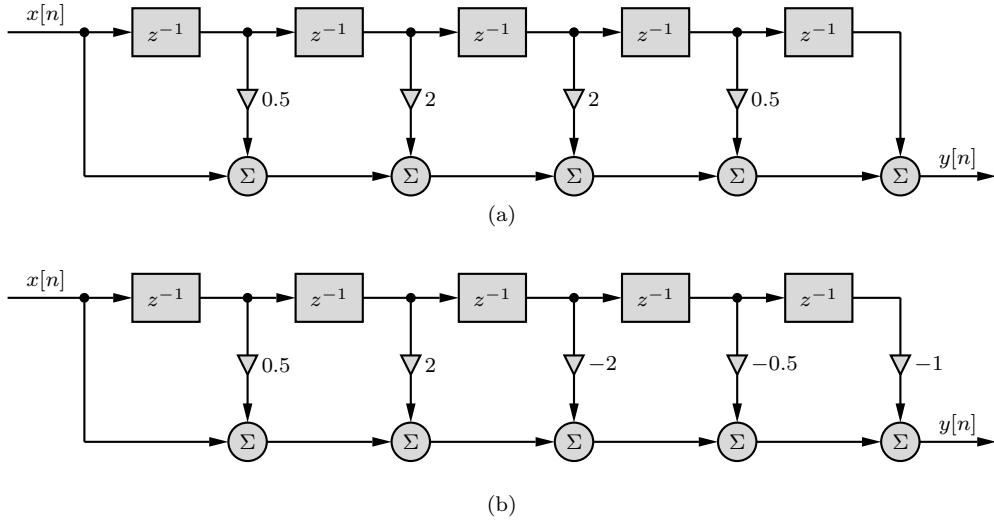


Figure P7.6-8

response $|H(e^{j\Omega})|$ is constant with frequency.

- (b) Generalize this result to describe how to construct a K th-order allpass filter.

7.6-16 Let the impulse responses of two LTIID system be related as

$$h_2[n] = (-1)^n h_1[n].$$

- (a) Show that

$$H_2(z) = H_1(e^{j\pm\pi} z).$$

How does the frequency response $H_2(e^{j\Omega})$ relate to $H_1(e^{j\Omega})$?

- (b) If $H_1(z)$ represents an ideal lowpass filter with cutoff frequency Ω_c , sketch $H_2(e^{j\Omega})$. What type of filter is $H_2(z)$?

7.7-1 (a) Determine the transfer function $H_{\text{direct}}(z)$ of the direct form second-order IIR system given in Fig. 7.32.

- (b) Determine the transfer function $H_{\text{coupled}}(z)$ of the coupled form second-order IIR system given in Fig. 7.34.

- (c) Compare $H_{\text{direct}}(z)$ and $H_{\text{coupled}}(z)$. How are these two systems the same? How are they different?

7.7-2 The 6th-order lowpass Chebyshev filter of Ex. 7.21 can be realized as a single-section

direct form system with transfer function

$$H(z) = \frac{\sum_{l=0}^6 b_l z^{-l}}{1 + \sum_{k=1}^6 a_k z^{-k}}.$$

As in Ex. 7.21, assume that $F_s = \frac{100}{\pi}$.

- (a) Determine the 13 coefficients comprising $H(z)$, and generate a pole-zero plot for this system.
 (b) Plot the magnitude response $|H(e^{j\Omega})|$ over $0 \leq \Omega/T \leq 100$, and verify that it matches Fig. 7.36.
 (c) Quantize the 13 coefficients comprising $H(z)$ using 12-, 10-, 8-, and 6-bit word lengths. Present your results in a form similar to Table 7.3.
 (d) Investigate 12-, 10-, 8-, and 6-bit coefficient quantization on the locations of the system poles and zeros. Identify those cases, if any, that produce realizations that are not stable.
 (e) Investigate 12-, 10-, 8-, and 6-bit coefficient quantization on the magnitude response of this system. Identify those cases, if any, that produce realizations that do not closely match the original filter.
 (f) Determine the smallest word length that produces a realization that essentially matches the original filter. Hint: The smallest word length may be a value other than 12, 10, 8, or 6.

Chapter 8

Digital Filters

Chapter 2 presents techniques for the systematic design of analog filters. Chapters 4, 5, 6, and 7 introduce various aspects of digital filters, primarily from a qualitative perspective.[†] In this chapter, we discuss techniques for the systematic design of digital filters to meet quantitative design specifications.

Digital filters are classified as either *infinite impulse response* (IIR) or *finite impulse response* (FIR). As discussed in Ch. 4, a typical K th-order LTID system is described by the difference equation

$$y[n] = -a_1y[n-1] - \cdots - a_{K-1}y[n-(K-1)] - a_Ky[n-K] + b_0x[n] + b_1x[n-1] + \cdots + b_{L-1}x[n-(L-1)] + b_Lx[n-L]. \quad (8.1)$$

If the feedback coefficients $a_k = 0$ for $k = 1, 2, \dots, K$, then the system is FIR. If even one of these feedback coefficients is nonzero, however, then the system is IIR.

For a given set of specifications, IIR filters tend to be much lower order than FIR filters, which can make them much more efficient to implement. On the other hand, it is often simpler to design FIR filters with arbitrary frequency response characteristics. Unlike IIR filters, FIR filters are also guaranteed stable and can be made to have linear phase, both desirable characteristics. In any case, IIR and FIR filters require different design methods. In this chapter, we shall begin with the design of IIR filters and then conclude with the design of FIR filters.

8.1 Infinite Impulse Response Filters

Most commonly, IIR filters are designed to follow the difference equation form of Eq. (8.1). In this case, there are $K + L + 1$ unknowns: $L + 1$ feedforward coefficients (b_l) and K feedback coefficients (a_k). Equivalently, IIR filter design requires locating L zeros and K poles and setting the overall filter gain. Particularly for high-order designs, systematic procedures are necessary to successfully design systems to meet given specifications.

From Ch. 2, we know how to design a good selection of analog IIR filters, including Butterworth, Chebyshev, inverse Chebyshev, and others. We can capitalize on this knowledge in the design of digital IIR filters. Through transformation, we can convert an analog IIR filter into digital form. Notice, however, that the *periodic* frequency response of a digital filter is necessarily an approximation of the *aperiodic* frequency response of an analog filter. Different criteria lead to different approximations and, therefore, different transformation rules. We present here several popular methods to convert analog IIR filters into digital IIR filters.

[†]Following common terminology, we use “analog” and “digital” filters to refer to what are more accurately termed “continuous-time” and “discrete-time” filters.

8.1.1 The Impulse Invariance Method Revisited

In Sec. 6.5.2, we studied the impulse invariance method to realize analog filters using digital systems, as depicted in Fig. 8.1. For bandlimited responses, the impulse invariance method yields exact results. Unfortunately, all practical systems are non-bandlimited.[†] What happens if the impulse invariance method is used for any practical design? The resulting design will be an approximation of what we want. We shall now consider the nature of this approximation and the error in the resulting design. This will enable us to find a procedure that will result in a design within given error specifications.

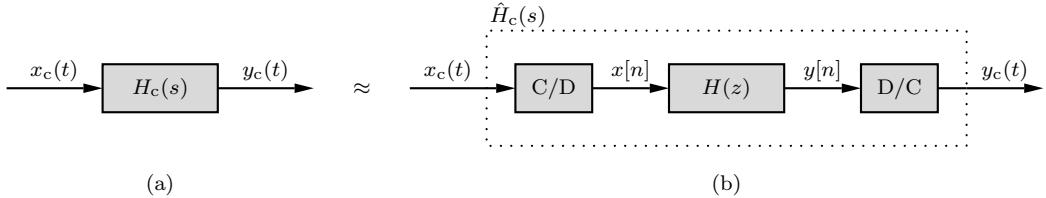


Figure 8.1: Realization of (a) a desired CT filter using (b) a DT filter and interfaces.

Realization of Rational $H(s)$

Let us begin with a simple case. Suppose that we wish to realize a first-order continuous-time filter with transfer function

$$H_c(s) = \frac{c}{s - \lambda}.$$

The impulse response $h_c(t)$, given by the inverse Laplace transform of $H_c(s)$, is

$$h_c(t) = ce^{\lambda t}u(t).$$

The corresponding digital filter unit impulse response $h[n]$, as per Eq. (6.55), is

$$h[n] = Th_c(nT) = Tce^{\lambda nT}u(nT).$$

Recognizing that $u(0) = 0.5$ for the CT unit step but $u[0] = 1$ for the DT unit step, this simplifies to

$$h[n] = Tc(e^{\lambda T})^n u[n] - \frac{Tc}{2}\delta[n]. \quad (8.2)$$

Figure 8.2 shows an example $h_c(t)$ and the corresponding $h[n]$.

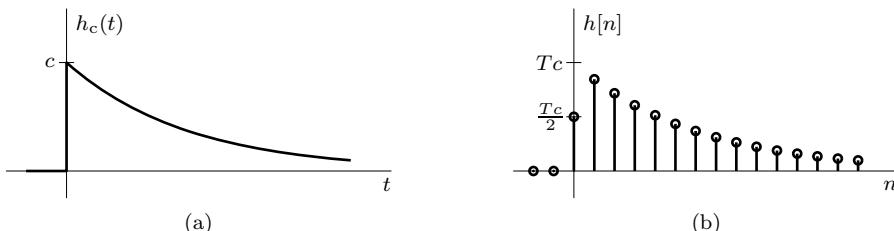


Figure 8.2: Impulse invariance filter design: (a) CT impulse response and (b) DT impulse response.

At this point, it is reasonable to ask whether the term $-\frac{Tc}{2}\delta[n]$ in Eq. (8.2) is really needed. Looking at Fig. 8.2, this term makes $h[0]$ half as large as we might intuitively expect. In fact, most

[†]Any practical CT system is causal, which is to say that $h_c(t) = h_c(t)u(t)$. The abrupt transition $u(t)$ implicit in $h_c(t)$ ensures that the frequency response $H_c(j\omega)$ is non-bandlimited.

signal processing references follow this logic and do not include $-\frac{Tc}{2}\delta[n]$. Unfortunately, intuition steers us wrong in this case. When using the impulse invariance method, we should always use the midpoint when sampling any point of discontinuity. As we shall see in Ex. 8.1, such a strategy generates superior results [1,4].

With the help of Table 7.1, the z -transform of the impulse response of Eq. (8.2) is

$$H(z) = \frac{Tc}{2} \left(\frac{z + e^{\lambda T}}{z - e^{\lambda T}} \right). \quad (8.3)$$

This result is readily extended to higher-order systems. First, we express a K th-order CT transfer function $H_c(s)$ as a sum of partial fractions as[†]

$$H_c(s) = \sum_{k=1}^K \frac{c_k}{s - \lambda_k}. \quad (8.4)$$

The corresponding $H(z)$ is given by

$$H(z) = \frac{T}{2} \sum_{k=1}^K c_k \left(\frac{z + e^{\lambda_k T}}{z - e^{\lambda_k T}} \right). \quad (8.5)$$

This transfer function can be readily realized, as explained in Sec. 7.5. Table 8.1 lists several pairs of $H_c(s)$ and their corresponding $H(z)$, including cases involving repeated poles. Notice that the $-\frac{Tc}{2}\delta[n]$ term occurs in $h[n]$ only when $h_c(t)$ contains a discontinuity (pairs 1 and 4). Furthermore, as T becomes small, $-\frac{Tc}{2}\delta[n]$ has diminishing impact, and good results can be obtained even if the term is omitted. Regardless, the responses $H_c(j\omega)$ in Eq. (8.4) and Table 8.1 are not bandlimited, which causes distortion in the corresponding DT filter responses $H(e^{j\Omega})$. Our next task is to better understand the nature of this distortion.

$H_c(s)$	$h_c(t)$	$h[n]$	$H(z)$
1. $\frac{1}{s}$	$u(t)$	$Tu[n] - \frac{T}{2}\delta[n]$	$\frac{T}{2} \left(\frac{z+1}{z-1} \right)$
2. $\frac{1}{s^2}$	$tu(t)$	$nT^2 u[n]$	$\frac{T^2 z}{(z-1)^2}$
3. $\frac{1}{s^3}$	$\frac{t^2}{2}u(t)$	$\frac{n^2 T^3}{2} u[n]$	$\frac{T^3 z(z+1)}{2(z-1)^3}$
4. $\frac{1}{s-\lambda}$	$e^{\lambda t}u(t)$	$T(e^{\lambda T})^n u[n] - \frac{T}{2}\delta[n]$	$\frac{T}{2} \left(\frac{z+e^{\lambda T}}{z-e^{\lambda T}} \right)$
5. $\frac{1}{(s-\lambda)^2}$	$te^{\lambda t}u(t)$	$nT^2(e^{\lambda T})^n u[n]$	$\frac{T^2 z e^{\lambda T}}{(z-e^{\lambda T})^2}$
6. $\frac{1}{(s-\lambda)^3}$	$\frac{t^2}{2}e^{\lambda t}u(t)$	$\frac{n^2 T^3}{2}(e^{\lambda T})^n u[n]$	$\frac{T^3 z e^{\lambda T} (z+e^{\lambda T})}{2(z-e^{\lambda T})^3}$

Table 8.1: Selected impulse invariance pairs.

Impact of the Sampling Interval T on the Impulse Invariance Method

The impulse invariance criterion in Eq. (6.55) seeks a time-domain equivalence of the two systems in Fig. 8.1, which is to say that the DT impulse response $h[n]$ relates directly to samples of the CT impulse response $h_c(t)$ as

$$h[n] = Th_c(nT).$$

[†]Equation (8.4) assumes that $H_c(s)$ has only simple poles. For repeated poles, the form changes accordingly, such as shown in Table 8.1.

Let us see how this time-domain equivalence holds up in the frequency domain.

Consider a hypothetical frequency response $H_c(j\omega)$ (Fig. 8.3a) that we wish to realize using a digital filter. From the discussion in Secs. 6.4 and 6.5, we know that the spectrum $H(e^{j\Omega})$ is the frequency-scaled spectrum $H_c(j\omega)$ repeating periodically with period $2\pi/T$, as shown in Fig. 8.3b. Because all practical filters are non-bandlimited, the overlap among various cycles and the consequent aliasing cannot be avoided. Thus, the frequency response $\hat{H}_c(j\omega)$ that we realize is a distorted version of $H_c(j\omega)$ (Fig. 8.3c). The distortion occurs because of two factors. First, there is a loss of the spectral tail consisting of the components of $H_c(j\omega)$ beyond π/T (light shading), and second, this very tail folds back at the folding frequency and reappears through aliasing to distort frequencies below π/T (dark shading).

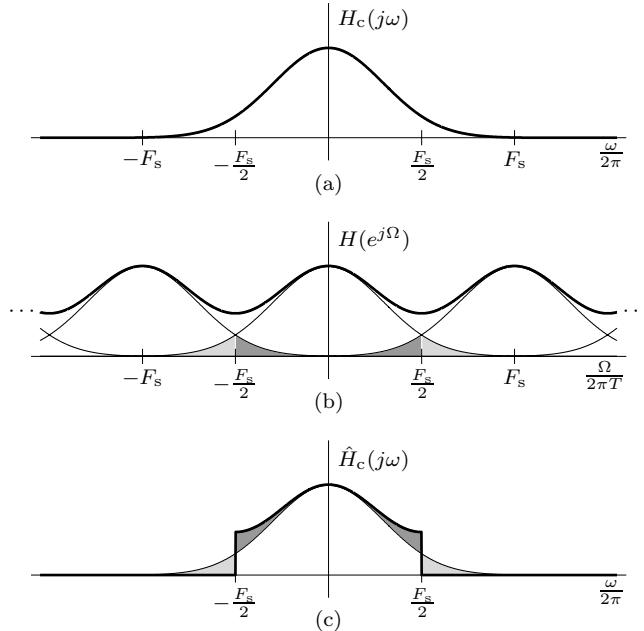


Figure 8.3: Aliasing in digital filters: (a) desired CT filter $H_c(j\omega)$, (b) DT filter $H(e^{j\Omega})$ with aliasing, and (c) realized CT filter $\hat{H}_c(j\omega)$ with distortion.

To reduce the distortions in $H(e^{j\Omega})$ as well as $\hat{H}_c(j\omega)$, aliasing must be minimized by choosing a sufficiently small sampling interval T . As a rule of thumb, we choose T such that the spectrum $|H_c(j\omega)|$ at the folding frequency $F_s/2$ becomes a negligible fraction, such as 1% of the peak value of $|H_c(j\omega)|$. For a lowpass filter with peak amplitude at $\omega = 0$, for example, a 1% criterion suggests that we select T such that

$$|H(j\pi/T)| \leq 0.01|H(j0)|.$$

If $H_c(j\omega)$ is bandlimited to B Hz, there is no spectral overlap and, of course, no aliasing, provided that we choose T such that $B \leq 1/2T$. The realized filter matches the desired response $H_c(j\omega)$ without error, as proven in Sec. 6.5.2. For some filters, such as highpass and bandstop filters, no value of T , no matter how small, can be found to avoid aliasing. As a result, the impulse invariance method cannot be used in these cases.[†]

We should also give due consideration to the rate at which the spectrum $|H_c(j\omega)|$ decays with frequency. If K and L represent the orders of the denominator and numerator polynomials of $H_c(s)$, then, generally, this rate of decay at higher frequencies is $6(K-L)$ dB/octave or $20(K-L)$

[†]We can see this fact in the time domain as well. Highpass and bandstop filters have a $\delta(t)$ term in their impulse responses that, regardless of T , cannot be properly sampled using the impulse invariance criterion of $h[n] = Th_c(nT)$.

dB/decade. Thus, higher values of $K - L$ can tolerate higher values of $|H_c(j\pi/T)|$, and we can get by with somewhat larger T .

Limitations of the Impulse Invariance Method

The impulse invariance method is handicapped by aliasing. Consequently, this method can be used to design filters where $H_c(j\omega)$ becomes negligible beyond some frequency B Hz. This condition restricts the procedure to lowpass and bandpass filters. The impulse invariance method cannot be used for highpass or bandstop filters. Moreover, to reduce aliasing effects, the sampling rate has to be very high, which makes its implementation costly. In applications where the sampling rate is fixed, aliasing may produce unacceptable levels of distortion. In general, the frequency-domain method discussed in the next section is superior to the impulse invariance method.

► **Example 8.1 (Impulse Invariance Method: Digital Butterworth LPF)**

Using the impulse invariance method, design a digital filter to realize a first-order Butterworth lowpass filter with 3-dB cutoff frequency $\omega_c = 10^5$ rad/s.

From Ch. 2, the transfer function of a first-order Butterworth lowpass filter with 3-dB cutoff frequency ω_c is

$$H_c(s) = \frac{\omega_c}{s + \omega_c}.$$

According to Eq. (8.3) (or pair 4 in Table 8.1), the transfer function $H(z)$ of the corresponding digital filter is

$$H(z) = \frac{\omega_c T}{2} \left(\frac{z + e^{-\omega_c T}}{z - e^{-\omega_c T}} \right). \quad (8.6)$$

Typically, we would next select the value of T according to the criterion where the gain at $\omega = \pi/T$ drops to 1% of the maximum filter gain. However, this choice results in such a good design that aliasing is nearly imperceptible. The resulting magnitude response is so close to the desired response that we can hardly notice the aliasing effect in our plot. For the sake of demonstrating the aliasing effect, we shall deliberately select a 10% criterion instead of a 1% criterion.

Setting $s = j\omega$ in $H_c(s)$, we have

$$|H_c(j\omega)| = \frac{\omega_c}{\sqrt{\omega^2 + \omega_c^2}}.$$

In this case, the maximum value of $|H_c(j\omega)|$ is 1, which occurs at $\omega = 0$. Applying a 10% criterion leads to $|H_c(j\pi/T)| = 0.1$. Observe that

$$|H_c(j\omega)| \approx \frac{\omega_c}{\omega} \quad \text{for } \omega \gg \omega_c.$$

Hence,

$$|H_c(j\pi/T)| \approx \frac{\omega_c}{\pi/T} = 0.1 \implies \pi/T = 10\omega_c = 10^6.$$

Thus, a 10% criterion yields $T = 10^{-6}\pi$ (a 1% criterion yields $T = 10^{-7}\pi$, which increases the sampling frequency $F_s = 1/T$ by one full order of magnitude). Substitution of $T = 10^{-6}\pi$ in Eq. (8.6) yields

$$H(z) = 0.1571 \left(\frac{z + 0.7304}{z - 0.7304} \right) = \frac{0.1571z + 0.1147}{z - 0.7304}.$$

A canonical TDFII realization of this filter is shown in Fig. 8.4.

To confirm proper filter behavior, the frequency responses of the digital and analog filters are compared using MATLAB. As Fig. 8.5 makes clear, the digital filter (solid line) does a respectable job in behaving like the analog filter (dashed line), although aliasing distortions are apparent, particularly at higher frequencies (near $F_s/2$).

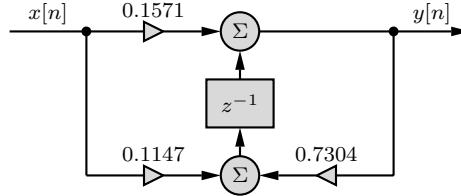


Figure 8.4: Canonical TDFII realization of $H(z) = \frac{0.1571z+0.1147}{z-0.7304}$.

```

01 omegac = 10^5; T = 10^(-6)*pi; Omega = linspace(0,pi,1001);
02 omega = Omega/T; f = omega/(2*pi); Hc = @(s) omegac./(s+omegac);
03 H = @(z) omegac*T/2*(z+exp(-omegac*T))./(z-exp(-omegac*T));
04 subplot(121); plot(f,abs(H(exp(1j*Omega))),'k',f,abs(Hc(1j*omega)),'k--');
05 subplot(122); plot(f,angle(H(exp(1j*Omega))),'k',f,angle(Hc(1j*omega)),'k--');

```

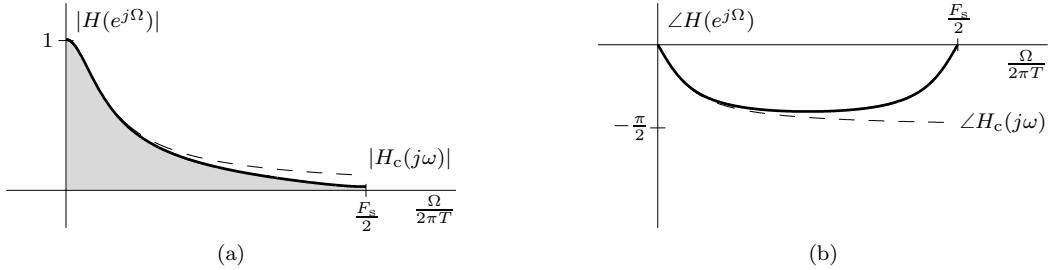


Figure 8.5: Impulse invariance method: digital (solid) and analog (dashed) filters' (a) magnitude and (b) phase responses.

Before concluding this example, we wish to again stress the importance of using the midpoint when sampling discontinuities in $h_c(t)$. Consider what happens in the present example if we assign $u(0) = 1$ (as done in most signal processing texts) or, alternatively, if we assign the left limit $u(0) = 0$. As Fig. 8.6 demonstrates, the responses of the resulting digital filters ($u(0) = 1$ case dashed, $u(0) = 0$ case dotted) are significantly degraded from the $u(0) = 0.5$ case (solid). As the magnitude response plots show, these degradations are not just limited to high frequencies but also can significantly impact filter behavior at low frequencies.

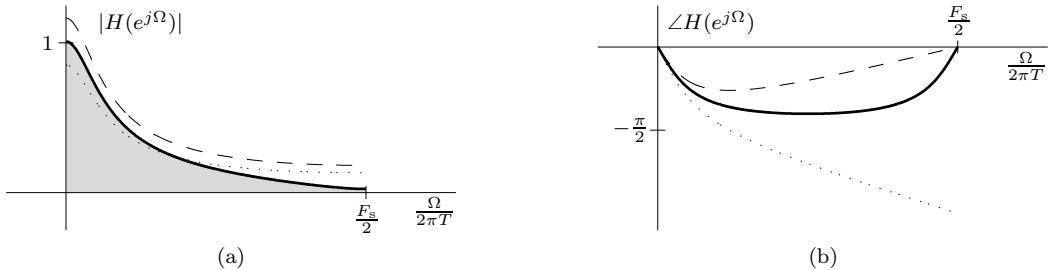


Figure 8.6: Frequency responses for impulse invariance method with $u(0) = 0.5$ (solid), $u(0) = 1$ (dashed), and $u(0) = 0$ (dotted).

▷ **Drill 8.1 (Impulse Invariance Method: Digital Butterworth LPF)**

Using a 10% rule, design a DT system to implement a Butterworth lowpass filter with a 3-dB cutoff frequency of 250 Hz. The DT system must operate at a sampling frequency $F_s \leq 10000$ Hz. What is the effect of using a 1% rule on the resulting digital filter?

◀

8.1.2 The Bilinear Transform

Impulse invariance is a time-domain method that converts a CT impulse response $h_c(t)$ to a DT impulse response $h[n]$. We now develop the bilinear transform, which is a frequency-domain method that converts an analog filter transfer function $H_c(s)$ into a digital filter $H(z)$. In filtering problems where the magnitude response is piecewise constant, the bilinear transform is preferable to the impulse invariance method. Hence, the bilinear transform is commonly employed to design lowpass, bandpass, highpass, bandstop, and other filters where the gains are constant over certain bands. This method yields sharper rolloff characteristics compared with the impulse invariance method. Moreover, the one-to-one mapping from the s -plane to the z -plane inherent in this method avoids the aliasing problem. The bilinear transform is suitable to design analog filters that use digital processors with A/D and D/A converters as well as digital filters that process digital signals directly.

Let us start with the design of an analog filter using a digital processor with A/D and D/A converters, as in Fig. 8.1. We first use one of the IIR filter design methods of Ch. 2 to determine an analog filter $H_c(s)$ that meets design requirements. To obtain the desired digital filter $H(z)$, we then apply a suitable transformation $s = f(z)$ that maps the s -plane to the z -plane. By substituting $s = f(z)$ into $H_c(s)$, we obtain the desired DT filter $H(z)$. Section 7.8 suggests that the z -transform can be considered as a Laplace transform with a change of variable $z = e^{sT}$ or $s = \ln(z)/T$, where T is the sampling interval. It is tempting, therefore, to convert a continuous-time filter to a discrete-time filter by the substitution $H(z) = H_c(s)|_{s=\ln(z)/T}$. Unfortunately, this approach is impractical since the resulting $H(z)$ is not rational and therefore cannot be implemented using standard adder, scalar multiplier, and delay blocks.[†] We require a transformation that produces a rational transfer function $H(z)$. How do we choose such a $f(z)$? There are several approaches to this problem.

The Backward Difference Transform

We indirectly touched on one such approach in Sec. 4.4, which, to realize an analog system digitally, approximates a differential equation (Eq. (4.48)) specifying the analog system with a difference equation of the same order (Eq. (4.49)). This method uses the backward difference approximation to a derivative, given as[‡]

$$\frac{d}{dt}y(t) \approx \frac{y[n] - y[n-1]}{T}.$$

The left-hand side is the CT derivative operation with transfer function s . The right-hand side is the DT backward difference operation with transfer function $(1 - z^{-1})/T$. This method, therefore, leads to the *backward difference transform*

$$s = \frac{1 - z^{-1}}{T} = \frac{z - 1}{Tz}. \quad (8.7)$$

[†]Although not considered here, the so-called matched z -transform relies on the relationship $z = e^{sT}$ to convert the poles and zeros of $H_c(s)$ to a corresponding digital design $H(z)$, so this relationship does find relevance to digital filter design.

[‡]For most functions $y(t)$, the approximation improves toward equality in the limit,

$$\frac{d}{dt}y(t) = \lim_{T \rightarrow 0} \frac{y[n] - y[n-1]}{T}.$$

From Eq. (4.51), we observe that the second derivative can be approximated as

$$\frac{d^2}{dt^2}y(t) \approx \frac{1}{T^2} \left(y[n] - 2y[n-1] + y[n-2] \right).$$

Consistent with Eq. (8.7), this leads to the transformation

$$s^2 = \frac{1 - 2z^{-1} + z^{-2}}{T^2} = \left(\frac{1 - z^{-1}}{T} \right)^2.$$

Similarly, we can verify from Eq. (4.52) that, for any integer k , the transformation for s^k is

$$s^k = \left(\frac{1 - z^{-1}}{T} \right)^k.$$

Thus, to realize a rational function $H_c(s)$, we apply the backward difference transform of Eq. (8.7) to obtain a DT system transfer function $H(z)$ as

$$H(z) = H_c(s)|_{s=\frac{1}{T}(1-z^{-1})}. \quad (8.8)$$

The digital realizations of an analog differentiator and integrator found in Exs. 4.9 and 4.10, respectively, conform with this transformation: the analog differentiator $H_c(s) = s$ is realized with $H(z) = (1 - z^{-1})/T$, and the analog integrator $H_c(s) = 1/s$ is realized with $H(z) = T/(1 - z^{-1})$.

Although the backward difference transform of Eq. (8.7) works reasonably well for low frequencies (lowpass and bandpass filters), it has a drawback in that it maps the s -plane ω -axis (representing CT sinusoids) not onto the z -plane unit circle (representing DT sinusoids) but onto the unit-diameter circle centered at $z = 0.5$.[†] For this reason, Eq. (8.7) is ineffective for highpass and bandstop filters. Further, even lowpass and bandpass filters may require higher-than-desired sampling rates. Put bluntly, the backward difference transform is rather crude and not very desirable in practice.

Trapezoidal Rule: The Bilinear Transform

To improve the performance of the backward difference transform, we look for a better approximation to the derivative operation. This is obtained by considering the trapezoidal rule for integration found in Ex. 4.10 (Eq. (4.46)) as

$$y[n] - y[n-1] = \frac{T}{2} \left(x[n] + x[n-1] \right).$$

The z -transform of this equation yields

$$\frac{Y(z)}{X(z)} = \frac{T}{2} \left(\frac{z+1}{z-1} \right).$$

This expression corresponds to an analog integrator $H_c(s) = 1/s$ and suggests using the transformation[‡]

$$s = \frac{2}{T} \left(\frac{z-1}{z+1} \right). \quad (8.9)$$

[†]To show this, set $s = j\omega$ in Eq. (8.7) and solve for z as

$$z = \frac{1}{1 - j\omega T} = \underbrace{\left(\frac{1}{1 + \omega^2 T^2} \right)}_x + j \underbrace{\left(\frac{\omega T}{1 + \omega^2 T^2} \right)}_y = x + jy.$$

Observing that $(x - 0.5)^2 + y^2 = (0.5)^2$ confirms that $s = j\omega$ produces values z that lie on a circle with radius 0.5 centered at the point $(0.5, 0)$.

[‡]The transformation of Eq. (8.9) is a *conformal*, or angle-preserving, mapping, a topic that requires some background in the theory of complex variables. For our purposes, we need not worry about the details.

Equation (8.9) is the desired *bilinear transform*, from which we obtain

$$z = \frac{1 + \frac{sT}{2}}{1 - \frac{sT}{2}}. \quad (8.10)$$

From these relationships, we see that the bilinear transform is a one-to-one mapping between the s -and z -planes.

Unlike the backward difference transform, the bilinear transform maps the s -plane ω -axis onto the z -plane unit circle, which allows it to handle highpass and bandstop filters. Furthermore, it maps the left-half s -plane inside the z -plane unit circle and the right-half s -plane outside the z -plane unit circle. As a consequence, the bilinear transform maps a stable CT filter to a stable DT filter. To show these mapping properties, we use Eq. (8.10) with $s = \sigma + j\omega$ to obtain

$$z = \frac{2 + \sigma T + j\omega T}{2 - \sigma T - j\omega T}. \quad (8.11)$$

From this expression, we see that $|z|$ is greater than unity if $\sigma > 0$ and is less than unity if $\sigma < 0$. This means that the left and right halves of the s -plane, respectively, map into the inside and outside of the z -plane. When $s = j\omega$ (implying that $\sigma = 0$), Eq. (8.11) shows that $|z| = 1$, implying that the ω -axis in the s -plane maps onto the unit circle in the z -plane.

To find the relationship between analog frequency ω and the corresponding digital frequency Ω , let us consider Eq. (8.9) with $s = j\omega$ and $z = e^{j\Omega}$, that is,

$$j\omega = \frac{2}{T} \left(\frac{e^{j\Omega} - 1}{e^{j\Omega} + 1} \right) = \frac{2}{T} \left(\frac{e^{j\Omega/2} - e^{-j\Omega/2}}{e^{j\Omega/2} + e^{-j\Omega/2}} \right) = \frac{2}{T} \left(\frac{j \sin(\Omega/2)}{\cos(\Omega/2)} \right) = j \frac{2}{T} \tan\left(\frac{\Omega}{2}\right).$$

In other words, the bilinear transform causes

$$\omega = \frac{2}{T} \tan\left(\frac{\Omega}{2}\right) \quad \text{and} \quad \Omega = 2 \tan^{-1}\left(\frac{\omega T}{2}\right). \quad (8.12)$$

As ω varies from 0 to ∞ , Ω varies from 0 to π . Since every value ω one-to-one maps to a corresponding value Ω , the bilinear transform does not suffer from the aliasing problems that plague the impulse invariance method.

Like the backward difference transform, the bilateral transform starts with a known analog filter transfer function $H_c(s)$. Applying the substitution $s = \frac{2}{T} \left(\frac{z-1}{z+1} \right)$ yields the desired digital filter's transfer function as

$$H(z) = H_c(s)|_{s=\frac{2}{T}(\frac{z-1}{z+1})}. \quad (8.13)$$

Let us consider the impact of this transform on a rational $H_c(s)$ with L zeros z_l and K poles p_k , written in factored form as (see Eq. (1.52))

$$H_c(s) = \left(\frac{b_L}{a_K} \right) \frac{\prod_{l=1}^L (s - z_l)}{\prod_{k=1}^K (s - p_k)}. \quad (8.14)$$

Applying the bilinear transform and simplifying the result yield (see Prob. 8.1-26)

$$H(z) = \left(\frac{b_L \prod_{l=1}^L (\frac{2}{T} - z_l)}{a_K \prod_{k=1}^K (\frac{2}{T} - p_k)} \right) \frac{\prod_{l=1}^L \left(z - \frac{1+z_l T/2}{1-z_l T/2} \right)}{\prod_{k=1}^K \left(z - \frac{1+p_k T/2}{1-p_k T/2} \right)} (z + 1)^{K-L}. \quad (8.15)$$

From Eq. (8.15), we see that the bilinear transform maps the CT zeros z_l to DT zeros $\frac{1+z_l T/2}{1-z_l T/2}$ and the CT poles p_k to DT poles $\frac{1+p_k T/2}{1-p_k T/2}$. Further, the bilinear transform maps the $K - L$ zeros at ∞

to $z = -1$. Each pole or zero of the CT filter independently maps to a corresponding pole or zero of the DT filter; only the overall gain term of the DT filter depends on a combined interaction of terms.

The frequency response of the digital filter of Eq. (8.13) is obtained by setting $z = e^{j\Omega}$. Using the result in Eq. (8.12), we obtain

$$H(e^{j\Omega}) = H_c \left(j \frac{2}{T} \tan \left[\frac{\Omega}{2} \right] \right). \quad (8.16)$$

We observed earlier that as ω varies from 0 to ∞ , Ω varies from 0 to π . Hence, as Ω varies from 0 to π , $H(e^{j\Omega})$ takes on exactly the same values as that of $H_c(j\omega)$ as ω varies from 0 to ∞ . Thus, the analog filter frequency response over the entire frequency range of $0 \leq \omega < \infty$ is compressed to the range $0 \leq \Omega < \pi$ for the digital filter frequency response. This *frequency warping* is the basic difference between the bilateral transform and the impulse invariance technique. In the latter, the mapping from analog to digital frequency is linear, and therefore, aliasing occurs for all frequencies ω that map beyond $\Omega = \pi$. Conversely, the bilinear transform, while beset with frequency warping, is not tormented by aliasing. Furthermore, the distortions caused by warping are tolerable in many applications. For example, frequency warping can be compensated in filter applications that require ideal piecewise-constant magnitude responses.

Impact of the Sampling Interval T on the Bilinear Transform

The absence of aliasing in the bilinear transform is not a license to use an arbitrarily large sampling interval T (or low sampling frequency F_s). We still must ensure that the input is sampled at a rate that avoids aliasing. If the highest frequency to be processed is f_{\max} Hz, then to avoid signal aliasing, we must use

$$T \leq \frac{1}{2f_{\max}}. \quad (8.17)$$

Further, the value of T impacts the way that frequencies are warped, as suggested by Eq. (8.12). Let us clarify all these ideas with an example. Later, we shall learn how to better remedy the ills of frequency warping.

▷ Example 8.2 (Bilinear Transform: Digital Butterworth LPF)

Using the bilinear transform, design a digital filter to realize a first-order Butterworth lowpass filter with 3-dB cutoff frequency $\omega_c = 10^5$ rad/s.

Example 8.1 solves this problem using the impulse invariance method. Here, we shall use the bilinear transform and then compare the results of the two methods.

As shown in Ex. 8.1, the transfer function of a first-order Butterworth lowpass filter with 3-dB cutoff frequency ω_c is $H_c(s) = \omega_c/(s + \omega_c)$. This transfer function is already in the factored form of Eq. (8.14). Noting that $b_L/a_K = \omega_c$, $L = 0$, $K = 1$, and $p_1 = -\omega_c$, we use Eq. (8.15) to obtain

$$H(z) = \left(\frac{\omega_c}{\frac{2}{T} + \omega_c} \right) \frac{z + 1}{z - \frac{1 - \omega_c T/2}{1 + \omega_c T/2}}.$$

Normally, we should use Eq. (8.17) to select a suitable value for T based on the highest input frequency f_{\max} . However, to facilitate comparison with the impulse invariance method, we choose the Ex. 8.1 value of $T = \frac{\pi}{10\omega_c}$. Making this substitution into the previous equation yields

$$H(z) = 0.1358 \left(\frac{z + 1}{z - 0.7285} \right).$$

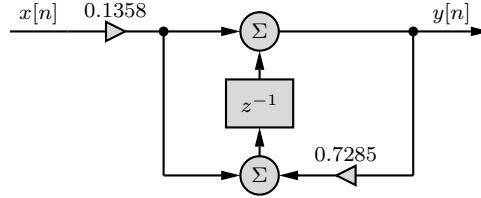


Figure 8.7: Canonical TDFII-like realization of $H(z) = 0.1358 \frac{z+1}{z-0.7285}$.

A canonical TDFII-like realization of this filter is shown in Fig. 8.7. Compared with Fig. 8.4, we see in this case that the bilinear transform produces a solution that is very similar to the solution of the impulse invariance method.

To verify filter behavior, the frequency responses of the digital and analog filters are compared using MATLAB. The following code, while somewhat bulky for this simple first-order example, easily accommodates higher-order cases. This code also readily verifies the numerator and denominator coefficients of $H(z)$ computed earlier.

```

01 Omega = linspace(0,pi,1001); T = 10^(-6)*pi; omega = Omega/T; f = omega/(2*pi);
02 omegac = 10^5; Hc = omegac./(1j*omega+omegac);
03 Z = [] ; P = [-omegac]; bL = omegac; aK = 1; L = length(Z); K = length(P);
04 B = bL/aK*prod(2/T-Z)/prod(2/T-P)*poly([(1+Z*T/2)/(1-Z*T/2),-ones(1,K-L)])
B = 0.1358 0.1358
05 A = poly((1+P*T/2)/(1-P*T/2))
A = 1.0000 -0.7285
06 H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
07 subplot(121); plot(f,abs(H),'k',f,abs(Hc),'k--');
08 subplot(122); plot(f,angle(H),'k',f,angle(Hc),'k--');
```

As Fig. 8.8 makes clear, the bilinear transform produces a digital filter (solid line) that closely matches the analog filter (dashed line), although distortions from frequency warping are apparent at higher frequencies. In this example, both the bilinear transform (Fig. 8.8) and the impulse invariance method (Fig. 8.5) produce satisfactory approximations of the analog filter characteristics. Generally, as the phase responses illustrate in this case, the bilinear transform tends to outperform the impulse invariance method.

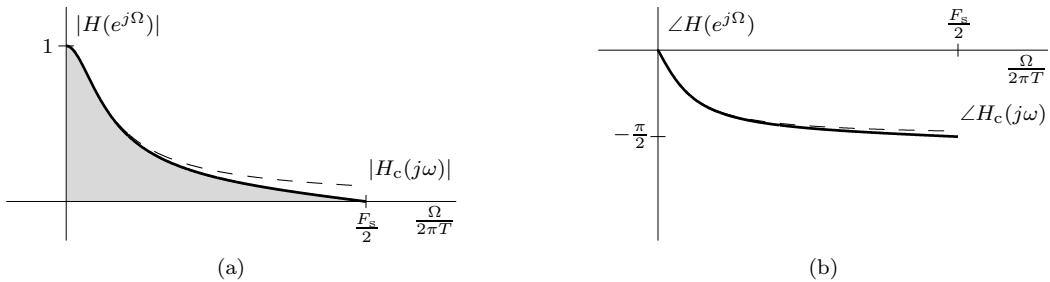


Figure 8.8: Bilinear transform: digital (solid) and analog (dashed) filters' (a) magnitude and (b) phase responses.

Every Possible Thrill, Adventure, Passion, and Sensation

Figure 8.8 shows that $|H(e^{j\Omega})| \approx |H_c(j\omega)|$ for small frequencies. As frequency increases toward the folding frequency $F_s/2$, the error increases. Moreover, $|H(e^{j\Omega})| = 0$ at the folding frequency $\Omega = \pi$. In fact, as observed earlier, the entire frequency band ($0 \leq \omega < \infty$) in $|H_c(j\omega)|$ is compressed within the range ($0 \leq \Omega < \pi$) in $|H(e^{j\Omega})|$. This is as if a promising 20-year-old youth, who, after learning that he has only a year to live, tries to crowd his last year with every possible thrill, adventure, passion, and sensation that a normal human being would have experienced in an entire lifetime.

Frequency Warping Inherent in the Bilinear Transform

To better understand this warping behavior, consider Fig. 8.9, which illustrates the bilinear transform graphically. The desired analog response $|H_c(j\omega)|^2$ (note the reverse orientation of the vertical axis) is shown in the lower right, the transformation rule is shown in the upper right, and the resulting digital response $|H(e^{j\Omega})|^2$ is shown in the upper left.

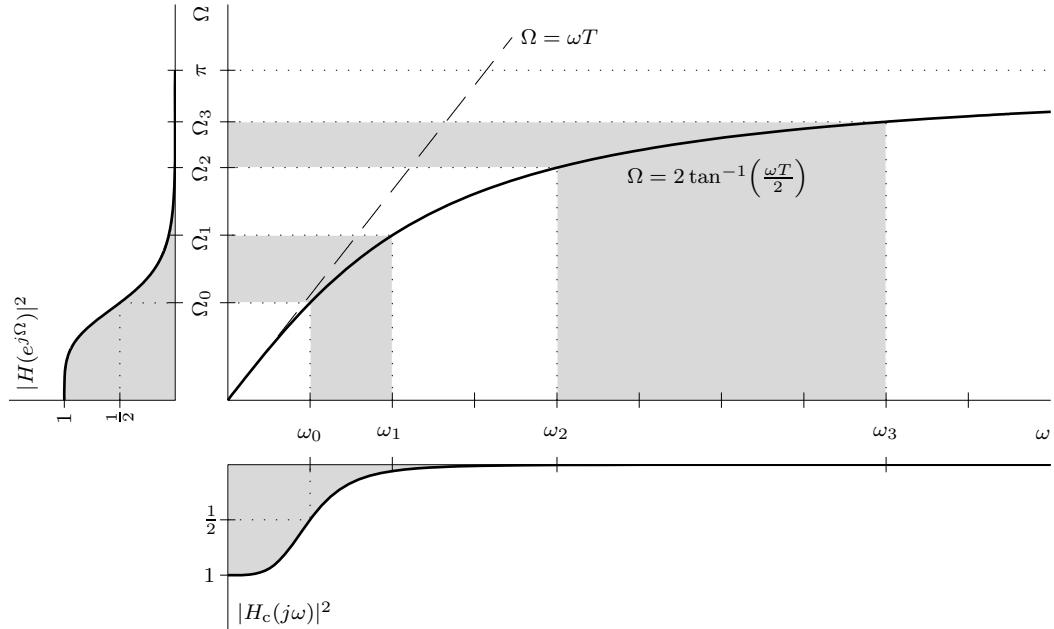


Figure 8.9: Analog-to-digital filter transformation using the bilinear transform.

Now, uniform sampling relates CT frequency ω to digital frequency Ω linearly as $\Omega = \omega T$ (dotted line in Fig. 8.9). The bilinear transform, however, maps the analog filter response at ω to $\Omega = 2 \tan^{-1}(\omega T / 2)$. For low frequencies, this mapping closely matches the linear relationship from sampling, and the filter behaves as hoped. As frequency increases, however, the bilinear transform squeezes an ever-increasing portion of the analog filter response into an ever-decreasing range of Ω . This compression causes premature response of the digital filter, particularly at high frequencies. For example, the digital filter's cutoff frequency Ω_0 occurs earlier than the expected $\omega_0 T$ since $\omega T \geq 2 \tan^{-1}(\omega T / 2)$. As Fig. 8.9 makes clear, warping distortion increases with frequency. Analog filter frequencies between ω_0 and ω_1 , for example, map to a wider interval of digital frequencies Ω than do the fourfold wider range of higher frequencies between ω_2 and ω_3 .

Because of the inherent frequency warping, the bilinear transform cannot be used to design ideal differentiators, where the magnitude response is a linear function of frequency. The linear frequency response in $H_c(j\omega)$ will be transformed into a digital filter with nonlinear magnitude response. For

such applications, the impulse invariance method is more appropriate since it produces a linear mapping from analog to digital frequency ($\Omega = \omega T$). The bilinear transform is better suited for filters where compression can be accepted, as in the case of filters approximating an ideal piecewise-constant magnitude response.

There are two common ways to combat frequency warping. The first is to reduce T (increase the sampling rate) so that the signal bandwidth is kept small enough that $\omega T \approx 2 \tan^{-1}(\omega T/2)$ over the desired frequency band. This step is easy to execute, but it requires a higher sampling rate (lower T) than necessary. The second approach, known as *prewarping*, solves the problem without unduly reducing T .

8.1.3 The Bilinear Transform with Prewarping

With prewarping, we start not with the desired $H_c(s)$ but with a prewarped version $H'_c(s)$. The idea is to prewarp the analog filter response in such a way as to exactly cancel the distortion caused by the bilinear transform. The idea is similar to the one used in prestressed concrete, in which a concrete beam is precompressed initially. The beam's built-in compression compensates for tension that occurs when the beam is loaded.

Usually, prewarping is done at certain critical frequencies rather than over the entire band. The final filter behavior exactly equals the desired behavior at these selected frequencies. Such an approach is suitable for the design of filters approximating ideal piecewise-constant magnitude responses, provided that the critical frequencies are properly chosen. If we require a digital filter to have particular response characteristics at critical frequencies $\omega_1, \omega_2, \dots, \omega_M$, then we must start with an analog filter $H'_c(s)$ with the same response characteristics at frequencies $\omega'_1, \omega'_2, \dots, \omega'_M$, where

$$\omega'_m = \frac{2}{T} \tan\left(\frac{\omega_m T}{2}\right), \quad m = 1, 2, \dots, M. \quad (8.18)$$

This results in the prewarped filter $H'_c(s)$. Applying the bilinear transform to this filter (Eq. (8.13)) yields a digital filter $H(z)$ with the desired response features at the correct frequencies $\Omega_m = \omega_m T$. This is because, according to Eq. (8.12), the critical frequencies ω'_m of the analog filter map to the digital frequencies

$$\Omega_m = 2 \tan^{-1}\left(\frac{\omega'_m T}{2}\right) = 2 \tan^{-1}\left(\frac{T}{2} \left[\frac{2}{T} \tan\left(\frac{\omega_m T}{2}\right) \right] \right) = \omega_m T.$$

In other words, the prewarping of Eq. (8.18) exactly cancels the warping of Eq. (8.12) that is inherent to the bilinear transform. We clarify these ideas with an example of a Butterworth lowpass filter.

▷ Example 8.3 (Bilinear Transform with Prewarping: Digital Butterworth LPF)

Using the bilinear transform with prewarping, design a digital filter to realize a Butterworth lowpass filter with unity gain at $\omega = 0$, $\alpha_p = 2$ dB of maximum passband attenuation over $0 \leq \omega \leq 8$, and $\alpha_s = 11$ dB of minimum stopband attenuation over $\omega \geq 15$. The maximum frequency to be processed is $\omega_{\max} = 30$ rad/s.

The highest frequency $\omega_{\max} = 30$ yields $T \leq \pi/30$. Let us use $T = \pi/35$. The Butterworth filter specifications for this design are $\omega_p = 8$, $\omega_s = 15$, $\alpha_p = 2$ dB ($\delta_p = 0.2057$), and $\alpha_s = 11$ dB ($\delta_s = 0.2818$). To begin, we prewarp the critical frequencies ω_p and ω_s according to Eq. (8.18), which yields

$$\omega'_p = \frac{2}{T} \tan\left(\frac{\omega_p T}{2}\right) = \frac{70}{\pi} \tan\left(\frac{4\pi}{35}\right) = 8.3625$$

and

$$\omega'_s = \frac{2}{T} \tan\left(\frac{\omega_s T}{2}\right) = \frac{70}{\pi} \tan\left(\frac{15\pi}{35}\right) = 17.7691.$$

Next, we use the techniques of Ch. 2 to design a Butterworth filter with critical frequencies ω'_p and ω'_s . Using Eq. (2.38), the minimum filter order K is computed as

$$K = \left\lceil \frac{\log [(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)]}{2 \log(\omega'_s/\omega'_p)} \right\rceil = \left\lceil \frac{\log [(10^{11/10} - 1) / (10^{2/10} - 1)]}{2 \log(17.7691/8.3625)} \right\rceil = \lceil 1.9811 \rceil = 2.$$

Using Eq. (2.39), the half-power frequency ω'_c must therefore satisfy

$$\frac{\omega'_p}{(10^{\alpha_p/10} - 1)^{1/2K}} = 9.5624 \leq \omega'_c \leq 9.6305 = \frac{\omega'_s}{(10^{\alpha_s/10} - 1)^{1/2K}}.$$

To ensure that both passband and stopband specifications are comfortably met, we choose a convenient middle value $\omega'_c = 9.5965$.

MATLAB simplifies the remaining computations of the design. First, we define a few basic parameters.

```
01 T = pi/35; omegacp = 9.5965; K = 2; k = [1:K];
```

Next, we use Eqs. (2.34) and (2.35) to determine the zeros, poles, and gain of our prewarped analog filter $H'_c(s)$.

```
02 Z = [] ; P = [1j*omegacp*exp(1j*pi*(2*k-1)/(2*K))] ;
03 bL = omegacp^K; aK = 1; L = length(Z); K = length(P);
```

We next apply the bilinear transform and expand the result of Eq. (8.15) to obtain the coefficients of $H(z) = B(z)/A(z)$.

```
04 B = bL/aK*prod(2/T-Z)/prod(2/T-P)*poly([(1+Z*T/2)./(1-Z*T/2), -ones(1,K-L)])
B = 0.1034 0.2067 0.1034
05 A = poly((1+P*T/2)./(1-P*T/2))
A = 1.0000 -0.9077 0.3212
```

Using these results, the desired digital filter has a transfer function

$$H(z) = \frac{0.1034(z^2 + 2z + 1)}{z^2 - 0.9077z + 0.3212}.$$

To verify that filter requirements are met, Fig. 8.10 plots the magnitude response versus $\Omega/T = \omega$.

```
06 Omega = linspace(0,pi,1001); omega = Omega/T;
07 H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
08 plot(omega,abs(H),'k');
```

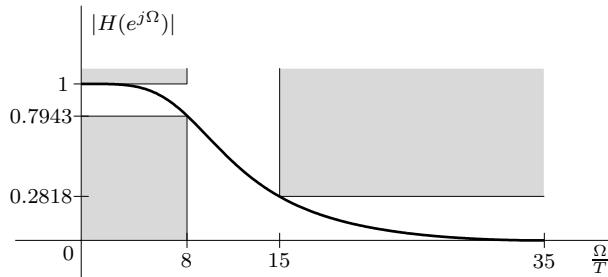


Figure 8.10: Magnitude response for $H(z) = \frac{0.1034(z^2 + 2z + 1)}{z^2 - 0.9077z + 0.3212}$.

A Simplified Procedure

In the bilinear transform with prewarping, all the critical frequencies ω_m are transformed (prewarped) by the equation $\omega'_m = \frac{2}{T} \tan\left(\frac{\omega_m T}{2}\right)$ (Eq. (8.18)). These prewarped frequencies are used to find the prewarped CT filter transfer function $H'_c(s)$. Finally, we substitute $s = \frac{2}{T} \left(\frac{z-1}{z+1} \right)$ (Eq. (8.9)) in $H'_c(s)$ to obtain the desired digital filter transfer function $H(z)$.

This procedure can be simplified by observing that the scaling factor $\frac{2}{T}$ is irrelevant to the combination of Eqs. (8.18) and (8.9). Instead of using these two equations, we can ignore the scaling factor $\frac{2}{T}$ and use the simplified equations

$$\omega'_m = \tan\left(\frac{\omega_m T}{2}\right) \quad \text{and} \quad s = \frac{z-1}{z+1}. \quad (8.19)$$

This simplification works because the factor $2/T$ in Eq. (8.18) is a frequency scaling factor, and ignoring it, as in Eq. (8.19), results in the pretransformed filter that is frequency scaled by a factor $2/T$. This scaling is undone by using $s = \frac{z-1}{z+1}$ instead of Eq. (8.9). Using these simplifications, the resulting digital filter transfer function is

$$H(z) = \left(\frac{b_L \prod_{l=1}^L (1 - z_l)}{a_K \prod_{k=1}^K (1 - p_k)} \right) \frac{\prod_{l=1}^L \left(z - \frac{1+z_l}{1-z_l} \right)}{\prod_{k=1}^K \left(z - \frac{1+p_k}{1-p_k} \right)} (z + 1)^{K-L}, \quad (8.20)$$

where z_l , p_k , and b_L/a_K are the zeros, poles, and gain factor of the prewarped filter $H'_c(s)$.

▷ Example 8.4 (Simplified Procedure for the Bilinear Transform with Prewarping)

Repeat Ex. 8.3 using the simplified procedure for the bilinear transform with prewarping.

To begin, we prewarp the critical frequencies ω_p and ω_s according to Eq. (8.19), which yields

$$\omega'_p = \tan\left(\frac{\omega_p T}{2}\right) = \tan\left(\frac{4\pi}{35}\right) = 0.3753 \quad \text{and} \quad \omega'_s = \tan\left(\frac{\omega_s T}{2}\right) = \tan\left(\frac{15\pi}{70}\right) = 0.7975.$$

The ratio of ω'_s/ω'_p is unchanged, so calculation of the minimum filter order remains $K = 2$. Again using Eq. (2.39), the half-power frequency ω'_c must satisfy

$$\frac{\omega'_p}{(10^{\alpha_p/10} - 1)^{1/2K}} = 0.4292 \leq \omega'_c \leq 0.4322 = \frac{\omega'_s}{(10^{\alpha_s/10} - 1)^{1/2K}}.$$

From this range, we select the middle value $\omega'_c = 0.4307$.

Given the low order and simplified transform, we complete the design using tables and hand calculations rather than MATLAB. From Table 2.2, a $K = 2$ normalized Butterworth filter has transfer function $1/(s^2 + \sqrt{2}s + 1)$. To shift the 3-dB cutoff from 1 to $\omega'_c = 0.4307$, we apply the lowpass-to-lowpass transformation $s \rightarrow s/0.4307$ (Eq. (2.25)) to obtain

$$H'_c(s) = \frac{1}{\left(\frac{s}{0.4307}\right)^2 + \sqrt{2}\left(\frac{s}{0.4307}\right) + 1} = \frac{0.1855}{s^2 + 0.6091s + 0.1855}.$$

Finally, we obtain $H(z)$ from $H'_c(s)$ using the simplified bilinear transform $s = (z-1)/(z+1)$ (Eq. (8.19)), which yields

$$H(z) = H'_c(s)|_{s=\frac{z-1}{z+1}} = \frac{0.1855}{s^2 + 0.6091s + 0.1855} \Big|_{s=\frac{z-1}{z+1}} = \frac{0.1034(z+1)^2}{z^2 - 0.9077z + 0.3212}.$$

This result is identical to that obtained in Ex. 8.3.

Example 8.4 ◀

Filter Design Using Discrete-Time Specifications

So far we have considered realizing an analog filter $H_c(s)$ using a digital system and filter $H(z)$. What if we want to design $H(z)$ not from continuous-time but rather discrete-time specifications? For example, our specifications may require a digital Butterworth lowpass filter with half-power cutoff frequency $\Omega_c = \pi/2$. Recognizing that digital frequencies are related to analog frequencies as $\Omega = \omega T$, we can once again design our filter $H(z)$ using the bilinear transform with prewarping. We only need to slightly modify the simplified procedure of Eq. (8.19) to reflect digital, rather than analog, specifications:

$$\omega'_m = \tan\left(\frac{\Omega_m}{2}\right) \quad \text{and} \quad s = \frac{z-1}{z+1}. \quad (8.21)$$

In the prewarping step, we apply Eq. (8.21) to the DT specifications to find the CT specifications. Next, we use any of the well-known analog techniques to find a prototype filter $H'_c(s)$ that meets these specifications. In the last step, we use the simplified bilinear transform $s = \frac{z-1}{z+1}$ to obtain the desired digital filter $H(z)$.

Notice that since filter specifications are in terms of digital frequencies Ω , there is no need to specify a value for the sampling interval T during the design of $H(z)$. In other words, the critical frequencies Ω_m of a digital filter are not affected by how fast (or slow) the system processes samples. Such is the nature of digital frequency Ω .

▷ Example 8.5 (Bilinear Transform Using Discrete-Time Specifications)

Using the bilinear transform with prewarping, design a digital Butterworth lowpass filter with unity gain at $\Omega = 0$, $\alpha_p = 2$ dB of maximum passband attenuation over $0 \leq \Omega \leq \frac{\pi}{2}$, and $\alpha_s = 11$ dB of minimum stopband attenuation over $\frac{3\pi}{4} \leq \Omega \leq \pi$.

The Butterworth filter specifications for this design are $\Omega_p = \frac{\pi}{2}$, $\Omega_s = \frac{3\pi}{4}$, $\alpha_p = 2$ dB ($\delta_p = 0.2057$), and $\alpha_s = 11$ dB ($\delta_s = 0.2818$). To begin, we use Eq. (8.21) to map our digital critical frequencies Ω_p and Ω_s to their corresponding prewarped analog critical frequencies

$$\omega'_p = \tan\left(\frac{\Omega_p}{2}\right) = \tan\left(\frac{\pi}{4}\right) = 1.0000 \quad \text{and} \quad \omega'_s = \tan\left(\frac{\Omega_s}{2}\right) = \tan\left(\frac{3\pi}{8}\right) = 2.4142.$$

Next, we use the techniques of Ch. 2 to design a Butterworth filter with critical frequencies ω'_p and ω'_s . Using Eq. (2.38), the minimum filter order K is computed as

$$K = \left\lceil \frac{\log[(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)]}{2 \log(\omega'_s/\omega'_p)} \right\rceil = \left\lceil \frac{\log[(10^{11/10} - 1) / (10^{2/10} - 1)]}{2 \log(2.4142)} \right\rceil = \lceil 1.6942 \rceil = 2.$$

Using Eq. (2.39), the half-power frequency ω'_c must therefore satisfy

$$\frac{\omega_p}{(10^{\alpha_p/10} - 1)^{1/2K}} = 1.1435 \leq \omega'_c \leq 1.3085 = \frac{\omega_s}{(10^{\alpha_s/10} - 1)^{1/2K}}.$$

Once again, we choose the middle value $\omega'_c = 1.2260$.

From Table 2.2, we find the normalized Butterworth polynomial for $K = 2$ and replace s with s/ω'_c to find $H'_c(s)$ as

$$H'_c(s) = \frac{1}{\left(\frac{s}{1.2260}\right)^2 + \sqrt{2}\left(\frac{s}{1.2260}\right) + 1} = \frac{1.5030}{s^2 + 1.7338s + 1.5030}.$$

Finally, we obtain $H(z)$ from $H'_c(s)$ using the substitution $s = (z-1)/(z+1)$ (Eq. (8.21)), which yields

$$H(z) = H'_c(s)|_{s=\frac{z-1}{z+1}} = \frac{1.5030}{s^2 + 1.7338s + 1.5030} \Big|_{s=\frac{z-1}{z+1}} = \frac{0.3548(z+1)^2}{z^2 + 0.2375z + 0.1816}.$$

To verify that specifications are met, Fig. 8.11 plots the filter's magnitude response $|H(e^{j\Omega})|$. This magnitude response is fixed, regardless of the operational sampling rate $F_s = 1/T$. In this case, the sampling interval T is completely irrelevant to the design of $H(z)$.

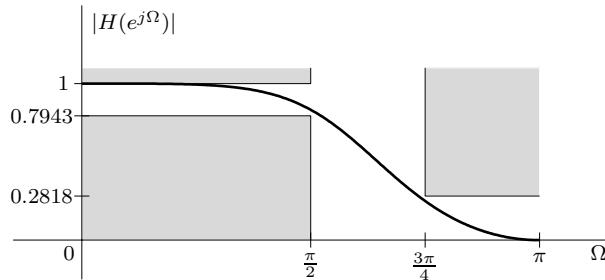


Figure 8.11: Magnitude response for $H(z) = \frac{0.3548(z+1)^2}{z^2+0.2375z+0.1816}$.

Example 8.5 □

▷ Drill 8.2 (Bilinear Transform Using Discrete-Time Specifications)

Using the bilinear transform with prewarping, design a second-order digital Chebyshev lowpass filter with a maximum passband ripple of $\alpha_p = 1$ dB for $0 \leq \Omega \leq \pi/3$.

□

8.1.4 Highpass, Bandpass, and Bandstop Filters

The same bilinear transform procedures to design digital lowpass filters also work to design digital highpass, bandpass, and bandstop filters. The main difficulty that arises is the sheer number of transformations that need to be taken. First, critical frequencies are prewarped, yielding critical frequencies of the analog highpass, bandpass, or bandstop filter. These critical frequencies are then related to the critical frequencies of an appropriate lowpass filter, which is then designed to meet these specifications. The lowpass prototype is transformed to an analog highpass, bandpass, or bandstop filter, which is then further transformed into a digital filter.

So many transformations can be a daunting task, particularly for high-order filters. The process can be streamlined, however, if we use the simplified form of the bilinear transform, combine certain transformations, use normalized analog lowpass prototype filters, and operate on the factored form of the filter transfer function. We next summarize these ideas and provide several design examples.

Digital Highpass IIR Filter Design

As given by Eq. (2.27), a normalized ($\omega_0 = 1$) analog lowpass prototype filter is converted to a highpass filter using the lowpass-to-highpass transformation $s \rightarrow \omega'_1/s$, where ω'_1 is the (prewarped) highpass filter frequency that corresponds to $\omega_0 = 1$ of the lowpass prototype. Combined with the simplified bilinear transform $s = (z - 1)/(z + 1)$, a normalized analog lowpass prototype $H_c(s)$ is converted to a digital highpass filter $H(s)$ by the substitution

$$s = \omega'_1 \frac{z + 1}{z - 1}. \quad (8.22)$$

Applied to the factored form of the lowpass prototype transfer function $H_c(s)$ (see Eq. (8.14)), the desired highpass digital filter is thus

$$H(z) = \left(\frac{b_L \prod_{l=1}^L (\omega'_1 - z_l)}{a_K \prod_{k=1}^K (\omega'_1 - p_k)} \right) \frac{\prod_{l=1}^L \left(z - \frac{z_l + \omega'_1}{z_l - \omega'_1} \right)}{\prod_{k=1}^K \left(z - \frac{p_k + \omega'_1}{p_k - \omega'_1} \right)} (z - 1)^{K-L}. \quad (8.23)$$

As Eq. (8.23) shows, the lowpass prototype's zeros z_l and poles p_k map to the zeros $\frac{z_l + \omega'_1}{z_l - \omega'_1}$ and poles $\frac{p_k + \omega'_1}{p_k - \omega'_1}$ of the digital highpass filter. Further, the $K - L$ zeros at ∞ map to zeros at $z = 1$. These mappings are relatively easy to compute, by calculator or computer, regardless of the filter order.

▷ Example 8.6 (Digital Highpass IIR Filter Design)

Using the bilinear transform with prewarping, design a digital filter to realize a Chebyshev highpass filter with $\alpha_p = 1$ dB of maximum passband ripple ($\delta_p = 0.1087$) for $\omega \geq 20$ rad/s and $\alpha_s = 20$ dB of minimum stopband attenuation ($\delta_s = 0.1$) over $0 \leq \omega \leq 15$. The maximum frequency to be processed is $\omega_{\max} = 80$ rad/s.

In order to avoid aliasing, Eq. (8.17) requires that the sampling interval T be chosen less than $\pi/\omega_{\max} = \pi/80$. Let us choose $T = \pi/100$. Due to the complexity of this problem, we shall use MATLAB for most of the computations.

To begin, the critical frequencies $\omega_s = 15$ and $\omega_p = 20$ are prewarped using Eq. (8.19).

```
01 T = pi/100; omegas = 15; omegap = 20; alphap = 1; alphas = 20;
02 omegasp = tan(omegas*T/2), omegapp = tan(omegap*T/2)
omegasp = 0.2401
omegapp = 0.3249
```

Thus, the analog highpass filter's prewarped critical frequencies are $\omega'_s = 0.2401$ and $\omega'_p = 0.3249$.

To determine the order K of the lowpass prototype, we need to determine how the analog highpass filter's critical frequencies map to the lowpass filter's stopband-to-passband ratio $\omega_{lp,s}/\omega_{lp,p}$. Using Eq. (2.27), this ratio is

$$\frac{\omega_{lp,s}}{\omega_{lp,p}} = \frac{\frac{\omega_0 \omega'_s}{-\omega'_s}}{\frac{\omega_0 \omega'_p}{-\omega'_p}} = \frac{\omega'_s}{\omega'_p}.$$

The order of the lowpass prototype is computed according to (see Eq. (2.46))

$$K = \left\lceil \frac{\cosh^{-1} \sqrt{(10^{\alpha_s/10} - 1) / (10^{\alpha_p/10} - 1)}}{\cosh^{-1}(\omega_{lp,s}/\omega_{lp,p})} \right\rceil.$$

```
03 K = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/acosh(omegapp/omegasp))
K = 5
```

Normalizing the passband frequency to 1, we now design our prototype Chebyshev lowpass filter using Eqs. (2.45), (2.47), and (2.48). Since K is odd, the dc gain of the prototype is 1 rather than $1/\sqrt{1 + \epsilon^2}$.

```
04 k = [1:K]; epsilon = sqrt(10^(alphap/10)-1); Z = [];
05 P = -sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
06     1j*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
07 bL = 1*prod(-P); aK = 1; L = length(Z); K = length(P);
```

With the prototype Chebyshev lowpass filter in place, we next use Eq. (8.23) to determine the final digital highpass filter. To do this, we first need to determine an appropriate frequency scaling

parameter ω'_1 . Setting $\omega'_1 = \omega'_p = 0.3249$ causes the resulting filter to exactly meet passband requirements and exceed stopband specifications. Most design tools choose this simple and convenient value. To realize a design with buffer zones on both the passband and stopband sides, a little more work is required.

To achieve the other extreme, we first use Eq. (2.46) to determine the actual stopband frequency $\omega_{lp,s}$ of the lowpass prototype.

```
08 omegalps = cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K);
```

From Eq. (2.27), we see that $\omega'_1 = \omega'_s \omega_{lp,s}$ causes the lowpass stopband frequency $\omega_{lp,s}$ to map exactly to the (prewarped) analog highpass filter's stopband frequency ω'_s . This results in a filter that exactly meets stopband requirements and exceeds passband specifications. Instead of using either extreme, we instead choose a middle value.

```
09 omega1p = mean([omegapp,omegapsp*omegalps])
omega1p = 0.3162
```

In this case, the balanced value $\omega'_1 = 0.3162$ is quite close to the extreme (but simple) choice $\omega'_1 = \omega'_p = 0.3249$. As filter order increases, the differences tend to become progressively smaller, and there is less justification to go through the extra work to determine the balanced value.

We next apply Eq. (8.23) and expand the result to obtain the coefficients of $H(z) = B(z)/A(z)$.

```
10 B = bL/aK*prod(omega1p-Z)/prod(omega1p-P)*...
11 poly([(Z+omega1p)./(Z-omega1p),ones(1,K-L)])
B = 0.2615 -1.3073 2.6145 -2.6145 1.3073 -0.2615
12 A = poly([(P+omega1p)./(P-omega1p)])
A = 1.0000 -2.4798 2.8611 -1.6181 0.4299 0.0224
```

From these results, the transfer function of the desired digital Chebyshev highpass filter is thus

$$H(z) = \frac{0.2615z^5 - 1.3073z^4 + 2.6145z^3 - 2.6145z^2 + 1.3073z - 0.2615}{z^5 - 2.4798z^4 + 2.8611z^3 - 1.6181z^2 + 0.4299z + 0.0224}.$$

To verify that filter requirements are met, Fig. 8.12 plots the magnitude response versus $\Omega/T = \omega$.

```
13 Omega = linspace(0,pi,1001);
14 H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
15 plot(Omega/T,abs(H),'k');
```

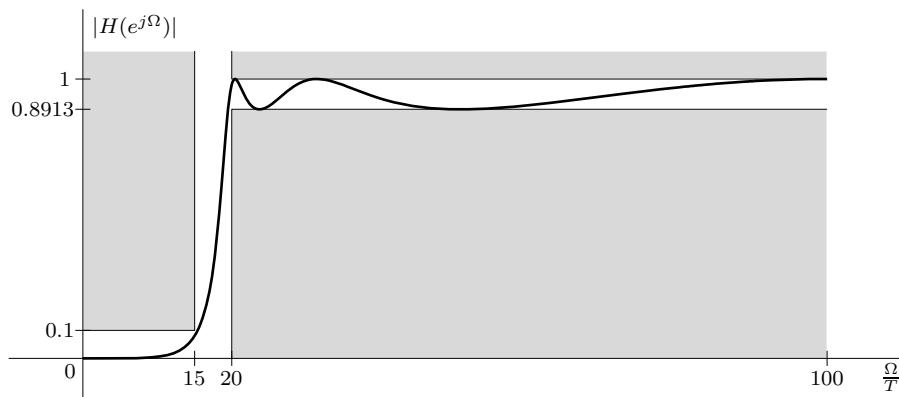


Figure 8.12: Magnitude response for $H(z) = \frac{0.2615z^5 - 1.3073z^4 + 2.6145z^3 - 2.6145z^2 + 1.3073z - 0.2615}{z^5 - 2.4798z^4 + 2.8611z^3 - 1.6181z^2 + 0.4299z + 0.0224}$.

Digital Bandpass IIR Filter Design

A normalized ($\omega_0 = 1$) analog lowpass prototype filter is converted to a bandpass filter using the Eq. (2.29) lowpass-to-bandpass transformation

$$s \rightarrow \frac{s^2 + \omega'_1 \omega'_2}{s(\omega'_2 - \omega'_1)},$$

where ω'_1 and ω'_2 are the (prewarped) bandpass filter frequencies that correspond to $\omega_0 = 1$ of the lowpass prototype. Combined with the simplified bilinear transform $s = (z-1)/(z+1)$, a normalized analog lowpass prototype $H_c(s)$ is converted to a digital bandpass filter $H(z)$ by the substitution

$$s = \frac{z^2 + 2c_1 z + 1}{c_2(z^2 - 1)}, \quad \text{where } c_1 = \frac{\omega'_1 \omega'_2 - 1}{\omega'_1 \omega'_2 + 1} \quad \text{and} \quad c_2 = \frac{\omega'_2 - \omega'_1}{\omega'_1 \omega'_2 + 1}. \quad (8.24)$$

Applied to the factored form of the lowpass prototype transfer function $H_c(s)$ (see Eq. (8.14)), the desired bandpass digital filter is thus

$$H(z) = \left(\frac{b_L \prod_{l=1}^L (\frac{1}{c_2} - z_l)}{a_K \prod_{k=1}^K (\frac{1}{c_2} - p_k)} \right) \frac{\prod_{l=1}^L \left(z^2 + \left(\frac{2c_1}{1-c_2 z_l} \right) z + \frac{1+c_2 z_l}{1-c_2 z_l} \right)}{\prod_{k=1}^K \left(z^2 + \left(\frac{2c_1}{1-c_2 p_k} \right) z + \frac{1+c_2 p_k}{1-c_2 p_k} \right)} (z^2 - 1)^{K-L}, \quad (8.25)$$

where

$$c_1 = \frac{\omega'_1 \omega'_2 - 1}{\omega'_1 \omega'_2 + 1} \quad \text{and} \quad c_2 = \frac{\omega'_2 - \omega'_1}{\omega'_1 \omega'_2 + 1}.$$

As Eq. (8.25) shows, each of the lowpass prototype's zeros z_l maps to a pair of zeros determined by the roots of $z^2 + \left(\frac{2c_1}{1-c_2 z_l} \right) z + \frac{1+c_2 z_l}{1-c_2 z_l}$. Similarly, each of the poles p_k maps to a pair of poles determined by the roots of $z^2 + \left(\frac{2c_1}{1-c_2 p_k} \right) z + \frac{1+c_2 p_k}{1-c_2 p_k}$. Further, each of the $K - L$ zeros at ∞ maps to a pair of zeros, one at $z = 1$ and one at $z = -1$.

▷ Example 8.7 (Digital Bandpass IIR Filter Design)

Design a digital filter to realize a Butterworth bandpass filter with $\alpha_p = 2.1$ dB of maximum passband ripple ($\delta_p = 0.2148$) for $1000 \leq \omega \leq 2000$ and $\alpha_s = 20$ dB of minimum stopband attenuation ($\delta_s = 0.1$) over both $0 \leq \omega \leq 450$ and $\omega \geq 4000$. Take $T = \pi/10000$, and let the digital design exactly meet passband specifications.

The passband and stopband critical frequencies are $\omega_{s_1} = 450$, $\omega_{p_1} = 1000$, $\omega_{p_2} = 2000$, and $\omega_{s_2} = 4000$. The corresponding prewarped critical frequencies are determined using Eq. (8.19).

```

01 T = pi/10000; omegas1 = 450; omegap1 = 1000; omegap2 = 2000; omegas2 = 4000;
02 omegas1p = tan(omegas1*T/2), omegap1p = tan(omegap1*T/2),
omegas1p = 0.0708
omegap1p = 0.1584
03 omegap2p = tan(omegap2*T/2), omegas2p = tan(omegas2*T/2),
omegap2p = 0.3249
omegas2p = 0.7265

```

Next, we find a lowpass prototype to meet these prewarped analog bandpass specifications. To determine the order K of the lowpass prototype, we need to determine the stopband-to-passband ratio ω_s/ω_p . To this end, we normalize $\omega_p = 1$ and then use Eq. (2.29) to solve for ω_s . Since there are two transition bands in the bandpass filter, two candidate values of ω_s are computed.

```

04 omegas = abs([(omegas1p^2-omegap1p*omegap2p)/(omegas1p*(omegap2p-omegap1p)),...
05 (omegas2p^2-omegap1p*omegap2p)/(omegas2p*(omegap2p-omegap1p))])
omegas = 3.9393 3.9374

```

Selecting the smaller (more restrictive) value, the order is computed using Eq. (2.38).

```
06 omegap = 1; omegas = min(omegas); alphap = 2.1; alphas = 20;
07 K = ceil(log((10^(alphas/10)-1)/(10^(alphap/10)-1))/(2*log(omegas/omegap)))
K = 2
```

To meet passband specifications exactly, we compute the 3-dB cutoff frequency ω_c of the prototype filter using the lower bound of Eq. (2.39).

```
08 omegac = omegap/((10^(alphap/10)-1)^(1/(2*K)))
omegac = 1.1261
```

Using Eqs. (2.34) and (2.35), we compute the zeros, poles, and gain of the lowpass prototype.

```
09 k = [1:K]; Z = []; P = 1j*omegac*exp(1j*pi*(2*k-1)/(2*K));
10 bL = omegac^K; aK = 1; L = length(Z); K = length(P);
```

Using Eq. (8.25), each pole of the prototype transforms into a pair of poles in the digital BPF. The zeros are at $z = \pm 1$ since the Butterworth prototype is an all-pole filter ($L = 0$).

```
11 c1 = (omegap1p*omegap2p-1)/(omegap1p*omegap2p+1);
12 c2 = (omegap2p-omegap1p)/(omegap1p*omegap2p+1);
13 Zdig = [ones(1,K-L), -ones(1,K-L)];
14 for i = 1:K, Pdig(i,:) = roots([1 2*c1./(1-c2*P(i)) (1+c2*P(i))./(1-c2*P(i))]); end
15 B = bL*aK*prod(1/c2-Z)/prod(1/c2-P)*poly(Zdig(:)'), A = poly(Pdig(:)');
B = 0.0248 0 -0.0495 0 0.0248
A = 1.0000 -3.1646 4.0431 -2.4558 0.6071
```

Thus, the desired digital Butterworth bandpass filter has transfer function

$$H(z) = \frac{B(z)}{A(z)} = \frac{0.0248(z^4 - 2z^2 + 1)}{z^4 - 3.1646z^3 + 4.0431z^2 - 2.4558z + 0.6071}.$$

The magnitude response plot of Fig. 8.13 verifies that $H(z)$ satisfies design requirements.

```
16 Omega = linspace(0,pi,1001); H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
17 plot(Omega/T,abs(H),'k-');
```

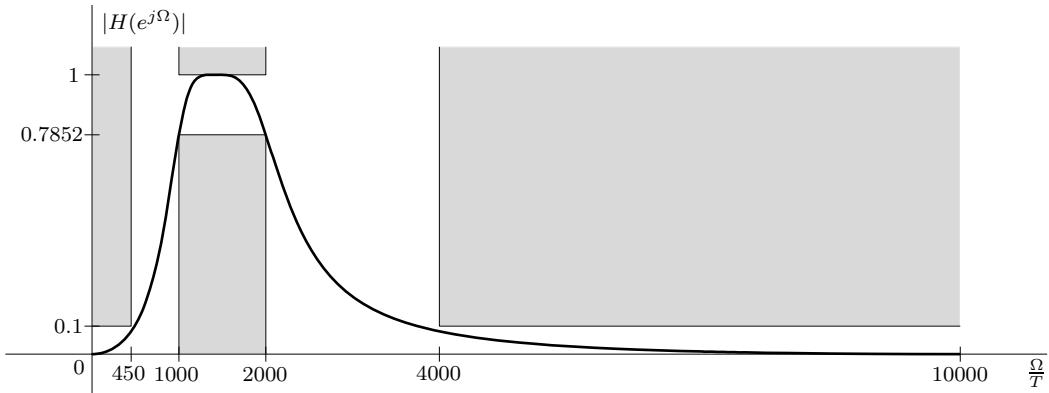


Figure 8.13: Magnitude response for $H(z) = \frac{0.0248(z^4 - 2z^2 + 1)}{z^4 - 3.1646z^3 + 4.0431z^2 - 2.4558z + 0.6071}$.

Digital Bandstop IIR Filter Design

The bandstop case is quite similar to the bandpass case. Using Eq. (2.31), a normalized ($\omega_0 = 1$) analog lowpass prototype filter is converted to a bandstop filter using the transformation

$$s \rightarrow \frac{s(\omega'_2 - \omega'_1)}{s^2 + \omega'_1\omega'_2},$$

where ω'_1 and ω'_2 are the (prewarped) bandstop filter frequencies that correspond to $\omega_0 = 1$ of the lowpass prototype. Combined with $s = (z-1)/(z+1)$, a normalized analog lowpass prototype $H_c(s)$ is converted to a digital bandstop filter $H(z)$ by the substitution

$$s = \frac{c_2(z^2 - 1)}{z^2 + 2c_1z + 1}, \quad \text{where } c_1 = \frac{\omega'_1\omega'_2 - 1}{\omega'_1\omega'_2 + 1} \quad \text{and} \quad c_2 = \frac{\omega'_2 - \omega'_1}{\omega'_1\omega'_2 + 1}. \quad (8.26)$$

Equation (8.26) is just the reciprocal of the lowpass-to-bandpass transformation of Eq. (8.24). Applied to the factored form of the lowpass prototype transfer function $H_c(s)$, the desired bandstop digital filter is thus

$$H(z) = \left(\frac{b_L \prod_{l=1}^L (c_2 - z_l)}{a_K \prod_{k=1}^K (c_2 - p_k)} \right) \frac{\prod_{l=1}^L \left(z^2 + \left(\frac{2c_1 z_l}{z_l - c_2} \right) z + \frac{z_l + c_2}{z_l - c_2} \right)}{\prod_{k=1}^K \left(z^2 + \left(\frac{2c_1 p_k}{p_k - c_2} \right) z + \frac{p_k + c_2}{p_k - c_2} \right)} (z^2 + 2c_1z + 1)^{K-L}, \quad (8.27)$$

where

$$c_1 = \frac{\omega'_1\omega'_2 - 1}{\omega'_1\omega'_2 + 1} \quad \text{and} \quad c_2 = \frac{\omega'_2 - \omega'_1}{\omega'_1\omega'_2 + 1}.$$

Equation (8.27) creates a bandstop digital filter that is double the order of the lowpass prototype, with two poles (or zeros) for every pole (or zero) of the lowpass prototype.

▷ Example 8.8 (Digital Bandstop IIR Filter Design)

Plot the magnitude response of a 10th-order digital Chebyshev bandstop filter that has $\alpha_p = 1$ dB of maximum passband ripple ($\delta_p = 0.1087$) for $0 \leq \Omega \leq \pi/4$ and $\pi/2 \leq \Omega \leq \pi$. Determine the critical stopband frequencies Ω_{s1} and Ω_{s2} that correspond to $\alpha_s = 20$ dB of stopband attenuation ($\delta_s = 0.1$).

A 10th-order digital Chebyshev bandstop filter requires a 5th-order lowpass prototype, which we can design with normalized passband frequency $\omega_p = 1$ in an identical fashion to Ex. 8.6.

```
01 alphap = 1; K = 5; k = 1:K; epsilon = sqrt(10^(alphap/10)-1); Z = [];
02 P = -sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
03 1j*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
04 bL = 1*prod(-P); aK = 1; L = length(Z); K = length(P);
```

The critical passband frequencies of the digital bandstop filter are $\Omega_{p1} = \pi/4$ and $\Omega_{p2} = \pi/2$. These digital specifications are prewarped according to Eq. (8.21).

```
05 Omegap1 = pi/4; omegap1p = tan(Omegap1/2); Omegap2 = pi/2; omegap2p = tan(Omegap2/2);
```

Next, we use Eq. (8.27) to determine the poles, zeros, gain, and transfer function coefficients of the digital filter.

```
06 c1 = (omegap1p*omegap2p-1)/(omegap1p*omegap2p+1);
07 c2 = (omegap2p-omegap1p)/(omegap1p*omegap2p+1);
08 Zdig = repmat(roots([1 2*c1 1]),K-L,1);
09 for i = 1:K, Pdig(i,:) = roots([1 2*c1*P(i)/(P(i)-c2) (P(i)+c2)/(P(i)-c2)]); end
10 B = bL/aK*prod(c2-Z)/prod(c2-P)*poly(Zdig(:)');
A = poly(Pdig(:)');
```

To determine the stopband frequencies Ω_{s_1} and Ω_{s_2} that correspond to $\alpha_s = 20$ dB of stopband attenuation, we first solve Eq. (2.46) for the stopband frequency of the prototype, that is,

$$\omega_s = \cosh\left(\frac{1}{K} \cosh^{-1}\left[\left(10^{\alpha_s/10} - 1\right) / \left(10^{\alpha_p/10} - 1\right)\right]^{1/2}\right).$$

From the lowpass-to-bandstop transformation of Eq. (2.31), ω_s maps to two stopband frequencies of the analog bandstop filter, which are determined by solving the quadratic

$$\omega_s \omega^2 + (\omega_{p_2} - \omega_{p_1})\omega - \omega_s \omega_{p_1} \omega_{p_2} = 0.$$

The bilinear transform then converts these values according to $\Omega = 2 \tan^{-1}(\omega)$.

```
11 alphas = 20; omegas = cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K);
12 Omegas = 2*atan(abs(roots([omegas omegap2p-omegap1p -omegas*omegap1p*omegap2p])));
Omegas = 1.4784 0.8529
```

Thus, the critical stopband frequencies are

$$\Omega_{s_1} = 0.8529 = 0.2715\pi \quad \text{and} \quad \Omega_{s_2} = 1.4784 = 0.4706\pi.$$

The magnitude response plot of Fig. 8.14 confirms these calculations as well as the overall behavior of the digital bandstop filter.

```
13 Omega = linspace(0,pi,1001);
14 H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
15 plot(Omega,abs(H),'k-');
```

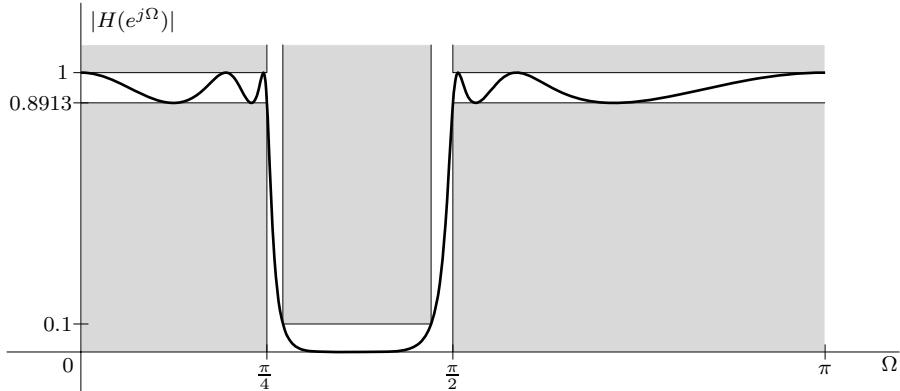


Figure 8.14: Magnitude response for 10th-order digital Chebyshev bandstop filter.

Example 8.8 ◁

▷ Drill 8.3 (Digital Bandstop IIR Filter Design)

Plot the magnitude response of a 12th-order digital inverse Chebyshev bandstop filter that has $\alpha_p = 0.5$ dB maximum passband ripple ($\delta_p = 0.0559$) for $0 \leq \Omega \leq \pi/3$ and $2\pi/3 \leq \Omega \leq \pi$. Set the minimum stopband attenuation to $\alpha_s = 40$ dB ($\delta_s = 0.01$), and determine the critical stopband frequencies Ω_{s_1} and Ω_{s_2} .

◁

8.1.5 Realization of IIR Filters

Section 7.5 presents several methods to realize constant-coefficient linear difference equations. For low-order designs implemented on floating-point processors, any of the direct realizations are likely to work, although the canonical TDFII realization is probably the most popular. For high-order designs or implementations on fixed-point processors, however, direct realizations are often unsatisfactory. Better performance is obtained using parallel or cascade structures. Next, we present a relatively simple procedure to generate a cascade realization that works well in most circumstances. More exotic methods are found in the literature.

Cascade of Second-Order Sections

Cascade realizations are usually considered the best general-purpose filter realization. Here, we present one strategy to generate a realization based on the cascade of second-order sections. We shall assume the more common case of real systems, where complex poles and zeros occur in complex-conjugate pairs. The idea is to pair poles with nearby zeros, thereby encouraging the magnitude response of each pole-zero pair to be as flat as possible [5,7].

1. Express $H(z)$ in factored form to determine system poles and zeros.
2. Pair conjugate roots and expand to produce second-order terms with real coefficients. Real roots, if any, can also be combined to form second-order terms.
3. Sort conjugate pole terms followed by real pole terms in decreasing order of pole magnitude.
4. Pair the complex poles nearest the unit circle with the nearest complex zeros to form a second-order section. Remove these poles and zeros from the list.
5. Repeat step 4 for each remaining pair of poles and zeros.
6. Retain or reverse section order. Both choices have advantages, although the reverse order, which orders poles closest to the unit circle last, is more common.
7. Realize each second-order section with a standard structure, such as TDFII. Overall filter gain is often applied to the first or last section or distributed across all sections.

We next demonstrate the procedure and the reasoning behind it with an example.

▷ Example 8.9 (Filter Realization Using a Cascade of Second-Order Sections)

Realize the 4th-order digital Butterworth bandpass filter of Ex. 8.7 using a cascade of second-order sections.

A cascade of two second-order stages is required to realize this 4th-order system. Factoring $H(z)$ from Ex. 8.7, the four system zeros are

$$z_1 = 1, z_2 = 1, z_3 = -1, z_4 = -1,$$

the four system poles are

$$p_1 = 0.8659 + j0.2876, p_2 = 0.8659 - j0.2876, p_3 = 0.7164 + j0.4648, p_4 = 0.7164 - j0.4648,$$

and the overall system gain is 0.0248. Figure 8.15 shows the pole-zero plot for this Butterworth bandpass filter.

Matching p_1 with its conjugate p_2 produces the quadratic $z^2 - 1.7318z^2 + 0.8325$, and matching p_3 with its conjugate p_4 produces $z^2 - 1.4328z^2 + 0.7293$. Grouping z_1 and z_2 produces $z^2 + 2z + 1$, while the remaining zeros z_3 and z_4 produce $z^2 - 2z + 1$.

Since p_1 and p_2 are closest to the unit circle, these poles are paired with the closest zeros z_1 and z_2 to form a second-order transfer function $(z^2 + 2z + 1)/(z^2 - 1.7318z^2 + 0.8325)$. The remaining poles p_3

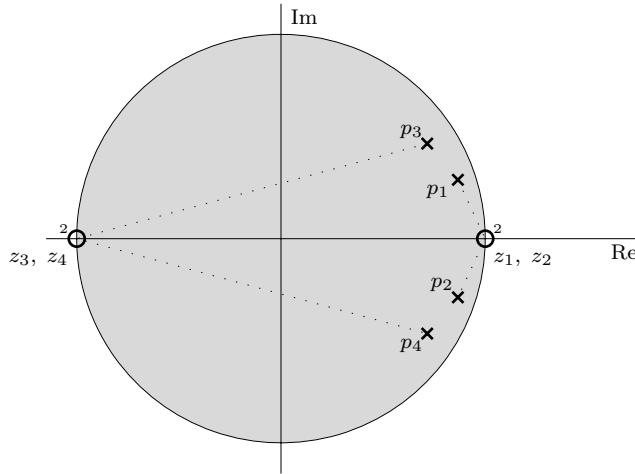


Figure 8.15: Pole-zero plot for 4th-order digital Butterworth BPF.

and p_4 are paired with the remaining zeros z_3 and z_4 to generate $(z^2 - 2z + 1)/(z^2 - 1.4328z^2 + 0.7293)$. These pole-zero pairings are shown in Fig. 8.15 using dotted lines.

In the final realization, we reverse the order of these second-order sections and combine the overall filter gain 0.0248 with the first section. The transfer functions of the first and second stages are thus

$$H_1(z) = \frac{0.0248(z^2 - 2z + 1)}{z^2 - 1.4328z^2 + 0.7293} \quad \text{and} \quad H_2(z) = \frac{z^2 + 2z + 1}{z^2 - 1.7318z^2 + 0.8325}.$$

Figure 8.16 shows the cascade realization of $H_1(z)$ followed by $H_2(z)$, where each stage is implemented using a TDFII structure.

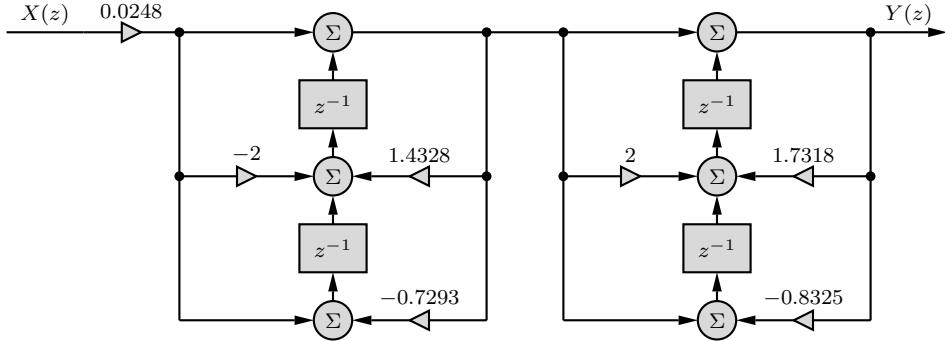


Figure 8.16: Second-order section cascade realization of 4th-order digital Butterworth BPF.

To better appreciate the cascade realization of Fig. 8.16, consider the magnitude response plots of Fig. 8.17. The magnitude response $|H_1(e^{j\Omega})|$ of the first stage is fairly flat and has a gain less than 1 everywhere. The second stage's magnitude response $|H_2(e^{j\Omega})|$ is more peaked and has a gain greater than 1 for most frequencies. Taken in combination, the overall response $|H(e^{j\Omega})| = |H_1(e^{j\Omega})H_2(e^{j\Omega})|$ produces the desired Butterworth BPF characteristics with maximum passband gain of unity. Even for fixed-point implementations, this realization is well behaved and unlikely to cause overflow.

Next, let us exchange the numerators of $H_1(z)$ and $H_2(z)$ to produce an alternate pair of second-order sections. The resulting two transfer functions are

$$H'_1(z) = \frac{z^2 + 2z + 1}{z^2 - 1.4328z^2 + 0.7293} \quad \text{and} \quad H'_2(z) = \frac{0.0248(z^2 - 2z + 1)}{z^2 - 1.7318z^2 + 0.8325}.$$

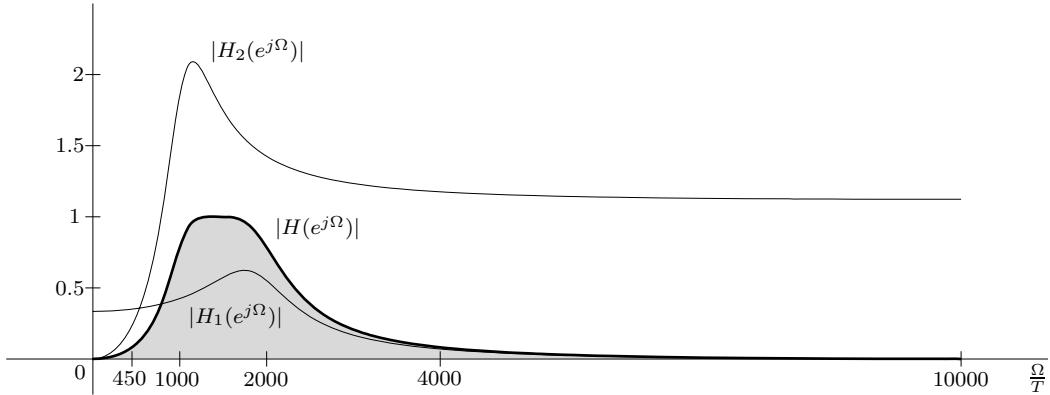


Figure 8.17: Magnitude responses for a cascade of second-order sections.

This cascade realization pairs the poles closest to the unit circle with the zeros that are farthest away, exactly opposite the recommended strategy. Figure 8.18 plots the corresponding magnitude response plots. Although the magnitude response $|H'_2(e^{j\Omega})|$ of the second stage is somewhat less peaked than that in Fig. 8.17, the first stage is significantly more peaked. Further, both $|H'_1(e^{j\Omega})|$ and $|H'_2(e^{j\Omega})|$ have gains greater than 1 for some frequencies. Ideally, the total response $|H'_1(e^{j\Omega})H'_2(e^{j\Omega})|$ exactly equals the total response $|H_1(e^{j\Omega})H_2(e^{j\Omega})|$ of the previous cascade realization. Practically, however, and especially for fixed-point implementations, this alternate structure possesses inferior behavior and can easily cause overflow errors. These differences tend to become only more pronounced as filter order increases.

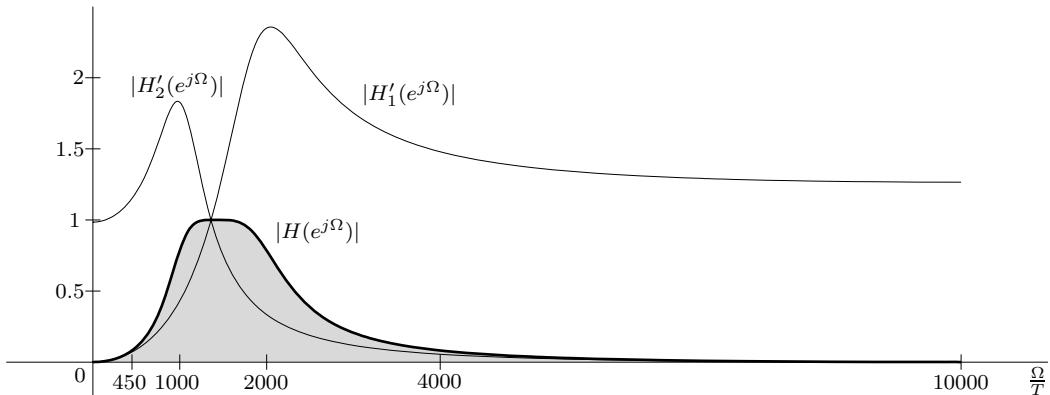


Figure 8.18: Magnitude responses for an alternate cascade of second-order sections.

Example 8.9 ◀

▷ Drill 8.4 (Filter Realization Using a Cascade of Second-Order Sections)

Realize the 10th-order digital Chebyshev bandstop filter of Ex. 8.8 using a cascade of second-order sections.



8.2 Finite Impulse Response Filters

Infinite impulse response (IIR) filters are very sensitive to coefficient accuracy. Implementation deficiencies, especially short word lengths, may drastically change IIR filter behavior and even make them unstable. Moreover, IIR filter design methods are well established only for magnitude responses that are piecewise constant, such as lowpass, bandpass, highpass, and bandstop filters. In contrast, FIR filters are all-zero systems that are guaranteed to be stable even with implementation limitations such as finite word lengths. Additionally, FIR filters can be readily designed to have arbitrarily shaped frequency responses. Unlike IIR filters, FIR filters can possess a linear phase response, thereby allowing constant group delay. This makes FIR filters well suited to applications where linear phase is important. On the other hand, when both FIR and IIR design solutions exist for a particular problem, IIR filters will have lower order, which means that IIR filters operate with less processing delay and require less memory than their FIR counterparts. When processing delay is not critical, FIR filters are the obvious choice. Let us now briefly review the nature of FIR filters.

As discussed in the beginning of this chapter, Eq. (8.1) represents a FIR filter if $a_1 = a_2 = \dots = a_K = 0$. Consequently, the transfer function of the resulting L th-order FIR filter is

$$H(z) = b_0 + b_1 z^{-1} + \dots + b_{L-1} z^{-(L-1)} + b_L z^{-L}. \quad (8.28)$$

By definition, $H(z)$ also equals the z -transform of $h[n]$, or

$$H(z) = \sum_{n=0}^{\infty} h[n] z^{-n} = h[0] + h[1] z^{-1} + \dots + h[L] z^{-L} + h[L+1] z^{-(L+1)} + \dots.$$

Comparison of this equation with Eq. (8.28) shows that

$$h[n] = \begin{cases} b_n & 0 \leq n \leq L \\ 0 & \text{otherwise} \end{cases}. \quad (8.29)$$

Thus, an FIR filter's impulse response $h[n]$ has a finite length of $L_h = L + 1$ elements, and the difference equation coefficients b_n serve as an FIR filter's impulse response $h[n]$, and vice versa.

Written in terms of the impulse response $h[n]$, Eq. (8.28) becomes

$$H(z) = h[0] + h[1] z^{-1} + \dots + h[L-1] z^{-(L-1)} + h[L] z^{-L}. \quad (8.30)$$

Inverting this equation, the impulse response $h[n]$ can be expressed as

$$h[n] = h[0] \delta[n] + h[1] \delta[n-1] + \dots + h[L] \delta[n-L]. \quad (8.31)$$

Substituting $e^{j\Omega}$ for z , Eq. (8.30) also yields an FIR filter's frequency response as

$$H(e^{j\Omega}) = h[0] + h[1] e^{-j\Omega} + \dots + h[L] e^{-jL\Omega} = \sum_{n=0}^L h[n] e^{-jn\Omega}. \quad (8.32)$$

8.2.1 Linear Phase FIR Filters

An important subclass of FIR filters consists of those with linear phase. Such filters are desirable as linear phase implies only a delay in the response, not phase distortion. We shall now show that certain symmetry conditions in $h[n]$ result in a generalized linear phase response.[†]

Consider a causal L th-order finite impulse response (FIR) filter described by the transfer function $H(z)$ of Eq. (8.30) and the corresponding impulse response $h[n]$ of Eq. (8.31). We shall now show that

[†]We restrict our attention here to real filters. By carefully modifying the discussion, one can also treat the more general case of complex filters with linear phase.

FIR filters with odd or even symmetry about the center point of $h[n]$ have linear phase characteristics. As summarized in Table 8.2 and depicted graphically in Fig. 8.19, four types of linear phase FIR filters are possible, depending on whether $h[n]$ is midpoint symmetric or antisymmetric and whether the filter order L is even or odd. Notice that when L is odd (type II or type IV), the midpoint of $h[n]$ does not occur at an integer value of n .

	Even Order L (Odd Length L_h)	Odd Order L (Even Length L_h)
Midpoint Symmetric $h[n]$	Type I	Type II
Midpoint Antisymmetric $h[n]$	Type III	Type IV

Table 8.2: Types of linear phase FIR filters.

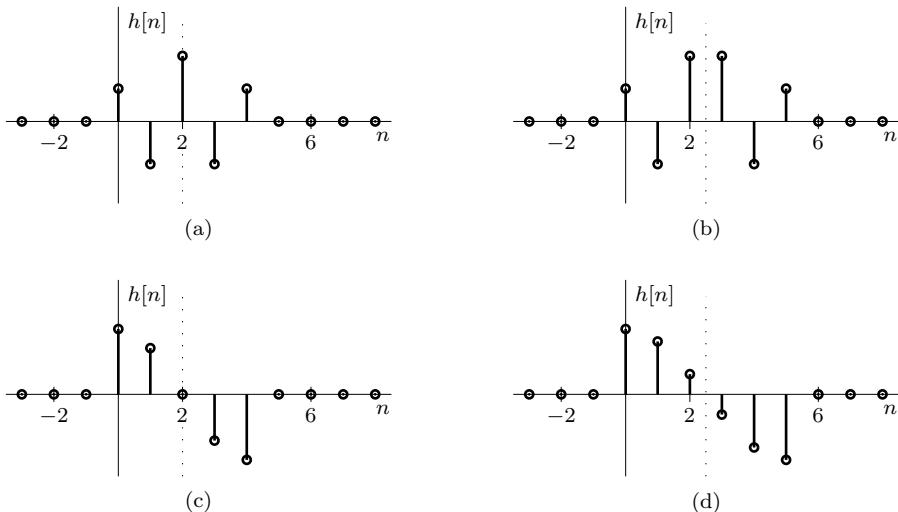


Figure 8.19: Representative linear phase filter impulse responses: (a) type I, (b) type II, (c) type III, and (d) type IV.

To show the phase linearity of these filters, we first consider a fourth-order ($L = 4$) filter that is symmetric about the center point $n = 2$. This is a type I filter, such as the one depicted in Fig. 8.19a. As given by Eq. (8.31), the filter impulse response is given as

$$h[n] = h[0]\delta[n] + h[1]\delta[n-1] + h[2]\delta[n-2] + h[3]\delta[n-3] + h[4]\delta[n-4].$$

Using Eq. (8.32), the frequency response is given by

$$\begin{aligned} H(e^{j\Omega}) &= h[0] + h[1]e^{-j\Omega} + h[2]e^{-j2\Omega} + h[3]e^{-j3\Omega} + h[4]e^{-j4\Omega} \\ &= e^{-j2\Omega} [h[0]e^{j2\Omega} + h[1]e^{j\Omega} + h[2] + h[3]e^{-j\Omega} + h[4]e^{-j2\Omega}]. \end{aligned}$$

Because $h[n]$ is symmetric about $n = 2$, $h[0] = h[4]$, $h[1] = h[3]$, and

$$\begin{aligned} H(e^{j\Omega}) &= e^{-j2\Omega} [h[0](e^{j2\Omega} + e^{-j2\Omega}) + h[2] + h[1](e^{j\Omega} + e^{-j\Omega})] \\ &= e^{-j2\Omega} \underbrace{[2h[0]\cos(2\Omega) + 2h[1]\cos(\Omega) + h[2]]}_{H_a(e^{j\Omega})}. \end{aligned}$$

The bracketed quantity $H_a(e^{j\Omega})$ is real and represents the filter's *amplitude response*; it can be positive over some bands of frequencies and negative over others. Thus, the phase response is

$$\angle H(e^{j\Omega}) = \begin{cases} -2\Omega & H_a(e^{j\Omega}) > 0 \\ -2\Omega - \pi & H_a(e^{j\Omega}) < 0 \end{cases}.$$

The occasional jumps of π in $\angle H(e^{j\Omega})$ caused by $H_a(e^{j\Omega})$ changing sign do not alter the linear nature of the phase response, which is what we sought to prove. Notice in this case that the delay of the filter, which is computed as the negative slope of the phase response, is 2 samples (see Eq. (6.16)).

Let us now consider an $L = 4$ type III filter. The antisymmetry of $h[n]$ requires that $h[0] = 0$ at the center point. Thus, $h[2] = 0$, $h[0] = -h[4]$, $h[1] = -h[3]$, and the frequency response becomes

$$\begin{aligned} H(e^{j\Omega}) &= e^{-j2\Omega} [h[0](e^{j2\Omega} - e^{-j2\Omega}) + h[1](e^{j\Omega} - e^{-j\Omega})] \\ &= je^{-j2\Omega} \underbrace{[2h[0]\sin(2\Omega) + 2h[1]\sin(\Omega)]}_{H_a(e^{j\Omega})}. \end{aligned}$$

As in the type I case, the amplitude response $H_a(e^{j\Omega})$ is real and only impacts the phase response with an occasional jump of π . Since $je^{-j2\Omega} = e^{-j(2\Omega-\pi/2)}$, the phase response is thus

$$\angle H(e^{j\Omega}) = \begin{cases} -2\Omega + \frac{\pi}{2} & H_a(e^{j\Omega}) > 0 \\ -2\Omega - \frac{\pi}{2} & H_a(e^{j\Omega}) < 0 \end{cases}.$$

Clearly, this phase response is linear in Ω . This system has a time delay of 2 samples, the same as in the symmetric case.

Similar procedures demonstrate the linear phase characteristics of type II and type IV filters. All these results are easily generalized to arbitrary order L . Table 8.3 summarizes the amplitude and phase responses for each type as a function of filter order L . Filter magnitude response is just the absolute value of the amplitude response, so the overall frequency response is computed as

$$H(e^{j\Omega}) = |H_a(e^{j\Omega})|e^{j\angle H(e^{j\Omega})}.$$

Filter Type	$H_a(e^{j\Omega})$	$\angle H(e^{j\Omega})$
Type I	$2 \sum_{n=0}^{\frac{L-2}{2}} h[n] \cos(\Omega [\frac{L}{2} - n]) + h[L/2]$	$\begin{cases} -\frac{L}{2}\Omega & H_a(e^{j\Omega}) > 0 \\ -\frac{L}{2}\Omega - \pi & H_a(e^{j\Omega}) < 0 \end{cases}$
Type II	$2 \sum_{n=0}^{\frac{L-1}{2}} h[n] \cos(\Omega [\frac{L}{2} - n])$	$\begin{cases} -\frac{L}{2}\Omega & H_a(e^{j\Omega}) > 0 \\ -\frac{L}{2}\Omega - \pi & H_a(e^{j\Omega}) < 0 \end{cases}$
Type III	$2 \sum_{n=0}^{\frac{L-2}{2}} h[n] \sin(\Omega [\frac{L}{2} - n])$	$\begin{cases} -\frac{L}{2}\Omega + \frac{\pi}{2} & H_a(e^{j\Omega}) > 0 \\ -\frac{L}{2}\Omega - \frac{\pi}{2} & H_a(e^{j\Omega}) < 0 \end{cases}$
Type IV	$2 \sum_{n=0}^{\frac{L-1}{2}} h[n] \sin(\Omega [\frac{L}{2} - n])$	$\begin{cases} -\frac{L}{2}\Omega + \frac{\pi}{2} & H_a(e^{j\Omega}) > 0 \\ -\frac{L}{2}\Omega - \frac{\pi}{2} & H_a(e^{j\Omega}) < 0 \end{cases}$

Table 8.3: Amplitude and phase responses of linear phase FIR filters.

From Table 8.3, the linear nature of each phase response is clear. Further, the delay of these filters equals half the filter order ($L/2$ samples).[†] As we shall later see, the equations of Table 8.3 also provide a foundation for the design of linear phase FIR filters.

[†]When L is odd, the filter delay $L/2$ is not an integer and thus does not truly represent a simple DT shift. Refer to Sec. 6.4 for a more correct interpretation of non-integer shifts.

Inherent Restrictions of Linear Phase FIR Filters

The behavior of an FIR filter is determined by the zeros of $H(z)$. For a real FIR system, we know that a complex zero must occur with its conjugate. For linear phase FIR filters, the symmetry of $h[n]$ imposes additional constraints on the system zeros. Let us investigate the implications of these restrictions. To begin, let us look at the transfer function of a symmetric (type I or II) FIR filter. Since the filter is symmetric, $h[n] = h[L - n]$, and

$$H(z) = \sum_{n=0}^L h[n]z^{-n} = \sum_{n=0}^L h[L - n]z^{-n}.$$

Applying the change of variable $m = L - n$ yields

$$H(z) = \sum_{m=0}^L h[m]z^{m-L} = z^{-L} \sum_{m=0}^L h[m](z^{-1})^{-m}$$

or

$$H(z) = z^{-L} H(z^{-1}). \quad (8.33)$$

In other words, symmetry in $h[n]$ forces $H(z)$ to be what is called a *mirror image polynomial*. Similarly, the transfer function of an antisymmetric (type III or IV) FIR filter is an *anti-mirror image polynomial*

$$H(z) = -z^{-L} H(z^{-1}). \quad (8.34)$$

According to both Eqs. (8.33) and (8.34), if z_l is a zero of $H(z)$, then the reciprocal $1/z_l$ must also be a zero of $H(z)$.

Now, consider a system with a general zero at $z = re^{j\theta}$. If the system is real, it must also include a zero at the conjugate location $z = re^{-j\theta}$. If the system is linear phase, the reciprocals $z = \frac{1}{r}e^{-j\theta}$ and $z = \frac{1}{r}e^{j\theta}$ must also be zeros of the system. Thus, a general zero of a real linear phase FIR filter always travels in the company of three other roots. Setting $r = 1$, a zero on the unit circle $z = e^{j\theta}$ need only travel with its conjugate $z = e^{-j\theta}$. The requirement for reciprocal roots is already met since $r = 1 = 1/r$. Similarly, a real zero $z = r$ need only travel with its reciprocal $z = \frac{1}{r}$ since real roots already equal their own conjugates. Zeros at $z = \pm 1$, which equal their own reciprocal and conjugate, can exist singly.

To see how this discussion helps us better understand linear phase FIR filters, consider a real type II FIR filter. Since the order L is odd, there must be an odd number of zeros. However, most zeros must travel in groups of two or four. The only way to obtain an odd number of zeros is for at least one system zero to be at either 1 or -1. It is straightforward to show that a type II FIR filter must possess an odd number of zeros at $z = -1$ (see Prob. 8.2-3). By constraint, type II filters cannot pass high frequencies and therefore are not suitable for highpass and bandstop filter applications. Table 8.4 summarizes the various restrictions for each type of linear phase FIR filter.

Filter Type	Zero Restrictions	Suitable Filter Types
Type I	None	LP, HP, BP, BS, multiband
Type II	Odd number of zeros at $z = -1$	LP, BP
Type III	Odd number of zeros at $z = 1$, Odd number of zeros at $z = -1$	BP
Type IV	Odd number of zeros at $z = 1$	HP, BP

Table 8.4: Restrictions to linear phase FIR filters.

As shown in Table 8.4, type I FIR filters have the fewest restrictions and are generally the most popular of the four types. There are reasons, however, to utilize the other types. The phase requirements of Hilbert transformers and differentiators, for example, are not satisfied by type I or type II filters, which makes a type III or type IV filter the logical choice. When a filter is realized in hardware, the most efficient memory storage and retrieval occurs if $h[n]$ has a length that is a power of 2; this requires that the order of the filter be odd, which favors type II and type IV filters. As is so often the case in engineering, good filter design needs to balance a wide variety of design considerations and constraints, and no simple “one-size-fits-all” design method works for every possible case.

8.2.2 Realization of FIR Filters

The FIR filter in Eq. (8.28) is a special case of the general filter structures, depicted in Figs. 7.14, 7.17, and 7.19, with all feedback (or recursive) coefficients set equal to zero. Therefore, the realization of an FIR filter is the same as that of an IIR filter with all the feedback connections omitted. Figure 8.20a shows the direct form realization for an L th-order FIR filter using Eq. (8.29) to define the feedforward coefficients in terms of the impulse response. With all the feedback coefficients zero, there is no difference between the DFI and DFII structures. The transposed direct form of an FIR filter, shown in Fig. 8.20b, is nearly identical to the direct form. We can readily verify from either of these two figures that for the input $\delta[n]$, the output is $h[n]$ given in Eq. (8.31).

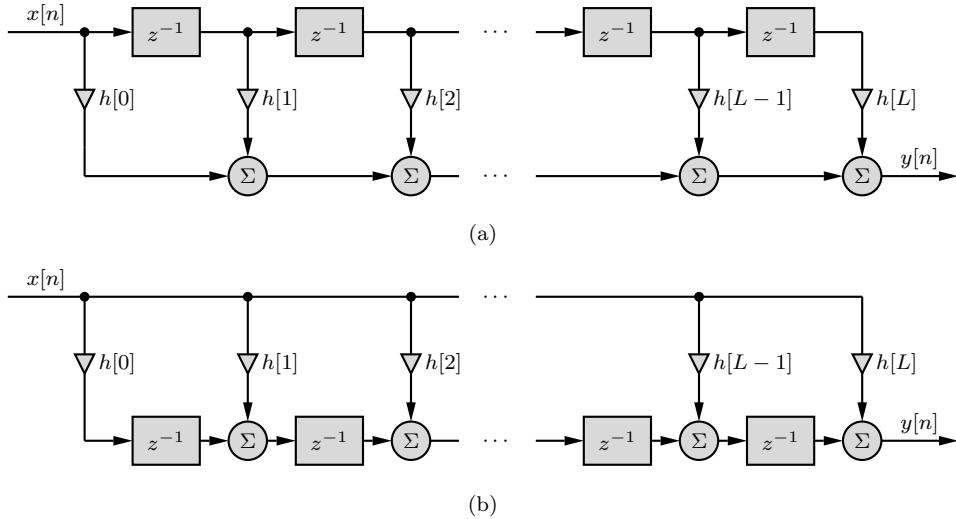


Figure 8.20: Canonical realizations of an L th-order FIR filter: (a) direct form and (b) transposed direct form.

As Fig. 8.20 illustrates, an FIR filter is a *tapped delay line* with successive taps at unit delay (T seconds). Such a filter is also known as a *transversal filter*. To obtain a cascade realization, we factor Eq. (8.30) into a combination of first- and second-order terms and then cascade their realizations.

Realization of Linear Phase Filters

For linear phase FIR filters, symmetry conditions reduce the number of coefficients to half or nearly half. Table 8.3 shows that the number of coefficients required for an L th-order FIR filter is either $L/2$ (type III), $(L+1)/2$ (types II and IV), or $(L+2)/2$ (type I). By summing appropriate pairs of delay chain outputs prior to coefficient multiplication, we can use symmetry to reduce by half or nearly half the number of multipliers needed by direct realization. For many hardware targets, multiply

operations are quite expensive, so such a substantial reduction in multipliers is very desirable. Figures 8.21a and 8.21b illustrate the idea for type I and type II FIR filters, respectively.

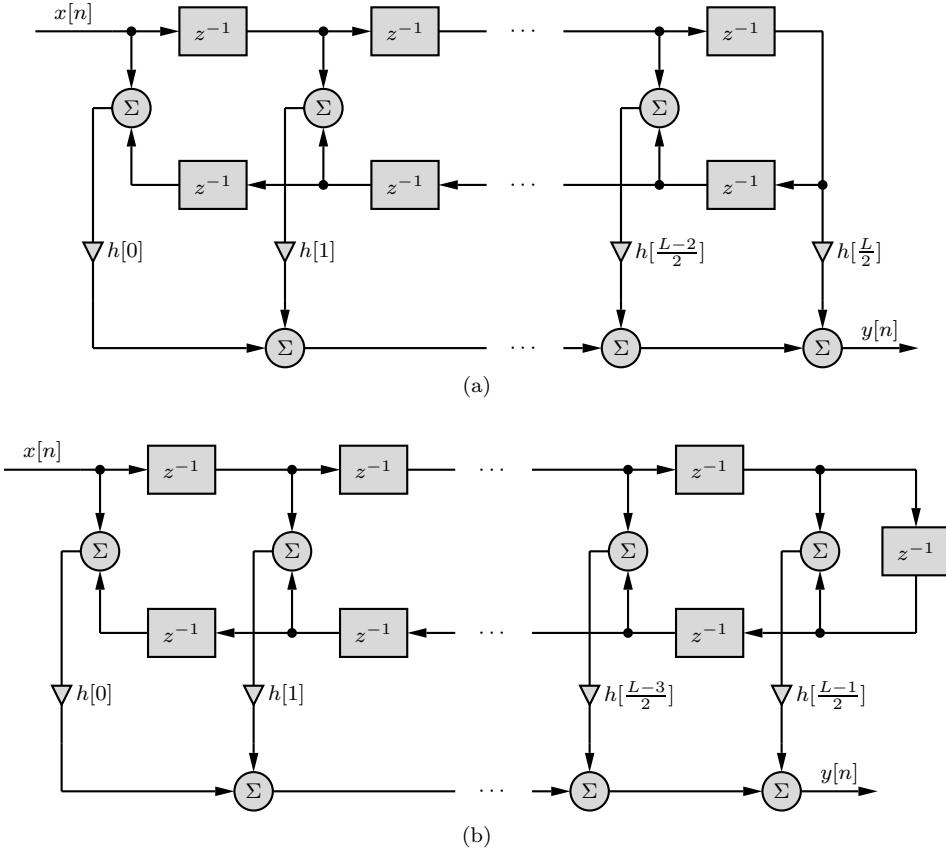


Figure 8.21: Multiplier-efficient realizations of linear phase FIR filters: (a) type I and (b) type II.

▷ Example 8.10 (Realization of a Comb Filter)

Determine the type and direct form realization of a sixth-order comb filter whose impulse response is given by

$$h[n] = \delta[n] - \delta[n - 6].$$

Additionally, determine the transfer function and plot the frequency response of this system.

Because the impulse response is finite duration and antisymmetric about $n = 3$, this is a type III linear phase FIR filter with $L = 6$. Also,

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n} = \sum_{n=-\infty}^{\infty} (\delta[n] - \delta[n - 6])z^{-n} = 1 - z^{-6}.$$

Figure 8.22a shows a direct form realization of this system.

Substituting $z = e^{j\Omega}$ into $H(z)$, the frequency response is given by

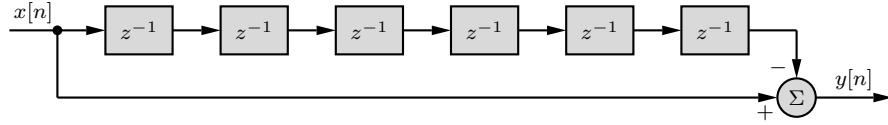
$$H(e^{j\Omega}) = 1 - e^{-j6\Omega} = e^{-j3\Omega}(e^{j3\Omega} - e^{-j3\Omega}) = 2je^{-j3\Omega} \sin(3\Omega) = 2e^{-j(3\Omega - \frac{\pi}{2})} \sin(3\Omega).$$

The magnitude response, shown in Fig. 8.22b, is shaped like a comb with periodic nulls. The phase response, shown Fig. 8.22c, verifies the linear phase nature of the system.

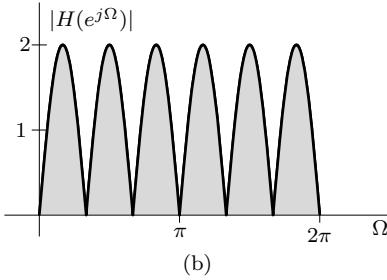
```

01 Omega = linspace(0,2*pi,1001); H = @(Omega) 1-exp(-1j*6*Omega);
02 subplot(121); plot(Omega,abs(H(Omega)),',k');
03 subplot(122); plot(Omega,angle(H(Omega)),',k');

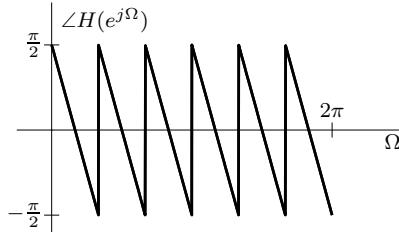
```



(a)



(b)



(c)

Figure 8.22: Sixth-order comb filter: (a) direct form realization, (b) magnitude response, and (c) phase response.

Using a similar procedure, we see that an L th-order comb filter has transfer function $H(z) = 1 - z^{-L}$ and frequency response

$$H(e^{j\Omega}) = 2e^{-j(\frac{L\Omega}{2} - \frac{\pi}{2})} \sin(L\Omega/2).$$

Over any 2π interval of frequency Ω , the magnitude response of an L th-order comb filter has L nulls and L peaks.

Example 8.10 \triangleleft

▷ **Drill 8.5 (Realization of Type III and Type IV FIR Filters)**

Modify the type I and type II structures of Fig. 8.21 to instead realize type III and type IV FIR filters.

\triangleleft

8.2.3 Windowing in FIR Filters

Since the impulse response of an FIR filter is finite in duration, we can always view $h[n]$ as being an implicitly windowed function, even if the window is simply rectangular. Further, when a desired filter response has infinite length, explicit windowing produces an FIR approximation of the system. In either case, windowing is inherent to FIR filters, and it is important to understand the nature of window functions and their impact on filter performance and behavior.

We observed in Sec. 2.4.1 that straight truncation of data creates serious problems such as spectral spreading and leakage. As explained in Sec. 2.4.3, we can partly remedy these problems by increasing the window length and tapering (smoothing) the windows. Much of the discussion in Ch. 2 about windows for continuous-time signals applies to discrete-time signals also.[†] A review of

[†]The reason for this similarity stems from the fact that DT windows are sampled versions of CT windows. Thus, as discussed in Ch. 3, DT window spectra are just periodically replicated versions of CT window spectra.

Sec. 2.4 is highly recommended for a better understanding of this section. We shall briefly survey the effects of windowing for the discrete-time case.

Consider a causal rectangular window function $w_{\text{rec}}[n]$ of length $L_w = 9$, depicted in Fig. 8.23a. This same window (centered at the origin) is analyzed in Ex. 6.1. As shown in Fig. 8.23b, the magnitude spectrum of this window, found from pair 7 of Table 6.1, is

$$|W_{\text{rec}}(e^{j\Omega})| = \left| \frac{\sin(L_w\Omega/2)}{\sin(\Omega/2)} \right|.$$

The main lobe (the shaded lobe centered at the origin) reaches zero at $\Omega = \pm 2\pi/9$ so that the main lobe width is $4\pi/9$. In general, the main lobe width is $4\pi/L_w$ for a rectangular window of length L_w . This may also be computed from pair 7 of Table 6.1. Observe the reciprocal nature between window length and bandwidth. For this window, the spectrum decays rather slowly with frequency (approximately as $1/\Omega$). Thus, the peak value of the second lobe of the spectrum (the highest side lobe) is only about 13.3 dB below the peak value of the main lobe, as shown in the normalized dB spectrum of Fig. 8.23c.

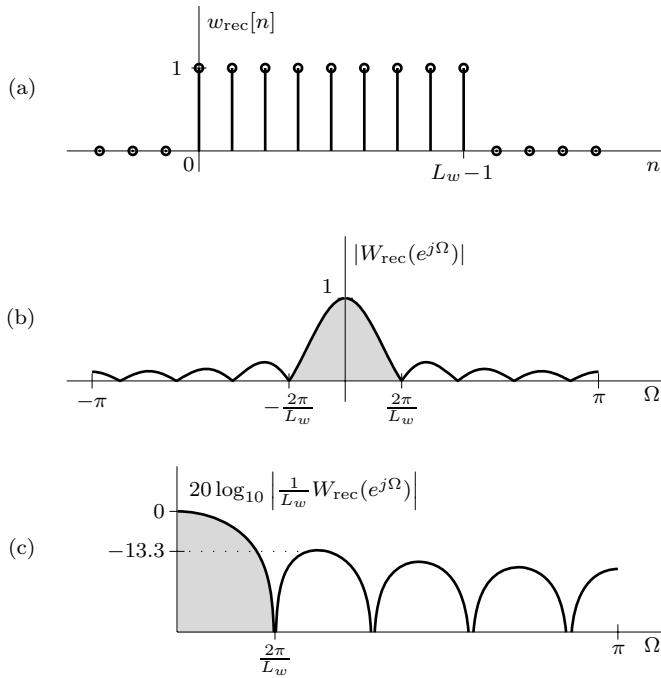


Figure 8.23: Length $L_w = 9$ rectangular window and its spectrum.

To truncate an impulse response $h[n]$ with a window $w[n]$, we multiply $h[n]$ by $w[n]$. The windowed (truncated) impulse response $h_w[n] = h[n]w[n]$ has finite length. According to the frequency-convolution property, the spectrum $H_w(e^{j\Omega})$ is $1/2\pi$ times the (circular) convolution of the two spectra $H(e^{j\Omega})$ and $W(e^{j\Omega})$. This convolution spreads or smears the spectrum $H(e^{j\Omega})$, mostly due to the main lobe of $W(e^{j\Omega})$. Thus, following the width property of convolution, truncation increases the width of $H(e^{j\Omega})$ by the width of $W(e^{j\Omega})$. Moreover, the window spectrum is generally not bandlimited, although it usually decays with Ω . A window function's spectral side lobes negatively impact the windowed filter response $H_w(e^{j\Omega})$. For example, windowing the impulse response $h[n]$ of an ideal lowpass filter spreads out its passband and produces unwanted ripple (nonzero gain) in the stopband. To minimize these spectral distortions, we desire a window spectrum with narrow main lobe (bandwidth) and small side lobes.

The ideal window is a rectangular window of infinite length. Such a window permits all the data to pass and hence causes no distortion. Over the fundamental band, the frequency response of this ideal window is $2\pi\delta(\Omega)$. Thus, an ideal window spectrum has zero width and side lobe level of zero. In practice, a window of infinite length defeats the very purpose of windowing to produce a finite-length filter (FIR filter). Hence, our goal should be to achieve a window spectrum as close to $\delta(\Omega)$ as possible yet remain finite in duration. In other words, we want a finite-duration window whose spectral width (spread) and side lobes are as small as possible.

Unfortunately, the two goals of narrow main lobe width and small side lobes are generally incompatible. For a given window length, if we try to improve the one, then the other deteriorates. For instance, low side lobes (reduced leakage) require a smooth (tapered) window that truncates data gradually rather than abruptly. But as window smoothness increases, so does spectral spreading. Recalling that window length is inversely proportional to the spectral spread, the way out of this dilemma is to increase the window length sufficiently to counter the increased spectral spread due to tapering. Thus, we choose a sufficiently smooth (tapered) window to obtain the desired side lobe (leakage) characteristics and then we increase the window length to compensate for the increased spectral spreading.

The relative importance of spectral spreading and leakage varies with application. In spectral analysis, for example, a smooth window is often desired since its small side lobes with fast decay help minimize interference between different frequency components. On the other hand, window smoothness is less important in filter applications that emphasize a narrow transition band over large stopband attenuation.

Some Useful Windows

A useful selection of discrete-time window functions is readily obtained by sampling the continuous-time window functions in Table 2.1 (page 110). For convenience, we make the DT window functions causal by appropriately delaying the CT window functions prior to sampling. Most of the discussion in Sec. 2.4 about continuous-time windows applies to the discrete-time (sampled) versions as well. Hence, we again stress a review of Sec. 2.4.

A causal length- L_w rectangular window is specified by

$$w_{\text{rec}}[n] = u[n] - u[n - L_w].$$

Similarly, to construct a causal length- L_w Hamming window, we delay the continuous-time Hamming window from Table 2.1 by $T/2$ to render it causal. Over $0 \leq t \leq T$, this yields

$$w_{\text{ham}}(t) = 0.54 + 0.46 \cos\left(\frac{2\pi[t - (T/2)]}{T}\right) = 0.54 - 0.46 \cos\left(\frac{2\pi t}{T}\right).$$

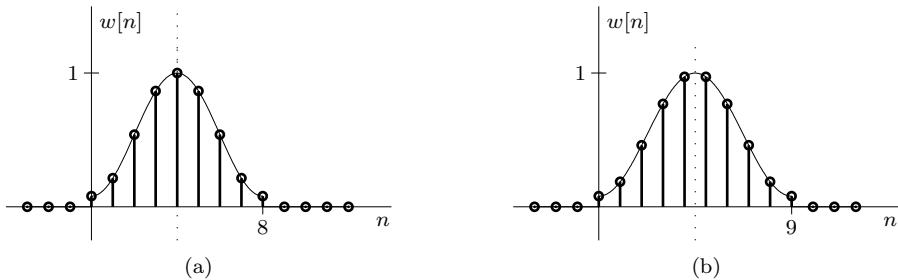
Now we sample this signal at intervals of $T/(L_w - 1)$ by substituting $t = nT/(L_w - 1)$ to obtain L_w uniformly spaced samples. This yields

$$w_{\text{ham}}[n] = \begin{cases} 0.54 - 0.46 \cos\left(\frac{2\pi n}{L_w - 1}\right) & 0 \leq n \leq L_w - 1 \\ 0 & \text{otherwise} \end{cases}.$$

Figures 8.24a and 8.24b show the Hamming window for $L_w = 9$ and $L_w = 10$, respectively.

In a similar way, we can construct discrete-time versions of the other continuous-time windows of Table 2.1. Table 8.5 summarizes the resulting discrete-time window functions.[†] Since the underlying CT windows are non-bandlimited, sampling them to produce the DT windows causes aliasing in the spectra. The effects of this aliasing change with L_w , so the main lobe width, rolloff rate, and peak side lobe values in Table 8.5 should be taken as approximations. Observe that the windows in Table 8.5 are symmetric about the center point $(L_w - 1)/2$, so the corresponding window spectra possess linear phase characteristics.

[†]In the literature, expressions for these same windows often appear different because they are either not causal or their expressions omit the end points with zero values.

Figure 8.24: Causal Hamming windows: (a) $L_w = 9$ and (b) $L_w = 10$.

Window $w[n]$ for $0 \leq n \leq L_w - 1$, 0 otherwise	Main Lobe Width	Rolloff Rate [dB/dec]	Peak Side Lobe Level [dB]
1. Rectangular:	$\frac{4\pi}{L_w}$	-20	-13.3
2. Triangular (Bartlett): $1 - \frac{ 2n-(L_w-1) }{L_w-1}$	$\frac{8\pi}{L_w}$	-40	-26.5
3. Hann: $\frac{1}{2} \left[1 - \cos\left(\frac{2\pi n}{L_w-1}\right) \right]$	$\frac{8\pi}{L_w}$	-60	-31.5
4. Hamming: $0.54 - 0.46 \cos\left(\frac{2\pi n}{L_w-1}\right)$	$\frac{8\pi}{L_w}$	-20	-42.7
5. Blackman: $0.42 - 0.5 \cos\left(\frac{2\pi n}{L_w-1}\right) + 0.08 \cos\left(\frac{4\pi n}{L_w-1}\right)$	$\frac{12\pi}{L_w}$	-60	-58.1
6. Kaiser: $\frac{I_0\left(\alpha \sqrt{1 - \left(\frac{2n-(L_w-1)}{L_w-1}\right)^2}\right)}{I_0(\alpha)}$	varies with α	-20	varies with α

Table 8.5: Common discrete-time window functions and their approximate characteristics.

Comments on Windows

Recall that a desirable window is one with narrow main lobe and small side lobes. For a given window type, peak side lobe levels are more or less fixed. Main lobe width, however, shrinks as window length increases. Thus, it is typical to choose a window based on side lobe characteristics and then adjust the window length to achieve a suitable main lobe width. Notice also that the complexity (order) of a windowed FIR filter depends on the length of the window. Particularly for real-time filtering applications, it is crucial to minimize the computational complexity of the filter. Thus, the shortest possible window length should be chosen.

For a given length L_w , the rectangular window has the smallest spectral spread ($4\pi/L_w$). Unfortunately, its peak side lobe is also highest (-13.3 dB relative to the main lobe peak). This undesirable behavior is the result of the abrupt truncation and consequent Gibbs phenomenon. These problems can be minimized by using a tapered window, although a longer window will be required to maintain a comparable level of spectral spread.

As discussed in Ch. 2, the triangular window (also called the *Bartlett window*) is one of the simplest tapered windows, but it is rarely used since its characteristics are largely inferior to other

window types. The relatively simple Hann window (also called the *Hanning window*), with modest spread ($8\pi/L_w$), reasonable peak side lobe (-31.5 dB), and excellent rolloff rate, is quite popular, particularly for spectral analysis applications. Both the triangular and Hann windows are defined with their end points as zero. Because of this feature, when using the triangular or Hann window, the filter order can be reduced by two for the same window length. Alternately, we can increase the window length by two points for the same length L_w .

Like the triangular and Hann windows, the Hamming window has a reasonable main lobe width of $8\pi/L_w$. Its relatively low side lobe level (-42.7 dB), achieved at the cost of poor rolloff, makes it a popular choice for filtering applications, where low side lobe level is more important than fast rolloff rates. The Blackman window achieves both low side lobes and fast rolloff at the expense of increased complexity and main lobe width ($12\pi/L_w$).

The Kaiser window is an *adjustable window* that is defined in terms of $I_0(\cdot)$, a 0th-order modified Bessel function of the first kind. Bessel functions are most easily evaluated with mathematical packages such as MATLAB but can also be computed according to

$$I_0(x) = \sum_{k=0}^{\infty} \left(\frac{x^k}{2^k k!} \right)^2.$$

By adjusting the parameter α , the smoothness of the Kaiser window can be controlled. Starting at 0, increasing α tends to increase the smoothness of the Kaiser window, reduce its peak side lobe level, and increase its main lobe width. Since a Kaiser window can be adjusted to meet almost any set of filter specifications, it is perhaps the most widely used window in signal processing.

Although Table 8.5 summarizes the most popular discrete-time window functions, there are many others. Additional examples include the cosine and Lanczos windows, as well as the adjustable Dolph-Chebyshev window. Each different window has its own special characteristics that make it suitable for particular applications. Details can be found in the literature and online.

8.2.4 Time-Domain Methods of FIR Filter Design

As in the case of IIR filters, FIR filters can be designed using time-domain or frequency-domain criteria. For time-domain criteria, we exactly or closely match a digital filter impulse response to the samples of a desired (analog or digital) filter impulse response over a finite range $0 \leq n \leq L_h - 1$. For frequency-domain criteria, we exactly or closely match a digital filter frequency response to a desired (analog or digital) filter frequency response at a selection of frequencies, normally at uniform spacing. Let us begin our investigation of these methods by looking at two time-domain methods of FIR filter design, the second of which is a special case of the first.

With the *window method*, a length- L_h window is applied to a desired impulse response to produce the desired FIR filter impulse response.[†] If the window is tapered, the FIR impulse response only approximately matches the desired impulse response over $0 \leq n \leq L_h - 1$. If the window is rectangular, however, the digital filter impulse response exactly matches the desired impulse response over $0 \leq n \leq L_h - 1$. This case, known as the *Fourier series method*, is similar to the impulse invariance method used for IIR filters discussed in Sec. 8.1.1, except that the FIR filter impulse response must be of finite length.

For the window and Fourier series methods, the FIR filter design problem begins as follows: we are given a digital frequency response $H(e^{j\Omega})$ that needs to be realized by an FIR filter. Alternately, we may be given a continuous-time filter frequency response $H_c(j\omega)$ that needs to be realized using a FIR filter. Since the former case can be easily converted to the latter, we shall consider only the case where we are required to realize a continuous-time frequency response $H_c(j\omega)$ using a length- L_h FIR filter.[‡]

[†]When an FIR filter is designed using the window method, the filter and window lengths are equal, $L_h = L_w$.

[‡]To convert $H(e^{j\Omega})$ to $H_c(j\omega)$, simply set $T = 1$ so that $\omega = \Omega$. Then, $H_c(\omega) = H(e^{j\omega})$ for $|\omega| \leq \pi$ and is zero otherwise.

Recall that a digital filter's frequency response is periodic, with the first period in the frequency range $-\frac{\pi}{T} \leq \omega < \frac{\pi}{T}$. Hence, the best we can hope for is to realize the equivalence of $H_c(j\omega)$ over this range. Using this range, the bandlimited CT impulse response is given by the inverse Fourier transform as

$$h_c(t) = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} H_c(j\omega) e^{j\omega t} d\omega. \quad (8.35)$$

Remember, by integrating from $-\frac{\pi}{T}$ to $\frac{\pi}{T}$ rather than from $-\infty$ to ∞ , we effectively bandlimit $H_c(j\omega)$, which ensures that we can use the impulse invariance method of Eq. (6.55) to convert $h_c(t)$ to a discrete-time form. This condition does not pose any limitations in our realization of the continuous-time filter because by selecting smaller T , we can encompass as much range of ω as we wish.

The CT response $h_c(t)$ is generally centered at the origin and of infinite length. To convert this noncausal response into a causal DT sequence of finite length, we must delay $h_c(t)$ by $T(L_h - 1)/2$, convert the response to DT by $h[n] = Th_c(nT)$ (Eq. (6.55)), and then truncate the response using a suitable window $w[n]$ of length L_h .[†] The combination of these operations results in a causal impulse response with finite length L_h , that is,

$$h[n] = Th_c([n - \frac{L_h-1}{2}]T)w[n]. \quad (8.36)$$

Substituting Eq. (8.35) into the right-hand side of this equation yields

$$h[n] = \left(\frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} H_c(j\omega) e^{j\omega(n - \frac{L_h-1}{2})T} d\omega \right) w[n]. \quad (8.37)$$

This equation describes the window method of FIR filter design. Notice that Eq. (8.37) produces a causal impulse response with finite length L_h . Knowing $h[0], h[1], h[2], \dots, h[L_h - 1]$, we can determine $H(z)$ using Eq. (8.30) and can realize the filter using a standard structure such as those shown in Fig. 8.20.

Optimality of the Rectangular Window

If the truncating window is rectangular, the procedure outlined here is optimum in the sense that the energy of the error (difference) between the desired frequency response $H_c(j\omega)$ and the realized frequency response $H(e^{j\omega T})$ is the minimum for a given L_h . This conclusion follows from the fact that the resulting filter frequency response $H(e^{j\omega T})$ is given by

$$H(e^{j\omega T}) = \sum_n h[n] e^{-j\omega n T}.$$

This frequency response is an approximation of the desired frequency response $H_c(j\omega)$ because of the finite length of $h[n]$. Thus,

$$H_c(j\omega) \approx \sum_n h[n] e^{-j\omega n T}. \quad (8.38)$$

How do we select $h[n]$ for the best approximation in the sense of minimizing the energy of the error $H_c(j\omega) - H(e^{j\omega T})$? The preceding equation shows that the right-hand side is the finite-term exponential Fourier series for $H_c(j\omega)$ with period $2\pi/T$. As seen from Eq. (8.37), $h[n]$ are the Fourier coefficients, which explains why the rectangular window method is also called the *Fourier series method*. According to the *finality property* (see page 37), a finite Fourier series is the optimum (in the sense of minimizing the error energy) for a given L_h . In other words, for a given number of terms (L_h), any choice for $h[n]$ other than the Fourier coefficients in Eq. (8.38) will lead to higher

[†]When L_h is odd, we can order the delay operation after truncation. However, for even L_h , we must delay the sequence before sampling and truncation. Otherwise, the elements of $h[n]$ are located at fractional values of n .

error energy. For windows other than rectangular, this minimum mean square error optimality deteriorates, and the filter is somewhat suboptimal.

Next, let us investigate some concrete examples of window method FIR filter design. To emphasize procedure rather than a jungle of data, the initial examples have small L_h and no rigid specifications. Later, we shall consider design problems with all the usual filter specifications.

▷ **Example 8.11 (Window Method Lowpass Filter Design)**

Using rectangular and Hamming windows, design length-7 FIR filters to approximate an audio lowpass filter with cutoff frequency $f_c = 20$ kHz. Plot the magnitude response of each filter. Set the sampling frequency F_s equal to four times the cutoff frequency f_c .

In this case, the impulse response length (and thus the window length) is $L_h = 7$, and the sampling interval T is computed as

$$F_s = \frac{1}{T} = 4f_c = 80000 \quad \Rightarrow \quad T = 12.5 \times 10^{-6}.$$

Recall that a continuous-time frequency ω appears as a discrete-time frequency $\Omega = \omega T$. Thus, the desired CT cutoff frequency $\omega_c = 2\pi f_c = \frac{\pi F_s}{2} = \frac{\pi}{2T}$ appears at $\Omega_c = \frac{\pi}{2}$, which is halfway to the highest (non-aliased) digital frequency $\Omega = \pi$.

Although we could now use Eq. (8.37) to find the desired impulse responses, we shall derive the results in steps. This approach, although more lengthy, will help clarify the basic design concepts. Let us consider the rectangular window (Fourier series method) first.

Using pair 8 of Table 1.1, the impulse response of the desired ideal (zero-phase) lowpass filter is

$$h_c(t) = \frac{1}{2T} \text{sinc}\left(\frac{t}{2T}\right).$$

Next, we delay $h_c(t)$ by $\frac{(L_h-1)T}{2} = 3T$ to obtain

$$h_c(t - 3T) = \frac{1}{2T} \text{sinc}\left(\frac{t - 3T}{2T}\right).$$

This response is now scaled by T , truncated (windowed), and sampled by setting $t = nT$ to obtain the desired FIR impulse response

$$h_{\text{rec}}[n] = Th_c(nT - 3T)w_{\text{rec}}[n] = \frac{1}{2} \text{sinc}\left(\frac{n - 3}{2}\right) w_{\text{rec}}[n].$$

This result also follows directly from Eq. (8.36).

To visualize these three steps, consider Fig. 8.25. Using the impulse invariance method, the ideal but noncausal DT impulse response is $Th_c(nT) = 0.5 \text{sinc}(n/2)$ (Fig. 8.25a). To obtain a causal and finite-duration solution, we first delay the CT response by $(L_h - 1)T/2 = 3T$ (Fig. 8.25b) and then truncate it with a length-7 rectangular window (Fig. 8.25c). In this case, the noncausal filter is made realizable at a cost of a $3T$ -second delay. Over the window duration $0 \leq n \leq 6$, the samples $h_{\text{rec}}[n]$ exactly match the underlying CT impulse response. In this case, $h_{\text{rec}}[n]$ is also midpoint symmetric and thus a type I linear phase FIR filter.

Using Eq. (8.30) and computing each value of $h_{\text{rec}}[n]$, the transfer function is

$$H_{\text{rec}}(z) = \sum_{n=0}^6 h_{\text{rec}}[n]z^{-n} = -\frac{1}{3\pi} + \frac{1}{\pi}z^{-2} + \frac{1}{2}z^{-3} + \frac{1}{\pi}z^{-4} - \frac{1}{3\pi}z^{-6}.$$

Setting $z = e^{j\Omega}$, the frequency response is

$$H_{\text{rec}}(e^{j\Omega}) = \sum_{n=0}^6 h_{\text{rec}}[n]e^{-j\Omega n} = -\frac{1}{3\pi} + \frac{1}{\pi}e^{-j2\Omega} + \frac{1}{2}e^{-j3\Omega} + \frac{1}{\pi}e^{-j4\Omega} - \frac{1}{3\pi}e^{-j6\Omega}.$$

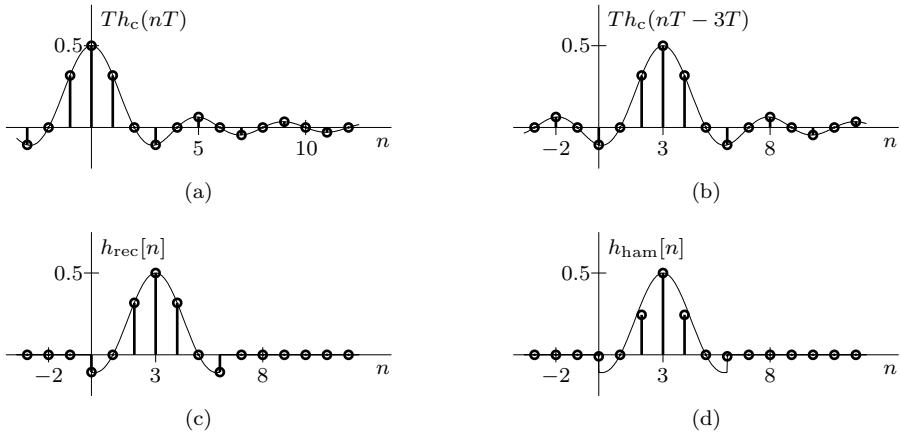


Figure 8.25: Discrete-time LPF impulse responses: (a) ideal, (b) shifted ideal, (c) rectangular windowed, and (d) Hamming windowed.

Using the amplitude and phase responses of Table 8.3, we can alternately express the frequency response as

$$H_{rec}(e^{j\Omega}) = \left[\frac{1}{2} + \frac{2}{\pi} \cos(\Omega) - \frac{2}{3\pi} \cos(3\Omega) \right] e^{-j3\Omega}.$$

The right-hand term $e^{-j3\Omega}$ confirms the filter's linear phase characteristics and represents 3 samples (3T seconds) of delay. Computed using MATLAB, Fig. 8.26 shows the resulting magnitude response $|H_{rec}(e^{j\Omega})|$ as well as the ideal lowpass response (shaded) for reference. The magnitude response exhibits oscillatory behavior that decays rather slowly over the stopband. Although increasing L_h improves the frequency response, the oscillatory nature persists due to Gibbs phenomenon.

```

01 Lh = 7; fc = 20000; T = 1/(4*fc); n = 0:Lh-1;
02 hc = @(t) 1/(2*T)*sinc(t/(2*T));
03 wrec = @(n) 1.0*((n>=0)&(n<=Lh-1));
04 hrec = T*hc(n*T-(Lh-1)/2*T).*wrec(n)
      hrec = -0.1061 -0.0000  0.3183  0.5000  0.3183  0.0000 -0.1061
05 Omega = linspace(-pi,pi,1001);
06 Hrec = polyval(hrec,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega); plot(Omega,abs(Hrec));

```

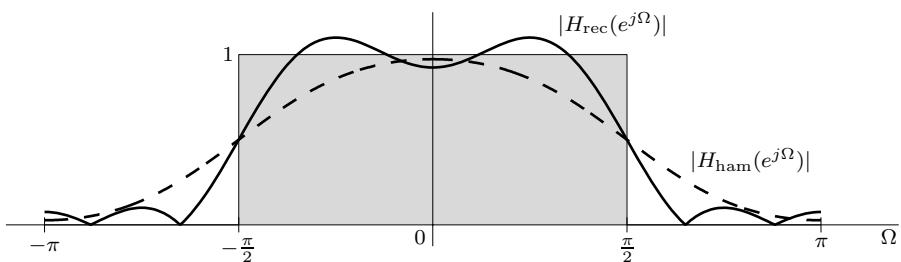


Figure 8.26: Window method LPF magnitude responses: rectangular (solid) and Hamming (dashed).

The side lobes in the rectangular window are rather large (-13 dB), which results in a filter with poor stopband attenuation. The Hamming window, which has smaller side lobes (-43 dB), produces a filter with better stopband attenuation but wider transition band. Using a Hamming

window rather than a rectangular window, the desired impulse response is

$$h_{\text{ham}}[n] = Th_c(nT - 3T)w_{\text{ham}}[n] = \frac{1}{2}\text{sinc}\left(\frac{n-3}{2}\right)w_{\text{ham}}[n].$$

Using Table 8.5 to define $w_{\text{ham}}[n]$, MATLAB readily computes $h_{\text{ham}}[n]$, which is shown in Fig. 8.25d.

```
07 wham = @ (n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
08 hham = T*hc(n*T-(Lh-1)/2*T).*wham(n)
hham = -0.0085 -0.0000 0.2451 0.5000 0.2451 0.0000 -0.0085
```

Unlike the result obtained using the rectangular window, $h_{\text{ham}}[n]$ does not exactly match the underlying CT impulse response. The largest differences occur at the edges, where the taper of the Hamming window is greatest.

Using Eq. (8.32) and the computed values of $h_{\text{ham}}[n]$, the frequency response is

$$H_{\text{ham}}(e^{j\Omega}) = -0.0085 + 0.2451e^{-j2\Omega} + 0.5e^{-j3\Omega} + 0.2451e^{-j4\Omega} - 0.0085e^{-j6\Omega}.$$

The magnitude response $|H_{\text{ham}}(e^{j\Omega})|$ is shown dashed in Fig. 8.26.

```
09 Hham = polyval(hham,exp(1j*0mega)).*exp(-1j*(Lh-1)*0mega); plot(0mega,abs(Hham));
```

Observe that the transition band corresponding to the Hamming window is wider than that corresponding to the rectangular window. Both $h_{\text{rec}}[n]$ and $h_{\text{ham}}[n]$ are symmetric about $n = 3$ and, hence, are type I filters. Either can be realized with six delay elements using a structure of the form depicted in Fig. 8.21a.

Comments:

Referring to Fig. 8.26, the ideal (unwindowed) filter transitions from passband to stopband abruptly, resulting in a zero-width transition band. In the windowed filters, on the other hand, the spectrum spreads out, which results in a gradual transition from the passband to the stopband. For the rectangular window case, the distance (spectral spread) between the two peaks on either side of the transition region approximately equals $4\pi/L_h$, the main lobe width of the rectangular window spectrum. Clearly, increasing the window width decreases the transition width. This result is intuitive because a wider window means that we are accepting more data (closer approximation), which should cause smaller distortion (smaller spectral spreading). Smaller window length (poorer approximation) causes more spectral spreading (more distortion).

The windowed filters also possess passband and stopband ripple. These characteristics are a result of the side lobes of the window spectra. When the ideal impulse response is windowed, its spectrum is convolved with the window's spectrum, and the window's side lobes leak into the passband and stopband. To keep this spectral leakage small, the window side lobes must also be small and decay rapidly with Ω . It is the type of window, not its length, that largely determines side lobe levels and decay rates.

Example 8.11 ▶

▷ Drill 8.6 (Window Method Lowpass Filter Design)

Using rectangular and Hamming windows, design order-98 FIR filters to approximate an audio lowpass filter with cutoff frequency $f_c = 20$ kHz. Plot the magnitude response of each filter. Set the sampling frequency F_s equal to four times the cutoff frequency f_c .

◀

▷ **Drill 8.7 (Triangular Window Lowpass Filter Design)**

Using the window method and a length-9 triangular (Bartlett) window, show that the transfer function of a lowpass FIR filter with $\Omega_c = \pi/2$ is

$$H(z) = -0.0265z^{-1} + 0.2387z^{-3} + 0.5z^{-4} + 0.2387z^{-5} - 0.0265z^{-7}.$$

△

▷ **Example 8.12 (Window Method Differentiator Design)**

Using rectangular and Hamming windows, design length-8 and length-11 FIR filters to realize a digital differentiator. Plot the magnitude response of each filter.

The transfer function of an ideal continuous-time differentiator is $H_c(s) = s$, and its frequency response is $H_c(j\omega) = j\omega$. Restricting our attention to the frequencies of interest to our digital differentiator, the desired impulse response is found from Eq. (8.37) as

$$h[n] = \left(\frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} j\omega e^{j\omega(n-\frac{L_h-1}{2})T} d\omega \right) w[n].$$

Letting $x = (n - \frac{L_h-1}{2})T$, we have

$$\begin{aligned} h[n] &= \left(\frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} j\omega e^{j\omega x} d\omega \right) w[n] \\ &= \begin{cases} \frac{T}{2\pi} \left(\frac{\frac{2\pi x}{T} \cos(\pi x/T) - 2 \sin(\pi x/T)}{x^2} \right) w[n] & x \neq 0 \\ 0 & x = 0 \end{cases}. \end{aligned}$$

For integer n , observe that $\sin(\pi x/T) = \sin(\pi[n - \frac{L_h-1}{2}]) = 0$ when L_h is odd. Similarly, $\cos(\pi x/T) = \cos(\pi[n - \frac{L_h-1}{2}]) = 0$ when L_h is even. Hence, for $n \neq (L_h - 1)/2$,

$$h[n] = \begin{cases} \frac{\cos(\pi[n - \frac{L_h-1}{2}])}{[n - \frac{L_h-1}{2}]T} w[n] & L_h \text{ odd} \\ \frac{-\sin(\pi[n - \frac{L_h-1}{2}])}{\pi[n - \frac{L_h-1}{2}]^2 T} w[n] & L_h \text{ even} \end{cases}. \quad (8.39)$$

When L_h is odd, $h[n] = 0$ at the midpoint $n = (L_h - 1)/2$.

Case of Odd L_h

Let us first consider the case of $L_h = 11$ (L_h odd). To begin, we use MATLAB to compute $Th[n]$ using Eq. (8.39).

```
01 Lh = 11; n = 0:Lh-1; Omega = linspace(-pi,pi,1001);
02 wrec = @ (n) 1.0*((n>=0)&(n<=Lh-1));
03 wham = @ (n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
04 Threc = cos(pi*(n-(Lh-1)/2))./(n-(Lh-1)/2).*wrec(n)
    Threc = 0.200 -0.250 0.333 -0.500 1.000 Inf -1.000 0.500 -0.333 0.250 -0.200
05 Thham = cos(pi*(n-(Lh-1)/2))./(n-(Lh-1)/2).*wham(n)
    Thham = 0.016 -0.042 0.133 -0.341 0.912 Inf -0.912 0.341 -0.133 0.042 -0.016
06 Threc(n==(Lh-1)/2) = 0; Thham(n==(Lh-1)/2) = 0;
```

In each of the two cases, the $n = (L_h - 1)/2 = 5$ point is initially computed incorrectly as ∞ . It is a simple matter to adjust this point to the correct value of 0 (line 06). Since the impulse responses are odd length and antisymmetric, both are type III FIR filters.

Figures 8.27a and 8.27b show the (scaled) impulse responses $Th_{rec}[n]$ and $Th_{ham}[n]$, and Fig. 8.27c compares their corresponding magnitude responses (rectangular solid, Hamming dashed) to an ideal differentiator (shaded). Notice once again that the rectangular window produces oscillatory behavior (Gibbs phenomenon). The tapered Hamming window produces a much smoother response.

```

07 subplot(221); stem(n,Threc); subplot(222); stem(n,Thham);
08 THrec = polyval(Threc,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
09 THham = polyval(Thham,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
10 subplot(212); plot(Omega,abs(THrec),Omega,abs(THham),'--k');

```

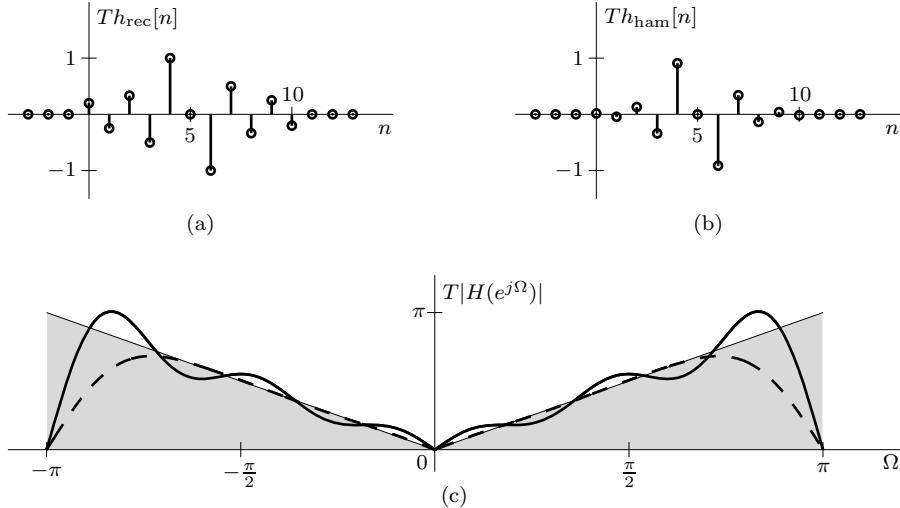


Figure 8.27: $L_h = 11$ digital differentiators: (a) $Th_{rec}[n]$, (b) $Th_{ham}[n]$, and (c) $T|H_{rec}(e^{j\Omega})|$ (solid) and $T|H_{ham}(e^{j\Omega})|$ (dashed).

Since these filters are type III FIR filters, closed-form expressions for the magnitude and phase responses are easily obtained using Table 8.3. Except for occasional jumps of $\pm\pi$ and a linear phase component of -5Ω (corresponding to time delay of $5T$ seconds), both filters exactly match the phase response of an ideal differentiator. Notice that the Hamming-windowed filter is more accurate for low frequencies, but the rectangular-windowed filter operates over a broader frequency range. The taper of the Hamming window effectively shrinks the passband of the differentiator. This is generally true for all tapered windows. To compensate for this shrinkage, we usually start with a passband somewhat larger (typically 25% larger) than the design passband. For example, to ensure that our Hamming-windowed differentiator works in the 20-kHz audio range, we might select T such that

$$2\pi \times 20000 \leq \frac{2\pi}{3T} \quad \Rightarrow \quad T \leq 16.67 \mu\text{s}.$$

Case of Even L_h

We shall now consider the case of $L_h = 8$ (L_h even). Once again, we use MATLAB to compute $Th[n]$ using Eq. (8.39). The (scaled) impulse responses $Th_{rec}[n]$ and $Th_{ham}[n]$ are shown in Figures 8.28a and 8.28b, and the corresponding magnitude responses are shown in Fig. 8.28c.

```

11 Lh = 8; n = 0:Lh-1; Omega = linspace(-pi,pi,1001);
12 wrec = @ (n) 1.0*((n>=0)&(n<=Lh-1));
13 wham = @ (n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
14 Threc = -sin(pi*(n-(Lh-1)/2))./(pi*(n-(Lh-1)/2).^2).*wrec(n)
      Threc = -0.0260 0.0509 -0.1415 1.2732 -1.2732 0.1415 -0.0509 0.0260
15 Thham = -sin(pi*(n-(Lh-1)/2))./(pi*(n-(Lh-1)/2).^2).*wham(n)

```

```

16 Tham = -0.0021 0.0129 -0.0909 1.2152 -1.2152 0.0909 -0.0129 0.0021
16 subplot(221); stem(n,Threc); subplot(222); stem(n,Thham);
17 THrec = polyval(Threc,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
18 THham = polyval(Thham,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
19 subplot(212); plot(Omega,abs(THrec),Omega,abs(THham),'--k');

```

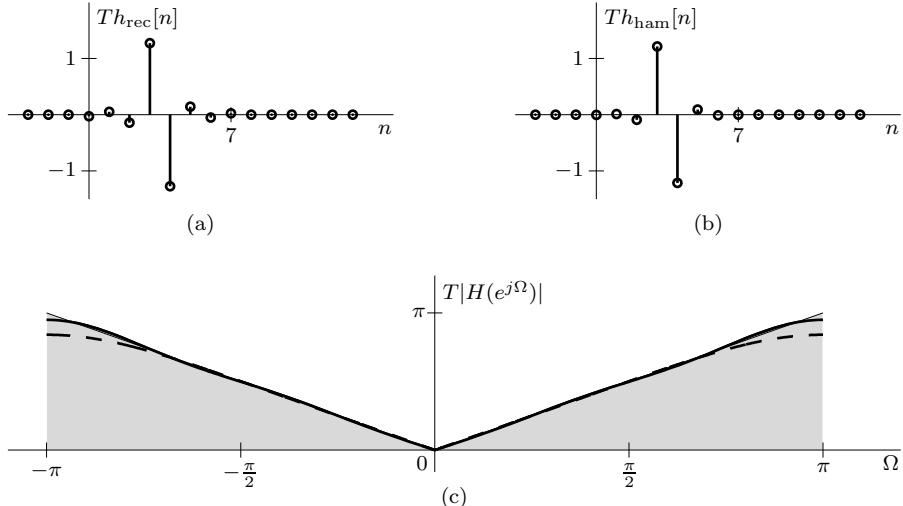


Figure 8.28: $L_h = 8$ digital differentiators: (a) $Th_{rec}[n]$, (b) $Th_{ham}[n]$, and (c) $T|H_{rec}(e^{j\Omega})|$ (solid) and $T|H_{ham}(e^{j\Omega})|$ (dashed).

A Surprising Success

It is interesting to compare the magnitude responses of the even-length filters (Fig. 8.28c) with those of the odd-length filters (Fig. 8.27c). Despite the advantage of being higher order, the $L_h = 11$ differentiators cannot compete with the $L_h = 8$ differentiators. The surprising success of the $L_h = 8$ cases can be explained in two ways.

First, we note that the $L_h = 8$ cases are type IV filters, whereas the $L_h = 11$ cases are type III filters. Consulting Table 8.4, we see that both type III and type IV filters are constrained to have at least one system zero at $z = 1$ (dc). This poses no problem since an ideal differentiator naturally blocks dc. The type III filter, however, has an additional restriction that it must also have at least one system zero at $z = -1$ (high frequency). Since differentiators should pass rather than block high-frequency signals, this restriction degrades the performance of odd-length differentiators.

Another way to view the discrepancy in performance is to look at Eq. (8.39) and the impulse response plots. These show that the impulse responses for even L_h decay rapidly (as $1/L_h^2$) from the center. But for odd L_h , the rate of decay is slower (only as $1/L_h$). Consequently, truncation (windowing) for even L_h does not affect the impulse response as much as the truncation for odd L_h . In other words, the truncated impulse responses for the even case behave very similar to the ideal impulse response. Such is not the case for odd L_h .

Example 8.12

▷ Drill 8.8 (Window Method Differentiator Design)

Using a Hann window, design length-49 and length-50 FIR filters to realize a digital differentiator. Plot the magnitude response of each filter.



▷ **Drill 8.9 (Another Window Method Design)**

Using a rectangular window, design a length-7 FIR filter to realize $H_c(j\omega) = \Lambda(\omega T/\pi)$. Plot the magnitude response of the filter.

△

8.2.5 Window Method FIR Filter Design for Given Specifications

To this point, we have applied the window method to design FIR filters without concern for meeting particular passband ripple, stopband ripple, or transition band requirements. We now consider how to use the window method to design FIR filters to meet given specifications.

For IIR and FIR digital filters, we designate passband edge as Ω_p (or ω_p), stopband edge as Ω_s (or ω_s), and stopband ripple as δ_s . However, FIR filters require a slight modification from IIR filters with regard to passband ripple. For IIR filters, the maximum gain over the passband is unity, and the minimum gain is $1 - \delta_p$ (see Fig. 2.22). The passband gain of FIR filters, in contrast, oscillates about unity between $1 + \frac{\delta_p}{2}$ and $1 - \frac{\delta_p}{2}$, as shown in Fig. 8.29 for the lowpass case. Although both possess δ_p of passband ripple, the reference levels are different.[†]

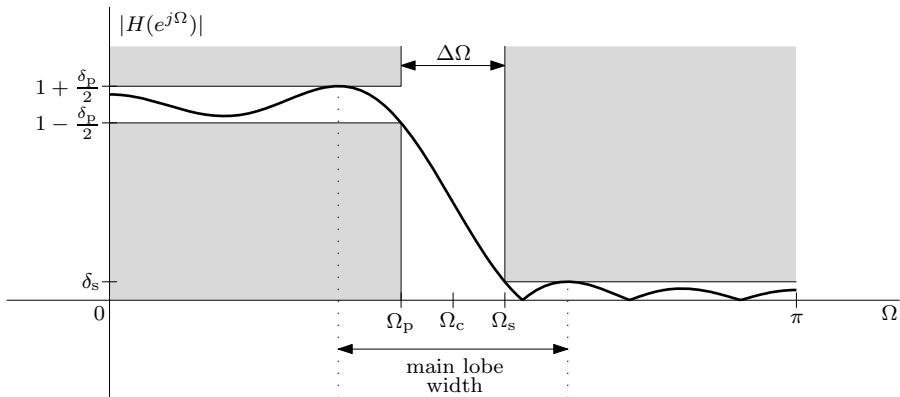


Figure 8.29: Lowpass FIR filter specifications.

As with IIR filters, FIR filter stopband attenuation is expressed in dB as $\alpha_s = -20 \log_{10}(\delta_s)$. Passband ripple, expressed in dB, is defined in terms of the ratio of maximum to minimum passband gains as

$$r_p = 20 \log_{10} \left(\frac{1 + \frac{\delta_p}{2}}{1 - \frac{\delta_p}{2}} \right). \quad (8.40)$$

The passband ripple parameter r_p serves essentially the same purpose as the passband attenuation parameter α_p for IIR filters, which is also the ratio of maximum to minimum passband gains expressed in a dB scale.

The window method causes ripple in the resulting filter response. While the amount of ripple depends on the window type, the filter's stopband ripple equals half its passband ripple, $\delta_s = \frac{\delta_p}{2}$. In other words, passband and stopband ripple cannot be independently specified when using the window method to design an FIR filter. To provide an example, a rectangular window produces a magnitude response $|H_{rec}(j\omega)|$ (such as the one shown in Fig. 8.26) with maximum passband gain of 1.09 ($\frac{\delta_p}{2} = 0.09$ and $r_p = 1.57$ dB) and maximum stopband ripple of $\delta_s = 0.09$ ($\alpha_s = 20.9$ dB).

[†]Similar adjustments in defining passband ripple are needed for highpass, bandpass, and bandstop FIR filters.

Thus, we see that a rectangular window produces ripple of $\frac{\delta_p}{2} = \delta_s = 0.09$. In fact, each window's ripple characteristics are more or less fixed, changing only very little with window length (or filter order).

Recall from Table 8.5 that the main lobe width for a rectangular window is $4\pi/L_h$. The main lobe width is equal to the spectral spread caused by the window. This should not be confused with the transition band $\Delta\Omega = \Omega_s - \Omega_p$, although the two are often very close. For the rectangular window, the main lobe width equals the difference between the frequencies where the passband and stopband gains are maximum. The transition band $\Delta\Omega$, on the other hand, is the difference between Ω_s (the start of the stopband where the gain drops to δ_s) and Ω_p (the end of the passband where the gain drops to $1 - \frac{\delta_p}{2}$). This difference is clearly depicted in Fig. 8.29. Table 8.6 (adapted from [6]) shows some useful windows and their approximate filtering characteristics.

Window	Main Lobe Width	Peak Side Lobe Level [dB]	$\Delta\Omega = \Omega_s - \Omega_p$	$\delta_s, \delta_p/2$	α_s [dB]	r_p [dB]
1. Rectangular	$\frac{4\pi}{L_h}$	-13.3	$\frac{1.84\pi}{L_h-1}$	0.090	20.9	1.57
2. Hann	$\frac{8\pi}{L_h}$	-31.5	$\frac{6.22\pi}{L_h-1}$	0.0064	43.9	0.11
3. Hamming	$\frac{8\pi}{L_h}$	-42.7	$\frac{6.64\pi}{L_h-1}$	0.0019	54.5	0.033
4. Blackman	$\frac{12\pi}{L_h}$	-58.1	$\frac{11.13\pi}{L_h-1}$	0.00017	75.3	0.0030
5. Kaiser			← varies with α →			

Table 8.6: Approximate filtering characteristics of common windows.

Let us now return to our goal of using the window method to design FIR filters to meet given specifications. The design procedure is straightforward. First, we specify an ideal response $H_c(j\omega)$ by setting its cutoff frequency to the middle of the transition band, $\Omega_c = (\Omega_p + \Omega_s)/2$. From this ideal response, we use Eq. (8.35) to determine the ideal impulse response $h_c[n]$. Next, we determine the type and length of window required. From the given passband ripple r_p and minimum stopband attenuation α_s (or δ_p and δ_s), we select from Table 8.6 a window type to satisfy both requirements. Knowing Ω_p and Ω_s , we determine the transition band $\Delta\Omega = \Omega_s - \Omega_p$, which, in turn, allows us to determine the window length L_h , also from Table 8.6. With $h_c[n]$, window type, and window length in hand, the desired filter is designed using the window method of Eq. (8.36). Lastly, since the characteristics in Table 8.6 are approximate, we must verify that design specifications are met. If the designed filter does not meet specifications, we simply adjust window type or length until specifications are met. Let us next demonstrate the procedure with an example.

▷ Example 8.13 (Window Method FIR LPF Design for Given Specifications)

Using the window method, design a FIR lowpass filter with cutoff frequency $f_c = 20$ kHz, passband frequency $f_p \geq 19$ kHz, passband ripple $r_p \leq 0.1$ dB, stopband frequency $f_s \leq 21$ kHz, and stopband attenuation $\alpha_s \geq 50$ dB. Plot the magnitude response of the filter using a dB scale. Set the sampling frequency to be four times the cutoff frequency, $F_s = 4f_c = 80$ kHz.

From Table 8.6, it follows that the Hamming window can satisfy the specifications because it has $r_p = 0.033$ and $\alpha_s = 54.5$. Using $\Omega = \omega T$, we determine that

$$\Omega_c = \frac{2\pi(20000)}{4(20000)} = \frac{\pi}{2}, \quad \Omega_p = \frac{19\pi}{40}, \quad \text{and} \quad \Omega_s = \frac{21\pi}{40}.$$

Thus, $\Delta\Omega = \Omega_s - \Omega_p = 2\pi/40 = 0.05\pi$. Consulting Table 8.6 for the Hamming window,

$$\Delta\Omega = 0.05\pi = \frac{6.64\pi}{L_h - 1} \implies L_h = 133.8.$$

We take $L_h = 134$. As in Ex. 8.11, $\omega_c = \pi/2T$ and $h_c(t) = \frac{1}{2T} \text{sinc}(t/2T)$. Due to the relatively high order of this design, the necessary design computations are done using MATLAB. The resulting impulse and magnitude responses are shown in Fig. 8.30. Careful inspection of the magnitude response verifies that all design specifications are met.

```

01 fc = 20000; fp = 19000; fs = 21000; Fs = 4*fc; T = 1/Fs;
02 Omegac = 2*pi*fc*T; Omegap = 2*pi*fp*T; Omegas = 2*pi*fs*T;
03 DeltaOmega = Omegas-Omegap; Lh = ceil(6.64*pi/DeltaOmega +1)
Lh = 134
04 n = 0:Lh-1; Omega = linspace(0,pi,10001);
05 wham = @(n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
06 hc = @(t) 1/(2*T)*sinc(t/(2*T)); h = T*hc(n*T-(Lh-1)/2*T).*wham(n);
07 subplot(211); stem(n,h);
08 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
09 subplot(212); plot(Omega,20*log10(abs(H)));

```

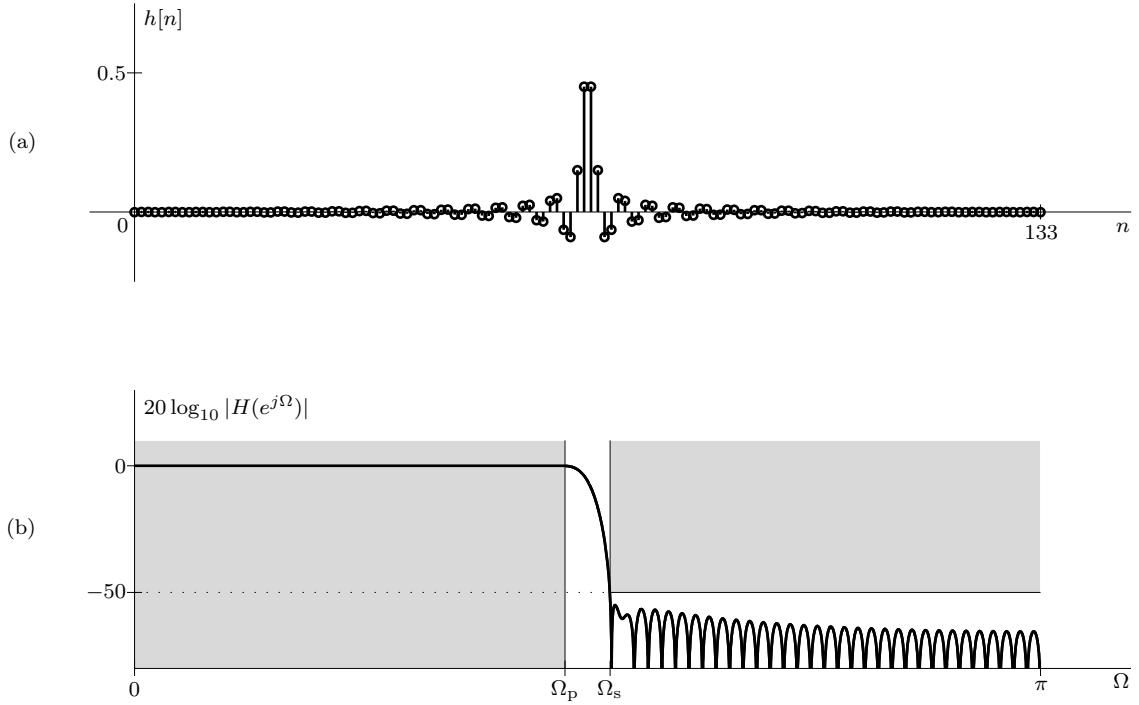


Figure 8.30: Length-134 Hamming LPF: (a) impulse response and (b) magnitude response.

Example 8.13 \triangleleft

▷ **Drill 8.10 (Choosing between Small and Large Ripple Windows)**

Explain why a Hamming window is preferred over a Blackman window for the filter design in Ex. 8.13, even though a Blackman window more easily meets the given ripple requirements.

△

Filter Design Using the Kaiser Window

One deficiency of the window method is that passband and stopband ripple cannot be independently controlled. For typical designs, either the passband or stopband ripple is more restrictive than the other. By meeting the more stringent ripple requirement, the other ripple requirement is exceeded. This tends to increase filter order more than is truly necessary. The problem is actually worse for windows with fixed-ripple characteristics, since *both* ripple requirements are generally exceeded, and the filter order becomes higher still. In Ex. 8.13, for example, the Hamming window produces $r_p = 0.033$ and $\alpha_s = 54.5$ dB, both of which exceed the respective design requirements of 0.1 and 50.

While it still does not allow passband and stopband ripple to be independently controlled, the adjustable Kaiser window can be shaped so that the resulting filter very closely meets the more stringent ripple requirement. In this way, the Kaiser window tends to produce a lower-order filter than does a window with fixed ripple characteristics. To realize this advantage, we need to know how to set the Kaiser window parameter α . Kaiser found an approximate relationship between α and α_s as [2]

$$\alpha = \begin{cases} 0 & \alpha_s < 21 \\ 0.5842(\alpha_s - 21)^{0.4} + 0.0788(\alpha_s - 21) & 21 \leq \alpha_s \leq 50 \\ 0.1102(\alpha_s - 8.7) & \alpha_s > 50 \end{cases}. \quad (8.41)$$

Kaiser also found that the filter length L_h needed to satisfy the specifications α_s and $\Delta\Omega$ is approximately given by

$$L_h = 1 + \frac{\alpha_s - 8}{2.285\Delta\Omega}. \quad (8.42)$$

Equations (8.41) and (8.42) are written in terms of α_s rather than r_p , assuming that α_s is more restrictive than r_p . However, should the passband ripple be more restrictive than the stopband ripple, it is a simple matter to express Eqs. (8.41) and (8.42) in terms of r_p rather than α_s (see Prob. 8.2-21).

▷ **Example 8.14 (Lowpass Filter Design Using the Kaiser Window)**

Redo Ex. 8.13 using the Kaiser window.

Using Eqs. (8.41) and (8.42), MATLAB computes the Kaiser parameter α as well as the necessary filter order L_h .

```
01 alphas = 50; alpha = 0.5842*(alphas-21)^(0.4)+0.0788*(alphas-21)
    alpha = 4.5318
02 fc = 20000; fp = 19000; fs = 21000; Fs = 4*fc; T = 1/Fs;
03 Omegac = 2*pi*fc*T; Omegap = 2*pi*fp*T; Omegas = 2*pi*fs*T;
04 DeltaOmega = Omegas-Omegap; Lh = ceil(1+(alphas-8)/(2.285*DeltaOmega))
    Lh = 119
```

Defining the Kaiser window $w[n]$ as in Table 8.5, we next design the Kaiser filter using Eq. (8.36). The resulting impulse and dB magnitude response are shown in Fig. 8.31.

```
05 n = 0:Lh-1; Omega = linspace(0,pi,10001);
06 wkai = @(n) besseli(0,alpha*sqrt(1-((2*n-(Lh-1))/(Lh-1)).^2))...
```

```

07      besseli(0,alpha).*((n>=0)&(n<=Lh-1));
08 hc = @t) 1/(2*T)*sinc(t/(2*T)); h = T*hc(n*T-(Lh-1)/2*T).*wkai(n);
09 subplot(211); stem(n,h);
10 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
11 subplot(212); plot(Omega,20*log10(abs(H)));

```

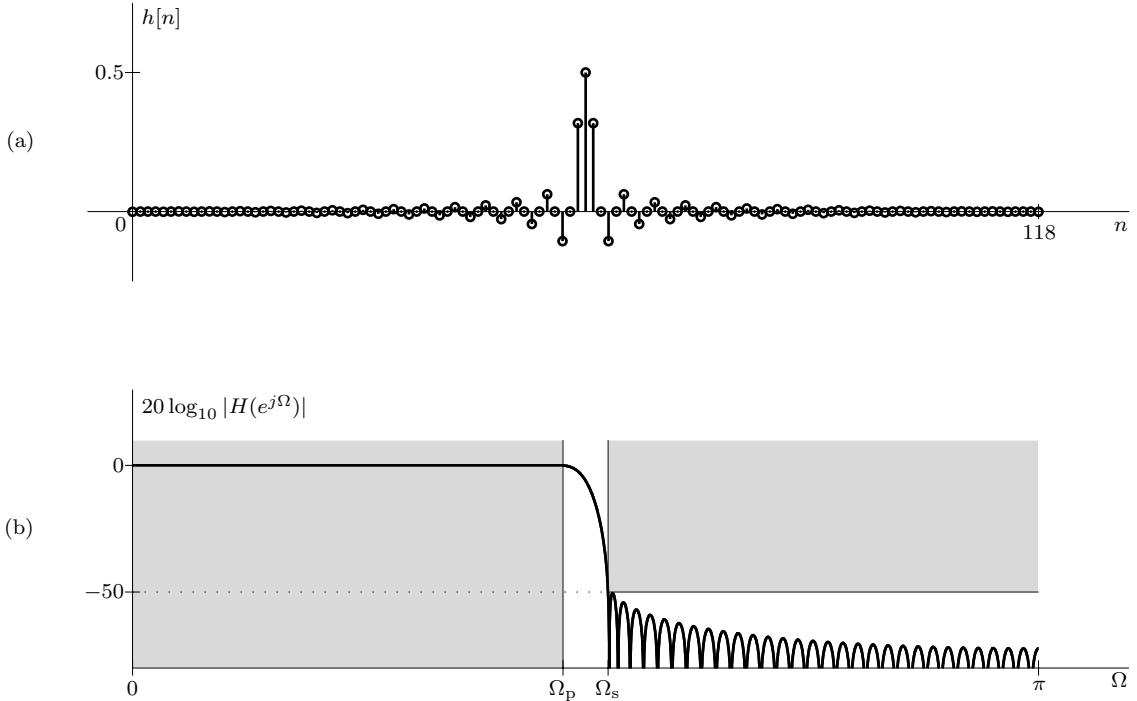


Figure 8.31: Length-119 Kaiser LPF: (a) impulse response and (b) magnitude response.

Compared with the length-134 Hamming filter in Ex. 8.13, the length-119 Kaiser window is about 12% shorter yet meets the same specifications. The principal reason for the shorter length is that the Kaiser filter more closely meets stopband ripple requirements (Fig. 8.13b), whereas the Hamming filter unnecessarily exceeds stopband ripple requirements (Fig. 8.30b).

Example 8.14 ▶

Window Method for Highpass, Bandpass, and Bandstop FIR Filters

Although we have discussed the window method using the example of lowpass filters, it can also be used to truncate the impulse responses of other types of filters. In dealing with filters that have piecewise-constant magnitude responses, the approximate relationships in Table 8.6 and Eqs. (8.41) and (8.42) apply regardless of the type of filter (lowpass, highpass, bandpass, bandstop). This is because the leakage and smearing distortions caused by windowing are almost identical for each case. The spectrum of a windowed lowpass filter is centered at $\Omega = 0$ and obtained by convolving $H_{lp}(e^{j\Omega})$ with $W(e^{j\Omega})$. The same is true for the bandpass case, except that the spectrum $H_{bp}(e^{j\Omega})$ has two blocks (passbands), each identical to $H_{lp}(e^{j\Omega})$ but centered at $\pm\Omega_{ctr}$. Hence, the spectrum for the windowed bandpass filter is identical to that for windowed lowpass case, except that there are two blocks rather than one. All the blocks have (approximately) the same values of r_p and α_s . Similar arguments apply for the cases of highpass and bandstop filters.

▷ **Example 8.15 (Window Method Bandpass Filter Design)**

Using the window method, design a digital bandpass filter with $r_p = 0.1$ dB maximum passband ripple for $1000 \leq \omega \leq 2000$ and $\alpha_s = 50$ dB minimum stopband attenuation over both $0 \leq \omega \leq 450$ and $\omega \geq 4000$. Take $T = \pi/10000$.

In this design, the critical analog frequencies are $\omega_{s_1} = 450$, $\omega_{p_1} = 1000$, $\omega_{p_2} = 2000$, and $\omega_{s_2} = 4000$. While many choices are possible to satisfy the design constraints, let us take the ideal CT BPF to have cutoff frequencies set midway between passband and stopband edges, that is,

$$\omega_{c_1} = \frac{\omega_{s_1} + \omega_{p_1}}{2} = 725 \quad \text{and} \quad \omega_{c_2} = \frac{\omega_{s_2} + \omega_{p_2}}{2} = 3000.$$

Thus, the ideal BPF has center frequency ω_{ctr} and bandwidth $2B$ given as

$$\omega_{ctr} = \frac{\omega_{c_1} + \omega_{c_2}}{2} = 1862.5 \quad \text{and} \quad 2B = \omega_{c_2} - \omega_{c_1} = 2275.$$

The ideal BPF frequency response is

$$H_c(j\omega) = \Pi\left(\frac{\omega - \omega_{ctr}}{2B}\right) + \Pi\left(\frac{\omega + \omega_{ctr}}{2B}\right) = \Pi\left(\frac{\omega - 1862.5}{2275}\right) + \Pi\left(\frac{\omega + 1862.5}{2275}\right).$$

Using the modulation property (Eq. (1.90)), the impulse response of this filter is

$$h_c(t) = \frac{2B}{\pi} \operatorname{sinc}(Bt/\pi) \cos(\omega_{ctr}t) = \frac{2275}{\pi} \operatorname{sinc}(1137.5t/\pi) \cos(1862.5t).$$

The lower transition band of $H_c(j\omega)$ has width $\Delta\omega_1 = \omega_{p_1} - \omega_{s_1} = 550$, whereas the upper transition band has width $\Delta\omega_2 = \omega_{s_2} - \omega_{p_2} = 2000$. Since it is more narrow, the lower transition band $\Delta\omega_1 = 550$ is the more restrictive.

Recalling that $\Omega = \omega T$, the digital filter has transition band $\Delta\Omega = \Delta\omega T = \frac{55\pi}{1000}$. Given $r_p = 0.1$ and $\alpha_s = 50$, we find from Table 8.6 that a Hamming window should be adequate. Further, the Kaiser window, being a window for all occasions, can also be used. We shall consider both.

Hamming Window

To simplify the process, we use MATLAB to perform all calculations. First, we define the design parameters and the ideal CT BPF impulse response.

```

01 rp = 0.1; alphas = 50; T = pi/10000;
02 omegas1 = 450; omegap1 = 1000; omegap2 = 2000; omegas2 = 4000;
03 omegac1 = (omegas1+omegap1)/2; omegac2 = (omegas2+omegap2)/2;
04 B = (omegac2-omegac1)/2; omegatr = (omegac1+omegac2)/2;
05 hc = @(t) 2*B/pi*sinc(B*t/pi).*cos(omegatr*t);
06 DeltaOmega = min([omegap1-omegas1, omegas2-omegap2])*T;
```

Referring to Table 8.6, we next compute L_h from $\Delta\Omega$. Then, defining the Hamming window as in Table 8.5, we use Eq. (8.36) to compute the desired impulse response $h[n]$ (Fig. 8.32a), from which it is a simple matter to compute the corresponding magnitude response (Fig. 8.32b).

```

07 Lh = ceil(6.64*pi/DeltaOmega+1)
Lh = 122
08 wham = @(n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
09 n = 0:Lh-1; h = T*hc(n*T-(Lh-1)/2*T).*wham(n);
10 subplot(211); stem(n,h);
11 Omega = linspace(0,pi,10001); H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
12 subplot(212); plot(Omega/T,20*log10(abs(H)));
```

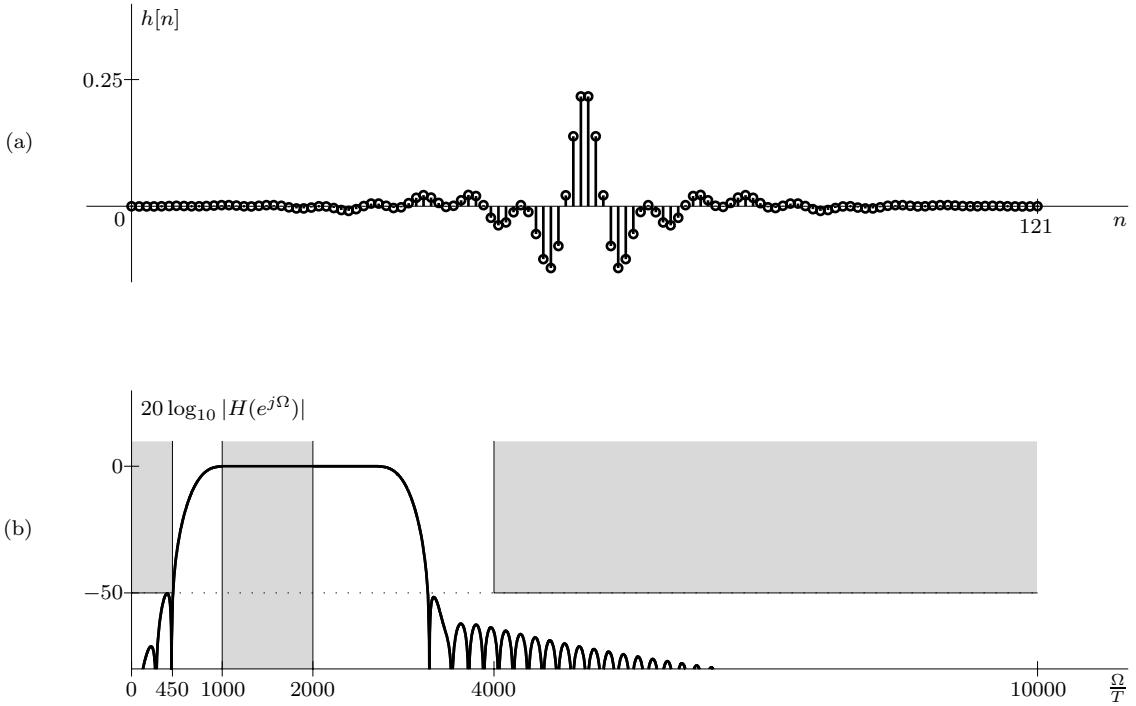


Figure 8.32: Length-122 Hamming BPF: (a) impulse response and (b) magnitude response.

Although Table 8.6 suggests that the Hamming window should produce 54.5 dB of stopband attenuation, we see from Fig. 8.32b that, in this design, the lower stopband barely achieves the required 50 dB of attenuation. We once again emphasize the approximate nature of the window filtering characteristics of Table 8.6 as well as the need to verify every design to ensure specifications are met.

Kaiser Window

Using Eqs. (8.41) and (8.42), we compute the Kaiser window parameter α and length L_h .

```

13 alpha = 0.5842*(alphas-21)^(0.4)+0.0788*(alphas-21)
    alpha = 4.5318
14 Lh = ceil(1+(alphas-8)/(2.285*DeltaOmega))
    Lh = 108

```

Defining the Kaiser window as in Table 8.5, we use Eq. (8.36) to compute the desired impulse response $h[n]$ and then the magnitude response $|H(e^{j\Omega})|$.

```

15 wkai = @(n) besseli(0,alpha*sqrt(1-((2*n-(Lh-1))/(Lh-1)).^2))...
16     besseli(0,alpha).*(n>=0)&(n<=Lh-1);
17 n = 0:Lh-1; h = T*hc(n*T-(Lh-1)/2*T).*wkai(n);
18 subplot(211); stem(n,h);
19 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
20 subplot(212); plot(Omega/T,20*log10(abs(H)));

```

Unfortunately, inspecting the magnitude response, we find that the resulting FIR filter does not quite meet specifications. Increasing the length by 1 to $L_h = 109$ and again executing lines 15 through 20 produces the impulse response of Fig. 8.33a and magnitude response of Fig. 8.33b. In this case, all design specifications are met. Due to its adjustable parameter α , the Kaiser filter is

shorter ($L_h = 109$) than the corresponding Hamming filter ($L_h = 122$). Both possess very similar impulse responses that require high-precision digital implementations to preserve desired filter behavior.

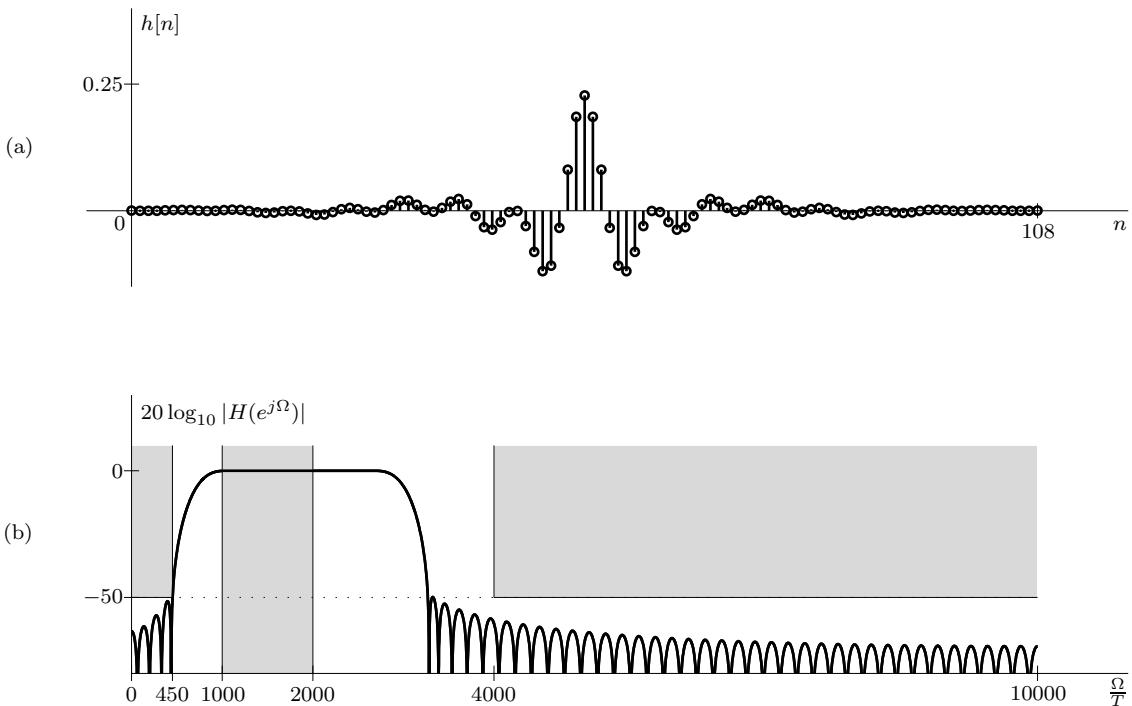


Figure 8.33: Length-109 Kaiser BPF: (a) impulse response and (b) magnitude response.

Example 8.15 □

▷ Drill 8.11 (A Guaranteed Failure of the Window Method)

Suppose that the specifications for a real highpass filter suggest using a Hamming window of length $L_h = 20$. Explain why the resulting FIR filter is *virtually guaranteed* not to meet specifications. Would it help to double the filter length? Explain.

□

Final Comments on the Window Method of FIR Filter Design

When using the window method, passband ripple δ_p (r_p) and stopband ripple δ_s (α_s) cannot be specified independently. Instead, stopband ripple is approximately half as large as passband ripple, $\delta_s = \frac{\delta_p}{2}$. When a design calls for two independent values, we select the more stringent of the two ripple specifications. It is also difficult to specify precisely the band edges and maximum ripple sizes. For all these reasons, the window method is suboptimal, and all designs must be verified to ensure that design specifications are met. Later in this chapter we shall investigate more optimal methods such as frequency-weighted least squares and, briefly, equiripple FIR filter design [8].

8.2.6 Frequency-Domain Methods of FIR Filter Design

It is more natural to describe filters in the frequency domain than in the time domain. For example, it is easier to speak of a lowpass filter (a frequency-domain description) than a sinc response (a time-domain description). In fact, the elementary descriptions of filters as lowpass, highpass, bandpass, and bandstop are all intrinsically frequency-domain descriptions. Not surprisingly, frequency-domain methods of FIR filter design are generally more intuitive than their time-domain counterparts.

Conceptually, an analog filter $H_c(s)$ can be realized by digital means, as depicted in Fig. 6.21a, if we select $h[n] = Th_c(nT)$ (Eq. (6.55)). This time-domain criterion is the impulse invariance method of digital filter design. The equivalent frequency-domain criterion is to cause $H(e^{j\Omega}) = H_c(j\Omega/T)$ over $|\Omega| \leq \pi$ (Eq. (6.54)). Since $\Omega = \omega T$, this is equivalently stated as $H(e^{j\omega T}) = H_c(j\omega)$ over $|\omega| \leq \pi/T$. These criteria typically lead to noncausal and infinite-length impulse responses.

For a length- L_h FIR filter, we have only L_h degrees of freedom. Thus, as we saw in the preceding section, the Fourier series method forces $h[n] = Th_c(nT)$ only at L_h values of n .[†] Similarly, the *frequency sampling method*, also called the *spectral sampling method*, forces $H(e^{j\Omega}) = H_c(j\Omega/T)$ only at L_h frequencies. To cover the 2π spectral width of $H_c(j\Omega/T)$, we typically choose these frequencies to be uniformly spaced $\Omega_0 = \frac{2\pi}{L_h}$ apart; that is,

$$\Omega_0 = \frac{2\pi}{L_h} \quad \text{and, equivalently,} \quad \omega_0 = \frac{2\pi}{L_h T}. \quad (8.43)$$

Since $H(e^{j\Omega})$ is 2π -periodic, we can proceed using any 2π interval of Ω . There are advantages, however, to using the interval $0 \leq \Omega < 2\pi$. Over all frequencies Ω , $H(e^{j\Omega})$ equals the 2π -periodic replication of $H_c(j\Omega/T)$. Thus, over $0 \leq \Omega < 2\pi$, the frequency sampling method requires that

$$H[k] \equiv H(e^{jk\Omega_0}) = \begin{cases} H_c(jk\Omega_0/T) & 0 \leq k \leq \lceil \frac{L_h-1}{2} \rceil \quad (0 \leq k\Omega_0 \leq \pi) \\ H_c(j\frac{k\Omega_0-2\pi}{T}) & \lceil \frac{L_h-1}{2} \rceil + 1 \leq k \leq L_h - 1 \quad (\pi < k\Omega_0 < 2\pi) \end{cases}. \quad (8.44)$$

In other words, Eq. (8.44) seeks to exactly match the frequency responses of the digital and (periodically replicated) analog filters at the L_h frequencies $\Omega = k\Omega_0$ ($0 \leq k \leq L_h - 1$). To determine $H(e^{j\Omega})$ over $\pi < \Omega < 2\pi$, Eq. (8.44) copies (replicates) $H_c(j\Omega/T)$ from $-\pi < \Omega < 0$. For real filters, this causes $H[k]$ to possess conjugate symmetry about the (possibly non-integer) point $k = \frac{L_h}{2}$, which is to say that $H[k] = H^*[L_h - k]$.

Now, to determine our DT FIR filter's impulse response $h[n]$, recall that

$$H(e^{j\Omega}) = \sum_{n=0}^{L_h-1} h[n] e^{-j n \Omega}$$

so that

$$H(e^{jk\Omega_0}) = H[k] = \sum_{n=0}^{L_h-1} h[n] e^{-j n k \Omega_0}, \quad k = 0, 1, 2, \dots, L_h - 1. \quad (8.45)$$

These L_h linear simultaneous equations in L_h unknowns $h[0], h[1], h[2], \dots, h[L_h - 1]$ can be solved in the usual manner to find $h[n]$. This same problem is solved in the next chapter. The solution is given by (see Eq. (9.5))

$$\begin{aligned} h[n] &= \frac{1}{L_h} \sum_{k=0}^{L_h-1} H(e^{jk\Omega_0}) e^{j n k \Omega_0} \\ &= \frac{1}{L_h} \sum_{k=0}^{L_h-1} H[k] e^{j \frac{2\pi n k}{L_h}}, \quad n = 0, 1, 2, \dots, L_h - 1. \end{aligned} \quad (8.46)$$

[†]Actually, causality requires the Fourier series method to force equality of the shifted impulse response, $h[n] = Th_c([n - \frac{L_h-1}{2}]T)$, for $0 \leq n \leq L_h - 1$.

Together, Eqs. (8.45) and (8.46) form the well-known discrete Fourier transform (DFT) pair. We call $H[k] = H(e^{jk\Omega_0})$ the direct DFT of $h[n]$, and $h[n]$ is the inverse DFT (IDFT) of $H[k]$. The *fast Fourier transform* (FFT) and *inverse fast Fourier transform* (IFFT), which are efficient algorithms to compute the DFT and IDFT, make this method of design especially simple.

▷ **Example 8.16 (Frequency Sampling Method Ideal Lowpass Filter Design)**

Using the frequency sampling method, design an ideal lowpass FIR filter with cutoff frequency $\Omega_c = \frac{\pi}{2}$ and length $L_h = 7$.

MATLAB makes short work of this design.

```

01 Lh = 7; Omega0 = 2*pi/Lh; k = 0:Lh-1; n = 0:Lh-1; Omega = linspace(-pi,2*pi,1001);
02 Hk = 1.0*(k*Omega0<=pi/2); Hk(1,fix(Lh/2)+2:end) = conj(Hk(1,round(Lh/2):-1:2))
    Hk = 1 1 0 0 0 0 1
03 h = ifft(Hk)
    h = 0.4286 0.3210 0.0793 -0.1146 -0.1146 0.0793 0.3210
04 subplot(211); stem(n,h);
05 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
06 subplot(212); plot(Omega,abs(H),k*Omega0,abs(Hk),'ko');
```

Line 01 establishes the basic design parameters. Although we can theoretically use Eq. (8.44) to compute $H[k]$, computer round-off sometimes causes seemingly minor errors that destroy the conjugate symmetry required of $H[k]$. Thus, line 02 determines $H[k]$ using $H_c(jk\Omega_0/T)$ over $0 \leq k\Omega_0 \leq \pi$ and then computes the remaining values to ensure that $H[k]$ is conjugate symmetric about $k = L_h/2 = 3.5$ ($k\Omega_0 = \pi$). Rather than compute $h[n]$ by direct evaluation of Eq. (8.46), it is more simple to use MATLAB's built-in inverse fast Fourier transform command, `ifft`. Figure 8.34 shows the resulting impulse and magnitude responses.

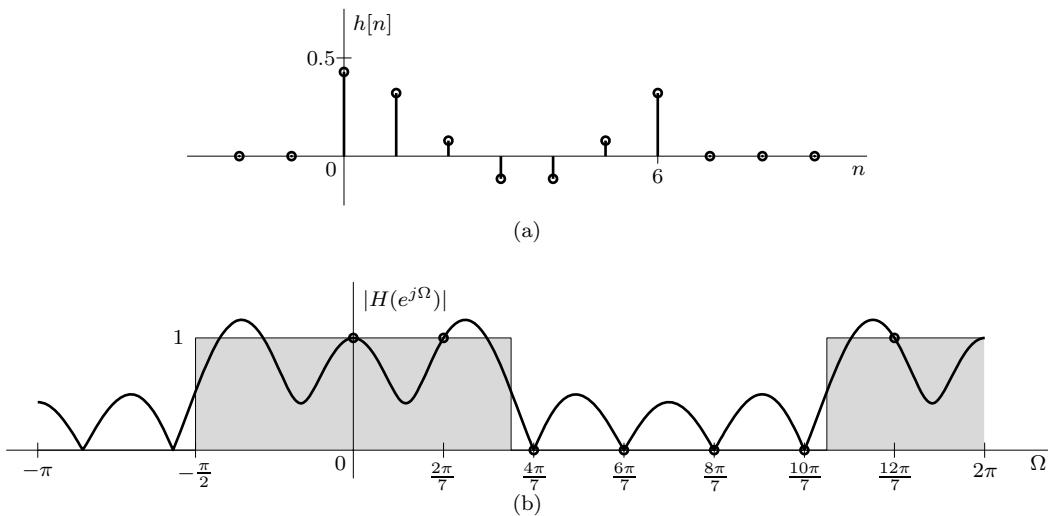


Figure 8.34: Frequency sampling method ideal LPF: (a) impulse response and (b) magnitude response.

While $|H(e^{j\Omega})|$ exactly matches the ideal response at the seven frequencies $\Omega = k2\pi/7$ ($0 \leq k \leq 6$), we see from Fig. 8.34b that the magnitude response is an overall poor approximation of the ideal (shaded). The culprit is the underlying 0-phase nature of the ideal response. In essence, we are trying to obtain a good magnitude response without any filter delay. As discussed in Sec. 2.3, our causal (realizable) filter cannot meet such unreasonable expectations. We can also predict this poor

filter behavior based on $h[n]$. Instead of being sinc-like, the impulse response of Fig. 8.34a is highly distorted with the bulk of its energy at the edges rather than the middle; these abrupt edges cause severe spectral leakage, which explains the significant passband and stopband ripple of the filter. As we shall next see, we can greatly improve the magnitude response by modifying the spectral sampling method to deliver a linear phase filter.

Example 8.16 □

Linear Phase (Constant-Delay) Filters

Ideally, we desire $H(e^{j\Omega}) = H_c(j\Omega/T)$ for $|\Omega| \leq \pi$. A filter designed using Eqs. (8.44) and (8.46) satisfies this condition only at L_h uniformly spaced values of Ω . Between samples, the frequency response, both magnitude and phase, can deviate considerably. As shown in Ex. 8.16, such deviations are quite large when trying to force a causal FIR filter to have a zero phase response. Better results are obtained by modifying the frequency sampling method to encourage a linear phase response.

Under usual circumstances, the ideal response $H_c(j\Omega/T)$ has zero (or constant $\pm\frac{\pi}{2}$) phase response. The corresponding impulse response $h_c(t)$ is generally centered at $t = 0$. To best realize this noncausal filter with a length- L_h FIR filter, we want to delay $h_c(t)$ by $T(L_h - 1)/2$, exactly as done in the window method. Such a delay amounts to multiplying $H_c(j\Omega/T)$ by $e^{-j\frac{L_h-1}{2}\Omega}$. This delay does not alter the filter magnitude response but changes the phase response by $-\Omega(L_h - 1)/2$ (a linear function of Ω). In this way, we encourage, and often achieve, a linear phase result.

Now, when $\Omega = k\Omega_0 = k2\pi/L_h$, $H_c(j\Omega/T)e^{-j\frac{L_h-1}{2}\Omega}$ becomes

$$H_c(jk\Omega_0/T)e^{-jk\pi\frac{L_h-1}{L_h}}.$$

Similarly, the shifted version $H_c(j(\Omega - 2\pi)/T)e^{-j\frac{L_h-1}{2}(\Omega - 2\pi)}$ becomes

$$H_c\left(j\frac{k\Omega_0 - 2\pi}{T}\right)e^{-jk\pi(L_h-1)(\frac{k}{L_h}-1)} = H_c\left(j\frac{k\Omega_0 - 2\pi}{T}\right)e^{-jk\pi\frac{L_h-1}{L_h}}(-1)^{L_h-1}.$$

Using these results, Eq. (8.44) thus becomes

$$H[k] = \begin{cases} H_c(jk\Omega_0/T)e^{-jk\pi\frac{L_h-1}{L_h}} & 0 \leq k \leq \lceil \frac{L_h-1}{2} \rceil \\ H_c\left(j\frac{k\Omega_0 - 2\pi}{T}\right)e^{-jk\pi\frac{L_h-1}{L_h}}(-1)^{L_h-1} & \lceil \frac{L_h-1}{2} \rceil + 1 \leq k \leq L_h - 1 \end{cases}. \quad (8.47)$$

For real filters, Eq. (8.47) can also be computed as

$$H[k] = \begin{cases} H_c(jk\Omega_0/T)e^{-jk\pi\frac{L_h-1}{L_h}} & 0 \leq k \leq \lceil \frac{L_h-1}{2} \rceil \\ H^*[L_h - k] & \lceil \frac{L_h-1}{2} \rceil + 1 \leq k \leq L_h - 1 \end{cases}. \quad (8.48)$$

When there is a choice, Eq. (8.48) is generally preferred over Eq. (8.47) since the resulting $H[k]$ is guaranteed to be conjugate symmetric, even in the presence of computer round-off errors. Let us now demonstrate the approach and its advantages by redesigning the filter of Ex. 8.16.

▷ Example 8.17 (Frequency Sampling Method Linear Phase Lowpass Filter Design)

Using the frequency sampling method, repeat Ex. 8.16 and design a linear phase lowpass FIR filter with cutoff frequency $\Omega_c = \frac{\pi}{2}$ and length $L_h = 7$.

In this case, we use Eq. (8.48) rather than Eq. (8.44), as used in Ex. 8.16. Over $0 \leq k\Omega_0 \leq \pi$, the only difference between these equations is that Eq. (8.47) includes the extra term $e^{-jk\pi\frac{L_h-1}{L_h}}$.

```

01 Lh = 7; Omega0 = 2*pi/Lh; k = 0:Lh-1; n = 0:Lh-1; Omega = linspace(-pi,2*pi,1001);
02 Hk = 1.0*(k*Omega0<=pi/2).*exp(-1j*k*pi*(Lh-1)/Lh);
03 Hk(1,fix(Lh/2)+2:end) = conj(Hk(1,round(Lh/2):-1:2))
Hk = 1.0000 -0.9010-0.4339i 0 0 0 0 -0.9010+0.4339i
04 h = ifft(Hk)
h = -0.1146 0.0793 0.3210 0.4286 0.3210 0.0793 -0.1146
05 subplot(211); stem(n,h);
06 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
07 subplot(212); plot(Omega,abs(H),k*Omega0,abs(Hk),'ko');

```

As line 03 demands, $H[k]$ is conjugate symmetric about $k = L_h/2 = 3.5$. The impulse response $h[n]$, shown in Fig. 8.35a, is more sinc-like than that in Ex. 8.16 and has the bulk of its energy in the middle rather than the edges. The natural taper of $h[n]$ helps ensures a relatively smooth magnitude response, as shown in Fig. 8.35b. Although both exactly match the ideal response at the seven frequencies $\Omega = k2\pi/7$ ($0 \leq k \leq 6$), the linear phase filter more closely approximates the ideal (shaded) than the Fig. 8.34b response of Ex. 8.16. Clearly, allowing a little filter delay greatly improves the filter's magnitude response.

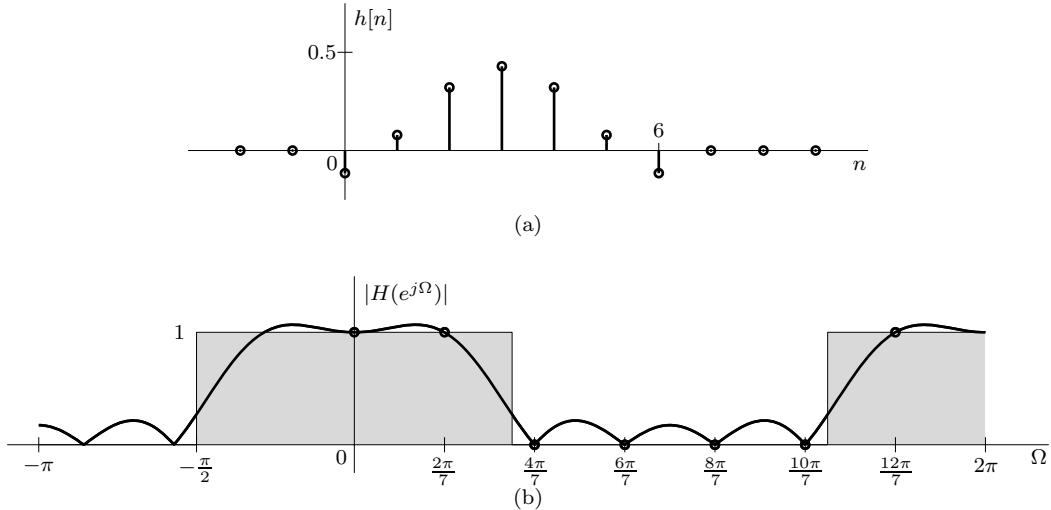


Figure 8.35: Frequency sampling method linear phase LPF: (a) impulse response and (b) magnitude response.

Before concluding this example, it is worth comparing this filter with that obtained using the Fourier series (rectangular window) method (see Figs. 8.25c and 8.26). The impulse responses of both filters are very similar, as are the magnitude responses. What, then, is the difference in these two filters? The Fourier series method optimizes the design with respect to all frequencies. It minimizes the mean square error between the desired and realized frequency responses. The frequency sampling method, in contrast, realizes a filter whose frequency response exactly matches the desired frequency response at L_h uniformly spaced frequencies. The mean square error in this design will generally be higher than that in the Fourier series (rectangular window) method.

Example 8.17 \triangleleft

An Alternate Approach to the Frequency Sampling Method

For each of the four linear phase filter types, Table 8.3 expresses the amplitude response $H_a(e^{j\Omega})$ in terms of the impulse response $h[n]$. These equations form the basis of a frequency sampling approach

to design FIR filters that are guaranteed to have linear phase. All that needs to be done is to set $H_a(e^{j\Omega})$ equal to a desired response $H_c(j\Omega/T)$ at a suitable number of frequencies Ω_k and then solve the resulting linear system of equations.

To provide an example, consider a type I filter. From Table 8.3, we see that

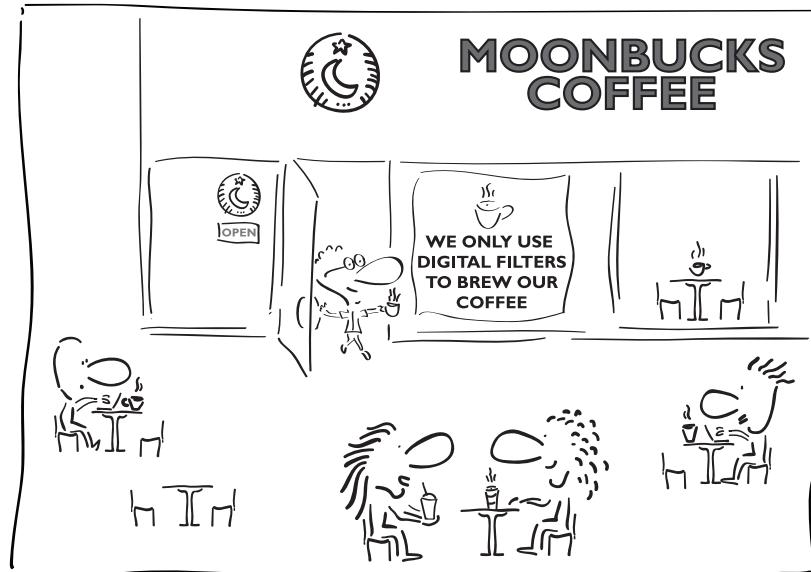
$$H_a(\Omega_k) = \sum_{n=0}^{\frac{L_h-1}{2}} a_{k,n} h[n], \quad \text{where } a_{k,n} = \begin{cases} 2 \cos(\Omega_k [\frac{L_h-1}{2} - n]) & n = 0, \dots, \frac{L_h-3}{2} \\ 1 & n = \frac{L_h-1}{2} \end{cases}. \quad (8.49)$$

For $k = 0, 1, \dots, \frac{L_h-1}{2}$, we get a set of $\frac{L_h+1}{2}$ linear equations in $\frac{L_h+1}{2}$ unknown values of $h[n]$ (the remaining values are known from symmetry). Expressed in matrix form and letting $H_c(j\Omega_k/T) = H_a(\Omega_k)$, the solution of this system of equations is

$$\underbrace{\begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[\frac{L_h-1}{2}] \end{bmatrix}}_h = \underbrace{\begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,\frac{L_h-1}{2}} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,\frac{L_h-1}{2}} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\frac{L_h-1}{2},0} & a_{\frac{L_h-1}{2},1} & \cdots & a_{\frac{L_h-1}{2},\frac{L_h-1}{2}} \end{bmatrix}}_{A^{-1}}^{-1} \underbrace{\begin{bmatrix} H_c(j\Omega_0/T) \\ H_c(j\Omega_1/T) \\ \vdots \\ H_c(j\Omega_{\frac{L_h-1}{2}}/T) \end{bmatrix}}_H. \quad (8.50)$$

Expressions for type II, III, and IV filters are similar.

There are several convenient features of this alternate approach of frequency sampling as compared with using Eqs. (8.46) and (8.47). To begin, when $\Omega_k = k\Omega_0 = k2\pi/L_h$, all values of $h[n]$ can be found using $H_c(j\Omega/T)$ over the simple range $0 \leq \Omega \leq \pi$; there is no need to hassle with the shifted replicate $H_c(j(\Omega - 2\pi)/T)$ as in Eq. (8.47) or the conjugate symmetry of Eq. (8.48). Additionally, it is somewhat simpler with this method to accommodate nonuniformly spaced frequencies Ω_k , although such spacings are not commonly used.



▷ **Example 8.18 (Alternate Frequency Sampling Method Type I LPF Design)**

Using the frequency sampling method of Eqs. (8.49) and (8.50), repeat Ex. 8.17 and design a type I lowpass FIR filter with cutoff frequency $\Omega_c = \frac{\pi}{2}$ and length $L_h = 7$.

MATLAB is particularly well suited to matrix computations such as that in Eq. (8.50).

```

01 Lh = 7; n = (0:1:(Lh-1)/2); k = n'; Omega0 = 2*pi/Lh; Omegak = k*Omega0;
02 A = 2*cos(Omegak*((Lh-1)/2-n(1:end-1))); A = [A,ones(length(Omegak),1)];
03 H = abs(Omegak)<=pi/2; h = inv(A)*H; h = [h;fliplr(h(1:end-1))]', 
h = -0.1146 0.0793 0.3210 0.4286 0.3210 0.0793 -0.1146

```

The resulting filter $h[n]$ is identical to that obtained in Ex. 8.17 using Eqs. (8.47) and (8.46).

Example 8.18 ◀

▷ **Drill 8.12 (Alternate Frequency Sampling Method Allpass Filter Designs)**

Using frequency sampling based on the equations in Table 8.3, design $L_h = 7$ or 8 type I, II, III, and IV allpass filters. Plot the magnitude responses to confirm the conclusions of Table 8.4.

◀

Frequency Sampling Filters

Any FIR filter can be realized using a direct realization, such as those shown in Fig. 8.20. A filter designed using the frequency sampling method, however, can be realized using an alternate structure comprised of an L_h -order comb filter in cascade with a parallel bank of $L_h - 1$ first-order filters. This structure forms what is called a *frequency sampling filter*. Let us begin with the transfer function $H(z)$ of this filter, obtained by taking the z -transform of $h[n]$ as defined by Eq. (8.46):

$$\begin{aligned} H(z) &= \sum_{n=0}^{L_h-1} h[n]z^{-n} \\ &= \sum_{n=0}^{L_h-1} \left(\frac{1}{L_h} \sum_{k=0}^{L_h-1} H[k] e^{j \frac{2\pi n k}{L_h}} \right) z^{-n} \\ &= \frac{1}{L_h} \sum_{k=0}^{L_h-1} H[k] \sum_{n=0}^{L_h-1} (e^{j \frac{2\pi k}{L_h}} z^{-1})^n. \end{aligned}$$

The second sum on the right-hand side is a geometric series. Using entry 1 of Table 5.1, we have

$$H(z) = \frac{1}{L_h} \sum_{k=0}^{L_h-1} H[k] \left(\frac{1 - z^{-L_h} e^{j 2\pi k}}{1 - e^{j \frac{2\pi k}{L_h}} z^{-1}} \right).$$

Hence,

$$H(z) = \underbrace{\frac{1 - z^{-L_h}}{L_h}}_{H_1(z)} \underbrace{\sum_{k=0}^{L_h-1} \frac{H[k]}{1 - e^{j \frac{2\pi k}{L_h}} z^{-1}}}_{H_2(z)}. \quad (8.51)$$

Observe that we do not need to perform IDFT (or IFFT) computations to obtain the desired filter transfer function. All we need is the values of the frequency samples $H[k]$, which are known. Equation (8.51) shows that the desired filter is realized as a cascade of two filters with transfer functions $H_1(z)$ and $H_2(z)$. Also, $H_1(z)$ is the transfer function of an L_h -th-order comb filter (see

Ex. 8.10). The second filter with transfer function $H_2(z)$ is a parallel combination of L_h first-order filters whose poles lie on the unit circle at $e^{j\frac{2\pi k}{L_h}}$ ($k = 0, 1, 2, \dots, L_h - 1$). Particularly for lowpass and bandpass filters, many of the coefficients $H[k]$ appearing in $H_2(z)$ are zero. In Ex. 8.17, for example, four of seven coefficients are zero. Thus, in practice, the final filter is usually much simpler than it appears in Eq. (8.51). As a result, this realization may require fewer computations (multiplications and additions) than a direct form realization.

When we realize an FIR system using a frequency sampling filter, we make the observation that the $H_2(z)$ subsystem is an IIR filter. This strange fact should alert us to the possibility of something interesting going on in this realization. For an FIR filter, there can be no poles other than those at the origin. In the frequency sampling filter of Eq. (8.51), in contrast, $H_2(z)$ has L_h poles at $e^{jk\Omega_0}$ ($k = 0, 1, 2, \dots, L_h - 1$). All these poles lie on the unit circle at equally spaced points. These poles simply *cannot remain* in our FIR filter. They must somehow get canceled along the way. This is precisely what happens. The zeros of $H_1(z)$ are exactly where the poles of $H_2(z)$ are because

$$z^{L_h} - 1 = (z - e^{j0\frac{2\pi}{L_h}})(z - e^{j\frac{2\pi}{L_h}})(z - e^{j2\frac{2\pi}{L_h}}) \cdots (z - e^{j(L_h-1)\frac{2\pi}{L_h}}).$$

Thus, in theory at least, the poles of $H_2(z)$ are canceled by the zeros of $H_1(z)$, which results in the desired FIR filter.

The pole-zero cancellation in the frequency sampling filter is a potential cause for mischief because such a perfect cancellation assumes exact realization of both $H_1(z)$ and $H_2(z)$. Such a realization requires the use of infinite-precision arithmetic, which is a practical impossibility because of quantization effects. Imperfect pole-zero cancellation means that there will still be poles on the unit circle and the overall filter cannot be FIR. More serious, however, is the fact that the resulting system will be marginally stable. Such a system provides no damping of the round-off noise that is introduced in the computations. In fact, such noise tends to increase with time and may render the filter useless. We can partially mitigate this problem by moving both the poles (of $H_2(z)$) and zeros (of $H_1(z)$) to a circle of radius $r = 1 - \epsilon$, where ϵ is a small positive number. This artifice will make the overall system asymptotically stable.

It is also worth emphasizing that the first-order systems that comprise $H_2(z)$ are complex. This is because the poles lie on the unit circle, and the scaling coefficients $H[k]$ are, in general, complex. Although it is relatively simple to realize complex systems digitally, it is possible to avoid the need for complex filters by combining conjugate poles into real second-order systems. These points are clarified in the next example.

▷ Example 8.19 (Lowpass FIR Filter Realization by a Frequency Sampling Filter)

Realize the length-7 lowpass FIR filter of Ex. 8.17 using a frequency sampling filter.

Referring to Ex. 8.17, we see that $H[0] = 1$, $H[1] = e^{-j6\pi/7}$, $H[6] = e^{j6\pi/7}$, and $H[2] = H[3] = H[4] = H[5] = 0$. Substituting these values in the frequency sampling filter's transfer function of Eq. (8.51), we obtain

$$H(z) = \underbrace{\frac{1 - z^{-7}}{7}}_{H_1(z)} \underbrace{\left(\frac{1}{1 - z^{-1}} + \frac{e^{-j6\pi/7}}{1 - e^{j2\pi/7}z^{-1}} + \frac{e^{j6\pi/7}}{1 - e^{-j2\pi/7}z^{-1}} \right)}_{H_2(z)}.$$

In this form, two of the three first-order filters in $H_2(z)$ are complex. To implement this system

using only real filters, we combine the two conjugate pole terms to obtain

$$\begin{aligned} H(z) &= \frac{1-z^{-7}}{7} \left(\frac{1}{1-z^{-1}} + \frac{2\cos(6\pi/7) - 2\cos(8\pi/7)z^{-1}}{1-2\cos(2\pi z/7)z^{-1}+z^{-2}} \right) \\ &= \underbrace{\frac{1-z^{-7}}{7}}_{H_1(z)} \underbrace{\left(\frac{1}{1-z^{-1}} - \frac{1.802(1-z^{-1})}{1-1.247z^{-1}+z^{-2}} \right)}_{H_2(z)}. \end{aligned}$$

This frequency sampling filter realization only requires three scalar multipliers, as opposed to the seven scalar multipliers needed using the direct form realization of Fig. 8.20. This is also more efficient than the four scalar multipliers needed using the form of Fig. 8.21.

Example 8.19 ◀

Spectral Sampling with Windowing

The frequency sampling method can be modified to take advantage of windowing. We first design our filter using a length L_h' that is much larger than the design value L_h . The result is a filter that matches the desired frequency response at a very large number (L_h') of points. Then we use a suitable L_h -point window to truncate the L_h' -point impulse response. This procedure yields the final design of a desired order L_h . As we shall see in the next section with frequency-weighted least-squares filter design, this idea of using more frequency-domain points than seems needed can help improve the overall behavior of our filter.

A Beauty More than Skin Deep

There is a story of a conceited prince who would only marry a woman of perfect complexion. Though he searched high and low, every woman he found had some flaw, whether it was one freckle too many or one too few. In the end, the prince died alone having never known the joy of even an imperfect love. How much happier would the prince have been had he learned that beauty is more than skin deep and that life often benefits from a bit of compromise. As we shall see next, these same lessons also apply to digital FIR filter design.

8.2.7 Frequency-Weighted Least-Squares FIR Filter Design

The Fourier series method forces an FIR impulse response to exactly match L_h values of the ideal. Such rigid expectations rarely lead to the best filter. The window method, on the other hand, seeks an impulse response that is only similar to the ideal. Through this compromise, the window method can often produce satisfactory filters when the Fourier series method cannot.

The story is the same when using frequency-domain methods for FIR filter design. The frequency sampling method wants to exactly match L_h values of the ideal frequency response. To exactly match these values, the remaining values of the frequency response may deviate so much from the ideal so as to render the filter worthless. Would it not be wiser to try and do well at all frequencies, even if most do not exactly match the ideal? This is precisely the approach taken by the frequency-weighted least-squares method. In order to derive the approach, however, we need to first cover some background mathematics.

Linear Vector Spaces, Inner Products, and Orthogonality

Here, we briefly summarize the concepts of linear vector spaces, inner products, and orthogonality. A comprehensive treatment of these topics would likely distract more than aid our goal of FIR filter design. Thus, we only cover the bare minimum needed for our later derivations.

A *linear vector space* is a set of elements (vectors) together with the operations of addition and scalar multiplication. For our purposes, we are most concerned with elements (vectors) that take the form of a function, either of time or of frequency.

Let x , y , and z designate elements in a linear vector space and α be some constant. We define the *inner product* $\langle \cdot, \cdot \rangle$ as an operation that satisfies four properties:

1. $\langle x, y \rangle = \langle y, x \rangle^*$,
2. $\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$,
3. $\langle \alpha x, y \rangle = \alpha \langle x, y \rangle$, and
4. $\langle x, x \rangle \geq 0$ with $\langle x, x \rangle = 0$ if and only if $x = 0$.

An inner product results in a (possibly complex) constant, which is to say that $\langle x, y \rangle \in \mathcal{C}$. For any linear vector space, there are many possible ways (infinite) to define an inner product.

It is convenient to view the inner product $\langle x, y \rangle$ as some measure of similarity between x and y . If x is quite similar to y , then the inner product is large. If x is not similar to y , then the inner product is small. We say that elements x and y are *orthogonal* (or perpendicular) if their inner product is zero. That is,

$$\text{if } \langle x, y \rangle = 0, \quad \text{then } x \perp y. \quad (8.52)$$

Orthogonal vectors are independent of one another: knowing one tells you nothing about the other. One can also interpret the inner product as a projection of one vector onto another. If the vectors are perpendicular to one another (orthogonal), then the projection results in zero. Designated as $\|x\|$, the *norm* of a vector is a positive real quantity that indicates the vector's size. The quantity $\sqrt{\langle x, x \rangle} = \sqrt{\|x\|^2}$ is a valid norm.

To provide a simple example, the set of all continuous-time energy signals is a linear vector space. For two energy signals $x(t)$ and $y(t)$, one valid inner product is

$$\langle x(t), y(t) \rangle = \int_{-\infty}^{\infty} x(t)y^*(t) dt.$$

From this definition, we see that a causal energy signal $x(t)u(t)$ is orthogonal to an anti-causal energy signal $y(t)u(-t)$. Such signals have nothing in common. Finally, we note that signal energy $E_x = \langle x(t), x(t) \rangle = \|x(t)\|^2$ is a valid norm.

The Orthogonality Principle

Next, consider a linear vector space that contains a known set of K *basis vectors* (or *basis signals*) ϕ_k ($1 \leq k \leq K$). One can think of the basis vectors ϕ_k as building blocks with which to construct or estimate any vector x that is in the linear vector space. We construct an estimate \hat{x}_K of vector x as a linear combination of the K basis vectors ϕ_k , that is,

$$\hat{x}_K = \sum_{k=1}^K c_k \phi_k. \quad (8.53)$$

What coefficients c_k produce the best possible estimate \hat{x}_K of x ? To answer this question, we must first know what we mean by best.

A sensible measure of estimate quality is the norm-squared error between x and \hat{x}_K ,

$$\mathcal{E}_K = \|x - \hat{x}_K\|^2 = \langle x - \hat{x}_K, x - \hat{x}_K \rangle.$$

In this sense, the best coefficients c_k minimize \mathcal{E}_K , the norm-squared error between \hat{x}_K and x . According to the *orthogonality principle*, this occurs when the error is orthogonal to the data used in the estimate, or

$$(\text{error} = x - \hat{x}_K) \perp \phi_l, \quad l = 1, 2, \dots, K.$$

Consequently,

$$\langle x - \hat{x}_K, \phi_l \rangle = 0, \quad l = 1, 2, \dots, K$$

or

$$\langle \hat{x}_K, \phi_l \rangle = \langle x, \phi_l \rangle, \quad l = 1, 2, \dots, K.$$

Substituting for \hat{x}_K , we find that the coefficients c_k that minimize \mathcal{E}_K must satisfy

$$\sum_{k=1}^K c_k \langle \phi_k, \phi_l \rangle = \langle x, \phi_l \rangle, \quad l = 1, 2, \dots, K. \quad (8.54)$$

These K linear equations in K unknown coefficients c_k are solved in the standard manner. However, if $\{\phi_k\}_{k=1}^K$ is an orthogonal set of basis vectors (meaning that $\langle \phi_l, \phi_k \rangle = 0$ for $l \neq k$), then the coefficients c_k can be individually solved as

$$c_k = \frac{\langle x, \phi_k \rangle}{\|\phi_k\|^2}. \quad (8.55)$$

This equation is particularly nice and intuitive. The coefficient c_k depends on the similarity between x and the basis function ϕ_k . If ϕ_k is very similar to x , then the coefficient c_k will be relatively large. If ϕ_k is not very similar to x , then the coefficient c_k will be small.

Let us prove the orthogonality principle. First, let us define an estimate x_K^\perp as one that satisfies the orthogonality principle. That is,

$$x_K^\perp = \sum_{k=1}^K c_k^\perp \phi_k \quad \text{and} \quad \langle x - x_K^\perp, \phi_l \rangle = 0 \text{ for } l = 1, 2, \dots, K.$$

Using a little give and take, we next write the norm-squared error of an arbitrary estimate \hat{x}_K in terms of x_K^\perp as

$$\begin{aligned} \mathcal{E}_K &= \|x - \hat{x}_K\|^2 \\ &= \|(x - x_K^\perp) + (x_K^\perp - \hat{x}_K)\|^2 \\ &= \|x - x_K^\perp\|^2 + \langle x - x_K^\perp, x_K^\perp - \hat{x}_K \rangle + \langle x_K^\perp - \hat{x}_K, x - x_K^\perp \rangle + \|x_K^\perp - \hat{x}_K\|^2 \\ &= \|x - x_K^\perp\|^2 + 2\operatorname{Re} \{ \langle x - x_K^\perp, x_K^\perp - \hat{x}_K \rangle \} + \|x_K^\perp - \hat{x}_K\|^2. \end{aligned}$$

Now, since x_K^\perp and \hat{x}_K are both linear combinations of $\{\phi_k\}_{k=1}^K$, and since $\langle x - x_K^\perp, \phi_l \rangle = 0$, we have

$$\langle x - x_K^\perp, x_K^\perp - \hat{x}_K \rangle = 0.$$

Consequently,

$$\mathcal{E}_K = \|x - x_K^\perp\|^2 + \|x_K^\perp - \hat{x}_K\|^2 \geq \|x - x_K^\perp\|^2.$$

Thus, the norm-squared error \mathcal{E}_K of some estimate \hat{x}_K must always be greater than or equal to the norm-squared error of the estimate x_K^\perp that satisfies the orthogonality principle. In fact, the only way the norm-squared error \mathcal{E}_K of \hat{x}_K equals the norm-squared error of x_K^\perp is if $\hat{x}_K = x_K^\perp$. In other words, the orthogonality principle is guaranteed to deliver the estimate with the minimum norm-squared error.

Let us demonstrate the orthogonality principle with a simple graphical example. As depicted in Fig. 8.36a, we wish to estimate the vector x using the horizontal direction vector (basis function) ϕ_1 . That is, we seek $\hat{x}_1 = c_1 \phi_1$ to estimate x . From trigonometry, we know that the best (minimum squared error) estimate \hat{x}_1 is one that causes the error vector $x - \hat{x}_1$ to be perpendicular to \hat{x}_1 (Fig. 8.36b). That is, we require that the error $x - \hat{x}_1$ be orthogonal to the basis function ϕ_1 that comprises \hat{x}_1 . This is exactly the conclusion of the orthogonality principle.



Figure 8.36: Graphical demonstration of the orthogonality principle.

► **Example 8.20 (Using the Orthogonality Principle to Derive the Fourier Series)**

Use the orthogonality principle to derive the Fourier series.

The set of all continuous-time T_0 -periodic power functions forms a linear vector space, on which we can define an inner product as

$$\langle x(t), y(t) \rangle = \frac{1}{T_0} \int_{T_0} x(t)y^*(t) dt.$$

In this way, signal power $P_x = \langle x(t), x(t) \rangle = \|x(t)\|^2$.

The Fourier series represents a continuous-time T_0 -periodic power signal $x(t)$ using an orthonormal set of harmonically related complex exponential basis functions,

$$x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}.$$

This representation of $x(t)$ follows the form of Eq. (8.53), where the basis functions $\phi_k = e^{jk\omega_0 t}$. The term *orthonormal* means that the basis functions are mutually orthogonal ($\langle \phi_l, \phi_k \rangle = 0$ for $l \neq k$) and of unit norm ($\|\phi_k\|^2 = \langle \phi_k, \phi_k \rangle = 1$).

According to the orthogonality principle, the best (minimum norm-squared error) coefficients X_k of this representation are given by Eq. (8.55) as

$$X_k = \frac{\langle x, \phi_k \rangle}{\|\phi_k\|^2} = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk\omega_0 t} dt.$$

This is exactly the Fourier series analysis equation (see Eq. (1.58)).

Example 8.20 ◀

We are now equipped to develop the frequency-weighted least-squares (FWLS) method of FIR filter design. From Eq. (8.32), recall that a length- L_h FIR filter with impulse response $h[n]$ has frequency response given by

$$H(e^{j\Omega}) = \sum_{n=0}^{L_h-1} h[n] e^{-jn\Omega}. \quad (8.56)$$

This equation is identical in form to Eq. (8.53). From this perspective, an FIR filter's frequency response approximates some desired frequency response (call it $H_d(e^{j\Omega})$) using a linear combination of complex exponential basis functions. To find the optimal coefficients $h[n]$, all we need is to define an appropriate inner product and then use the orthogonality principle.

Let us define our inner product between arbitrary functions $X(\Omega)$ and $Y(\Omega)$ as

$$\langle X(\Omega), Y(\Omega) \rangle = \frac{1}{2\pi} \int_0^{2\pi} X(\Omega) Y^*(\Omega) Q(\Omega) d\Omega. \quad (8.57)$$

The weighting function $Q(\Omega)$, which is taken to be real and nonnegative, allows different frequency regions to be emphasized in different ways. As a basic rule of thumb, the weighting function is typically set inversely proportional to the allowable squared ripple,

$$Q(\Omega) = \begin{cases} \left(\frac{1}{\delta_p/2}\right)^2 & \text{passband } \Omega \\ \left(\frac{1}{\delta_s}\right)^2 & \text{stopband } \Omega \\ 0 & \text{transition band } \Omega \end{cases}. \quad (8.58)$$

Defining $Q(\Omega)$ in this way, the filter's norm-squared error is more heavily weighted where ripple requirements are most restrictive (small). Thus, unlike the other FIR design methods we have discussed, an FWLS FIR filter has the notable advantage that it can achieve different levels of passband and stopband ripple.

Using the inner product of Eq. (8.57), the norm-squared error of our filter is given as

$$\mathcal{E} = \langle H_d(e^{j\Omega}) - H(e^{j\Omega}), H_d(e^{j\Omega}) - H(e^{j\Omega}) \rangle = \frac{1}{2\pi} \int_0^{2\pi} |H_d(e^{j\Omega}) - H(e^{j\Omega})|^2 Q(\Omega) d\Omega.$$

To minimize \mathcal{E} , the orthogonality principle requires that the error be orthogonal to the data (basis functions); that is,

$$\langle H_d(e^{j\Omega}) - H(e^{j\Omega}), e^{-jl\Omega} \rangle = 0 \quad \text{for } l = 0, 1, \dots, L_h - 1.$$

Thus,

$$\frac{1}{2\pi} \int_0^{2\pi} (H_d(e^{j\Omega}) - H(e^{j\Omega})) e^{jl\Omega} Q(\Omega) d\Omega = 0, \quad l = 0, 1, \dots, L_h - 1. \quad (8.59)$$

To help simplify this equation, let us define

$$a_l = \frac{1}{2\pi} \int_0^{2\pi} e^{jl\Omega} Q(\Omega) d\Omega$$

and

$$b_l = \frac{1}{2\pi} \int_0^{2\pi} H_d(e^{j\Omega}) e^{jl\Omega} Q(\Omega) d\Omega.$$

Using Eq. (8.56), we next rewrite Eq. (8.59) in terms of the known quantities a_l and b_l as

$$\sum_{n=0}^{L_h-1} a_{l-n} h[n] = b_l, \quad l = 0, 1, \dots, L_h - 1.$$

This is a system of L_h linear equations in L_h unknowns. Expressed in matrix form, the solution of this system of equations is

$$\underbrace{\begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[L_h-1] \end{bmatrix}}_h = \underbrace{\begin{bmatrix} a_0 & a_{-1} & \cdots & a_{1-L_h} \\ a_1 & a_0 & \cdots & a_{2-L_h} \\ \vdots & \vdots & \ddots & \vdots \\ a_{L_h-1} & a_{L_h-2} & \cdots & a_0 \end{bmatrix}}_{A^{-1}}^{-1} \underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{L_h-1} \end{bmatrix}}_b. \quad (8.60)$$

Since \mathbf{A} is a Toeplitz matrix, efficient algorithms exist to compute its inverse. Still, there are several significant disadvantages to this form of the FWLS method. First, in cases such as real-time channel equalization, the desired response $H_d(e^{j\Omega})$ may only be known for a finite number of points, not a continuum of Ω . Further, even if $H_d(e^{j\Omega})$ is known for all Ω , it is not particularly convenient to evaluate the integrals that define a_l and b_l .

To overcome both these problems, we discretize the FWLS procedure. Rather than looking at Ω over a continuum of points, we instead look at a large but finite number of frequencies $K \gg L_h$. Typically, these frequencies are uniformly spaced $2\pi/K$ apart. Thus, replacing integrals with sums, our norm-squared error over these points is

$$\mathcal{E}_K = \frac{1}{K} \sum_{k=0}^{K-1} |H_d[k] - H(e^{jk2\pi/K})|^2 Q(k2\pi/K).$$

Orthogonality again yields a solution, but now our expressions for a_l and b_l become

$$a_l = \frac{1}{K} \sum_{k=0}^{K-1} e^{jkl2\pi/K} Q(k2\pi/K) \quad (8.61)$$

and

$$b_l = \frac{1}{K} \sum_{k=0}^{K-1} H_d[k] e^{jkl2\pi/K} Q(k2\pi/K). \quad (8.62)$$

The desired FWLS FIR filter is again found using Eq. (8.60). As before, it is important that $H_d[k]$ be specified so that a linear phase solution is preferred. Similar to Eq. (8.48), for example, we can define a linear phase $H_d[k]$ for a real filter in terms of an ideal continuous-time response as

$$H_d[k] = \begin{cases} H_c(j\frac{k2\pi}{KT})e^{-jk\pi\frac{L_h-1}{K}} & 0 \leq k \leq \lceil \frac{K-1}{2} \rceil \\ H_d^*[L_h - k] & \lceil \frac{K-1}{2} \rceil + 1 \leq k \leq K-1 \end{cases}. \quad (8.63)$$

It is also crucial for real filters that the weighting function $Q(k2\pi/K)$ possess midpoint symmetry. To avoid problems from computer round-off, this is best done explicitly, as done for $H_d[k]$ in Eq. (8.63). Let us conclude with an FWLS design example.

▷ Example 8.21 (Frequency-Weighted Least-Squares Lowpass Filter Design)

Using the frequency-weighted least-squares method, design a length $L_h = 20$ lowpass FIR filter with $\Omega_p = \pi/3$, $\Omega_s = \pi/2$, $\delta_p = 1$, and $\delta_s = 0.01$. Specify the desired frequency response to have linear phase.

Let us use MATLAB to complete the design.

```

01 Omegap = pi/3; Omegas = pi/2; deltap = 1; deltas = .01;
02 Lh = 20; K = 10*Lh; k = (0:K-1); Omegak = k*2*pi/K;
03 Q = (Omegak<=Omegap)/((deltap/2)^2)+(Omegak>=Omegas)/(deltas^2);
04 Q(fix(K/2)+2:end) = Q(round(K/2):-1:2);
05 Hd = 1.0*(Omegak<=Omegap).*exp(-1j*k*pi*(Lh-1)/K);
06 Hd(fix(K/2)+2:end) = conj(Hd(round(K/2):-1:2));
07 l = (0:Lh-1)'; a = exp(1j*l*Omegak)*Q./K; b = exp(1j*l*Omegak)*(Hd.*Q/K)';
08 A = toeplitz(a); h = (A\b);
09 n = (0:Lh-1)'; subplot(211); stem(n,h);
10 Omega = linspace(0,pi,1001); H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
11 subplot(212); plot(Omega,20*log10(abs(H)));

```

Line 01 establishes the basic filter parameters, and line 02 selects $K = 10L_h$ points Ω_k to represent the continuum of points over $0 \leq \Omega < 2\pi$. Line 03 defines the frequency-weighting function $Q(\Omega_k)$

over $0 \leq \Omega_k \leq \pi$ according to Eq. (8.58), whereas line 04 uses midpoint symmetry to compute $Q(\Omega_k)$ for the remaining frequencies $\pi < \Omega_k < 2\pi$. Line 05 uses Eq. (8.63) to compute the desired (linear phase) response over $0 \leq \Omega_k \leq \pi$, and line 06 computes the remaining values using the midpoint conjugate symmetry of $H_d[k]$. Line 07 computes the a_l and b_l using Eqs. (8.61) and (8.62). Line 08 forms the Toeplitz matrix \mathbf{A} from the values a_l and then computes the FIR impulse response $h[n]$ using Eq. (8.60). Here, the computationally expensive matrix inverse is avoided by using the simpler left divide command. Lines 09–11 plot this impulse response (Fig. 8.60a) and its corresponding dB magnitude response (Fig. 8.60b). Notice that this FWLS solution achieves different levels of passband and stopband ripple.

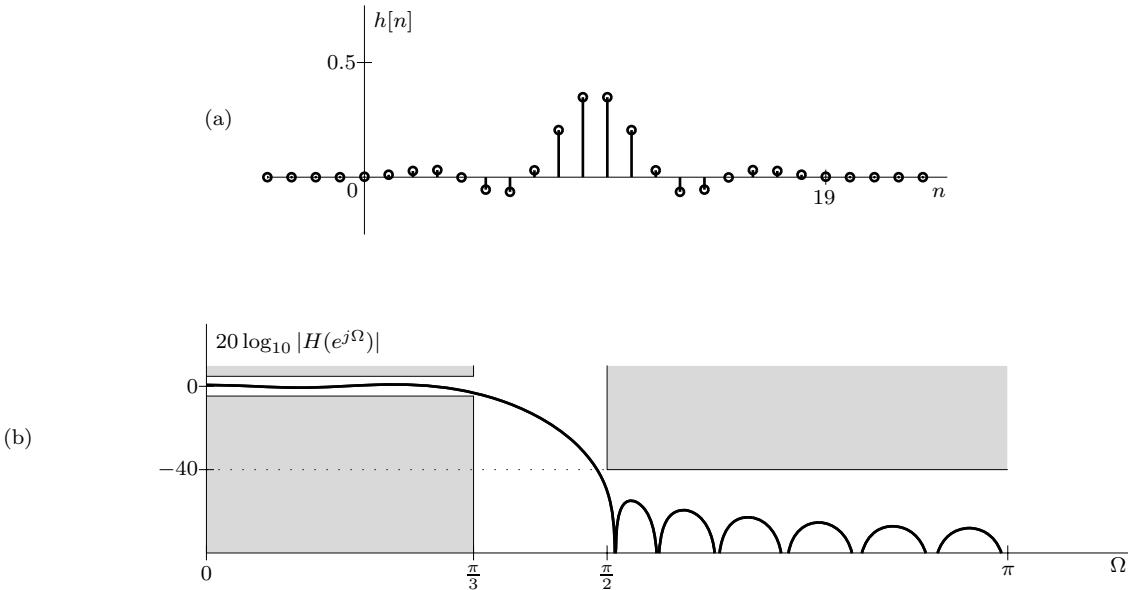


Figure 8.37: Frequency-weighted least-squares LPF: (a) impulse response and (b) magnitude response.

Although this design meets specifications on the first try, this will not always be the case. Thus, the length L_h or weighting function $Q(\Omega)$ may require adjustment to achieve a satisfactory design.

Example 8.21 \triangleleft

▷ Drill 8.13 (Frequency-Weighted Least-Squares Highpass Filter Design)

Using the frequency-weighted least-squares method, design a length $L_h = 101$ highpass FIR filter with $\Omega_p = \pi/2$ and $\Omega_s = 3\pi/8$. Specify the desired frequency response to have linear phase and $Q(\Omega)$ to be 10 times smaller in the passband than in the stopband. What is the passband and stopband ripple of the filter?

\triangleleft

Equiripple FIR Filter Design

As discussed in Ch. 2, the equiripple elliptic filter is generally lower order than other IIR filter types. Working just hard enough to satisfy design requirements, an equiripple solution just meets specifications over the entire passband and stopband regions. Non-equiripple solutions are more wasteful (higher order) and tend to exceed specifications over portions of passband, stopband, or both. These

same principles hold for FIR filters as well. To help reduce FIR filter length (complexity), we seek an equiripple solution.

As with FWLS FIR filters, consider a frequency-weighting function $Q(\Omega)$ from which we can define a frequency-weighted error $E(\Omega)$ given as

$$E(\Omega) = (H_d(e^{j\Omega}) - H(e^{j\Omega})) \sqrt{Q(\Omega)}.$$

Using this notation, the FWLS method minimizes the mean (integral) square of $E(\Omega)$. Now rather than mean square error, another sensible measure of error is maximum absolute error,

$$\begin{aligned} \mathcal{E}_{\max} &= \max_{\Omega} |E(\Omega)| \\ &= \max_{\Omega} |(H_d(e^{j\Omega}) - H(e^{j\Omega})) \sqrt{Q(\Omega)}|. \end{aligned}$$

Although the proof is outside the scope of this text, an equiripple design results from minimizing this maximum error (a min-max optimization problem).

The *Remez exchange algorithm* is a computation procedure for the general min-max optimization problem, and the *Parks-McClellan algorithm* is a popular variation specifically suited for FIR filter design. The details of these algorithms are rather complicated, and we do not cover them here. Fortunately, MATLAB's Signal Processing Toolbox provides built-in functions that implement the Parks-McClellan algorithm, as well as many other FIR filter design methods. Let us demonstrate the design of an equiripple FIR filter with a simple example.

▷ Example 8.22 (Equiripple FIR Filter Design)

Using functions from MATLAB's Signal Processing Toolbox, design a length $L_h = 20$ equiripple lowpass FIR filter with $\Omega_p = 2\pi/5$ and $\Omega_s = \pi/2$.

Here, we use MATLAB's `firpm` command to design our equiripple FIR filter. Details on this command, which implements the Parks-McClellan algorithm, are available using MATLAB's help facilities.

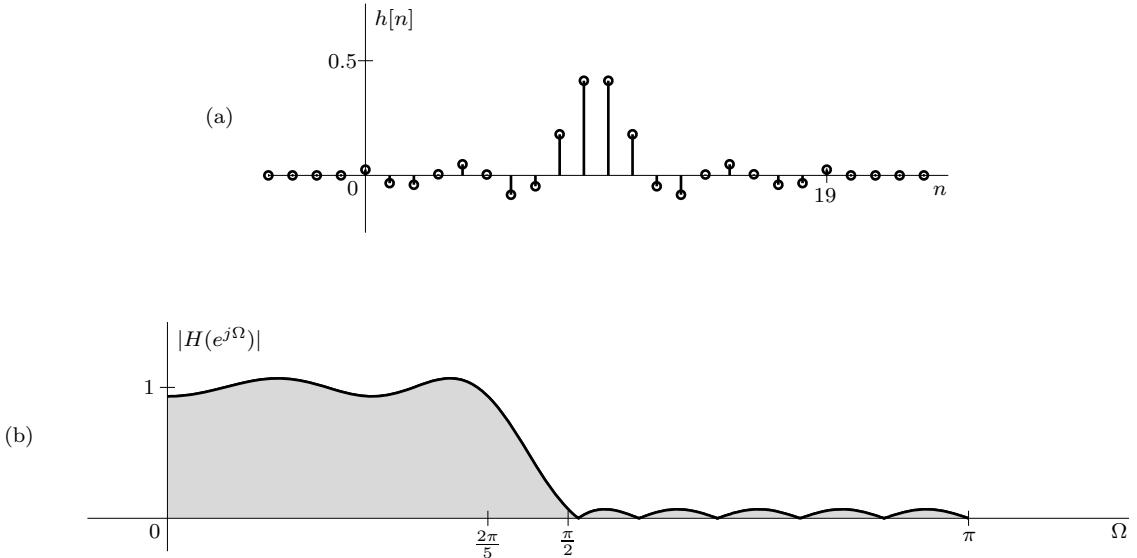


Figure 8.38: Equiripple LPF: (a) impulse response and (b) magnitude response.

```

01 Omegap = 2*pi/5; Omegas = pi/2; Lh = 20;
02 h = firpm(Lh-1,[0 Omegap Omegas pi]/pi,[1 1 0 0]);
03 n = (0:Lh-1); subplot(211); stem(n,h);
04 Omega = linspace(0,pi,1001); H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
05 subplot(212); plot(Omega,abs(H));

```

As shown in Fig. 8.38a, the resulting impulse response is finite in length and sinc-like in character. Further, the even symmetry of $h[n]$ ensures a linear phase response. The magnitude response of Fig. 8.38b confirms the equiripple nature of the response in both the passband and the stopband.

Example 8.22 ◀

8.3 Summary

Digital filters are classified as IIR or FIR. The duration of the impulse response of an IIR filter is infinite; that of a FIR filter is finite. A digital filter can simulate the time-domain or frequency-domain behavior of a desired filter. Thus, common digital filter design procedures are based on time- or frequency-domain criteria.

Digital IIR filters are typically derived from analog IIR filters. The time-domain criterion yields the impulse invariance method, and the frequency-domain criterion yields the bilinear transform. The impulse invariance method is handicapped by aliasing and cannot be used for highpass and bandstop filters. The bilinear transform, which is generally superior to the impulse invariance method, suffers from frequency warping. This effect, however, can be neutralized by prewarping. To design digital IIR highpass, bandpass, and bandstop filters, careful combinations of transformations are applied to normalized analog lowpass prototypes. To mitigate the ill effects of finite word lengths, digital IIR filters are usually realized using cascades of second-order sections.

Because FIR filters are a special case of IIR filters, we expect the performance of IIR filters to be superior. This statement is true in the sense that a given magnitude response can be achieved by an IIR filter of an order smaller than that required for the corresponding FIR filter. To their advantage, however, FIR filters can easily realize arbitrarily shaped magnitude responses, are guaranteed stable, and can, through proper midpoint symmetry in the impulse response $h[n]$, possess linear phase. Further, the realization of FIR filters is more simple and robust than the realization of IIR filters. For FIR filters, the time-domain criterion leads to the window method, of which the Fourier series method is a special case. Smoother windows yield smaller passband and stopband ripple but require longer lengths to achieve a given transition band. The frequency-domain criterion leads to the frequency sampling and frequency-weighted least-squares methods, both of which are particularly well suited to the design of filters with arbitrarily shaped magnitude responses.

References

1. Jackson, L. B., “A Correction to Impulse Invariance,” *IEEE Signal Processing Letters*, Vol. 7, No. 10, October 2000.
2. Kaiser, J. F., “Nonrecursive Digital Filter Design Using the I_0 -sinh Window Function, *Proc. 1974 IEEE International Symposium on Circuits and Systems*, San Francisco, CA, April 1974, pp. 20–23.
3. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
4. Mecklenbrauker, W. F. G., “Remarks on and Correction to the Impulse Invariant Method for the Design of IIR Digital Filters,” *Signal Processing*, Vol. 80, No. 8, August 2000, pp. 1687–1690.

5. Mitra, S. K., *Digital Signal Processing: A Computer-Based Approach*, 3rd Ed., McGraw-Hill, New York, 2006.
6. Mitra, S. K., and Kaiser, J. F., *Handbook for Digital Signal Processing*, Wiley, New York, 1993.
7. Porat, B., *A Course in Digital Signal Processing*, Wiley, New York, 1997.
8. Selesnick, I. W., et al., “Classical Filter Design Methods,” in *The Digital Signal Processing Handbook*, V. K. Madisetti and D. B. Williams, eds., CRC Press, Boca Raton, FL, 1998, pp. **11-18–11-30**.

Problems

- 8.1-1** In Ch. 4, we approximate the first-order differential equation $\frac{d}{dt}y(t) + cy(t) = x(t)$ [Eq. (4.48)] using backward differences as

$$y[n] + \frac{-1}{1+cT}y[n-1] = \frac{T}{1+cT}x[n].$$

Compare this solution with the one resulting from the impulse invariance method. Show that one result is a close approximation of the other and that the approximation improves as $T \rightarrow 0$.

- 8.1-2** Using the impulse invariance criterion, design a digital filter to realize a continuous-time filter with transfer function

$$H_c(s) = \frac{7s + 20}{2(s^2 + 7s + 10)}.$$

Use a 1% criterion for the choice of T . Show a canonical and a parallel realization of the filter.

- 8.1-3** Using the impulse invariance criterion, design a digital filter to realize the second-order continuous-time Butterworth filter with transfer function

$$H_c(s) = \frac{1}{s^2 + \sqrt{2}s + 1}.$$

Use a 1% criterion for the choice of T .

- 8.1-4** Design a digital integrator using the impulse invariance method. Determine and plot the magnitude response, and compare it with that of an ideal integrator. If this integrator is used primarily for integrating audio signals (whose bandwidth is 20 kHz), determine a suitable value for T .

- 8.1-5** By definition, an oscillator is a source (no input) that generates a sinusoid of a certain frequency ω_0 . Therefore, an oscillator is a system whose zero-input response is a sinusoid of the desired frequency. Choosing T so that there are 10 samples in each cycle of the sinusoid, consider the design of a 10-kHz digital oscillator.

- (a) Determine $H(z)$ directly so that its zero-input response is a discrete-time sinusoid of frequency $\Omega = \omega T$ corresponding to 10 kHz.

- (b) Choose $H_c(s)$ whose zero-input response is an analog sinusoid of 10 kHz. Now use the impulse invariance method to determine $H(z)$.

- (c) Show a canonic realization of the oscillator.

- 8.1-6** A variant of the impulse invariance method is the *step invariance method* of digital filter synthesis. In this method, for a given $H_c(s)$, we design $H(z)$ in Fig. 8.1b such that its output $y[n]$ is identical to $y_c(nT)$ in Fig. 8.1a for the input $x_c(t) = u(t)$.

- (a) For the step invariance method, show that

$$H(z) = (1 - z^{-1})\mathcal{Z}\left\{\left(\mathcal{L}^{-1}\frac{H_c(s)}{s}\right)_{t=nT}\right\}.$$

Hint: Assume that $x_c(t) = u(t)$ is sampled to produce $x[n] = u[n]$.

- (b) Using the step invariance method, determine $H(z)$ when

$$H_c(s) = \frac{\omega_c}{s + \omega_c}.$$

- (c) Use the step invariance method to synthesize a discrete-time integrator. Compare its magnitude response with that of an ideal integrator.

- 8.1-7** In the *ramp invariance method* for a given $H_c(s)$, we design $H(z)$ in Fig. 8.1b such that its output $y[n]$ is identical to $y_c(nT)$ in Fig. 8.1a for the input $x_c(t) = tu(t)$. Use this method to synthesize a discrete-time differentiator.

- 8.1-8** Repeat Prob. 8.1-7 to design a discrete-time integrator.

- 8.1-9** In an impulse invariance design, show that if $H_{mc}(s)$ is a transfer function of a stable system, the corresponding $H(z)$ is also a transfer function of a stable system.

- 8.1-10** Consider the design of a digital differentiator $H(z)$.

- (a) Determine $H(z)$ using the bilinear transform.

- (b) Show a realization of this filter.
- (c) Find and sketch the magnitude response of this filter, and compare it with that of an ideal differentiator.
- (d) If this filter is used primarily for processing audio signals (voice and music) up to 20 kHz, determine a suitable value for T .

8.1-11 Repeat Prob. 8.1-10 for a digital integrator.

8.1-12 Using the bilinear transform with pre-warping, design a digital lowpass Butterworth filter that satisfies $\alpha_p = 2$ dB, $\alpha_s = 11$ dB, $\omega_p = 100\pi$ rad/s, and $\omega_s = 200\pi$ rad/s. The highest significant frequency is 250 Hz. If possible, it is desirable to oversatisfy the requirement of α_s .

8.1-13 Repeat Prob. 8.1-12 for a Chebyshev filter.

8.1-14 Repeat Prob. 8.1-12 using $\alpha_p = 0.5$ dB and $\alpha_s = 40$ dB. Using a cascade of second-order sections, show a realization of this filter.

8.1-15 Using the bilinear transform with pre-warping, design a digital Butterworth low-pass filter with unity gain at $\Omega = 0$, $\alpha_p = 1$ dB maximum passband attenuation over $0 \leq \Omega \leq \frac{\pi}{3}$, and $\alpha_s = 20$ dB minimum stopband attenuation over $\frac{2\pi}{3} \leq \Omega \leq \pi$.

8.1-16 Using the bilinear transform with pre-warping, design a digital highpass Butterworth filter that satisfies $\alpha_p = 2$ dB, $\alpha_s = 10$ dB, $\omega_p = 150\pi$ rad/s, and $\omega_s = 100\pi$ rad/s. The highest significant frequency is 200 Hz.

8.1-17 Repeat Prob. 8.1-16 using an inverse Chebyshev filter.

8.1-18 Repeat Prob. 8.1-16 using a Chebyshev filter for $\alpha_p = 0.5$ dB and $\alpha_s = 40$ dB. Using a cascade of second-order sections, show a realization of this filter.

8.1-19 Using the bilinear transform with pre-warping, design a digital bandpass Butterworth filter that satisfies $\alpha_p = 2$ dB, $\alpha_s = 12$ dB, $\omega_{p1} = 120$ rad/s, $\omega_{p2} = 300$ rad/s, $\omega_{s1} = 45$ rad/s, and $\omega_{s2} = 450$ rad/s. The highest significant frequency is 500 Hz.

8.1-20 Repeat Prob. 8.1-19 for a Chebyshev filter.

8.1-21 Repeat Prob. 8.1-19 using $\alpha_p = 0.5$ dB, $\alpha_s = 40$ dB, $\omega_{p1} = 100$ rad/s, $\omega_{p2} = 300$ rad/s, $\omega_{s1} = 50$ rad/s, and $\omega_{s2} = 400$ rad/s.

8.1-22 Using the bilinear transform with pre-warping, design a digital bandstop Butterworth filter that satisfies $\alpha_p = 1$ dB, $\alpha_s = 22$ dB, $\omega_{p1} = 40$ rad/s, $\omega_{p2} = 195$ rad/s, $\omega_{s1} = 80$ rad/s, and $\omega_{s2} = 120$ rad/s. The highest significant frequency is 200 Hz.

8.1-23 Repeat Prob. 8.1-22 for a Chebyshev filter.

8.1-24 An input ADC is used to collect input samples at a CD-quality rate $F_s = 44$ kHz. Input samples are processed with a digital filter to generate output samples, which are sent to a DAC at the same rate F_s . Every sample interval $T = 1/F_s$, the digital processor executes the following MATLAB-compatible code:

```

01 x = read_ADC;
02 mem(1) = x - mem(3)*9/16;
03 y = mem(1)*7/16 - mem(3)*7/16;
04 write_DAC = y;
05 mem(3) = mem(2);
06 mem(2) = mem(1);

```

(a) Does the code implement DFI, DFII, TDFI, or TDFII? Support your answer by drawing the appropriate block diagram labeled in a manner that is consistent with the code.

(b) Plot the filter impulse and magnitude response. What type of filter does this digital system implement?

(c) What frequency f_0 causes the analog input $x(t) = \cos(2\pi f_0 t)$ to generate the strongest response? What is the corresponding gain of the digital system?

8.1-25 The bilinear transform, given by Eq. (8.9), is actually a one-to-one mapping relationship between the s -plane and the z -plane.

(a) Show that the bilinear transform maps the $j\omega$ -axis in the s -plane to the unit circle in the z -plane.

- (b) Show that the bilinear transform maps the LHP and the RHP of the s -plane map to the interior and the exterior, respectively, of the unit circle in the z -plane.
- (c) Show that the bilinear transform maps a stable continuous-time system $H_c(s)$ to a stable discrete-time system $H(z)$.

8.1-26 Prove that applying the bilinear transform to the CT filter of Eq. (8.14) yields the DT filter of Eq. (8.15).

8.2-1 Show that the frequency response of a third-order ($L = 3$) type II FIR filter is given by

$$H(e^{j\Omega}) = 2e^{-j1.5\Omega} \times \\ (h[0] \cos(3\Omega/2) + h[1] \cos(\Omega/2)).$$

8.2-2 Show that the frequency response of a third-order ($L = 3$) type IV FIR filter is given by

$$H(e^{j\Omega}) = 2e^{j(\frac{\pi}{2}-1.5\Omega)} \times \\ (h[0] \sin(3\Omega/2) + h[1] \sin(\Omega/2)).$$

8.2-3 Show that the transfer function $H(z)$ of a type II FIR filter must have an odd number of zeros at $z = -1$.

8.2-4 Another form of comb filter is given by the transfer function

$$H(z) = 1 + z^{-L}.$$

For this problem, let $L = 6$.

- (a) Find the filter's impulse response $h[n]$.
- (b) Determine a canonical realization of this filter.
- (c) Find and plot the magnitude and phase responses of this filter.

8.2-5 Using the Fourier series method, design an $L_h = 11$ FIR filter to approximate the ideal bandpass filter shown in Fig. P8.2-5. Let $T = 2 \times 10^{-3}$.

8.2-6 Repeat Prob. 8.2-5 using the window design method and a triangular window.

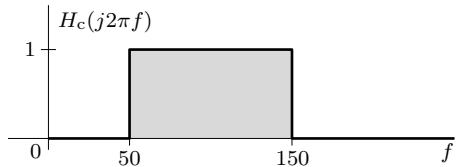


Figure P8.2-5

8.2-7 Using the Fourier series method, design an $L_h = 15$ FIR filter to approximate an ideal lowpass filter with a cutoff frequency of 20 kHz. Let the sampling frequency be $F_s = 200$ kHz.

8.2-8 Repeat Prob. 8.2-7 using the window design method and a Hamming window.

8.2-9 Consider the length-7 ($L_h = 7$) low-pass FIR filter of Ex. 8.11. Redesign this filter using a Hann window, and plot the resulting magnitude response $|H(e^{j\Omega})|$. How does this filter compare with those of Ex. 8.11?

8.2-10 Consider the $L_h = 8$ and $L_h = 11$ FIR differentiators of Ex. 8.12. Redesign these filters using a Blackman window, and plot the resulting magnitude response $|H(e^{j\Omega})|$. How does this filter compare with those of Ex. 8.12?

8.2-11 An ideal (bandlimited) $\pi/2$ phase shifter (Hilbert transformer) is given by

$$H_c(j\omega) = -j \operatorname{sgn}(\omega) \Pi \left(\frac{\omega}{2\pi/T} \right).$$

- (a) Sketch $|H_c(j\omega)|$ and $\angle H_c(j\omega)$.
- (b) Using rectangular and Hamming windows, use the window method to design $L_h = 15$ FIR filters to approximate $H_c(j\omega)$. Plot the resulting magnitude and phase responses, and compare them with the ideal.

8.2-12 Using the window method and a non-Kaiser window, design a digital lowpass filter with $\Omega_p = \pi/4$, $\Omega_s = \pi/3$, and $\alpha_s = 20$ dB.

8.2-13 Repeat Prob. 8.2-12 using $\alpha_s = 35$ dB.

8.2-14 Repeat Prob. 8.2-12 using $\alpha_s = 50$ dB.

8.2-15 Repeat Prob. 8.2-12 using $\alpha_s = 65$ dB.

8.2-16 Using the window method and a non-Kaiser window, design a digital highpass filter with $\Omega_s = 0.45\pi$, $\Omega_p = 0.5\pi$, and $\alpha_s = 60$ dB.

8.2-17 Using the window method with a Blackman window, design a length $L_h = 11$ digital bandpass filter with $\Omega_{p_1} = \pi/4$, $\Omega_{p_2} = 3\pi/8$, and $\alpha_s = 60$ dB. Plot the corresponding magnitude response, and determine the approximate widths of the two transition bands. Lastly, show an efficient block diagram realization of the filter.

8.2-18 Repeat Prob. 8.2-17 using a Kaiser window.

8.2-19 Using the frequency sampling method (with linear phase), design an $L_h = 11$ digital differentiator. Compare both $h[n]$ and $H(e^{j\Omega})$ of the resulting design with the $L_h = 11$ designs of Ex. 8.12.

8.2-20 Repeat Prob. 8.2-19 using $L_h = 8$.

8.2-21 For cases where passband ripple is more restrictive than stopband ripple, express Eqs. (8.41) and (8.42) in terms of r_p rather than α_s . Hint: Use the fact that $\delta_s = \frac{\delta_p}{2}$ for the window method.

8.2-22 Given a lowpass FIR filter with impulse response $h_{lp}[n]$, it is possible to obtain a highpass FIR filter by simply alternating the sign of every other value of $h_{lp}[n]$. That is,

$$h_{hp}[n] = (-1)^n h_{lp}[n].$$

- (a) Prove that this technique really does transform a lowpass FIR filter into a highpass FIR filter.
- (b) If $h_{lp}[n]$ has a cutoff frequency Ω_c , determine the corresponding cutoff frequency of $h_{hp}[n]$.
- (c) Using the window method and a Hann window, design an $L_h = 50$ lowpass filter with cutoff frequency $\Omega_c = \pi/3$. Plot the impulse response $h_{lp}[n]$ as well as the corresponding magnitude response $|H_{lp}(e^{j\Omega})|$.

(d) Transform the lowpass filter in part (c) into a highpass filter. Plot both $h_{hp}[n]$ and $|H_{hp}(e^{j\Omega})|$.

8.2-23 Given a lowpass FIR filter with impulse response $h_{lp}[n]$, it is possible to obtain a bandpass FIR filter with center frequency Ω_{ctr} according to

$$h_{bp}[n] = 2 \cos(n\Omega_{ctr}) h_{lp}[n].$$

- (a) Prove that this technique really does transform a lowpass FIR filter into a bandpass FIR filter.
- (b) If $h_{lp}[n]$ has a cutoff frequency Ω_c , determine the corresponding cutoff frequencies of $h_{bp}[n]$.
- (c) Using the window method and a triangular window, design an $L_h = 61$ lowpass filter with cutoff frequency $\Omega_c = \pi/4$. Plot the impulse response $h_{lp}[n]$ as well as the corresponding magnitude response $|H_{lp}(e^{j\Omega})|$.
- (d) Transform the lowpass filter in part (c) into a bandpass filter. Plot both $h_{bp}[n]$ and $|H_{bp}(e^{j\Omega})|$.

8.2-24 Using the frequency sampling method, design a linear phase lowpass FIR filter with cutoff frequency $\Omega_c = \pi/3$ and $L_h = 27$.

8.2-25 Using the frequency sampling method, design a linear phase lowpass FIR filter with cutoff frequency $\Omega_c = 2\pi/3$ and $L_h = 28$.

8.2-26 Using the frequency sampling method, design a linear phase highpass FIR filter with cutoff frequency $\Omega_c = \pi/3$ and $L_h = 15$. Show an efficient block diagram realization of the filter.

8.2-27 Using the frequency sampling method, design a linear phase bandpass FIR filter with cutoff frequencies $\Omega_{c_1} = \pi/4$, $\Omega_{c_2} = 5\pi/8$, and $L_h = 70$.

8.2-28 Using the frequency sampling method, design a linear phase bandstop FIR filter with cutoff frequencies $\Omega_{c_1} = \pi/3$, $\Omega_{c_2} = \pi/2$, and $L_h = 13$. Realize the design as a frequency sampling filter.

- 8.2-29** Using the frequency sampling method, design an $L_h = 21$ FIR filter to approximate the magnitude response shown in Fig. P8.2-29. Set the desired phase response to favor an appropriate linear characteristic.

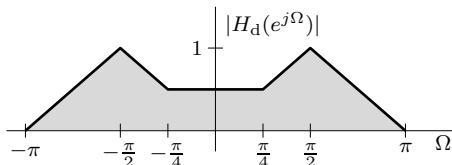


Figure P8.2-29

- 8.2-30** Defining $Q(\Omega) = 1$ for all Ω , repeat Prob. 8.2-29 using the frequency-weighted least-squares method. What happens if instead $Q(\Omega) = 2$ over all Ω ?

- 8.2-31** Defining $Q(\Omega) = 10$ for $|\Omega| < \pi/2$ and 1 elsewhere, repeat Prob. 8.2-29 using the frequency-weighted least-squares method.

- 8.2-32** Using the frequency-weighted least-squares method, design a length $L_h = 25$ highpass FIR filter with $\Omega_p = 2\pi/3$, $\Omega_s = \pi/2$, $\delta_p = 0.5$, and $\delta_s = 0.02$. Specify the desired frequency response to have linear phase. Does the filter meet specifications?

- 8.2-33** Using functions from MATLAB's Signal Processing Toolbox, design a length $L_h = 31$ equiripple highpass FIR filter with $\Omega_p = 3\pi/4$ and $\Omega_s = \pi/2$.

Chapter 9

Discrete Fourier Transform

In this chapter, we introduce the discrete Fourier transform (DFT), which may be viewed as an economy class DTFT and is applicable when $x[n]$ is of finite length (or made finite length by windowing). The DFT is one of the most important tools for digital signal processing, especially when we implement it using the efficient fast Fourier transform (FFT) algorithm, discussed in Sec. 9.7. The development of the FFT algorithm in the mid-sixties gave a huge impetus to the area of DSP. The DFT, using the FFT algorithm, is truly the workhorse of modern digital signal processing, and it is nearly impossible to exaggerate its importance. A solid understanding of the DFT is a must for anyone aspiring to work in the digital signal processing field. Not only does the DFT provide a frequency-domain representation of DT signals, it is also useful to numerous other tasks such as FIR filtering, spectral analysis, and solving partial differential equations.

Computation of the Direct and Inverse DTFT

As we saw in Ch. 6, frequency analysis of discrete-time signals involves determination of the discrete-time Fourier transform (DTFT) and its inverse (IDTFT). The DTFT analysis equation of Eq. (6.1) yields the frequency spectrum $X(\Omega)$ from the time-domain signal $x[n]$, and the synthesis equation of Eq. (6.2) reverses the process and constructs $x[n]$ from $X(\Omega)$. There are, however, two difficulties in the implementation of these equations on a digital processor or computer.

1. Equation (6.1) shows that determination of $X(\Omega)$ from $x[n]$ involves summing an infinite number of terms, which is not possible because it requires infinite computer time and memory. Moreover, such computations can determine only a finite number of sample values of $X(\Omega)$, not $X(\Omega)$ for all values of Ω .
2. Equation (6.2) shows that determination of $x[n]$ from $X(\Omega)$ requires integration that, most commonly, a computer only approximates by a finite sum. Further, a practical computer can compute $x[n]$ only for a finite number of n .

Both these difficulties are neatly resolved if $x[n]$ has a finite length N . In such a case, the infinite sum in Eq. (6.1) reduces to a finite sum. In addition, we shall show that only N uniformly spaced samples of $X(\Omega)$ are needed to specify $X(\Omega)$ completely. As a consequence, the integral in Eq. (6.2) reduces to a finite sum in terms of the samples of $X(\Omega)$. Therefore, for $x[n]$ of finite length N , the needed spectrum is a set of N samples of $X(\Omega)$. The direct and inverse relationships between a length- N signal $x[n]$ and the N samples of $X(\Omega)$ define the *discrete Fourier transform* (DFT). The DFT proves extremely useful in numerical computations of the direct and inverse DTFT.

The DFT can process finite-length $x[n]$ exactly, without any error. As we shall demonstrate, the DFT for a finite length sequence is not an approximation. It gives exact values of the DTFT at N uniform points, and we can increase N as large as we wish. Moreover, the DTFT can also be computed exactly from these N DFT points in the same manner as we can compute a bandlimited

continuous-time signal exactly from its Nyquist samples. Of course, not all signals of interest are finite in length. What about infinite-length $x[n]$? Every practical $x[n]$ must necessarily have finite energy and, therefore, must approach 0 and become negligible for n greater than some $n \geq N$. We can truncate this signal for $n \geq N$ and then use the DFT. This truncation produces (controllable) error in the results. But, as seen earlier, by its very nature, numerical computations for infinite-length $x[n]$ generally involve approximations no matter what method is used.

9.1 The Discrete Fourier Transform

Let $X(\Omega)$ be the DTFT of a finite-length signal $x[n]$ that is zero outside $0 \leq n \leq N - 1$. Let $X[k]$ be the uniform samples of $X(\Omega)$ at a frequency interval of $\Omega_0 = 2\pi/N$,

$$X[k] = X(k\Omega_0) = X\left(\frac{2\pi k}{N}\right), \quad 0 \leq k \leq N - 1. \quad (9.1)$$

Thus, $X[k]$ are the N samples of $X(\Omega)$ that are spaced uniformly at intervals of $\Omega_0 = 2\pi/N$ over the frequency range of 0 to 2π (one full period of $X(\Omega)$). Next, we develop the equations relating $x[n]$ and $X[k]$ that define the discrete Fourier transform (DFT).

Using Eq. (6.1) and the fact that $X[k] = X(k\Omega_0)$ with $\Omega_0 = 2\pi/N$, we obtain

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn}, \quad 0 \leq k \leq N - 1. \quad (9.2)$$

This *analysis equation* of the DFT (direct DFT) identifies the spectral components of $x[n]$. Since the spectrum $X[k]$ is constructed exclusively from N -periodic functions $e^{-j\Omega_0 kn}$, it is inherently an N -periodic function of k , although we only concern ourselves with the single period $0 \leq k \leq N - 1$. This is a satisfying result as we expect all DT signals to have periodic spectra.

To obtain the inverse relationship, we multiply both sides of Eq. (9.2) by $e^{j\Omega_0 km}$ and sum over $0 \leq k \leq N - 1$ to obtain

$$\sum_{k=0}^{N-1} X[k]e^{j\Omega_0 km} = \sum_{k=0}^{N-1} \left[\sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn} \right] e^{j\Omega_0 km}.$$

Multiplying by $\frac{1}{N}$ and interchanging the order of summation on the right-hand side yield

$$\frac{1}{N} \sum_{k=0}^{N-1} X[k]e^{j\Omega_0 km} = \frac{1}{N} \sum_{n=0}^{N-1} x[n] \left[\sum_{k=0}^{N-1} e^{j\Omega_0 k(m-n)} \right]. \quad (9.3)$$

To simplify this expression, let us investigate the right-hand-side bracketed sum. First note that since $\Omega_0 N = 2\pi$, $e^{j(m-n)\Omega_0 k} = 1$ when $m - n = 0, \pm N, \pm 2N, \dots$. More compactly, $e^{j(m-n)\Omega_0 k} = 1$ for $\langle m - n \rangle_N = 0$, where $\langle m - n \rangle_N$ denotes $m - n$ modulo N .[†] Thus, $\sum_{k=0}^{N-1} e^{j\Omega_0 k(m-n)} = \sum_{k=0}^{N-1} 1 = N$ for $\langle m - n \rangle_N = 0$. To compute the remaining values of $m - n$, we note that the sum is a geometric progression with common ratio $e^{j\Omega_0(m-n)}$. Thus, using entry 1 of Table 5.1, we see for $\langle m - n \rangle_N \neq 0$ that

$$\sum_{k=0}^{N-1} e^{j\Omega_0 k(m-n)} = \frac{1 - e^{j\Omega_0(m-n)N}}{1 - e^{j\Omega_0(m-n)}} = \frac{1 - e^{j2\pi(m-n)}}{1 - e^{j\Omega_0(m-n)}} = \frac{1 - 1}{1 - e^{j\Omega_0(m-n)}} = 0.$$

Taken together,

$$\begin{aligned} \sum_{k=0}^{N-1} e^{j\Omega_0 k(m-n)} &= \begin{cases} N & \langle m - n \rangle_N = 0 \\ 0 & \langle m - n \rangle_N \neq 0 \end{cases} \\ &= N\delta[\langle m - n \rangle_N]. \end{aligned} \quad (9.4)$$

[†]More intuitively, the modulo- N operation $\langle m - n \rangle_N$ is the remainder after dividing $m - n$ by N , assuming $m - n \geq 0$. This operation is quite important to DFT studies and is covered in detail in Sec. 9.2.1.

Substituting Eq. (9.4) into Eq. (9.3) yields

$$\frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 km} = \sum_{n=0}^{N-1} x[n] \delta[\langle m - n \rangle_N].$$

The right-hand sum has only one nonzero term at $n = m$, so this simplifies to

$$\frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 km} = x[m], \quad 0 \leq m \leq N-1.$$

Replacing the variable m with the more conventional n , we obtain the inverse relationship we seek:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 kn}, \quad 0 \leq n \leq N-1. \quad (9.5)$$

This *synthesis equation* of the DFT (inverse DFT) uses the spectral samples $X[k]$ to construct $x[n]$ over the range $0 \leq n \leq N-1$.

Equations (9.2) and (9.5) define the DFT pair. We call $X[k]$ the (direct) discrete Fourier transform (DFT) of $x[n]$. Conversely, $x[n]$ is the inverse discrete Fourier transform (IDFT) of $X[k]$. This can be represented as

$$X[k] = \text{DFT}\{x[n]\} \quad \text{and} \quad x[n] = \text{IDFT}\{X[k]\}$$

or, more compactly, as

$$x[n] \longleftrightarrow X[k].$$

To avoid needless confusion, we use \longleftrightarrow to indicate the DFT pair rather than \iff , which is reserved for DTFT pairs. We stress once again that the number of points in both $x[n]$ and $X[k]$ are identical and equal to N . Commonly, DFT frequencies are referred to as DFT *bins*.

▷ Example 9.1 (3-Point DFT of a Length-3 Signal)

Find the DFT of the length-3 signal $x[n] = 3\delta[n] + 2\delta[n-1] + 3\delta[n-2]$ shown in Fig. 9.1. Compare the DFT spectrum $X[k]$ with the corresponding DTFT spectrum $X(\Omega)$.

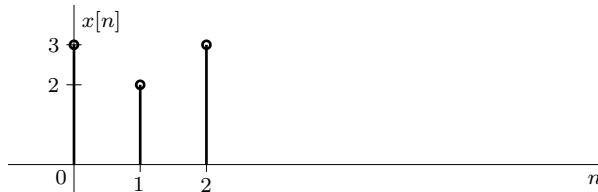


Figure 9.1: Length-3 signal $x[n] = 3\delta[n] + 2\delta[n-1] + 3\delta[n-2]$.

In this case, $N = 3$, and therefore, $\Omega_0 = 2\pi/3$. From Eq. (9.2), we obtain

$$X[k] = \sum_{n=0}^2 x[n] e^{-j(\frac{2\pi}{3})kn} = 3 + 2e^{-j\frac{2\pi}{3}k} + 3e^{-j\frac{4\pi}{3}k}.$$

Therefore,

$$X[0] = 3 + 2 + 3 = 8,$$

and

$$X[1] = 3 + 2e^{-j\frac{2\pi}{3}} + 3e^{-j\frac{4\pi}{3}} = 3 + \left(-1 - j\sqrt{3}\right) + \left(-\frac{3}{2} + j\frac{3\sqrt{3}}{2}\right) = \left(\frac{1}{2} + j\frac{\sqrt{3}}{2}\right) = e^{j\frac{\pi}{3}}.$$

Similarly,

$$X[2] = 3 + 2e^{-j\frac{4\pi}{3}} + 3e^{-j\frac{8\pi}{3}} = e^{-j\frac{\pi}{3}}.$$

In table form, the magnitudes and angles of $X[k]$ are thus

k	0	1	2
$ X[k] $	8	1	1
$\angle X[k]$	0	$\frac{\pi}{3}$	$-\frac{\pi}{3}$

Observe that $X[1] = X[2]^*$. This follows from the conjugate symmetry property of $X[k]$, given later by Eq. (9.25).[†] Because $x[n]$ is a 3-point signal ($N = 3$), $X[k]$ is also a 3-point sequence.

Since $x[n]$ is finite duration, the DFT equals sample values of the DTFT $X(\Omega)$. By comparing the DFT with the DTFT, we can determine whether the DFT has enough samples to give a reasonably good representation of the DTFT. The DTFT for $x[n]$ is given by

$$X(\Omega) = \sum_{n=0}^2 x[n]e^{-j\Omega n} = 3 + 2e^{-j\Omega} + 3e^{-j2\Omega} = e^{-j\Omega}[2 + 3e^{j\Omega} + 3e^{-j\Omega}] = e^{-j\Omega}[2 + 6\cos(\Omega)].$$

The magnitude and phase spectra are given by

$$|X(\Omega)| = |2 + 6\cos(\Omega)| \quad \text{and} \quad \angle X(\Omega) = \begin{cases} -\Omega & 2 + 6\cos(\Omega) > 0 \\ -\Omega - \pi & 2 + 6\cos(\Omega) < 0 \end{cases}.$$

Figure 9.2 compares $|X[k]|$ and $\angle X[k]$ with $|X(\Omega)|$ and $\angle X(\Omega)$. Observe that the DFT values are exactly equal to DTFT values at the sampling frequencies; there is no approximation. This is always true for the DFT of a finite-length $x[n]$. However, if $x[n]$ is obtained by truncating or windowing a longer sequence, we shall see that the DFT gives only approximate sample values of the DTFT.

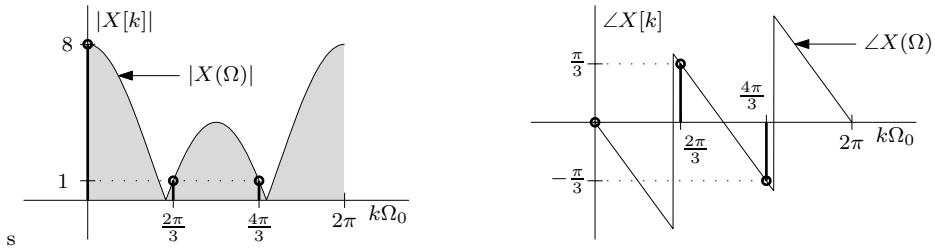


Figure 9.2: 3-point DFT of $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$ and its DTFT comparison.

The DFT in this example has too few points to give a reasonable picture of the DTFT. The peak of DTFT appearing between the second and third samples (between $k = 1$ and 2), for instance, is missed by the DFT. The two valleys of the DTFT are also missed. We definitely need more points in the DFT for an acceptable resolution. This goal is accomplished by *zero padding*, explained next.

Example 9.1 \triangleleft

[†]Recall that $X[k]$ are the samples of $X(\Omega)$ over the interval $0 \leq \Omega \leq 2\pi$. Moreover, $X(\Omega)$ is 2π -periodic so that $X(2\pi - \Omega) = X(-\Omega)$. Also, for real $x[n]$, $X(\Omega) = X^*(-\Omega) = X^*(2\pi - \Omega)$. If we let $\Omega = k\Omega_0$ and use the fact that $N\Omega_0 = 2\pi$, we obtain $X[k] = X^*[N - k]$. Here, $N = 3$, so $X[1] = X^*[3 - 1] = X^*[2]$.

▷ **Drill 9.1 (3-Point DFT of a Length-3 Signal)**

A length-3 signal is specified by $x[n] = 2\delta[n] + \delta[n - 1] + \delta[n - 2]$. Show that the ($N = 3$)-point DFT of this signal is $X[0] = 4$, $X[1] = 1$, and $X[2] = 1$. Find $X(\Omega)$, the DTFT of this signal, and graphically verify that the DFT is equal to the samples of $X(\Omega)$ at intervals of $\Omega_0 = 2\pi/N = 2\pi/3$.

△

9.1.1 The Picket Fence Effect and Zero Padding

The DFT yields samples of the DTFT spaced at frequency intervals of $\Omega_0 = 2\pi/N$. In this way, the DFT size N determines the frequency resolution Ω_0 . Seeing the DTFT through the DFT is like viewing $X(\Omega)$ through a *picket fence*. Only the spectral components at the sampled frequencies (which are integral multiples of Ω_0) are visible. All the remaining frequency components are hidden, as though behind the pickets of a fence. If the DFT has too few points, then peaks, valleys, and other details of $X(\Omega)$ that exist between the DFT points (sampled frequencies) will not be seen, thus giving an erroneous view of the spectrum $X(\Omega)$. This is precisely the case in Ex. 9.1. Actually, using the interpolation formula (discussed later), it is possible to compute any number of values of the DTFT from the DFT. But having to use the interpolation formula really defeats the purpose of the DFT. We therefore seek to reduce Ω_0 so that the number of samples is increased for a better view of the DTFT.

Because $\Omega_0 = 2\pi/N$, we can reduce Ω_0 by increasing N , the length of $x[n]$. But how do we increase the length of a signal when its length is given as N ? Simple! For an N -point signal, its values outside the range $0 \leq n \leq N - 1$ are zero. By simply taking $x[n]$ to include some of the zero-valued points beyond $N - 1$, we can artificially increase its length. This process is referred to as *zero padding*. By appending zero-valued samples to $x[n]$, we can increase N as much as we wish, thereby increasing the number of points of the DFT. Zero padding allows us to obtain more samples of $X(\Omega)$ for clarity. Furthermore, this process does not change the underlying signal $x[n]$, which means that zero padding does not affect the underlying spectrum $X(\Omega)$. Returning to our picket fence analogy, zero padding increases the number of pickets in our fence (N), but it cannot change what is behind the fence ($X(\Omega)$). To demonstrate the idea, we next rework Ex. 9.1 using zero padding to double the number of samples of the DTFT.

▷ **Example 9.2 (6-Point DFT of a Length-3 Signal)**

Using zero padding, find the 6-point DFT of the length-3 signal of Ex. 9.1, $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$. Compare the 6-point DFT $X[k]$ with the corresponding DTFT spectrum $X(\Omega)$.

To compute a 6-point DFT from our length-3 signal, we must pad $x[n]$ with three zeros, as shown in Fig. 9.3. In this way, we treat $x[n]$ as a 6-point signal with $x[0] = 3$, $x[1] = 2$, $x[2] = 3$, and $x[3] = x[4] = x[5] = 0$.

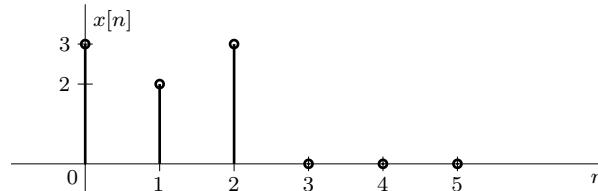


Figure 9.3: Zero padding $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$ to a length of six.

By padding $x[n]$ with three zeros, N increases to 6, and Ω_0 decreases to $2\pi/6 = \pi/3$. Computed

using Eq. (9.2), the magnitudes and angles of the resulting $X[k]$ are thus

k	0	1	2	3	4	5
$ X[k] $	8	5	1	4	1	5
$\angle X[k]$	0	$-\frac{\pi}{3}$	$\frac{\pi}{3}$	0	$-\frac{\pi}{3}$	$\frac{\pi}{3}$

Because of the conjugate symmetry property (Eq. (9.25), discussed later), $X[1] = X^*[5]$ and $X[2] = X^*[4]$. Observe that we now have a 6-point DFT, which provides 6 samples of the DTFT spaced at the frequency interval of $\pi/3$ (in contrast to the $2\pi/3$ spacing in Ex. 9.1). Here, the samples at $k = 0, 2$, and 4 are identical to the samples at $k = 0, 1$, and 2 in Ex. 9.1. The DFT spectrum in Fig. 9.4 contains all three samples appearing in Fig. 9.2 plus three additional samples interlaced in between. Clearly, zero padding allows us a better assessment of the DTFT. Notice in this case, however, that the valleys of $X(\Omega)$ are still largely missed by this 6-point DFT.

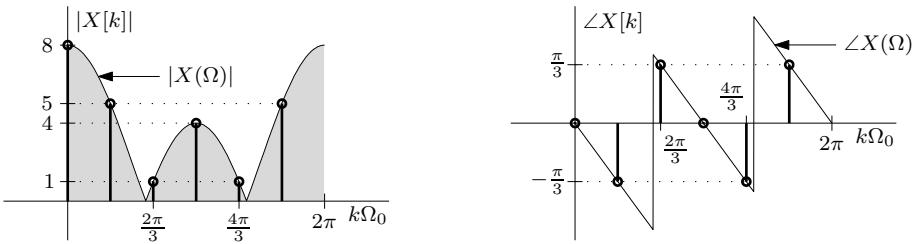


Figure 9.4: 6-point DFT of $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$ and its DTFT comparison.

Example 9.2 ◁

▷ Drill 9.2 (3-Point and 8-Point DFTs of a Length-3 Signal)

A length-3 signal $x[n]$ is specified by $x[0] = x[1] = x[2] = 1$.

- (a) Show that the DTFT of $x[n]$ is given by $X(\Omega) = e^{-j\Omega}[1 + 2\cos(\Omega)]$.
- (b) Show that the 3-point DFT of $x[n]$ is given by $X[0] = 3$ and $X[1] = X[2] = 0$. Graphically verify that the DFT values $X[k]$ are the samples of $X(\Omega)$ at intervals of $\Omega_0 = 2\pi/3$.
- (c) Show that the 8-point DFT of $x[n]$ is $X[k] = e^{-j\pi k/4}[1 + 2\cos(\pi k/4)]$. Observe the conjugate symmetry about $k = N/2 = 4$ by showing that $X[4+m] = X^*[4-m]$ for $m = 1, 2$, and 3 . Graphically verify that the DFT values $X[k]$ are the samples of $X(\Omega)$ at intervals of $\Omega_0 = \pi/4$.

▫

A Perplexing Puzzle of Padding

On close inspection, zero padding presents a somewhat perplexing puzzle: that through the addition of some innocuous zeros, which themselves never contribute to the DFT sum of Eq. (9.2), additional nonzero points in $X[k]$ are found. Consider, for example, a length-3 signal $x[n]$ that is defined over the range $0 \leq n \leq 2$ and is otherwise zero. Without zero padding, $N = 3$, and Eq. (9.2) produces a 3-point DFT. If we pad $x[n]$ with three zeros, however, then $N = 6$, and Eq. (9.2) produces a 6-point DFT. Since $x[n] = 0$ for $3 \leq n \leq 5$, we compute $X[k]$ using the exact same data ($x[n]$ for $0 \leq n \leq 2$) regardless of whether or not we use zero padding. Without zero padding, the zero-valued samples over the range $3 \leq n \leq 5$ are considered outside $x[n]$, whereas in the case with zero padding these samples are considered part of $x[n]$. This just appears to be a semantic sophistry. Yet it does produce results. Zero padding artificially increases the length N without changing the sequence, thereby reducing the sample interval $\Omega_0 = 2\pi/N$. This, in turn, increases the number of samples

$X[k]$ over the frequency range from 0 to 2π . Later, when we consider a periodic perspective of both the DFT and its inverse, we shall gain additional useful insight into this puzzle.

▷ **Example 9.3 (Zero Padding for Clarity)**

For the length-3 signal of Ex. 9.1, determine the zero padding needed to achieve a DFT frequency resolution of $\Omega_0 = \pi/10$. Write the corresponding expression for computing the DFT.

For $\Omega_0 = \pi/10$, the padded signal length is required to be $N = \frac{2\pi}{\Omega_0} = \frac{2\pi}{\pi/10} = 20$. Since the total length should be 20, we need to pad 17 zeros to $x[n]$. In this case,

$$X[k] = \sum_{n=0}^{19} x[n] e^{-j(\frac{\pi}{10})kn}, \quad 0 \leq k \leq 19.$$

However, the last 17 samples of $x[n]$ are zero. Therefore, the preceding equation reduces to

$$X[k] = \sum_{n=0}^2 x[n] e^{-j(\frac{\pi}{10})kn}, \quad 0 \leq k \leq 19.$$

The plot of this 20-point DFT, computed using MATLAB in Ex. 9.4, is shown in Fig. 9.5. As the DFT size N increases, it becomes progressively more difficult to compute the DFT (or its inverse) by hand. As we shall see next, matrix-based computations are more manageable, although not generally as efficient as the fast Fourier transform described in Sec. 9.7.

Example 9.3 □

9.1.2 Matrix Representation of the DFT and Its Inverse

For large N , whether the result of a long signal or much zero padding, it is cumbersome to manually evaluate the DFT and its inverse. Fortunately, both represent sets of N linear equations in N unknowns. Consequently, both can be expressed in matrix form, suitable for simple computer evaluation. Let us first consider the direct DFT. Expanding each sum for each value of k , Eq. (9.2) yields the matrix form of

$$\underbrace{\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix}}_X = \underbrace{\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{-j\Omega_0} & e^{-j\Omega_0 2} & \cdots & e^{-j\Omega_0(N-1)} \\ 1 & e^{-j\Omega_0 2} & e^{-j\Omega_0 4} & \cdots & e^{-j\Omega_0 2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j\Omega_0(N-1)} & e^{-j\Omega_0 2(N-1)} & \cdots & e^{-j\Omega_0(N-1)^2} \end{bmatrix}}_{D_N} \underbrace{\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix}}_x. \quad (9.6)$$

Known as the *discrete Fourier transform matrix*, the square matrix D_N has (kn) th element $e^{-j\Omega_0 kn}$. Furthermore, the DFT matrix D_N has a particularly convenient inverse $D_N^{-1} = \frac{1}{N}D_N^*$. Using this fact (or writing out Eq. (9.5)), the matrix form of the inverse DFT is

$$\underbrace{\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix}}_x = \frac{1}{N} \underbrace{\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{j\Omega_0} & e^{j\Omega_0 2} & \cdots & e^{j\Omega_0(N-1)} \\ 1 & e^{j\Omega_0 2} & e^{j\Omega_0 4} & \cdots & e^{j\Omega_0 2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\Omega_0(N-1)} & e^{j\Omega_0 2(N-1)} & \cdots & e^{j\Omega_0(N-1)^2} \end{bmatrix}}_{D_N^{-1}} \underbrace{\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix}}_X. \quad (9.7)$$

In compact matrix forms, the DFT is simply $\mathbf{X} = \mathbf{D}_N \mathbf{x}$, and the inverse DFT is $\mathbf{x} = \mathbf{D}_N^{-1} \mathbf{X}$ or, equivalently, $\mathbf{x} = \frac{1}{N} \mathbf{D}_N^* \mathbf{X}$.

The matrix forms of Eqs. (9.6) and (9.7) highlight the similarity between the DFT and its inverse. Except for a scale factor and a complex conjugation, the two are identical. As we shall explore later, this near-perfect symmetry leads to a *duality property* that is very similar to that of the FT.

▷ Example 9.4 (Matrix Evaluation of the DFT)

Use MATLAB and Eq. (9.6) to compute and plot the DFT of Ex. 9.3. Compare the DFT spectrum $X[k]$ with the corresponding DTFT spectrum $X(\Omega)$.

MATLAB readily computes and plots the required 20-point DFT and the DTFT for comparison.

```

01 N = 20; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1);
02 x = [3;2;3;zeros(N-3,1)];
03 DN = exp(-1j*Omega0*k'*n); X = DN*x;
04 Omega = linspace(0,2*pi,6001); XOmega = exp(-1j*Omega).*[2+6*cos(Omega));
05 subplot(121); stem(k*Omega0,abs(X)); line(Omega,abs(XOmega));
06 subplot(122); stem(k*Omega0,angle(X)); line(Omega,angle(XOmega));

```

Line 01 establishes the basic parameters of the problem. Line 02 defines the length-3 signal $x[n] = 3\delta[n] + 2\delta[n-1] + 3\delta[n-2]$ of Ex. 9.1 and pads it with $N-3 = 17$ zeros, as discussed in Ex. 9.3. Line 03 defines the 20-point DFT matrix \mathbf{D}_N and then computes the DFT using the matrix representation of Eq. (9.6). Line 04 defines the DTFT, derived in Ex. 9.1, as $X(\Omega) = e^{-j\Omega}[2 + 6 \cos(\Omega)]$. Finally, lines 05–06 plot the corresponding magnitude and phase spectra. As shown in Fig. 9.5, the 20-point DFT provides a good representation of the corresponding DTFT.

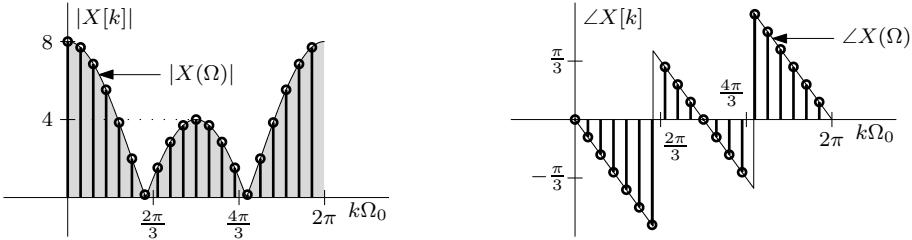


Figure 9.5: 20-point DFT of $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$ and its DTFT comparison.

Example 9.4 □

Signal Length Reduction and Aliasing

In zero padding, we effectively increase the length N . The opposite problem is to reduce the length N . We can examine the effect of such an operation on the DFT by considering $x[n]$ and $X(\Omega)$ in Ex. 9.1, where $x[n]$ is a length-3 sequence and its DFT is also a length-3 sequence. Recall that the DTFT in this case is

$$X(\Omega) = e^{-j\Omega}[2 + 6 \cos(\Omega)].$$

Let us examine what happens if we take only two samples of $X(\Omega)$, that is, if we reduce N from 3 to 2, which results in $\Omega_0 = \pi$. Thus, although $x[n]$ is a length-3 sequence, we are taking only two samples of its DTFT at intervals of π . Let these samples be denoted by $\hat{X}[k]$ for $k = 0$ and 1. Thus,

$$\hat{X}[k] = X(k\Omega_0) = X(k\pi) = e^{-jk\pi}[2 + 6 \cos(k\pi)], \quad k = 0, 1.$$

This yields $\hat{X}[0] = 8$ and $\hat{X}[1] = 4$. The IDFT corresponding to $\hat{X}[k]$ is $\hat{x}[n]$, given by

$$\hat{x}[n] = \frac{1}{2} \sum_{k=0}^1 \hat{X}[k] e^{j\pi k n}, \quad n = 0, 1.$$

This yields $\hat{x}[0] = 6$ and $\hat{x}[1] = 2$. The corresponding DTFT $\hat{X}(\Omega)$ is given by

$$\hat{X}(\Omega) = \sum_{n=0}^1 \hat{x}[n] e^{-jn\Omega} = 6 + 2e^{-j\Omega}.$$

Observe that reducing the number N of samples $X[k]$ from 3 to 2 changes everything. Zero padding, which increases the length N , does not change the basic signal $x[n]$ or the DTFT $X(\Omega)$. In contrast, a reduction of N defines a completely new signal $\hat{x}[n]$ and new DTFT $\hat{X}(\Omega)$. This behavior is closely related to undersampling a continuous-time signal. When a signal is undersampled, the samples cannot reconstruct the original signal (or the spectrum). Undersampling causes aliasing, and the reconstructed signal is entirely different from the original.

Minimum Number of DFT Points

Our discussion so far brings out the following very important fact: if $X(\Omega)$ is the DTFT of a length- N signal $x[n]$, then we can reconstruct $X(\Omega)$ from N' uniform samples of $X(\Omega)$ only if $N' \geq N$. Alternately, a minimum of N uniform samples of $X(\Omega)$ are needed to reconstruct $x[n]$. Using a smaller number of DFT points results in an entirely different $x[n]$. This is equivalent to constructing a signal from its sub-Nyquist rate samples, where aliasing prevents reconstruction of the original signal.

▷ Drill 9.3 (Signal Length Reduction and Aliasing)

The DTFT of the 3-point signal $x[n]$ specified in Drill 9.2 is $X(\Omega) = e^{-j\Omega}[1 + 2\cos(\Omega)]$. For this DTFT, consider the 2-point DFT obtained by sampling $X(\Omega)$ at intervals of π given by $\hat{X}[0] = 3$ and $\hat{X}[1] = 1$. Show that the IDFT of $\hat{X}[k]$ is given by $\hat{x}[0] = 2$ and $\hat{x}[1] = 1$. Show that the DTFT corresponding to $\hat{x}[n]$ is given by $\hat{X}(\Omega) = 2 + e^{-j\Omega}$.

△

9.1.3 DFT Interpolation to Obtain the DTFT

The length- N DFT of signal $x[n]$ yields N samples of its DTFT $X(\Omega)$. However, we can determine any number of samples of $X(\Omega)$ from the DFT of $x[n]$ by padding an adequate number of zero-valued samples to $x[n]$. If we pad a very large number of zeros to $x[n]$, we can practically obtain the entire $X(\Omega)$. This requires a large number of computations. We can also determine the entire $X(\Omega)$ from the DFT using interpolation. Recall that for the continuous-time case, we can obtain a signal's spectrum from its samples using spectral interpolation (Eq. (3.22)). We now discuss a similar technique for the discrete-time case. For a length- N $x[n]$, the DTFT is

$$\begin{aligned} X(\Omega) &= \sum_{n=0}^{N-1} x[n] e^{-jn\Omega} \\ &= \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 k n} \right] e^{-jn\Omega} \\ &= \frac{1}{N} \sum_{k=0}^{N-1} X[k] \sum_{n=0}^{N-1} e^{-j(\Omega - \Omega_0 k)n}. \end{aligned} \tag{9.8}$$

The inner sum on the right-hand side of Eq. (9.8) is a geometric progression with common ratio $e^{-j(\Omega - \Omega_0 k)}$. Using entry 1 of Table 5.1 and substituting $\Omega_0 = 2\pi/N$, we obtain

$$\begin{aligned} \sum_{n=0}^{N-1} e^{-j(\Omega - \Omega_0 k)n} &= \frac{1 - e^{-j(\Omega N - 2\pi k)}}{1 - e^{-j(\Omega - 2\pi k/N)}} \\ &= \frac{(e^{j(\Omega N - 2\pi k)/2} - e^{-j(\Omega N - 2\pi k)/2}) e^{-j(\Omega N - 2\pi k)/2}}{(e^{j(\Omega N - 2\pi k)/2N} - e^{-j(\Omega N - 2\pi k)/2N}) e^{-j(\Omega N - 2\pi k)/2N}} \\ &= \frac{\sin\left(\frac{\Omega N - 2\pi k}{2}\right)}{\sin\left(\frac{\Omega N - 2\pi k}{2N}\right)} e^{-j(\Omega N - 2\pi k)(N-1)/2N}. \end{aligned} \quad (9.9)$$

Substitution of Eq. (9.9) in Eq. (9.8) yields the desired interpolation formula

$$X(\Omega) = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \frac{\sin\left(\frac{\Omega N - 2\pi k}{2}\right)}{\sin\left(\frac{\Omega N - 2\pi k}{2N}\right)} e^{-j(\Omega N - 2\pi k)(N-1)/2N}. \quad (9.10)$$

Using this interpolation formula, we can compute $X(\Omega)$ from its N uniform samples $X[k]$ (the DFT).

▷ Example 9.5 (DFT Interpolation to Obtain the DTFT)

Using interpolation, determine the DTFT $X(\Omega)$ if the DFT of the corresponding 3-point $x[n]$ is given by $X[0] = 8$, $X[1] = e^{j\pi/3}$, and $X[2] = e^{-j\pi/3}$.

From Eq. (9.10), we obtain

$$\begin{aligned} X(\Omega) &= \frac{1}{3} \sum_{k=0}^2 X[k] \frac{\sin\left(\frac{3\Omega - 2\pi k}{2}\right)}{\sin\left(\frac{3\Omega - 2\pi k}{6}\right)} e^{-j(\Omega - \frac{2\pi k}{3})} \\ &= e^{-j\Omega} \frac{\sin(3\Omega/2)}{3} \left(\frac{8}{\sin(\Omega/2)} + \frac{1}{\sin(\frac{\Omega}{2} - \frac{\pi}{3})} - \frac{1}{\sin(\frac{\Omega}{2} + \frac{2\pi}{3})} \right). \end{aligned}$$

Using trigonometric identities and after some manipulation, we obtain

$$X(\Omega) = e^{-j\Omega}[2 + 6 \cos(\Omega)].$$

An Alternate Procedure

Obtaining $X(\Omega)$ from $X[k]$ generally requires heavy manipulation because of the complicated nature of the Eq. (9.10) interpolation formula. When N is small, it may be a simpler alternative to use Eq. (9.8) for the same purpose. This procedure can be broken down into two steps: first, determine $x[n]$ from $X[k]$, and second, determine $X(\Omega)$ from $x[n]$. In the present example, for instance, we find $x[n]$ as

$$x[n] = \frac{1}{3} \sum_{k=0}^2 X[k] e^{j2\pi kn/3} = \frac{1}{3} \left(8 + e^{j\pi/3} e^{j2\pi n/3} + e^{-j\pi/3} e^{j4\pi n/3} \right), \quad 0 \leq n \leq 2.$$

This yields $x[0] = 3$, $x[1] = 2$, and $x[2] = 3$. From these values of $x[n]$, we determine $X(\Omega)$ as

$$X(\Omega) = \sum_{k=0}^2 x[n] e^{-j\Omega n} = 3 + 2e^{-j\Omega} + 3e^{-j2\Omega} = e^{-j\Omega}[2 + 3e^{j\Omega} + 3e^{-j\Omega}] = e^{-j\Omega}[2 + 6 \cos(\Omega)].$$

Both procedures yield the same result, which also matches the result of Ex. 9.1.

Example 9.5 □

▷ **Drill 9.4 (DFT Interpolation to Obtain the DTFT)**

Use the interpolation formula to show that $X(\Omega) = 2 + e^{-j\Omega}$ is the DTFT underlying the 2-point DFT given by $X[0] = 3$ and $X[1] = 1$.

△

9.2 Uniqueness: Why Confine $x[n]$ to $0 \leq n \leq N - 1$?

In all the discussion so far, we have restricted the length- N sequence $x[n]$ to the range $0 \leq n \leq N - 1$. What happens if a sequence falls outside this range? Is the DFT for such a signal defined? Even when a signal is restricted to the range $0 \leq n \leq N - 1$, this signal, when time shifted in the traditional manner, will go out of this range. How, then, can one find the DFT of a shifted signal? Moreover, in the convolution of two signals, one of the signals is time-reversed and then shifted. A signal restricted to the range $0 \leq n \leq N - 1$, when time-reversed, also goes out of this range. Does this mean that the DFT of the convolution of two signals does not exist or is not meaningful? What use is such a restricted form of DFT that cannot handle signal shifting or convolution, which are the daily bread-and-butter issues in signal processing? Most important of all, for the purpose of defining the DFT, why are we restricting a signal to the range $0 \leq n \leq N - 1$? Why not let it have a general range such as $N_1 \leq n \leq N_2$? The answers to all these questions have something to do with the uniqueness of the DFT.

The Importance of Range on DFT Uniqueness

Let us see what happens if we remove the restriction on the range $0 \leq n \leq N - 1$. Consider a length- N signal $x[n]$ defined for $0 \leq n \leq N - 1$ and its N -point DFT $X[k]$, which has a similar range $0 \leq k \leq N - 1$. We use Eq. (9.2) to compute $X[k]$ as

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn}, \quad 0 \leq k \leq N - 1.$$

We make one interesting observation. In the sum on the right-hand side, the contribution of the sample $x[r]$, located at $n = r$, is $x[r]e^{-j\Omega_0 kr}$. If this sample $x[r]$, located at $n = r$, is shifted by some integer multiple of N , that is, shifted to $n = r + mN$, its contribution remains unchanged because

$$x[r]e^{-j\Omega_0 k(r+mN)} = x[r]e^{-j\Omega_0 kr}e^{-j2\pi m} = x[r]e^{-j\Omega_0 kr}.$$

Clearly, if we remove the restriction on the range $0 \leq n \leq N - 1$, the shift of any individual samples by some integer multiple of N results in the same $X[k]$. Note that each sample may be shifted by different integer multiples of N , although the more common case is when all $x[n]$ are shifted by the same integer multiple of N , which yields the signal $x[n-mN]$ defined over $mN \leq n \leq mN+N-1$. Either way, the only requirement to ensure that the N values of $X[k]$ remain unchanged is that all shifts be integer multiples of N . Using a similar argument, we can show that if we shift individual samples of $X[k]$ any way we want, as long as the shift of each sample is some integer multiple of N , the resulting $x[n]$ is the same.

To provide a graphical example, consider the 4-point signal $x_a[n]$ of Fig. 9.6a, defined over $0 \leq n \leq 3$. Shifting all $x_a[n]$ to the left by 4 yields the signal $x_b[n]$, defined over $-4 \leq n \leq -1$ and depicted in Fig. 9.6b. Figure 9.6c shows a case where different samples are shifted by different multiples of N : $x[0]$ is shifted from $n = 0$ to $n = 0 + 4 = 4$, $x[1]$ is shifted from $n = 1$ to $n = 1 + (-2 \times 4) = -7$, and the remaining two samples $x[2]$ and $x[3]$ are not shifted at all. Since their samples differ only by shifts that are integer multiples of N , the spectral samples $X[k]$ for all three signals in Fig. 9.6 are identical.

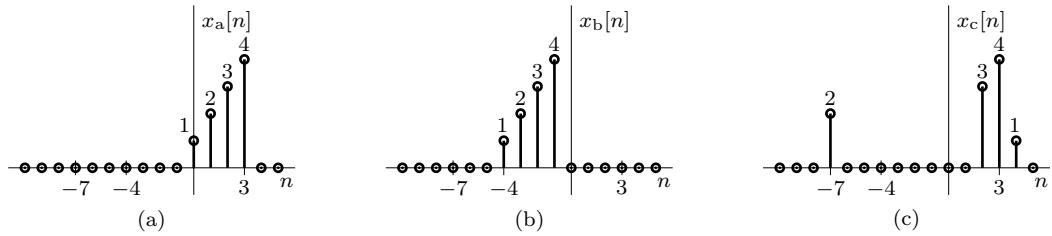


Figure 9.6: Three ($N = 4$)-point signals that differ only by shifting samples by multiples of N .

Do You Know My Niece Nicole?

Clearly, unless we restrict our signal to some accepted range such as $0 \leq n \leq N - 1$, we will not have a one-to-one correspondence between $x[n]$ and $X[k]$. Without such a convention, the DFT and IDFT are meaningless. It is like the case of an elderly gentleman who decides to visit his niece Nicole, who lives in New York City. Because of advanced age, he has forgotten her address, her phone number, and even her last name. Upon arriving in New York, he starts asking every stranger he meets whether they know his niece Nicole. Every stranger gives the same answer, “I know many people named Nicole, but I do not know whether any is your niece.” Poor uncle! If he had at least known the address of his niece, he would have had some hope of finding her. It is the same situation for the DFT. Unless we restrict $x[n]$ to some agreed range, the poor DFT will never find its IDFT relative.

A Perspective of Periodicity

There is another perspective that helps clarify this nonuniqueness between $X[k]$ and various $x[n]$. Mathematically, we can construct an N -periodic signal $\tilde{x}[n]$ from an N -point signal $x[n]$ as[†]

$$\tilde{x}[n] = \sum_{m=-\infty}^{\infty} x[n - mN]. \quad (9.11)$$

Figure 9.7 shows the three signals of Fig. 9.6 with their respective N -periodic replications. The resulting three N -periodic signals, which we know to have the same $X[k]$, are identical. This is no coincidence. Any N -point signals with identical N -periodic replications will have identical $X[k]$. Thought of another way, signals with the same periodic replication are indistinguishable to the DFT (indistinguishable $X[k]$). *It is as if the DFT views all N -point signals as being inherently N -periodic.* On the surface, periodicity may seem absurd for a signal that is assumed to be zero outside of N points. Before dismissing the idea of periodicity as delusional, however, we might notice that, taken over all n , the DFT synthesis Eq. (9.5) produces an N -periodic signal $x[n]$.[†] Furthermore, the discrete nature of $X[k]$ also suggests a periodic view of $x[n]$ (see Table 1.4 on page 68). The periodicity perspective offers tremendous insight into the DFT and its properties. Later, we will discuss this periodicity perspective in greater detail.

The main point to the preceding discussion is that by restricting ranges from 0 to $N - 1$, there is a one-to-one correspondence (uniqueness) between $x[n]$ and $X[k]$. Without such a restriction, there are infinite possible N -point $x[n]$ for a given N -point $X[k]$, and vice versa. This is the reason why we shall restrict $x[n]$ to $0 \leq n \leq N - 1$ and $X[k]$ to $0 \leq k \leq N - 1$. But why should we worry about uniqueness at all?

Without uniqueness, DFT analysis is largely meaningless. For instance, if DFT analysis yields an output DFT as $Y[k]$, then we cannot know $y[n]$ without uniqueness. Unless $y[n]$ is unique for a

[†]If $x[n]$ is an N -point signal with range $0 \leq n \leq N-1$, we can also construct an N -periodic replication using the modulo- N operator as $\tilde{x}[n] = x[n]_N$. The modulo- N operator is discussed in greater detail in Sec. 9.2.1.

[†]Also notice that the DFT analysis Eq. (9.2) produces an N -periodic signal $X[k]$.

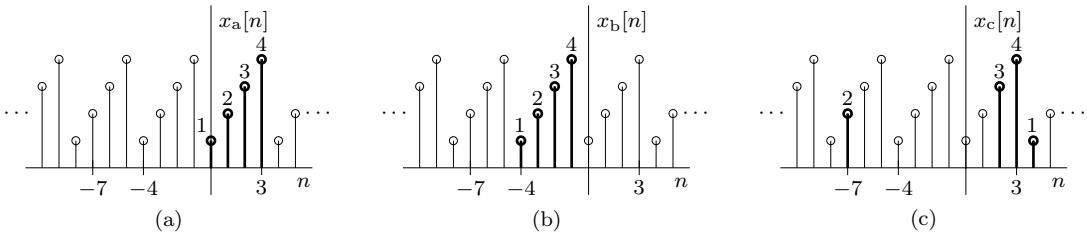


Figure 9.7: Three ($N = 4$)-point signals with equivalent periodic replications.

given $Y[k]$, we are overwhelmed by an infinite number of possible answers. For a signal $x[n]$ with a range of $0 \leq n \leq N - 1$, we have defined the corresponding DFT $X[k]$ to be the set of N uniform samples of $X(\Omega)$, the DTFT of $x[n]$. If an N -point sequence lies partly or wholly outside the range $0 \leq n \leq N - 1$, then $X[k]$, the set of N uniform samples of its DTFT, exists, but it cannot be called its DFT because the N -point DFT for such a signal is not defined. With these thoughts in mind, the DFT $X[k]$ of the signal in Fig. 9.6a is identical to the DTFT samples $X[k]$ for the signals in Figs. 9.6b and 9.6c, but $X[k]$ cannot properly be called the DFT of either of these signals.

Will It Play in Peoria?

As seen here, meaningful use of the DFT requires uniqueness that, in turn, demands signals be restricted to a certain range such as $0 \leq n \leq N - 1$. But, with such a severe restriction, can the DFT have any practical utility? Routine operations such as time shifting and reflection would make the resulting signal go outside the range $0 \leq n \leq N - 1$, and the operated signal would have no DFT. This state of affairs would render the DFT practically useless to many signals applications. So what use is the vaunted DFT if it won't play in Peoria? As it turns out, the DFT performs splendidly as long as we adopt modified operators that preserve the needed range. We shall soon see that the key to maintaining a range between 0 and $N - 1$ lies with the modulo- N shifting operation.

Origins of the Nonuniqueness Problem

One wonders why the problem of nonuniqueness arises in the DFT when the DFT is just a set of samples of the DTFT that has a one-to-one relationship to $x[n]$ without any restrictions on $x[n]$. The nonuniqueness problem originates not from the DTFT itself but rather from the sampling of the DTFT. To understand this point, let $X(\Omega)$ be the DTFT of a length- N $x[n]$. The DTFT of $x[n - mN]$ is $X(\Omega)e^{-j\Omega mN}$. Clearly, the DTFTs of $x[n]$ and $x[n - mN]$ are distinct. They differ by a linear phase term $e^{-j\Omega mN}$, which corresponds to a time delay of mN samples. But, as seen earlier, $X[k]$ for $x[n]$ and $x[n - mN]$ are identical. This means that the samples of these two distinct DTFTs $X(\Omega)$ and $X(\Omega)e^{-j\Omega mN}$, taken at intervals of Ω_0 , are identical. Why?

This can be explained by observing that the DTFTs of $x[n]$ and $x[n - mN]$ differ only by a linear phase shift factor $e^{-j\Omega mN}$. The samples of this factor at intervals of Ω_0 are $e^{-j\Omega_0 kmN} = e^{-j2\pi km} = 1$. Because all the samples of this term happen to be unity, the samples of $X(\Omega)$ and $X(\Omega)e^{-j\Omega mN}$ are identical, implying that $X[k]$ for $x[n]$ and $x[n - mN]$ are identical. A similar conclusion applies when individual samples of $x[n]$ are shifted by different amounts, as long as the shift of each sample is some integer multiple of N . This is left as an exercise for the reader.

▷ Drill 9.5 (Nonuniqueness of DTFT Samples)

As in Drill 9.1, the 3-point signal $x[n]$ specified as $x[0] = 2$ and $x[1] = x[2] = 1$ has DFT given by $X[0] = 4$ and $X[1] = X[2] = 1$. Show that this DFT is identical to the samples of the DTFT of the 3-point signal $y[n]$ specified as $y[0] = 2$ and $y[-1] = y[1] = 1$. Show that the DTFTs of $x[n]$ and $y[n]$ are not identical.



9.2.1 Modulo- N Operation

In DFT studies, the modulo- N operation proves very useful. We can think of the modulo- N value of number n as the remainder of the division of n by N , assuming nonnegative integers n and N . Thus, for example, the modulo-4 value of 7 is 3. The modulo- N value of n may also be obtained by subtracting (or adding) some integer multiple of N to n such that the resulting number lies in the range $0 \leq n \leq N - 1$. For instance, the modulo-4 values of 4 and 10 are $4 - (1 \times 4) = 0$ and $10 - (2 \times 4) = 2$, respectively.

Mathematically, we denote the modulo- N value of n as $\langle n \rangle_N$. Using $0 \leq n \leq 10$, the following table shows n modulo- N for $N = 3, 4$, and 7:

n	0	1	2	3	4	5	6	7	8	9	10
$\langle n \rangle_3$	0	1	2	0	1	2	0	1	2	0	1
$\langle n \rangle_4$	0	1	2	3	0	1	2	3	0	1	2
$\langle n \rangle_7$	0	1	2	3	4	5	6	0	1	2	3

Observe that $\langle 0 \rangle_N = 0$. Moreover, $\langle 1 \rangle_3 = 1$ because $1 - (0 \times 3) = 1$. In fact, $\langle n \rangle_N = n$ as long as $0 \leq n < N$. The modulo- N operation works equally well with negative arguments. For example, the modulo-3 values of $-4, -5$, and -6 are 2, 1, and 0, respectively, and the modulo-4 values of $-4, -5$, and -6 are 0, 3, and 2, respectively. The modulo- N value of n can be computed using an integer value r according to

$$\langle n \rangle_N = n - rN, \quad \text{where} \quad rN \leq n \leq (r + 1)N - 1. \quad (9.12)$$

Observe that the modulo- N value of any integer n always remains within the range from 0 to $N - 1$. Furthermore, notice that $\langle n \rangle_N$ is an N -periodic function of n . As we shall see shortly, this connection between periodicity and $\langle n \rangle_N$ is very useful to the DFT.

Modulo- N Shift

One useful application of the modulo- N operation is to help shift an arbitrary N -point sequence into the range 0 to $N - 1$. The *modulo- N shift* relocates the samples of an N -point signal, no matter its range, to the DFT-favored range of 0 to $N - 1$. To find out which sample goes where, we just perform a modulo- N operation on each sample's time index. Let us consider an example.

▷ Example 9.6 (Modulo- N Shift)

Find the signal $x[n]$ that is the modulo- N shift of the ($N = 6$)-point sequence $w[n]$, shown in Fig. 9.8. That is, modulo- N shift the samples of $w[n]$ so that $x[n]$ occupies $0 \leq n \leq N - 1$.

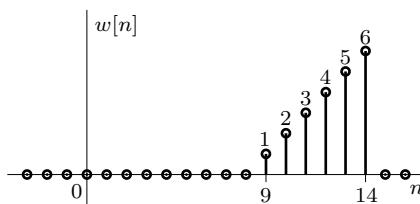


Figure 9.8: A 6-point signal $w[n]$ with range $9 \leq n \leq 14$.

The sequence $w[n]$ lies over the range $9 \leq n \leq 14$. The modulo-6 shift of the first sample $w[9]$ moves to $n = 3$ since $\langle 9 \rangle_6 = 3$. Over the entire $0 \leq n \leq 14$ range, the samples shift according to

n	9	10	11	12	13	14
$\langle n \rangle_6$	3	4	5	0	1	2

Figure 9.9a shows $x[n]$, the resulting modulo-6 shift of $w[n]$. Notice that the range of $x[n]$ is $0 \leq n \leq N - 1$, as desired.

There is another particularly simple way to compute $x[n]$. Figure 9.9b shows $\tilde{w}[n]$, the 6-periodic replication of $w[n]$. By simply retaining $\tilde{w}[n]$ over $0 \leq n \leq N - 1$, we likewise obtain the modulo- N shift of $w[n]$ (Fig. 9.9a).

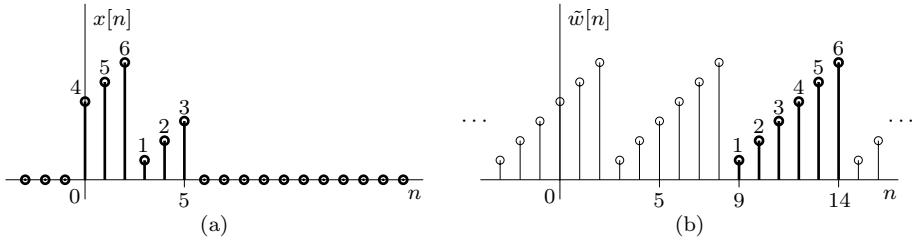


Figure 9.9: (a) Modulo- N shift of $w[n]$ and (b) N -periodic replication of $w[n]$.

Example 9.6 □

Effect of a Modulo- N Shift on DTFT Samples

Why do we bother with the modulo- N shift of a sequence? We can show that the N uniform samples of the DTFT of a length- N sequence remain unchanged under the modulo- N shift of the sequence. This follows from the fact that a modulo- N shift of a sample involves shifting it by some integer multiple of N . We saw earlier that such shifts have no effect on the N samples of a signal's DTFT.

In Ex. 9.6, the modulo- N shift of $w[n]$ yields $x[n]$. Hence, $W[k] = X[k]$, where $W[k]$ and $X[k]$ are the N uniform samples of the DTFT $W(\Omega)$ and $X(\Omega)$, respectively. As noted earlier, observe that $X[k]$ is the DFT of $x[n]$, but, strictly speaking, $W[k]$ cannot be labeled as the DFT of $w[n]$ because the DFT of a signal like $w[n]$ (with range outside $0 \leq n \leq N - 1$) is not defined.

There is also a dual result. If we consider N samples $W[k]$ that lie partly or wholly outside the range $0 \leq k \leq N - 1$, we can modulo- N shift $W[k]$ to obtain the set $X[k]$ that lies within the range $0 \leq k \leq N - 1$. Using an identical argument, we can show that the IDTFTs of $X[k]$ and $W[k]$ are identical.

▷ Drill 9.6 (Modulo- N Shift and DTFT Sample Equivalence)

Recall from Drill 9.5 that the three DTFT samples of $x[n]$, defined as $x[0] = 2$ and $x[1] = x[2] = 1$, are identical to the three DTFT samples of $y[n]$, defined as $y[0] = 2$ and $y[-1] = y[1] = 1$. Show that a modulo-3 shift of $y[n]$ yields $x[n]$. Find another 3-point signal $z[n]$ that is modulo-3 shift equivalent to $x[n]$.

□

9.2.2 Circular Representation of an N -Length Sequence

As we have seen, there is advantage and utility to viewing N -point signals as being N -periodic. Still, it can be a bit cumbersome to work with and display such signals. Additionally, a periodic extension of an N -point sequence seems to be adding many new samples, even if those samples are just copies of the originals. It seems somehow absurd to change an N -point signal into one of infinite duration.

A circular representation avoids all these difficulties while retaining the advantages of a periodic view. It adds no samples and is no more difficult to display than any N -point sequence. With a circular representation, time reversal and shifting are naturally modulo- N operations that retain a signal's $0 \leq n \leq N - 1$ range, which is not only desirable but necessary for DFT analysis. Still, despite their apparent differences, the circular representation of an N -point signal $x[n]$ is really identical to the N -periodic replication of the same N -point signal.

In the circular representation of a length- N sequence, data is displayed on a circular scale rather than a linear scale. We locate the N samples of a sequence $x[n]$ uniformly along the circumference of a circular dial in the clockwise direction, usually starting with $x[0]$ at the 0 hour. Figure 9.10a shows a length-6 sequence $x[n]$, and Fig. 9.10b shows the circular representation of $x[n]$. For this length-6 sequence, the points $n = 0, 1, 2, 3, 4$, and 5 are located at the 0, 2, 4, 6, 8, and 10 o'clock positions, respectively. While Fig. 9.10b explicitly labels n for clarity, such labeling is not necessary as the time index can be inferred by position.

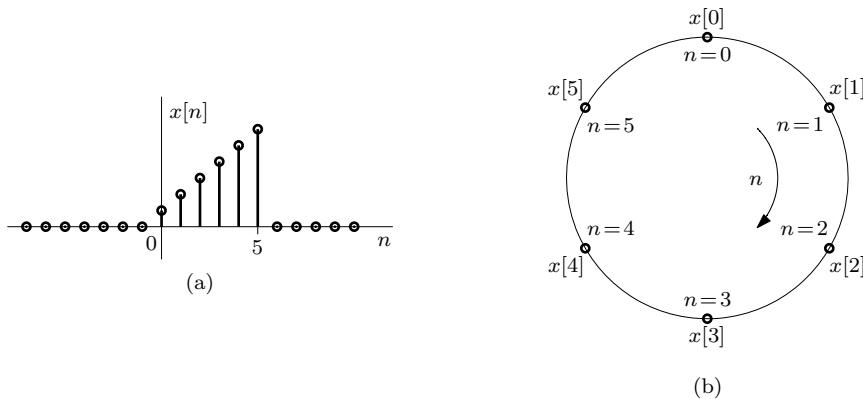


Figure 9.10: Length-6 signal $x[n]$: (a) linear representation and (b) circular representation.

▷ Drill 9.7 (Circular Representations of 3-Point Signals)

Show that the circular representations of the 3-point signals $x[n]$ and $y[n]$ in Drill 9.6 are identical.

△

Modulo- N Shift of $x[-n]$: The Modulo- N Reflection $x[\langle -n \rangle_N]$

If an N -point sequence $x[n]$ exists for $0 \leq n \leq N - 1$, then the time-reversed sequence $x[-n]$ will lie not in the range $0 \leq n \leq N - 1$ but in the range $-(N - 1) \leq n \leq 0$. Instead, we prefer the modulo- N signal corresponding to $x[-n]$, which allows us to retain the range $0 \leq n \leq N - 1$. This signal, denoted $x[\langle -n \rangle_N]$, is called the *modulo- N reflection* of $x[n]$. Using Eq. (9.12), the sequence $x[\langle -n \rangle_N]$ is obtained by shifting the samples of $x[-n]$ to their modulo- N values. Alternatively, we can obtain $x[\langle -n \rangle_N]$ as the periodic replication of $x[-n]$. Since periodic replication is no different than a circular representation, it is also possible to find $x[\langle -n \rangle_N]$ using the circular representation of $x[-n]$. Let us find $x[\langle -n \rangle_N]$ in all three of these ways.

Using $r = 0$ for $n = 0$ and $r = -1$ for $1 \leq n \leq N - 1$, Eq. (9.12) yields[†]

$$x[\langle -n \rangle_N] = \begin{cases} x[0] & n = 0 \\ x[N - n] & 1 \leq n \leq N - 1 \end{cases}. \quad (9.13)$$

[†]Taken over all n , notice that $x[\langle -n \rangle_N]$ yields an N -periodic signal, which further strengthens a periodic perspective of N -point sequences.

To illustrate, consider the 6-point signal $x[n]$ depicted in Fig. 9.10a. Figure 9.11a shows the reflection $x[-n]$. Over $0 \leq n \leq 5$ (heavy lines), Fig. 9.11b shows $x[\langle -n \rangle_N]$ obtained by transporting the samples of $x[-n]$ to their modulo-6 locations, as per Eq. (9.13). Note that $x[N-n]$ is $x[-n]$ right shifted by N . Thus, except for the sample at $n = 0$, which remains unchanged, the rest of $x[-n]$ are right shifted by $N = 6$. The sample at -1 goes to $n = 5$ (the modulo-6 value of -1), the sample at $n = -2$ goes to $n = 4$, and so on.

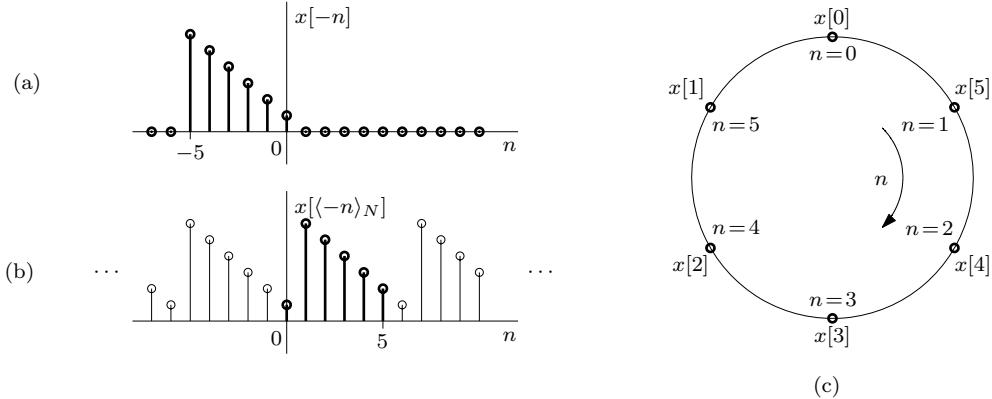


Figure 9.11: Modulo- N reflection: (a) $x[-n]$, (b) $x[\langle -n \rangle_N]$, and (c) circular representation of $x[\langle -n \rangle_N]$.

In the second method, we exploit the connection between a modulo- N shift and periodic replication to obtain $x[\langle -n \rangle_N]$. Using Eq. (9.11) and the footnote on page 569, we have

$$x[\langle -n \rangle_N] = \sum_{k=-\infty}^{\infty} x[-n + kN]. \quad (9.14)$$

Equation (9.14) just defines a periodic replication of $x[-n]$. To illustrate, the periodic replication of $x[-n]$ in Fig. 9.11a is shown in Fig. 9.11b, which is identical to the first method considered.

In the third method, we use a circular representation of $x[-n]$, as depicted in Fig. 9.11c. This can be done by placing the samples of $x[n]$ in a *counterclockwise direction* along the dial to represent $x[-n]$. We now read this diagram in a *clockwise direction* to obtain $x[\langle -n \rangle_N]$. We read the values at $n = 0, 1, 2, 3, 4$, and 5 as $x[0], x[5], x[4], x[3], x[2]$, and $x[1]$, respectively. This matches Fig. 9.11b, the result of the first two methods. Notice that just as with a linear representation, the circular representation of time reversal is just a reflection about the vertical axis.

Because we can obtain $x[\langle -n \rangle_N]$ from $x[-n]$ through a modulo- N shift, the N uniform samples of the DTFT of $x[-n]$ are identical to the DTFT of $x[\langle -n \rangle_N]$. It is in this sense that we can perform time reflection and maintain the range $0 \leq n \leq N - 1$ required of the DFT. Next, we consider another case of a modulo- N shift so as to accommodate $x[n - m]$.

▷ Drill 9.8 (Modulo- N Reflection)

Using the three methods described, find the modulo- N reflection $x[\langle -n \rangle_N]$ of the 3-point signal $x[n]$ specified by $x[1] = 1$, $x[2] = 2$, and $x[3] = 3$.

△

Modulo- N Shift of $x[n - m]$: The Circular Shift $x[\langle n - m \rangle_N]$

Consider an N -point sequence $x[n]$ in the range $0 \leq n \leq N - 1$. A linear shift of this signal by m samples yields the signal $x[n - m]$, which is a length- N signal with a range $m \leq n \leq m + N - 1$.

As with the case of reflection, we instead seek a form that will retain the range $0 \leq n \leq N - 1$. A modulo- N shift of $x[n - m]$, also called a *circular shift*, accomplishes exactly this task. Further, because $x[\langle n - m \rangle_N]$ is obtained from $x[n - m]$ through a modulo- N shift, the set of N uniform samples of the DTFT of $x[n - m]$ is identical to the DFT of $x[\langle n - m \rangle_N]$.

If $0 \leq m \leq N - 1$, a modulo- N shift operation does not affect the samples $x[n - m]$ over the range $m \leq n \leq N - 1$. That is, $x[\langle n - m \rangle_N] = x[n - m]$ for $m \leq n \leq N - 1$. The remaining samples over the range $N \leq n < N + m$ are shifted to the range $0 \leq n < m$ by the modulo- N operation. Over $0 \leq n \leq N - 1$, Eq. (9.12) yields

$$x[\langle n - m \rangle_N] = \begin{cases} x[n - m] & 0 \leq m \leq n \leq N - 1 \\ x[N + n - m] & 0 \leq n < m \leq N - 1 \end{cases}. \quad (9.15)$$

We can also compute the modulo- N shift of $x[n - m]$, for any value m , by computing the N -periodic replication of $x[n - m]$ and then retaining the samples over $0 \leq n \leq N - 1$. That is,

$$x[\langle n - m \rangle_N] = \sum_{k=-\infty}^{\infty} x[n - m - kN], \quad 0 \leq n \leq N - 1. \quad (9.16)$$

As a third method, and the one that is probably most intuitive, we simply adopt a circular representation of $x[n - m]$ to obtain $x[\langle n - m \rangle_N]$.

To illustrate these three methods, let us consider an $m = 2$ shift of the 6-point signal $x[n]$ depicted in Fig. 9.10a. Figure 9.12a shows the shift $x[n - m]$. Over $0 \leq n \leq 5$ (heavy lines), Fig. 9.12b shows $x[\langle n - m \rangle_N]$ obtained by transporting the samples of $x[n - m]$ to their modulo-6 locations, as per Eq. (9.15). In this case, the samples $2 \leq n \leq 5$ of $x[n - m]$ do not move, but samples $n = 6$ and $n = 7$ of $x[n - m]$ are moved left by 6 to their modulo- N positions. Figure 9.12b also shows the periodic replication of $x[n - m]$, which, over $0 \leq n \leq N - 1$, matches the modulo- N shift of $x[n - m]$.

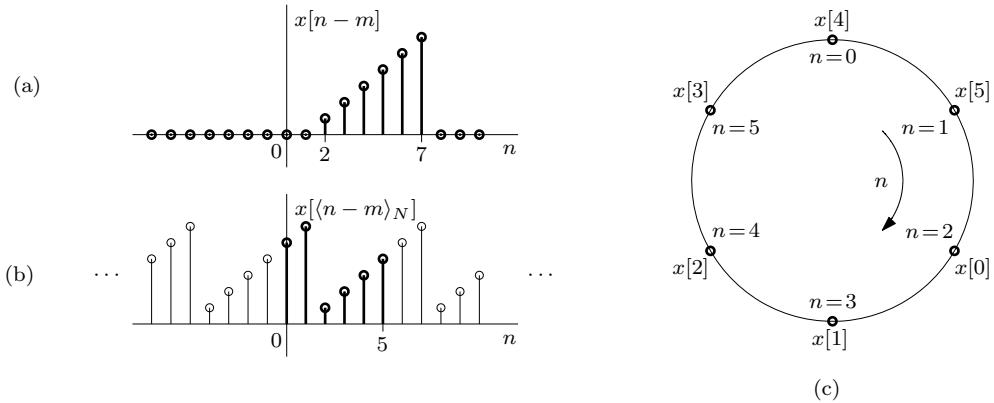


Figure 9.12: Circular shift by $m = 2$: (a) $x[n - m]$, (b) $x[\langle n - m \rangle_N]$, and (c) circular representation of $x[\langle n - m \rangle_N]$.

To understand the circular representation, let us first consider how we might interpret $x[n - m]$ on a linear scale. If $m > 0$, we shift $x[n]$ right by sliding it m positions rightward along our scale. If $m < 0$, we shift $x[n]$ left by sliding it $|m|$ positions leftward along our scale. The circular shift $x[\langle n - m \rangle_N]$ is accomplished in a similar fashion using the circular representation, except that we rotate rather than slide our values. If $m > 0$ (right shift), we rotate our values clockwise by m positions. If $m < 0$ (left shift), we rotate our values counterclockwise. In the present example, $m = 2$ is positive, so we rotate the circular representation (Fig. 9.10b) 2 positions clockwise, as shown in Fig. 9.12c. The results again match those of the two earlier methods. That is, at $n = 0$ (0 hour), we obtain $x[4]$; at $n = 1$, we obtain $x[5]$; at $n = 2$, we obtain $x[0]$; and so forth.

A Game of Musical Chairs

Observe that shifting an N -point signal $x[n]$ by one unit results in $x[n - 1]$ for which all the samples are right shifted by one unit. The slot $n = 0$ becomes vacant, and the sample at $n = N$ falls outside the range $0 \leq n \leq N - 1$. With a modulo- N shift, however, the sample at $n = N$ comes back to $n = 0$. This is the circular representation, where all samples are placed in a circle and rotated one unit, somewhat similar to the game of musical chairs. A rotation by one unit causes all the samples to shift by one position and the last sample to become the first sample. Every unit shift repeats this process. From the circular representation, it is easy to see why $x[\langle n - m \rangle_N]$ is called a *circular shift*.

Using a 6-point signal $x[n]$, Fig. 9.13 shows $x[\langle n - m \rangle_N]$ for $0 \leq m \leq N - 1$. For the $m = 0$ case of Fig. 9.13a, the signal has no shift at all. Shifting by $m = 1$, we rotate Fig. 9.13a clockwise by one position to obtain Fig. 9.13b. Rotating once again yields the $m = 2$ shift of Fig. 9.13c. The process continues, one step at a time, until we reach the $m = 5$ shift of Fig. 9.13f. Shifting once more will return us to Fig. 9.13a. Thus, a circular shift of $m = N$ is no different than not shifting at all. In fact, the N shifts found using $0 \leq m \leq N - 1$ cover all possible circular shifts of an N -point sequence. Thus, a circular shift by any value m is equivalent to a circular shift by $\langle m \rangle_N$, which occupies this range from 0 to $N - 1$. In this way, we can adapt Eq. (9.15) to work for all possible m .

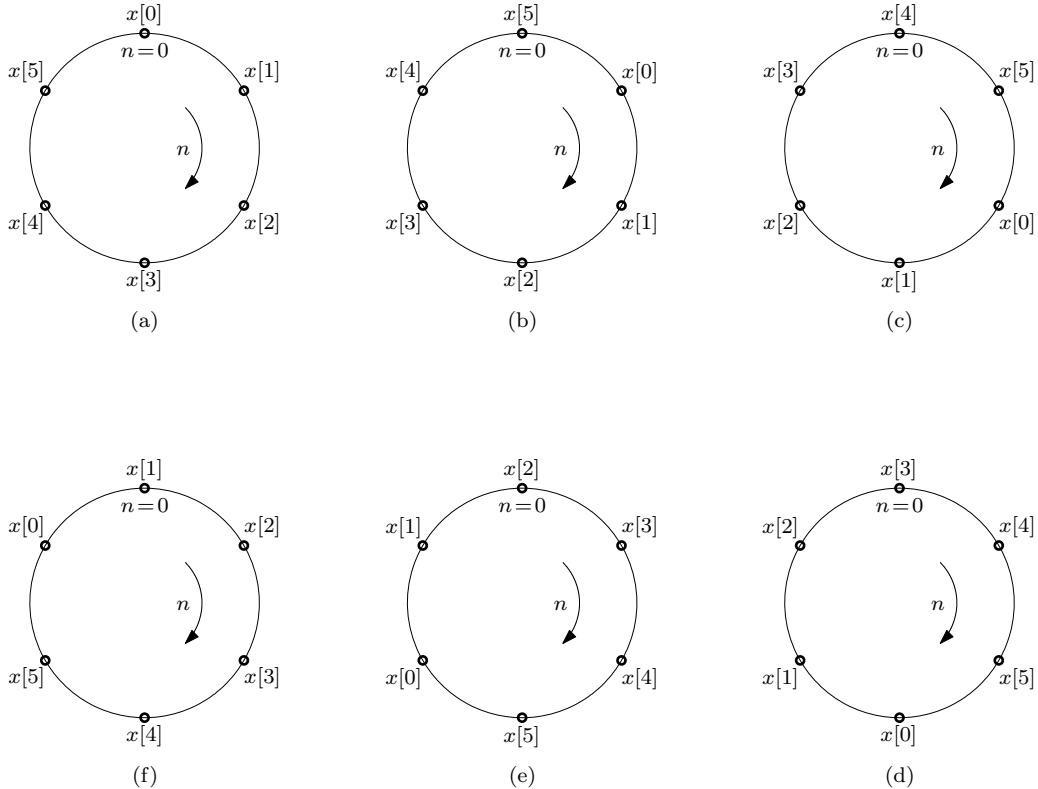


Figure 9.13: Circular shift $x[\langle n - m \rangle_N]$: (a) $m = 0$, (b) $m = 1$, (c) $m = 2$, (d) $m = 3$, (e) $m = 4$, and (f) $m = 5$.

▷ **Drill 9.9 (Circular Shift)**

Using the three methods described, find the $m = 4$ circular shift $x[\langle n - 4 \rangle_N]$ of the 3-point signal $x[n]$ specified by $x[0] = 2$ and $x[1] = x[2] = 1$. What is the result for a circular shift of $m = -1019$?

△

▷ **Drill 9.10 (Combined Modulo- N Reflection and Shift)**

Using any method, find $x[\langle -n - 2 \rangle_N]$ of the 6-point signal $x[n]$ shown in Fig. 9.10a.

△

Views of the DFT

If we take an N -point signal $x[n]$ to be zero outside the range 0 to $N - 1$, the DFT can be viewed as the N uniform samples of the DTFT of $x[n]$, as we have seen. This offers useful insight since we can leverage all our knowledge of the DTFT to better understand the DFT. We have also seen that viewing $x[n]$ as an N -periodic signal is useful in cases such as modulo- N shifting. Thus, it seems that we can view $x[n]$ as an N -periodic signal or as being 0 outside the range of 0 to $N - 1$. Which view is correct?

Actually, the question itself is faulty. In the most basic sense, the DFT maps N points in time to N points in frequency. The DFT is not at all concerned with anything other than these N points, which we take to occupy the range 0 to $N - 1$. We can view our N -point signal as 0 outside of the range 0 to $N - 1$, or we can view it as being N -periodic. Since we only use signal values in the range $0 \leq n \leq N - 1$, values outside $0 \leq n \leq N - 1$ are irrelevant to the structure of the DFT analysis and synthesis equations. Thus, we are free to adopt whatever view is most useful. Sometimes it is best to view our signals as being 0 outside the range 0 to $N - 1$, and sometimes it is best to view our signals as being N -periodic. A periodicity perspective, despite its phantom nature, is particularly useful in understanding DFT properties, and it deserves additional attention.

Phantom Periodicity

Ignoring restrictions on range, observe the interesting fact that the DFT Eqs. (9.2) and (9.5) defining $X[k]$ and $x[n]$ are both N -periodic. That is,

$$\begin{aligned} X[k] &= X[k + N] && \text{for all } k \\ \text{and} \quad x[n] &= x[n + N] && \text{for all } n. \end{aligned} \tag{9.17}$$

Here, periodicity is guaranteed since both $X[k]$ and $x[n]$ are exclusively constructed of scaled sums of functions $e^{\pm j\Omega_0 n k}$ that are N -periodic in both k and n .

Recall that the DTFT $X(\Omega)$ is periodic. Viewed as the N uniform samples of the DTFT, the periodicity of $X[k]$ is therefore understandable. It is more surprising that $x[n]$ also seems periodic, particularly since we derived the DFT by assuming $x[n]$ to be zero outside $0 \leq n \leq N - 1$. Again, the key to this mystery is that the DFT is only concerned with samples in the range 0 to $N - 1$. How we view $x[n]$ (or $X[k]$) outside this range is irrelevant to the inner workings of the DFT and its inverse. We can adopt whatever view best helps our understanding and intuition of the DFT. Later, in Sec. 9.8, we shall rely on a periodic view of $x[n]$ to derive the discrete-time Fourier series (DTFS) equations, and the phantom periodicity of $x[n]$ will seem even more natural.

One of the consequences of the phantom periodicity of $x[n]$ in Eq. (9.17) is that $x[0] = x[N]$. Strictly speaking, as observed earlier, this result is meaningless because $x[N]$ is not one of the N values that describes our N -point signal $x[n]$. However, if we compute $x[N]$ using Eq. (9.5), we find it to be $x[0]$. It is in this sense that we shall accept the existence of $x[N]$ and other values outside

the range $0 \leq n \leq N - 1$. An interesting consequence of accepting this result is that it simplifies the expressions for $x[\langle -n \rangle_N]$ and $X[\langle -n \rangle_N]$. As seen from Eq. (9.13),

$$x[\langle -n \rangle_N] = \begin{cases} x[0] & n = 0 \\ x[N-n] & 1 \leq n \leq N-1 \end{cases}.$$

If we accept the fact that $x[0] = x[N]$, then this equation can be simplified to

$$x[\langle -n \rangle_N] = x[N-n], \quad 0 \leq n \leq N-1. \quad (9.18)$$

In a similar way, we obtain

$$X[\langle -k \rangle_N] = X[N-k], \quad 0 \leq k \leq N-1. \quad (9.19)$$

▷ Drill 9.11 (Simplified Modulo-N Reflection)

Use Eq. (9.18) and $x[N] = x[0]$ to find $x[\langle -n \rangle_N]$ for a length-3 signal $x[n]$ specified by $x[0] = 3$, $x[1] = 2$, and $x[2] = 1$.

△

9.3 Properties of the DFT

Properties are useful to better understand and utilize the DFT. Because the DFT can be viewed as samples of the corresponding DTFT, DFT properties bear similarity to those of the DTFT. The proofs of several DFT properties follow directly from the corresponding properties for the DTFT and viewing the DFT as samples of the DTFT, that is, $X[k] = X(k\Omega_0)$. These proofs can also be derived directly from the definition of the DFT. In some cases, DFT properties are best understood from a periodicity perspective. We shall use both perspectives while discussing DFT properties.

9.3.1 Duality Property

Much like how the near-perfect symmetry of the FT produces the duality property of Eq. (1.80), the near-perfect symmetry between the DFT analysis and synthesis equations also produces a duality property. Thus, for any result or relationship between $x[n]$ and $X[k]$, there exists a dual result or relationship that is obtained by interchanging the roles of $x[n]$ and $X[k]$. More formally, the DFT duality property states that

$$\text{if } x[n] \longleftrightarrow X[k], \text{ then } X[n] \longleftrightarrow Nx[\langle -k \rangle_N]. \quad (9.20)$$

There is a striking relationship between the DFT duality property of Eq. (9.20) and the FT duality property of Eq. (1.80) on page 51. A proof of this property is outlined in Prob. 9.8-1 and is also found in the literature, such as [4].

9.3.2 Linearity Property

The DFT is linear. That is, if

$$x[n] \longleftrightarrow X[k] \quad \text{and} \quad y[n] \longleftrightarrow Y[k],$$

then, for any constants a and b ,

$$ax[n] + by[n] \longleftrightarrow aX[k] + bY[k]. \quad (9.21)$$

The proof is trivial and follows directly from the definition of DFT. Here, we are assuming that the lengths of both $x[n]$ and $y[n]$ are identical. If the lengths of $x[n]$ and $y[n]$ are different, then we only need to pad the shorter signal with zeros to use the linearity property. Of course, the linearity property easily extends to any finite number of terms.

9.3.3 Complex-Conjugation Property

Let us start with the frequency-domain complex-conjugation property. This property states that

$$x^*[\langle -n \rangle_N] \longleftrightarrow X^*[k]. \quad (9.22)$$

Using our earlier argument of letting $x[N] = x[0]$ leads to $x^*[\langle -n \rangle_N] = x^*[N - n]$. Hence, Eq. (9.22) simplifies to

$$x^*[N - n] \longleftrightarrow X^*[k]. \quad (9.23)$$

To prove this property, note that $x^*[\langle -n \rangle_N]$ is the modulo- N shift of $x^*[-n]$. Thus, the DFT of $x^*[\langle -n \rangle_N]$ is the same as the N uniform samples of the DTFT for $x^*[-n]$. Now, from Eqs. (6.14) and (6.15), the DTFT of $x^*[-n]$ is $X^*(\Omega)$. The N uniform samples of $X^*(\Omega)$ are $X^*[k]$. Hence, it follows that

$$x^*[\langle -n \rangle_N] \longleftrightarrow X^*[k].$$

Using the similar reasoning, we can find that the time-domain complex-conjugation property is

$$x^*[n] \longleftrightarrow X^*[\langle -k \rangle_N] = X^*[N - k]. \quad (9.24)$$

When $x[n]$ is real, $x^*[n] = x[n]$, and this equation reduces to

$$x[n] \longleftrightarrow X^*[N - k].$$

This implies that

$$X[k] = X^*[N - k]. \quad (9.25)$$

This is the (midpoint) *conjugate-symmetry property* of the DFT for real $x[n]$, which has been verified in Exs. 9.1 and 9.2.

▷ Drill 9.12 (Conjugate-Antisymmetry Property)

Prove that if $x[n]$ is imaginary, then $X[k]$ is midpoint conjugate antisymmetric, $X[k] = -X^*[N - k]$.

▫

9.3.4 Time-Reversal Property

The time-reversal property, which also serves as the frequency-reversal property, states that

$$\text{if } x[n] \longleftrightarrow X[k], \text{ then } x[\langle -n \rangle_N] \longleftrightarrow X[\langle -k \rangle_N] = X[N - k]. \quad (9.26)$$

To prove this property, recall that $x[\langle -n \rangle_N]$ is obtained by modulo- N shifting $x[-n]$. Hence, the set of N uniform samples of the DTFT of $x[-n]$ is the same as the DFT of $x[\langle -n \rangle_N]$. From Eq. (6.15), we find that $x[-n] \iff X(-\Omega)$, and the uniform sample set of $X(-\Omega)$ is $X[-k]$, which is equal to $X[N - k]$ because of the N -periodicity of $X[-k]$. Since $X[N - k] = X[\langle -k \rangle_N]$, the result follows. We can also use the simplified expression $x[N - n]$ for $x[\langle -n \rangle_N]$ to express an alternate form of Eq. (9.26) as

$$x[N - n] \longleftrightarrow X[N - k].$$

9.3.5 Circular Shifting Properties

The circular time-shifting property states that

$$\text{if } x[n] \longleftrightarrow X[k], \text{ then } x[\langle n - m \rangle_N] \longleftrightarrow X[k]e^{-j\Omega_0 km} \text{ for integer } m. \quad (9.27)$$

We again turn to the DTFT to prove this property. Because $x[\langle n - m \rangle_N]$ is obtained by a modulo- N shift of $x[n - m]$, the DFT of $x[\langle n - m \rangle_N]$ is the same as the N uniform samples of the DTFT of

$x[n - m]$. From Eq. (6.16), we obtain $x[n - m] \iff X(\Omega)e^{-j\Omega m}$. Hence, the N samples of the DTFT are $X[k]e^{-j\Omega_0 km}$, and the result in Eq. (9.27) follows.

The dual of the circular time-shifting property is the circular frequency-shifting property. This property states that

$$\text{if } x[n] \longleftrightarrow X[k], \text{ then } x[n]e^{j\Omega_0 mn} \longleftrightarrow X[\langle k - m \rangle_N] \text{ for integer } m. \quad (9.28)$$

To prove this property, let us use Eq. (9.2) to find the DFT of $x[n]e^{j\Omega_0 mn}$ as

$$\sum_{n=0}^{N-1} x[n]e^{j\Omega_0 mn}e^{-j\Omega_0 kn} = \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0(k-m)n} = X[\langle k - m \rangle_N].$$

The final step follows from the periodicity of $X[k]$.

▷ Example 9.7 (Circular Time Shifting)

Let a 3-point signal $x[n]$, defined as $x[0] = 2$ and $x[1] = x[2] = 1$, have DFT $X[k]$. Through explicit computation, show that $X[k]e^{-j\Omega_0 k}$ results in a one-unit circular shift of $x[n]$.

Let us use MATLAB to perform the necessary computations.

```
01 N = 3; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1); x = [2;1;1];
02 DN = exp(-1j*Omega0*k'*n); X = DN*x;
03 1/N*conj(DN)*(X.*exp(-1j*Omega0*k))
ans = 1 2 1
```

As in Ex. 9.4, line 01 establishes basic parameters. Line 02 defines the 3-point DFT matrix \mathbf{D}_N and computes the DFT of $x[n]$. Recalling that $\frac{1}{N}\mathbf{D}_N^*$ is the inverse DFT matrix, line 03 computes the IDFT of $X[k]e^{-j\Omega_0 k}$. The resulting sequence [1, 2, 1] is clearly a one-unit circular shift of the original sequence [2, 1, 1].

Example 9.7 ◀

9.3.6 Circular Convolution Properties

For two signals $x[n]$ and $y[n]$, each of length N , circular convolution is defined by

$$x[n] \circledast y[n] = \sum_{m=0}^{N-1} x[m]y[\langle n - m \rangle_N] = \sum_{m=0}^{N-1} y[m]x[\langle n - m \rangle_N]. \quad (9.29)$$

Circular convolution is like ordinary convolution except that it retains the 0 to $N - 1$ range by using the modulo- N operation. Circular convolution is very much a natural extension of regular convolution for N -point signals restricted to the range 0 to $N - 1$. Next, we introduce two circular convolution properties. Later, we shall provide a graphical interpretation of circular convolution and demonstrate some applications of circular convolution that are useful in DSP.

Let us begin with two N -point DFT pairs

$$x[n] \longleftrightarrow X[k] \quad \text{and} \quad y[n] \longleftrightarrow Y[k].$$

The time-domain circular convolution property states that

$$x[n] \circledast y[n] \longleftrightarrow X[k]Y[k]. \quad (9.30)$$

Equivalently, Eq. (9.30) tells us that by multiplying the DFTs of two N -point signals, the time-domain result is the circular convolution of the original two signals.

The proof of this property follows from the definition of the DFT,

$$\begin{aligned}\text{DFT}\{x[n] \circledast y[n]\} &= \sum_{n=0}^{N-1} \left[\sum_{m=0}^{N-1} x[m]y[\langle n-m \rangle_N] \right] e^{-j\Omega_0 kn} \\ &= \sum_{m=0}^{N-1} x[m] \left[\sum_{n=0}^{N-1} y[\langle n-m \rangle_N] e^{-j\Omega_0 kn} \right] \\ &= \sum_{m=0}^{N-1} x[m] [Y[k]e^{-j\Omega_0 km}] \\ &= Y[k] \sum_{m=0}^{N-1} x[m] e^{-j\Omega_0 km} \\ &= X[k]Y[k].\end{aligned}$$

The dual of the time-domain circular convolution property is the frequency-domain circular convolution property. This property states that

$$x[n]y[n] \longleftrightarrow \frac{1}{N} X[k] \circledast Y[k]. \quad (9.31)$$

This property is sometimes called the *multiplication property*, which references the time-domain side of Eq. (9.31). The proof of this property is similar to that for the time-domain circular convolution property.

9.3.7 Circular Correlation Property

Just as we extended linear convolution to circular convolution, we can use the modulo- N operation to extend linear correlation to circular (or cyclic) correlation. The circular time-domain correlation between N -point signals $x[n]$ and $y[n]$ is defined as

$$\rho_{x,y}[l] = \sum_{m=0}^{N-1} x[\langle m+l \rangle_N]y^*[m]. \quad (9.32)$$

The circular correlation function $\rho_{x,y}[l]$ is easily expressed in terms of circular convolution as

$$\rho_{x,y}[l] = x[l] \circledast y^*[\langle -l \rangle_N]. \quad (9.33)$$

Applying the complex-conjugation, time-reversal, and circular convolution properties yields the circular correlation property

$$\rho_{x,y}[l] = x[l] \circledast y^*[\langle -l \rangle_N] \longleftrightarrow X[k]Y^*[k]. \quad (9.34)$$

We stress that correlation order is important. That is, $\rho_{x,y}[l]$ does not generally equal $\rho_{y,x}[l]$.

A Frequency-Domain Representation of Signal Energy

Setting $x[n] = y[n]$ and $l = 0$ in Eq. (9.32) yields signal energy

$$\rho_{x,x}[0] = \sum_{n=0}^{N-1} x[n]x^*[n] = \sum_{n=0}^{N-1} |x[n]|^2 = E_x. \quad (9.35)$$

Since $\rho_{x,x}[l]$ has DFT $X[k]X^*[k]$, it can be synthesized using Eq. (9.5) as

$$\rho_{x,x}[l] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]X^*[k]e^{j\Omega_0 kl} = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 e^{j\Omega_0 kl}.$$

Substituting $l = 0$ and combining with Eq. (9.35), we obtain

$$E_x = \sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2. \quad (9.36)$$

Equation (9.36) is *Parseval's theorem* for the discrete Fourier transform, and it provides a frequency-domain representation of signal energy.

▷ **Example 9.8 (A Matrix Demonstration of Parseval's Theorem)**

Use the matrix representation of the DFT to demonstrate Parseval's theorem

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2.$$

Using the notation of Sec. 9.1.2, we first express the frequency-domain side in matrix form as

$$\frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 = \frac{1}{N} \mathbf{X}^T \mathbf{X}^* = \frac{1}{N} (\mathbf{D}_N \mathbf{x})^T (\mathbf{D}_N \mathbf{x})^* = \mathbf{x}^T \mathbf{D}_N^T \frac{1}{N} \mathbf{D}_N^* \mathbf{x}^*.$$

Since the DFT matrix is symmetric, $\mathbf{D}_N^T = \mathbf{D}_N$. Furthermore, from Sec. 9.1.2, we know that $\mathbf{D}_N^{-1} = \frac{1}{N} \mathbf{D}_N^*$. Thus,

$$\frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 = \mathbf{x}^T \mathbf{D}_N \mathbf{D}_N^{-1} \mathbf{x}^* = \mathbf{x}^T \mathbf{x}^* = \sum_{n=0}^{N-1} |x[n]|^2.$$

Example 9.8 ◇

Table 9.1 summarizes the discrete Fourier transform properties just discussed and provides a side-by-side comparison with the corresponding properties of the discrete-time Fourier transform. In most cases, the properties of the DFT and the DTFT are very similar.

9.4 Graphical Interpretation of Circular Convolution

Let us define $y[n]$ as the circular convolution between $x[n]$ and $h[n]$; that is,

$$y[n] = x[n] \circledast h[n] = \sum_{m=0}^{N-1} x[m] h[\langle n - m \rangle_N]. \quad (9.37)$$

This dry equation does not provide much intuition on computing $y[n]$. To help our understanding, let us consider a graphical interpretation of circular convolution. The circular convolution $y[n] = x[n] \circledast h[n]$ can be visualized by using circular representations of $x[n]$ and $h[n]$ in two concentric circles. The two sequences to be convolved must have the same length. If not, we should append a sufficient number of zeros to the smaller sequence to ensure equal lengths of both sequences. For simplicity's sake, we shall take both $x[n]$ and $h[n]$ as 4-point signals throughout this discussion.

To begin, notice in Eq. (9.37) that regardless of the value n of interest, the signal $x[m]$ remains fixed in the convolution sum. Thus, we take the inner circle as the fixed circular representation of $x[m]$, drawn in the regular clockwise manner. Starting at $n = 0$, the convolution sum requires $h[\langle 0 - m \rangle_N]$. Thus, we take the outer circle as the circular representation of $h[\langle -m \rangle_N]$, which we

Discrete Fourier Transform	Discrete-Time Fourier Transform
Synthesis: $x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 kn}, \quad \Omega_0 = \frac{2\pi}{N}$ Analysis: $X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\Omega_0 kn}, \quad \Omega_0 = \frac{2\pi}{N}$ Duality: if $x[n] \longleftrightarrow X[k]$, then $X[n] \longleftrightarrow Nx[\langle -k \rangle_N]$ Linearity: $ax[n] + by[n] \longleftrightarrow aX[k] + bY[k]$ Complex Conjugation: $x^*[n] \longleftrightarrow X^*[\langle -k \rangle_N]$ $x^*[\langle -n \rangle_N] \longleftrightarrow X^*[k]$ Reversal: $x[\langle -n \rangle_N] \longleftrightarrow X[\langle -k \rangle_N]$ Shifting: $x[\langle n - m \rangle_N] \longleftrightarrow X[k] e^{-j\Omega_0 km}, \quad \Omega_0 = \frac{2\pi}{N}$ $x[n] e^{j\Omega_0 mn} \longleftrightarrow X[\langle k - m \rangle_N], \quad \Omega_0 = \frac{2\pi}{N}$ Convolution: $x[n] \circledast y[n] \longleftrightarrow X[k] Y[k]$ $x[n] y[n] \longleftrightarrow \frac{1}{N} X[k] \circledast Y[k]$ Correlation: $\rho_{x,y}[l] = x[l] \circledast y^*[\langle -l \rangle_N] \longleftrightarrow X[k] Y^*[k]$ Parseval's: $E_x = \sum_{n=0}^{N-1} x[n] ^2 = \frac{1}{N} \sum_{k=0}^{N-1} X[k] ^2$	Synthesis: $x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega$ Analysis: $X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n}$ Duality: Linearity: $ax[n] + by[n] \iff aX(\Omega) + bY(\Omega)$ Complex Conjugation: $x^*[n] \iff X^*(-\Omega)$ $x^*[-n] \iff X^*(\Omega)$ Reversal: $x[-n] \iff X(-\Omega)$ Shifting: $x[n - m] \iff X(\Omega) e^{-j\Omega m}$ $x[n] e^{j\Omega_0 n} \iff X(\Omega - \Omega_0)$ Convolution: $x[n] * y[n] \iff X(\Omega) Y(\Omega)$ $x[n] y[n] \iff \frac{1}{2\pi} X(\Omega) \circledast Y(\Omega)$ Correlation: $\rho_{x,y}[l] = x[l] * y^*[-l] \iff X(\Omega) Y^*(\Omega)$ Parseval's: $E_x = \sum_{n=-\infty}^{\infty} x[n] ^2 = \frac{1}{2\pi} \int_{2\pi} X(\Omega) ^2 d\Omega$

Table 9.1: Properties of the DFT and the DTFT.

obtain by writing $h[m]$ in the counterclockwise direction. Figure 9.14a illustrates this $n = 0$ case. To compute $y[0]$, we multiply the corresponding points on our concentric circles and sum the results,

$$y[0] = x[0]h[0] + x[1]h[3] + x[2]h[2] + x[3]h[1].$$

To determine $y[1]$, the convolution sum requires $h[\langle 1 - m \rangle_N]$ rather than $h[\langle 0 - m \rangle_N]$. However, this can be simply obtained by rotating the outer ring in Fig. 9.14a one step clockwise, as shown in Fig. 9.14b. To compute $y[1]$, we again multiply the corresponding points on our concentric circles and sum the results,

$$y[1] = x[0]h[1] + x[1]h[0] + x[2]h[3] + x[3]h[2].$$

This process is continued, step by step, to compute the rest of $y[n]$. Figure 9.14c shows the second rotation, which yields

$$y[2] = x[0]h[2] + x[1]h[1] + x[2]h[0] + x[3]h[3].$$

Lastly, Fig. 9.14d shows the third rotation, which yields

$$y[3] = x[0]h[3] + x[1]h[2] + x[2]h[1] + x[3]h[0].$$

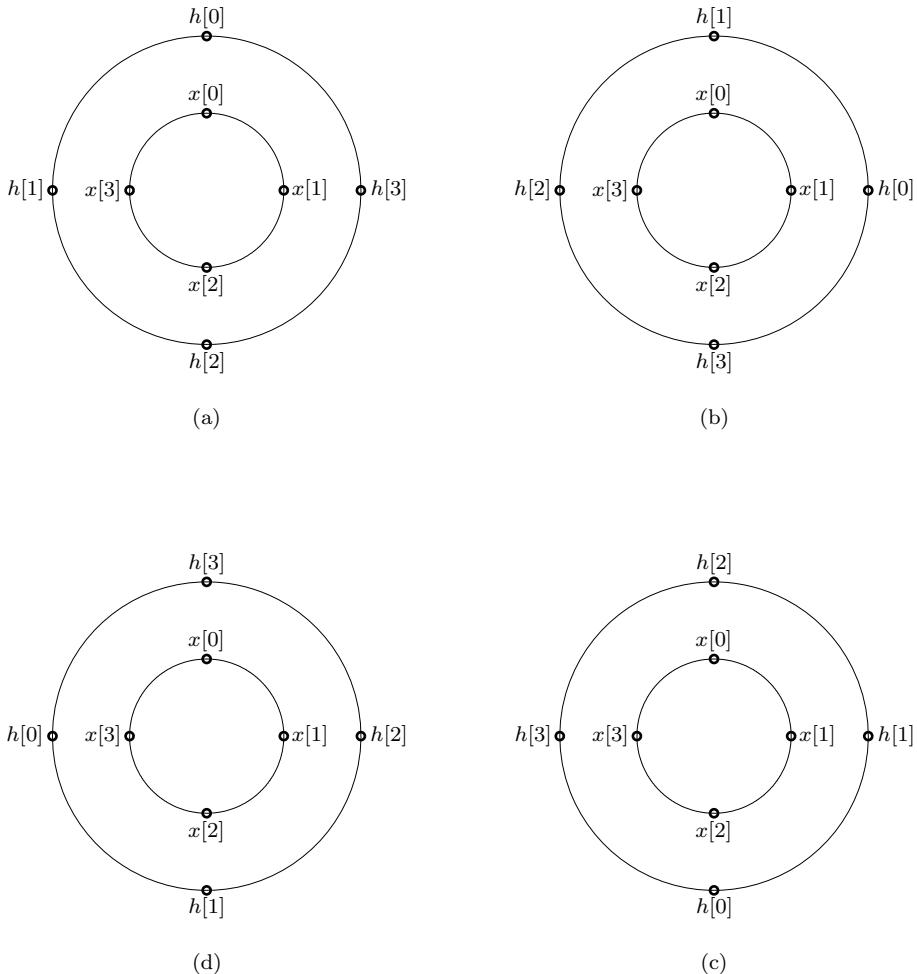


Figure 9.14: Graphical interpretation of $x[n] \otimes h[n]$: (a) $n = 0$, (b) $n = 1$, (c) $n = 2$, and (d) $n = 3$.

After the $(N - 1)$ th rotation, the pattern repeats. This reinforces an N -periodic view of the signals. Clearly, the circular convolution $y[n] = x[n] \otimes h[n]$ is N -periodic.

▷ **Drill 9.13 (Computing Circular Convolution Graphically)**

Over $0 \leq n \leq 5$, define the 6-point signals $x[n]$ and $h[n]$ as $[-1, 1, 2, -2, 1, 1]$ and $[1, 2, 3, 4, 5, 6]$, respectively. Letting $y[n] = x[n] \otimes h[n]$, use the graphical method to determine $y[2]$.

△

9.4.1 Circular and Linear Convolution

Circular convolution discussed here is different from the linear convolution of Eq. (5.22) studied in Ch. 5. In linear convolution, we place $x[m]$ along a straight line and place $h[m]$ in reverse direction along another straight line. Figures 5.5a and 5.5c (also Figs. 5.7a and 5.7c) show this graphically when $x[0]$ and $h[0]$ are aligned. Shifting by n , the h strip (now $h[n-m]$) moves along the fixed x strip ($x[m]$), where we multiply the overlapping numbers and add to obtain the convolution at n . Despite some similarities, a little reflection shows that linear convolution is not the same as the circular

variety shown in Fig. 9.14. For instance, in the linear convolution of two causal sequences $x[n]$ and $h[n]$, to obtain the output at $n = 0$, only $x[0]$ and $h[0]$ align, resulting in the single-term output $x[0]h[0]$. To obtain the convolution at $n = 1$, the h strip is shifted by one unit to the right, which results in two terms overlapping, yielding the two-term output $x[0]h[1] + x[1]h[0]$. In contrast, in circular convolution, all N numbers are aligned for every value of n . Hence, the circular convolution for any value of n consists of the sum of N terms.

For linear system analysis, as seen in Ch. 5, we need linear rather than circular convolution. Then what is the use of circular convolution? We shall now show that with suitable zero padding, we can make the results of circular convolution identical to those of linear convolution.

To determine the number of zeros to be padded, recall that the length of linear convolution $x[n] * h[n]$ is $N = N_x + N_h - 1$, where N_x and N_h are the lengths of $x[n]$ and $h[n]$, respectively. We also know that for circular convolution, both signals must have the same length. Thus, if linear and circular convolutions are to yield identical results, then a necessary condition is that the length of each signal $x[n]$ and $h[n]$ must be extended to $N = N_x + N_h - 1$. In other words, we must pad $N_h - 1$ zeros to $x[n]$ and pad $N_x - 1$ zeros to $h[n]$. This conclusion applies to $x[n]$ and $h[n]$ of any arbitrary lengths. The length N_x need not be equal to N_h . We shall give two proofs that the linear convolution of two finite-length sequences $x[n]$ and $h[n]$ (both starting at $n = 0$) is equal to the circular convolution of $x[n]$ and $h[n]$ after they are padded by $N_h - 1$ and $N_x - 1$ zeros, respectively.

Frequency-Domain Proof

Let us denote the length- N zero-padded versions of $x[n]$ and $h[n]$ by $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$, respectively. Since zero padding does not change a signal, $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$ are equivalent to $x[n]$ and $h[n]$. The linear convolution of $x[n]$ and $h[n]$ is given by

$$y_{\text{lin}}[n] = \sum_{m=0}^n x[m]h[n-m], \quad 0 \leq n \leq N-1.$$

The length of $y_{\text{lin}}[n]$ is $N = N_x + N_h - 1$. From the property of linear convolution, we know that the DTFT of $y_{\text{lin}}[n]$ is $Y_{\text{lin}}(\Omega) = X(\Omega)H(\Omega)$. Also, the N uniform samples of $Y_{\text{lin}}(\Omega)$ represent the DFT of $y_{\text{lin}}[n]$. Further, the N uniform samples of $X(\Omega)$ and $H(\Omega)$ correspond to the DFTs of the length- N zero-padded signals $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$, respectively. Hence, the DFT of $y_{\text{lin}}[n]$ is $Y_{\text{lin}}[k] = X_{\text{pad}}[k]H_{\text{pad}}[k]$ with length $N = N_x + N_h - 1$.

Let us now consider the circular convolution $y_{\text{cir}}[n] = x_{\text{pad}}[n] \circledast h_{\text{pad}}[n]$, where $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$ are zero-padded versions of $x[n]$ and $h[n]$. The lengths of $x_{\text{pad}}[n]$, $h_{\text{pad}}[n]$, and $y_{\text{cir}}[n]$ are each $N_x + N_h - 1$. Using the time-domain circular convolution property of Eq. (9.30), the DFT of $y_{\text{cir}}[n]$ is thus $Y_{\text{cir}}[k] = X_{\text{pad}}[k]H_{\text{pad}}[k]$. In this case, the linear and circular convolutions clearly yield the same result.

Alternate Time-Domain Proof

Let us next consider an alternative, time-domain proof of this result. The circular convolution of $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$ is given by

$$\begin{aligned} y_{\text{cir}}[n] &= \sum_{m=0}^{N-1} x_{\text{pad}}[m]h_{\text{pad}}[\langle n - m \rangle_N] \\ &= \sum_{m=0}^n x_{\text{pad}}[m]h_{\text{pad}}[\langle n - m \rangle_N] + \sum_{m=n+1}^{N-1} x_{\text{pad}}[m]h_{\text{pad}}[\langle n - m \rangle_N]. \end{aligned}$$

According to Eq. (9.15), this expression simplifies to

$$y_{\text{cir}}[n] = \sum_{m=0}^n x_{\text{pad}}[m]h_{\text{pad}}[n - m] + \sum_{m=n+1}^{N-1} x_{\text{pad}}[m]h_{\text{pad}}[N + n - m]. \quad (9.38)$$

Next, we show that the second sum on the right-hand side is zero. This follows from the fact that $h_{\text{pad}}[r]$ can be nonzero only when $0 \leq r$ and also $r \leq N_h - 1$ or, simply, $0 \leq r \leq N_h - 1$. Hence, $h_{\text{pad}}[N + n - m]$ can be nonzero only when $m \leq N + n$ and also $N_x + n \leq m$ or, simply,

$$N_x + n \leq m \leq N + n.$$

Moreover, by definition, $x_{\text{pad}}[m]$ can be nonzero only when

$$0 \leq m \leq N_x - 1.$$

Clearly, the product $x_{\text{pad}}[m]h_{\text{pad}}[N + n - m]$ can be nonzero only for those values of m that simultaneously satisfy these twin restrictions. Figure 9.15 shows the regions where $x_{\text{pad}}[m]$ (light shading) and $h_{\text{pad}}[N + n - m]$ (dark shading) can be nonzero. Since the regions are nonoverlapping, the two restrictions cannot be fulfilled simultaneously for any value m , and the product $x_{\text{pad}}[m]h_{\text{pad}}[N + n - m]$ must always be zero. Hence, the second sum on the righthand side of Eq. (9.38) is zero, and

$$y_{\text{cir}}[n] = \sum_{m=0}^n x_{\text{pad}}[m]h_{\text{pad}}[n - m].$$

This is identical to Eq. (5.23), the linear convolution $y_{\text{lin}}[n]$ of the causal signals $x[n]$ and $h[n]$.

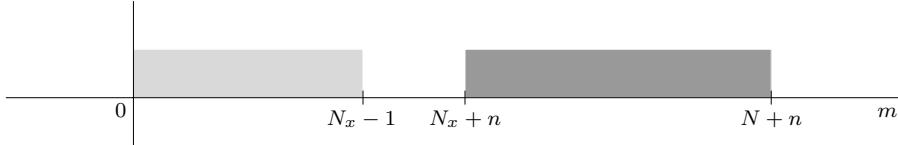


Figure 9.15: Range of m where $x_{\text{pad}}[m]$ (light shading) and $h_{\text{pad}}[N + n - m]$ (dark shading) can be nonzero.

▷ Example 9.9 (Linear Convolution by Circular Convolution)

Over $0 \leq n \leq 4$, define the 5-point signal $x[n]$ as $[1, 2, 3, 4, 5]$. Over $0 \leq n \leq 3$, define the 4-point signal $h[n]$ as $[1, 1, 2, 2]$. Compute the linear convolution $y_{\text{lin}}[n] = x[n] * h[n]$, and show that the same result is obtained by the $(N_x + N_h - 1)$ -point circular convolution $y_{\text{cir}}[n] = x_{\text{pad}}[n] * h_{\text{pad}}[n]$.

In this example, we use MATLAB to perform all computations. To begin, we perform the linear convolution using the `conv` command.

```
01 x = [1;2;3;4;5]; h = [1;1;2;2]; ylin = conv(x,h)
      ylin = 1   3   7   13  19  19  18   10
```

To perform the circular convolution, we first appropriately zero pad both $x[n]$ and $h[n]$. Since the lengths of $x[n]$ and $h[n]$ are different, different numbers of zeros are appended to make each have length $N_x + N_h - 1$.

```
02 Nx = length(x); Nh = length(h); N = Nx+Nh-1;
03 xpad = [x;zeros(Nh-1,1)]; hpad = [h;zeros(Nx-1,1)];
```

To compute the circular convolution, we compute the DFTs $X_{\text{pad}}[k]$ and $H_{\text{pad}}[k]$, multiply them together, and then take the inverse DFT.

```
04 Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1); DN = exp(-1j*Omega0*k'*n);
05 Xpad = DN*xpad; Hpad = DN*hpad; ycir = 1/N*conj(DN)*(Xpad.*Hpad)
      ycir = 1   3   7   13  19  19  18   10
```

As expected, the circular convolution of $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$ is the same as the linear convolution of $x[n]$ and $h[n]$.

Example 9.9 ▶

9.4.2 Aliasing in Circular Convolution

As we saw in the preceding section, to use circular convolution to compute the linear convolution of an N_x -point sequence $x[n]$ and an N_h -point sequence $h[n]$, we must pad $N_h - 1$ zeros to $x[n]$ and $N_x - 1$ zeros to $h[n]$. What happens if we pad fewer zeros so that the two sequences both have equal lengths N but N is less than $N_x + N_h - 1$? The resulting convolution and its DFT will have less than $N_x + N_h - 1$ samples. As seen earlier in Sec. 9.1.2, the use of fewer samples results in aliasing that generally causes erroneous results. But even under such aliasing, some samples of the two convolutions can be identical, and this fact is exploited in certain practical applications.

Consider the case where we do not zero pad $x[n]$ and pad $N_x - N_h$ zeros to $h[n]$ so that the length of both sequences is N_x (assuming $N_x \geq N_h$). Let us see if there is any connection between the linear convolution $y_{\text{lin}}[n] = x[n] * h[n]$ and $y'_{\text{cir}}[n]$, their circular convolution. The circular convolution in this case has only N_x samples, whereas the linear convolution has $N_x + N_h - 1$ samples. Obviously, the results of the two convolutions are not identical. Yet, we shall see that $N_x - N_h + 1$ samples of the two convolutions are identical. We use the notation $y'_{\text{cir}}[n]$ for this aliased N_x -point circular convolution to distinguish it from $y_{\text{cir}}[n]$, the $(N_x + N_h - 1)$ -point convolution without aliasing.

The linear convolution of $x[n]$ and $h[n]$ is given by

$$y_{\text{lin}}[n] = \sum_{m=0}^n x[m]h[n-m]. \quad (9.39)$$

Let the sequence $h[n]$ padded with $N_x - N_h$ zeros be denoted by $h_{\text{pad}}[n]$. The N_x -point circular convolution of $x[n]$ and $h_{\text{pad}}[n]$ is given by

$$y'_{\text{cir}}[n] = \sum_{m=0}^{N_x-1} x[m]h_{\text{pad}}[\langle n-m \rangle_{N_x}].$$

Following the development of Eq. (9.38), we have

$$y'_{\text{cir}}[n] = \sum_{m=0}^n x[m]h_{\text{pad}}[n-m] + \sum_{m=n+1}^{N_x-1} x[m]h_{\text{pad}}[N_x+n-m]. \quad (9.40)$$

The first sum on the right-hand side is identical to the linear convolution $y_{\text{lin}}[n]$ in Eq. (9.39).[†] Let us examine the second sum on the right-hand side, which contains the product $x[m]h_{\text{pad}}[N_x+n-m]$ and where m varies from $n+1$ to N_x-1 . Over this entire range of m , generally $x[m] \neq 0$. Let us now consider the term $h_{\text{pad}}[N_x+n-m]$. Because m varies from $n+1$ to N_x-1 , we have

$$N_x + n - m = \begin{cases} N_x - 1 & \text{when } m = n + 1 \\ n + 1 & \text{when } m = N_x - 1 \end{cases}.$$

Thus, $N_x + n - m$ varies from a low of $n+1$ to a high of N_x-1 ; that is,

$$n+1 \leq N_x + n - m \leq N_x - 1.$$

When $n < N_h - 1$, then $n+1 < N_h$, and according to the preceding equation, $N_x + n - m$ includes values less than N_h where $h_{\text{pad}}[N_x+n-m] \neq 0$. Thus, when $0 \leq n < N_h - 1$, the right-hand sum of Eq. (9.40) includes some nonzero terms. Hence, the linear and the circular convolutions are not equal for $0 \leq n < N_h - 1$ (the first $N_h - 1$ terms of the output). In the same way, we can show that $x[m]h_{\text{pad}}[N_x+n-m] = 0$ for $n+1 \geq N_h$, implying that the linear and the circular convolutions are identical for $n \geq N_h - 1$.

[†]Although they appear identical, there is a minor difference. While $y_{\text{lin}}[n]$ has $N_x + N_h - 1$ terms, $y'_{\text{cir}}[n]$ has only N_x terms. The equality is only for the N_x terms.

Observe that there are $N_x + N_h - 1$ terms in the linear convolution $y_{\text{lin}}[n]$. In contrast, there are only N_x terms in the circular convolution $y'_{\text{cir}}[n]$. We have shown that the first $N_h - 1$ terms of the linear and circular convolutions are generally different. The next $N_x - N_h + 1$ terms of both convolutions are identical. This exhausts all the terms of $y'_{\text{cir}}[n]$. In addition to the first N_x values, $y_{\text{lin}}[n]$ has extra $N_h - 1$ terms at the end. Thus, if we ignore the first $N_h - 1$ samples and the last $N_h - 1$ samples of the linear convolution, then the remaining middle $N_x - N_h + 1$ samples of the linear convolution are identical to the last $N_x - N_h + 1$ samples of the N_x -point circular convolution of $x[n]$ and $h_{\text{pad}}[n]$. Keep in mind that in this case no zeros are padded to $x[n]$, and $N_x - N_h$ zeros are padded on $h[n]$ to form $h_{\text{pad}}[n]$.

The observations in this section prove useful in block convolution, to be discussed in the next section.

▷ **Example 9.10 (Aliasing in Circular Convolution)**

Consider a 4-point sequence $x[n]$ and a 3-point sequence $h[n]$ respectively described as

$$\{x[n]\}_{n=0}^3 = \{x[0], x[1], x[2], x[3]\} \quad \text{and} \quad \{h[n]\}_{n=0}^2 = \{h[0], h[1], h[2]\}.$$

To demonstrate the aliasing that can occur in circular convolution, compare the linear convolution $y_{\text{lin}}[n] = x[n] * h[n]$ with the 4-point circular convolution $y'_{\text{cir}}[n] = x[n] \circledast h_{\text{pad}}[n]$, where no zeros are padded to $x[n]$ and one zero ($N_x - N_h = 1$) is padded to $h[n]$.

Figure 9.16a shows the two sequences on sliding tapes ready for linear convolution. We shift the h -tape forward by n , multiply the values of the overlapping samples, and then add to obtain $y_{\text{lin}}[n]$ as

$$\begin{aligned} y_{\text{lin}}[0] &= x[0]h[0], \\ y_{\text{lin}}[1] &= x[0]h[1] + x[1]h[0], \\ y_{\text{lin}}[2] &= x[0]h[2] + x[1]h[1] + x[2]h[0], \\ y_{\text{lin}}[3] &= x[1]h[2] + x[2]h[1] + x[3]h[0], \\ y_{\text{lin}}[4] &= x[2]h[2] + x[3]h[1], \\ \text{and} \quad y_{\text{lin}}[5] &= x[3]h[2]. \end{aligned}$$

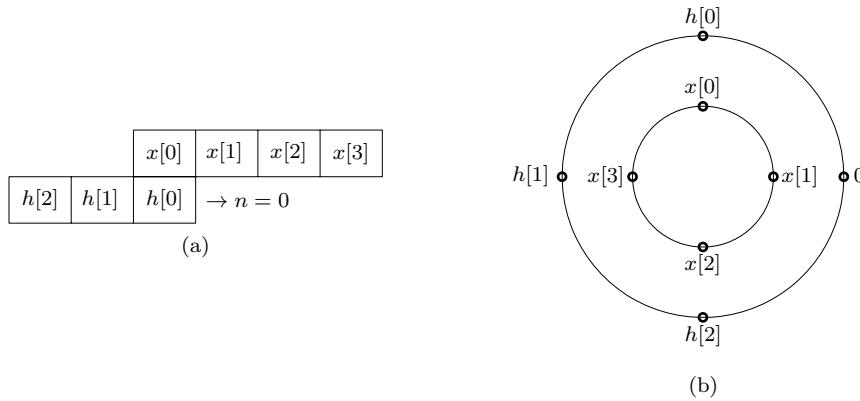


Figure 9.16: (a) Linear convolution and (b) circular convolution with aliasing.

Figure 9.16b shows the two 4-point sequences aligned along concentric circles ready for circular convolution. We shift the h -circle clockwise by n , multiply the values of the overlapping samples,

and then add to obtain $y'_{\text{cir}}[n]$ as

$$\begin{aligned}y'_{\text{cir}}[0] &= x[0]h[0] + x[2]h[2] + x[3]h[1], \\y'_{\text{cir}}[1] &= x[0]h[1] + x[1]h[0] + x[3]h[2], \\y'_{\text{cir}}[2] &= x[0]h[2] + x[1]h[1] + x[2]h[0], \\\text{and} \quad y'_{\text{cir}}[3] &= x[1]h[2] + x[2]h[1] + x[3]h[0].\end{aligned}$$

Observe that $y'_{\text{cir}}[0]$ and $y'_{\text{cir}}[1]$ are the inseparable sums of $y_{\text{lin}}[0] + y_{\text{lin}}[4]$ and $y_{\text{lin}}[1] + y_{\text{lin}}[5]$, respectively. This aliasing in the circular convolution causes the first $N_h - 1 = 2$ samples of the two convolutions to be different. The next $N_x - N_h + 1 = 2$ samples are identical, as expected.

Example 9.10 

▷ Drill 9.14 (Aliasing in Circular Convolution)

Using the DFT, show that the 3-point signal [15, 17, 16] is the circular convolution of the signals [3, 2, 3] and [1, 2, 3]. Verify your answer using the graphical method explained in Fig. 9.14. Show that the linear convolution of these two signals is the 5-point signal [3, 8, 16, 12, 9]. How can this linear convolution be used to obtain the previous 3-point circular convolution?



9.5 Discrete-Time Filtering Using the DFT

The DFT is useful in the computation of direct and inverse Fourier transforms, filtering, convolution, correlation, and others. It can perform these tasks not only for discrete-time systems but also for continuous-time signals and systems. The efficient FFT algorithm, discussed in Sec. 9.7, makes the DFT even more appealing. Let us focus our attention on using the DFT to perform filtering.

We generally think of filtering in terms of some hardware-oriented solution, such as using adders, multipliers, and delay elements. However, filtering also has a software-oriented solution that uses a computer algorithm to determine the filtered output $y[n]$ for a given input $x[n]$ and impulse response $h[n]$. Such filtering, which involves the convolution of $x[n]$ and $h[n]$, can be conveniently accomplished by using the DFT.

Filtering can be accomplished either in the time domain or in the frequency domain. In the time-domain approach, we compute the filter output $y[n]$ directly from a given input $x[n]$ and the known filter impulse response $h[n]$. This output is equal to the linear convolution of $x[n]$ with $h[n]$. It is also equal to the circular convolution of suitably zero-padded signals $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$. We shall later see that the number of computations can be drastically reduced by solving this problem in the frequency domain using the FFT algorithm. Using the FFT, we first find the DFTs $X_{\text{pad}}[k]$ and $H_{\text{pad}}[k]$ of the zero-padded signals $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$. The desired convolution (or correlation) is given by the IDFT of $X_{\text{pad}}[k]H_{\text{pad}}[k]$, found also by using the FFT algorithm.

The DFT-based procedure to find the linear convolution of two signals $x[n]$ and $h[n]$, whose lengths are N_x and N_h , respectively, is summarized in four steps as follows:

1. Pad $N_h - 1$ zeros to $x[n]$ to form $x_{\text{pad}}[n]$, and pad $N_x - 1$ zeros to $h[n]$ to form $h_{\text{pad}}[n]$.
2. Find $X_{\text{pad}}[k]$ and $H_{\text{pad}}[k]$, the DFTs of $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$.
3. Multiply $X_{\text{pad}}[k]$ by $H_{\text{pad}}[k]$ to obtain $Y[k]$.
4. The desired convolution $y[n]$ is the IDFT of $Y[k]$.

This procedure is known as *fast convolution* when the DFT and IDFT are computed using the efficient FFT algorithm.

▷ **Example 9.11 (Convolution Using the DFT)**

Using DFT-based convolution, determine the output $y[n]$ of an LTID filter with impulse response $h[n] = 2\delta[n] + 2\delta[n - 1]$ to an input $x[n] = 3\delta[n] + 2\delta[n - 1] + 3\delta[n - 2]$.

The output $y[n]$ is the convolution $x[n] * h[n]$, which we compute using the outlined four-step procedure.

1. In this case, $N_x = 3$ and $N_h = 2$. Therefore, we pad 1 zero to $x[n]$ and 2 zeros to $h[n]$, as depicted in Figs. 9.17a and 9.17b, respectively.

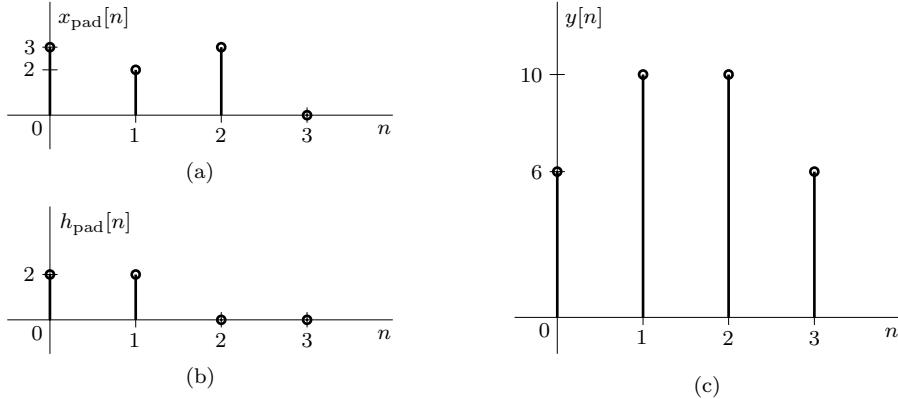


Figure 9.17: DFT-based convolution: (a) $x_{\text{pad}}[n]$, (b) $h_{\text{pad}}[n]$, and (c) $y[n] = \text{IDFT}\{X_{\text{pad}}[k]H_{\text{pad}}[k]\}$.

2. Now, $N = 4$ and $\Omega_0 = \pi/2$. The DFTs $X_{\text{pad}}[k]$ and $H_{\text{pad}}[k]$ of the zero-padded signals $x_{\text{pad}}[n]$ and $h_{\text{pad}}[n]$ are given by

$$X_{\text{pad}}[k] = \sum_{n=0}^3 x_{\text{pad}}[n]e^{-j\Omega_0 kn} = \sum_{n=0}^2 x[n]e^{-j\frac{\pi}{2}kn} = 3 + 2e^{-j\frac{\pi}{2}k} + 3e^{-j\pi k}$$

and

$$H_{\text{pad}}[k] = \sum_{n=0}^3 h_{\text{pad}}[n]e^{-j\Omega_0 kn} = \sum_{n=0}^1 h[n]e^{-j\frac{\pi}{2}kn} = 2 + 2e^{-j\frac{\pi}{2}k}.$$

Substituting $k = 0, 1, 2$, and 3 , we obtain

$$\{X_{\text{pad}}[k]\}_{k=0}^3 = \{8, -2j, 4, 2j\} \quad \text{and} \quad \{H_{\text{pad}}[k]\}_{k=0}^3 = \{4, 2\sqrt{2}e^{-j\frac{\pi}{4}}, 0, 2\sqrt{2}e^{j\frac{\pi}{4}}\}.$$

3. Next, we multiply $X_{\text{pad}}[k]$ by $H_{\text{pad}}[k]$ to obtain $Y[k]$ as

$$\{Y[k]\}_{k=0}^3 = \{32, 4\sqrt{2}e^{-j\frac{3\pi}{4}}, 0, 4\sqrt{2}e^{j\frac{3\pi}{4}}\}.$$

4. The desired convolution is the IDFT of $Y[k]$, given by

$$y[n] = \frac{1}{4} \sum_{k=0}^3 Y[k]e^{j\Omega_0 kn} = \frac{1}{4} \left(Y[0] + Y[1]e^{j\frac{\pi}{2}n} + Y[2]e^{j\pi n} + Y[3]e^{j\frac{3\pi}{2}n} \right), \quad 0 \leq n \leq 3.$$

Using the results of step 3 and noting that $e^{j\frac{3\pi}{2}} = e^{-j\frac{\pi}{2}}$, we obtain

$$y[n] = 8 + 4\sqrt{2}e^{-j\frac{3\pi}{4}+j\frac{\pi}{2}n} + 4\sqrt{2}e^{j\frac{3\pi}{4}-j\frac{\pi}{2}n} = 8 + 2\sqrt{2} \cos\left(\frac{\pi}{2}n - \frac{3\pi}{4}\right), \quad 0 \leq n \leq 3.$$

Evaluated over $0 \leq n \leq 3$, this equation yields

$$\{y[n]\}_{n=0}^3 = \{6, 10, 10, 6\}.$$

Figure 9.17c shows $y[n]$.

In this case, the results of our hand calculations are easily verified using MATLAB.

```
01 x = [3;2;3]; h = [2;2]; Nx = length(x); Nh = length(h);
02 xpad = [x;zeros(Nh-1,1)]; hpad = [h;zeros(Nx-1,1)];
03 N = Nx+Nh-1; n = (0:N-1); Omega0 = 2*pi/N; DN = exp(-1j*Omega0*k'*n);
04 Xpad = DN*xpad; Hpad = DN*hpad; y = conj(DN)/N*(Xpad.*Hpad)
y = 6 10 10 6
```

As an informative exercise, we encourage the reader to confirm these answers graphically by computing $y[n]$ as $x[n]*h[n]$ using the sliding-tape method and as $x_{\text{pad}}[n] \otimes h_{\text{pad}}[n]$ using two concentric circles, as in Fig. 9.14.

Example 9.11 ◀

Convolution Doublespeak?

From Ex. 9.11, the DFT-based convolution procedure appears much more laborious than the direct sliding-tape method or even the circular convolution method. To think that such a procedure leads to anything *fast* seems to be classical doublespeak. How can the task of convolution be simplified by adding the complexities of both the DFT and its inverse? In all honesty, there is little advantage to using DFT-based convolution for short signal lengths such as in Ex. 9.11. For relatively large N , however, DFT-based convolution implemented using the efficient FFT algorithm (fast convolution) can dramatically reduce the number of computations (multiplications and additions) needed for convolution-based filtering. With fewer computations, DSP execution time is shortened, and fast convolution lives up to its name.

▷ Example 9.12 (A Convolution Approximation Using the DFT)

Using a suitable truncation, use DFT-based convolution to approximate the linear convolution $y[n]$ between the infinite-duration signals $x[n] = (0.8)^n u[n]$ and $h[n] = (0.5)^n u[n]$.

Since both signals have infinite duration, we shall truncate them to a suitable finite length to use DFT-based convolution. Both signals decay exponentially, so let's truncate them once they are less than 1% of their peak amplitudes. For $x[n]$, this requires $0.8^n \leq 0.01$ or $n \geq \log(0.01)/\log(0.8) = 20.64$. Thus, we truncate $x[n]$ to have length $N_x = 21$. Similarly, $N_h = \lceil \log(0.01)/\log(0.5) \rceil = 7$. Now, we only need code similar to that in Ex. 9.11 to determine $\hat{y}[n]$, an approximation of $y[n] = x[n] * h[n]$.

```
01 Nx = ceil(log(0.01)/log(0.8)); x = (0.8).^(0:Nx-1)';
02 Nh = ceil(log(0.01)/log(0.5)); h = (0.5).^(0:Nh-1)';
03 xpad = [x;zeros(Nh-1,1)]; hpad = [h;zeros(Nx-1,1)];
04 N = Nx+Nh-1; n = (0:N-1); k=(0:N-1); Omega0 = 2*pi/N; DN = exp(-1j*Omega0*k'*n);
05 Xpad = DN*xpad; Hpad = DN*hpad; yhat = conj(DN)/N*(Xpad.*Hpad); stem(n,yhat);
```

The resulting approximation $\hat{y}[n]$, shown in Fig. 9.18, is visually indistinguishable from the true closed-form result (see pair 8 of Table 5.2),

$$y[n] = \frac{10}{3} [(0.8)^{n+1} - (0.5)^{n+1}] u[n].$$

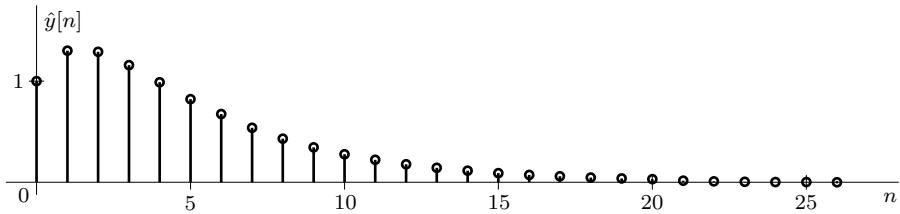


Figure 9.18: Approximation of $y[n] = x[n] * h[n]$ using DFT-based convolution.

Example 9.12 □

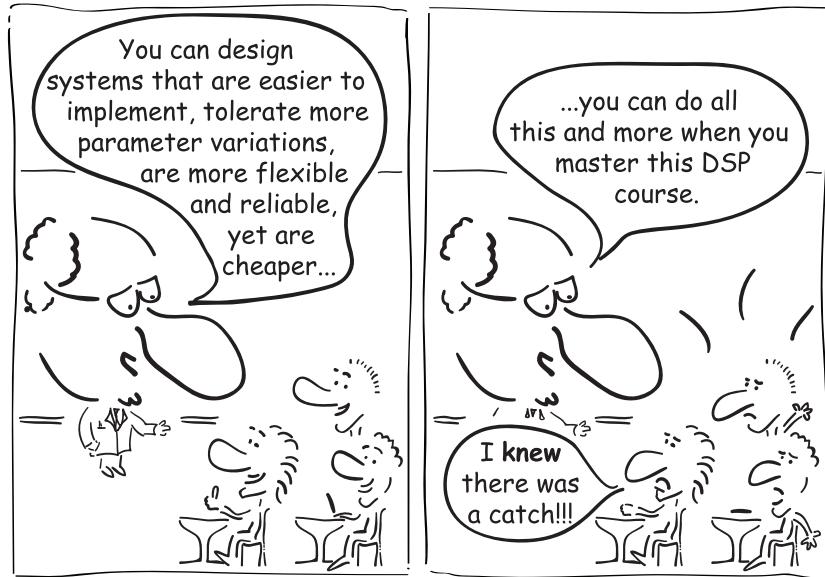
▷ Drill 9.15 (Convolution Using the DFT)

An LTID system has 3-point impulse response $h[n]$ and 4-point input $x[n]$ given by

$$\{h[n]\}_{n=0}^2 = \{1, 2, 1\} \quad \text{and} \quad \{x[n]\}_{n=0}^3 = \{1, 1, 1, 1\}.$$

Using the DFT-based procedure for convolution, show that the output $y[n]$ is the 6-point signal $[1, 3, 4, 4, 3, 1]$. Verify your answer by computing the linear convolution $x[n] * h[n]$.

□



Even with all DSP has to offer, there are no free lunches.

9.5.1 Block Convolution

Up to now, we have considered DFT-based convolution of two finite-length sequences. In FIR filtering and certain other convolution applications, it generally happens that one of the signals $h[n]$ is known a priori and is of fixed length N_h . The other signal $x[n]$, however, may be acquired in real time so that its total length may not be known. In such cases, a substantial wait may be required to compute $x[n] * h[n]$ using the DFT. Such delays are usually objectionable in real-time applications.

Furthermore, a large input length N_x requires a proportionally large number of zeros padded to $h[n]$, which unnecessarily increases the number of computations. These computations are not only very time-consuming but also require much larger amounts of memory, thereby increasing system cost and complexity.

To overcome these problems, we invoke the linearity property of convolution, which allows us to partition the input $x[n]$ into smaller consecutive data blocks $x_r[n]$, each of fixed length N_x . Partitioning data as it is received, we convolve each of the smaller blocks $x_r[n]$ with $h[n]$ and then add the results of all the convolutions, a process known as *block convolution*. Block convolution drastically reduces not only the delay time of the output but also the system memory requirements. Block convolution need not be performed using circular (DFT-based) convolution, but doing so can provide great computational benefits, particularly when the FFT algorithm is utilized. Thus, in our discussion of block convolution, we assume that all convolutions are performed in the frequency domain using the DFTs of the appropriately zero-padded signals, as explained earlier.

We shall discuss two such methods of block processing: the *overlap-and-add method* and *overlap-and-save method*. Either of the methods requires the same number of computations, so the choice of method is largely a matter of personal preference.

Overlap-and-Add Method

Consider a filtering operation where the output $y[n]$ is the convolution of the input $x[n]$ and an FIR filter impulse response $h[n]$ of length N_h . In practice, N_h is usually much smaller than the length of the input. Figure 9.19 illustrates a long input signal $x[n]$ sectioned into nonoverlapping blocks $x_r[n]$, each of a manageable length N_x . The number inside each block indicates its length. Let us assume that $N_x \gg N_h$. We now filter each block of the input data $x_r[n]$ using the impulse response $h[n]$. To be able to use circular convolution to perform these linear convolutions, we need to pad $N_h - 1$ zeros at the end of each data block.[†] Figure 9.19 shows each input data block augmented by $N_h - 1$ zeros (shaded). Observe that the zero-padded input blocks, each now of length $N_x + N_h - 1$, overlap.

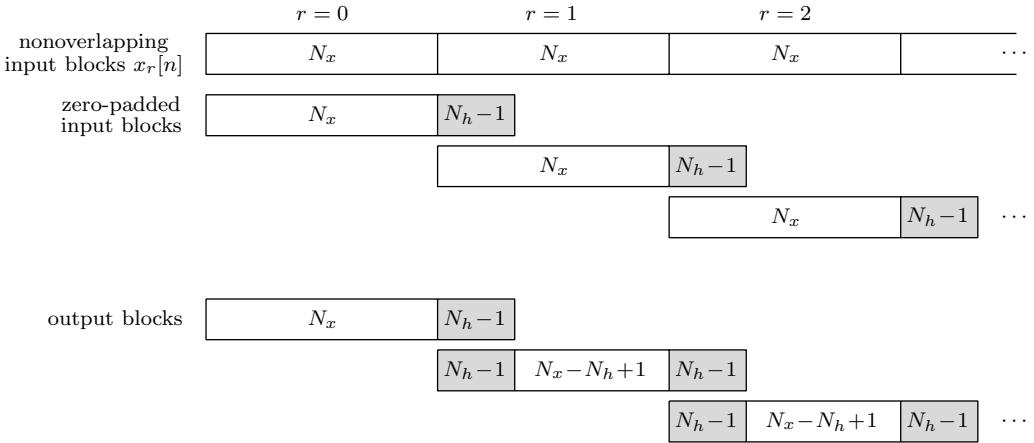


Figure 9.19: Overlap-and-add method of block filtering.

Since the circular convolution does not change the length of the signals involved, the output signals corresponding to the zero-padded input blocks each have a length $N_x + N_h - 1$. Consequently, the output blocks also overlap, as shown in Fig. 9.19. The total output is given by the sum of all these overlapping output blocks. Since the final output at any given instant n never requires summing

[†]We also pad $h[n]$ with $N_x - 1$ zeros so that the length of the padded $h[n]$ is $N_x + N_h - 1$.

more than two overlapping output blocks, the output lag is never much greater than the input segment length N_x . For obvious reasons, this method is known as the *overlap-and-add method*.

To explain the overlap-and-add method mathematically, consider a long signal $x[n]$ that is sectioned into R blocks, each of length N_x . The r th block $x_r[n]$ is given by

$$x_r[n] = \begin{cases} x[n] & rN_x \leq n \leq (r+1)N_x - 1 \\ 0 & \text{otherwise} \end{cases}.$$

These segments combine to form the overall signal $x[n]$ as

$$x[n] = \sum_{r=0}^{R-1} x_r[n].$$

If the system impulse response is $h[n]$, the system output $y[n]$ is given by

$$y[n] = x[n] * h[n] = \left(\sum_{r=0}^{R-1} x_r[n] \right) * h[n].$$

Invoking the linearity property of convolution, this expression simplifies to

$$y[n] = \sum_{r=0}^{R-1} \underbrace{x_r[n] * h[n]}_{y_r[n]}.$$

This is the result we seek. It shows that the convolution $y[n] = x[n] * h[n]$ can be performed by adding a series of individual block convolutions $y_r[n] = x_r[n] * h[n]$, each of which can be computed using efficient FFT-based fast convolution.

▷ Example 9.13 (Overlap-and-Add Method)

Using the overlap-and-add method with an $N_x = 3$ input block size, find the response $y[n]$ of an LTID system with the $N_h = 2$ finite impulse response $h[n]$ and input $x[n]$ shown in Fig. 9.20.

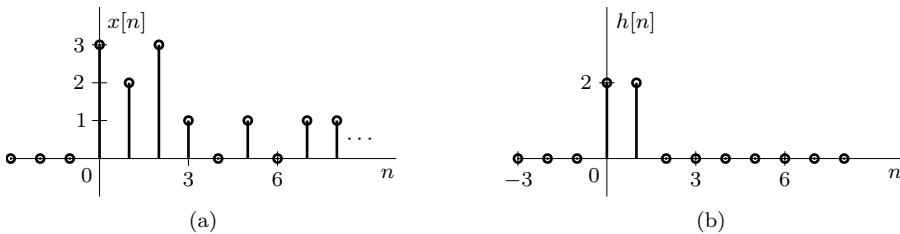


Figure 9.20: Block convolution: (a) input $x[n]$ and (b) impulse response $h[n]$.

Although the output $y[n]$ is a linear convolution of $x[n]$ and $h[n]$, we compute $y[n]$ using the overlap-and-add method of block convolution. In this example, the input block length is chosen as $N_x = 3$, and the impulse response length is $N_h = 2$. Hence, we need to pad $h[n]$ with $N_x - 1 = 2$ zeros. Once we partition the input into blocks of size $N_x = 3$, we pad each input block with $N_h - 1 = 1$ zero, as depicted in Fig. 9.21. We convolve each of these blocks with $h[n]$ using the DFT, as demonstrated in Ex. 9.11.

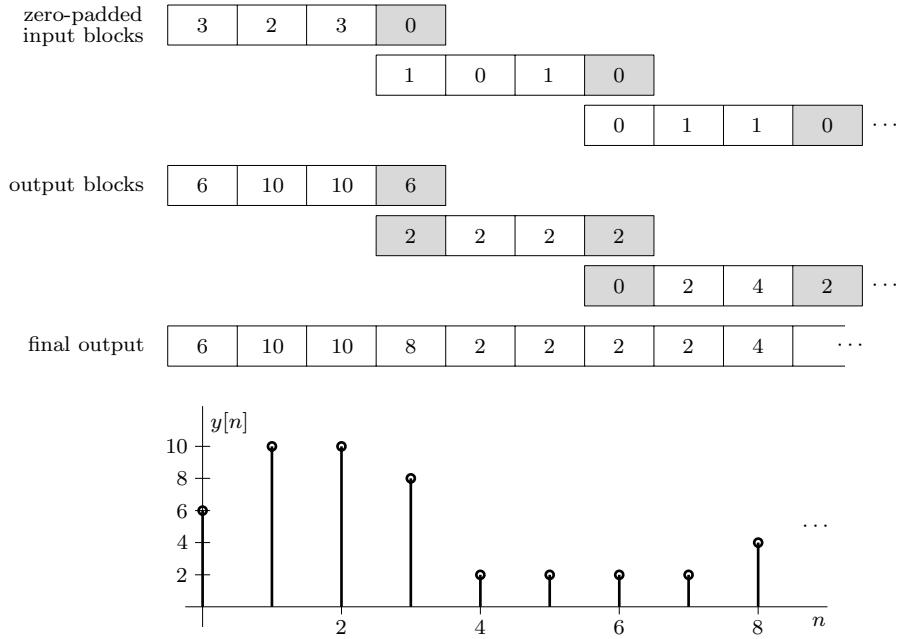


Figure 9.21: An example of the overlap-and-add method of block filtering.

In this case, the DFT size is $N = N_x + N_h - 1 = 4$ and $\Omega_0 = \pi/4$. Further, the transforms $X_{r,\text{pad}}[k]$ and $H_{\text{pad}}[k]$ of the zero-padded signals $x_{r,\text{pad}}[n]$ and $h_{\text{pad}}[n]$ are given by

$$X_{r,\text{pad}}[k] = \sum_{n=0}^2 x[n + rN_x] e^{-j\frac{\pi}{2}kn} \quad \text{and} \quad H_{\text{pad}}[k] = \sum_{n=0}^1 h[n] e^{-j\frac{\pi}{2}kn}.$$

Further, $Y_r[k] = X_{r,\text{pad}}[k]H_{\text{pad}}[k]$. We compute the values of $X_{r,\text{pad}}[k]$, $H_{\text{pad}}[k]$, and $Y_r[k]$ using these equations for each block.

For the first ($r = 0$) block, we have

$$X_{0,\text{pad}}[k] = 3 + 2e^{-j\frac{\pi}{2}k} + 3e^{-j\pi k}, \quad H_{\text{pad}}[k] = 2 + 2e^{-j\frac{\pi}{2}k}, \quad \text{and} \quad Y_0[k] = X_{0,\text{pad}}[k]H_{\text{pad}}[k].$$

Also,

$$y_0[n] = \frac{1}{4} \sum_{k=0}^3 Y_0[k] e^{j\frac{\pi}{2}kn}.$$

Even for the small N of this case, these equations are rather dull to evaluate by hand. Instead, we use MATLAB to perform the necessary calculations.

```
01 Nx = 3; Nh = 2; N = Nx+Nh-1; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1);
02 DN = exp(-1j*Omega0*k.*n); x0pad = [3;2;3;zeros(Nh-1,1)]; hpad = [2;2;zeros(Nx-1,1)];
03 X0pad = DN*x0pad; Hpad = DN*hpad; y0 = conj(DN)/N*(X0pad.*Hpad)
y0 = 6 10 10 6
```

We follow the same procedure for the second ($r = 1$) and third ($r = 2$) blocks.

```
04 x1pad = [1;0;1;zeros(Nh-1,1)]; X1pad = DN*x1pad; y1 = conj(DN)/N*(X1pad.*Hpad)
y1 = 2 2 2 2
05 x2pad = [0;1;1;zeros(Nh-1,1)]; X2pad = DN*x2pad; y2 = conj(DN)/N*(X2pad.*Hpad)
y2 = 0 2 4 2
```

Once properly aligned, the final output $y[n]$ is just the sum of the output blocks. Figure 9.21 shows the overlapping input blocks, output blocks, and final block convolution output.

In this example, computing each block convolution using the DFT is much more laborious than direct convolution by the sliding-tape method (Sec. 5.5.2). One reason is that we are not yet using the efficient FFT algorithm to compute the DFT and its inverse. Furthermore, the most dramatic reductions in the number of computations occur only for relatively large N . In practice, we generally deal with values of N much larger than 4, where the DFT-based approach (utilizing the FFT algorithm) really pays off.

Example 9.13 ▶

Overlap-and-Save Method

In the *overlap-and-save method*, the input signal is once again sectioned into manageable nonoverlapping blocks, this time of length $N_x - N_h + 1$. As before, each block is augmented (padded) with $N_h - 1$ data points. Unlike the previous method, however, this method places augmented points at the beginning of each block. Furthermore, it is not zeros that augment the signal but rather copies of the input signal itself, creating data-padded rather than zero-padded input blocks. The $N_h - 1$ augmented data points of a block are the same as the last $N_h - 1$ points of the previous block so that the last $N_h - 1$ data points of each block are saved and then reused as the first $N_h - 1$ data points of the succeeding block. The exception is the first block, where the first $N_h - 1$ data points are taken as zeros.[†] We also pad $h[n]$ with $N_x - N_h$ zeros to make a length- N_x signal $h_{\text{pad}}[n]$.

Once we form each input block, we compute the circular convolution between it and $h_{\text{pad}}[n]$. Each result has N_x samples. The first $N_h - 1$ samples are discarded, and the remaining $N_x - N_h + 1$ samples are saved. The saved samples of each output block are then concatenated together to obtain the final output. Figure 9.22 illustrates the general structure of this technique, which is commonly known as the *overlap-and-save method*. A better but less common name is the *overlap-and-discard method*. As with the overlap-and-add method, the DFT is used to perform the needed circular convolutions. Unlike the overlap-and-add method, however, these are circular convolutions with aliasing, as discussed in Sec. 9.4.2 and illustrated in Ex. 9.10.

Defining $x[n]$ as 0 for $n < 0$, the r th data-padded input block $x_r[n]$ is expressed as

$$x_r[n] = \begin{cases} x[n] & rN_x - N_h + 1 \leq n \leq (r+1)N_x - N_h \\ 0 & \text{otherwise} \end{cases}.$$

Each block $x_r[n]$ includes N_x values of the input $x[n]$ and shares $N_h - 1$ of these values with the following block $x_{r+1}[n]$. Next, we define each output block $y_r[n]$ as the N_x -point circular convolution between the N_x values of $x_r[n]$ and $h_{\text{pad}}[n]$. That is,

$$y_r[n] = x_r[n] \circledast h_{\text{pad}}[n].$$

To obtain the final result $y[n]$, the first $N_h - 1$ values of each $y_r[n]$ are discarded, the remaining $N_x - N_h + 1$ values are saved, and then the saved values from each output block are concatenated together. Let us now demonstrate the overlap-and-save method with an example.

Keys to Understanding the Overlap-and-Save Method

We have outlined the procedure of the overlap-and-save method, but what makes it work? The procedure is obviously based on the results of Sec. 9.4.2 on aliasing in circular convolution. When

[†]One might wonder why we partition the input into nonoverlapping length- N_x blocks for the overlap-and-add method but use a length of $N_x - N_h + 1$ in the overlap-and-save method. There are two primary reasons. First, this notation ensures that both methods use N_x input values for each processed block. Second, it helps readers directly use Sec. 9.4.2 (aliasing in circular convolution) to better understand how the overlap-and-save method actually works.

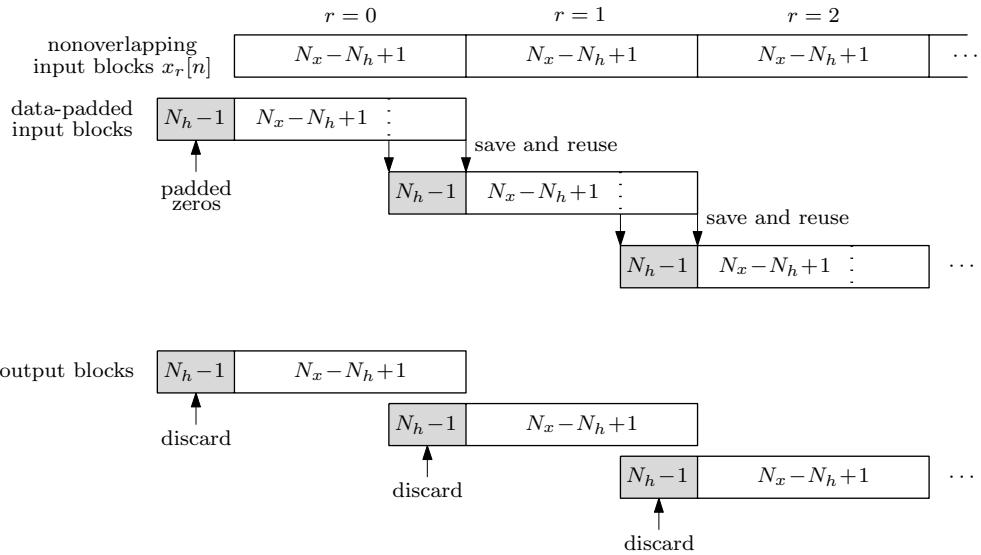


Figure 9.22: Overlap-and-save method of block filtering.

we perform the circular convolution between a length- N_x sequence $x_r[n]$ without zero padding and a length- N_h sequence $h[n]$ that is padded with $N_x - N_h$ zeros to the same length N_x , Sec. 9.4.2 shows that the first $N_h - 1$ terms of the linear and circular convolutions are not equal. However, the next $N_x - N_h + 1$ terms of the two convolutions are identical, assuming that $N_x \geq N_h$. We use this observation to find and save the correct $N_x - N_h + 1$ terms in each block computation. That is, the first $N_h - 1$ terms of the aliased circular convolution do not equal the desired linear convolution and are discarded. The remaining $N_x - N_h + 1$ values of the circular convolution do match the desired linear convolution and are saved. The (data-padded) input data blocks overlap by $N_h - 1$ values so that, once the first $N_h - 1$ values of each circular convolution are discarded, the remaining samples form the uninterrupted output. Viewed from this perspective of aliasing in circular convolution, the overlap-and-save method is much more intuitive.

There is another simple way to understand why the first $N_h - 1$ values of the circular convolution must be discarded. The output $y[n]$ of an FIR filter with an N_h -point impulse response requires the present input $x[n]$ and its past $N_h - 1$ values. The first $N_h - 1$ samples of any input block $x_r[n]$ lack at least some of the past $N_h - 1$ samples needed to compute their correct output values. It is not until the N_h sample of $x_r[n]$ that all required $N_h - 1$ previous samples are available. As a result, it is only beyond sample $N_h - 1$ that we can rely on the circular convolution output being correct.

▷ Example 9.14 (Overlap-and-Save Method)

Using the overlap-and-save method, repeat Ex. 9.13 and find the response $y[n]$ of an LTID system with the $N_h = 2$ finite impulse response $h[n]$ and input $x[n]$ shown in Fig. 9.20. As in Ex. 9.13, initially partition $x[n]$ into nonoverlapping blocks of length 3.

To partition $x[n]$ into nonoverlapping blocks of length 3 requires $N_x - N_h + 1 = 3$. Since $N_h = 2$, we must use $N_x = 4$. We follow the procedure illustrated in Fig. 9.22 to section the input data, as shown in Fig. 9.23. For the first data block, we pad $N_h - 1 = 1$ zero at the beginning. Thereafter, we save the last $N_h - 1 = 1$ point of each block and pad it to the beginning of the next block. Each length-4 input block is then circularly convolved (using the DFT) with $h_{\text{pad}}[n]$ to yield a 4-point output block. As in Ex. 9.13, these computations are easily performed in MATLAB.

```
01 Nx = 4; Nh = 2; N = Nx; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1);
```

```

02 DN = exp(-1j*Omega0*k'*n); x0 = [zeros(Nh-1,1);3;2;3]; hpad = [2;2;zeros(Nx-Nh,1)];
03 X0 = DN*x0; Hpad = DN*hpad; y0 = conj(DN)/N*(X0.*Hpad)
y0 = 6 6 10 10
04 x1 = [x0(end-(Nh-1)+1:end);1;0;1]; X1 = DN*x1; y1 = conj(DN)/N*(X1.*Hpad)
y1 = 8 8 2 2
05 x2 = [x1(end-(Nh-1)+1:end);0;1;1]; X2 = DN*x2; y2 = conj(DN)/N*(X2.*Hpad)
y2 = 4 2 2 4

```

Figure 9.23 graphically depicts these output blocks. Discarding the first $N_h - 1 = 1$ point of each output block, the remaining saved blocks are concatenated to provide the total output. As hoped, the result is identical to that obtained using the overlap-and-add method of Ex. 9.13.

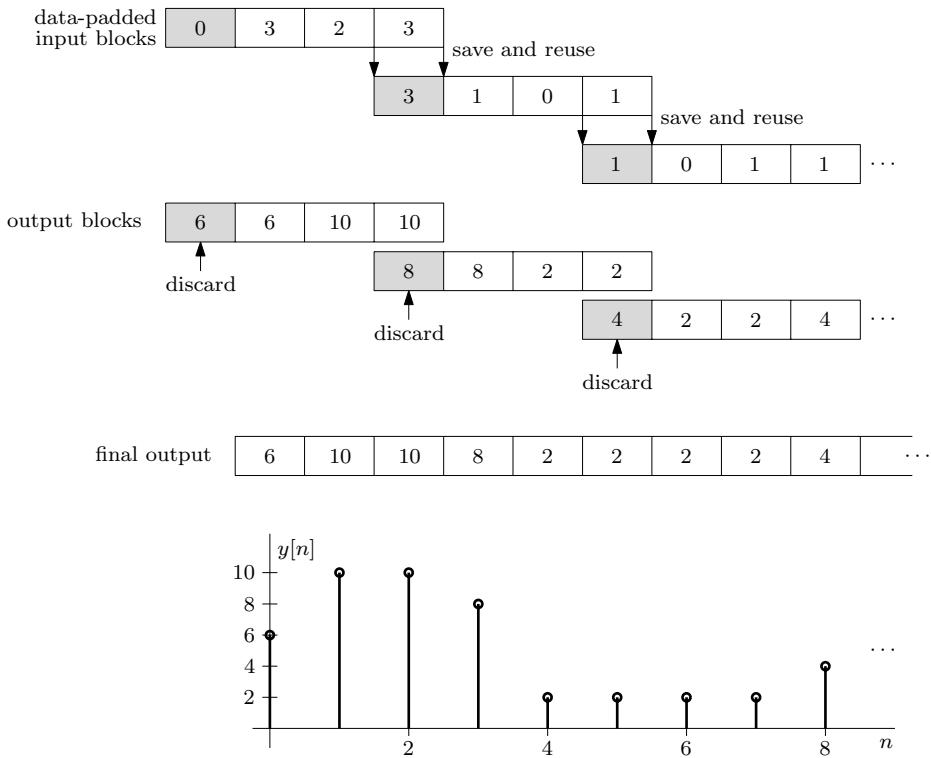


Figure 9.23: An example of the overlap-and-save method of block filtering.

Example 9.14 ◀

▷ Drill 9.16 (Block Convolution)

The signal $\{x[n]\}_{n=0}^{\infty} = \{1, 0, -1, 2, \dots\}$ is applied to an LTID system with FIR impulse response $h[n] = 3\delta[n] + 2\delta[n-1] + 3\delta[n-2]$. Segmenting the input into nonoverlapping blocks of length 2, use both methods of block convolution to show that the output is $\{y[n]\}_{n=0}^{\infty} = \{3, 2, 0, 4, \dots\}$.

◀

9.6 Goertzel's Algorithm

Assuming a complex-valued input $x[n]$, Eq. (9.2) requires N complex multiplications and $N - 1$ complex additions to directly compute one sample of $X[k]$. Each complex addition requires 2 real additions. A complex multiplication requires 4 real multiplications and 2 real additions. Thus, it takes $4N$ real multiplications and $2(N - 1) + 2N = 4N - 2$ real additions to compute each sample $X[k]$. To compute all N values of $X[k]$ ($k = 0, 1, \dots, N-1$) requires N^2 complex multiplications and $N(N - 1)$ complex additions or, equivalently, $4N^2$ real multiplications and $N(4N - 2)$ real additions. Computed in this way, the DFT requires order- N^2 multiplications and additions, denoted $\mathcal{O}(N^2)$. Notice that as we linearly increase our DFT size N , we geometrically increase our computational burden as $\mathcal{O}(N^2)$. It is not uncommon to encounter DFT sizes well in excess of $N = 1000$. For such large N , DFT computations can be prohibitively time-consuming, even for high-speed computers. Fortunately, there is much that we can do to reduce the computational burden of the DFT. Goertzel's algorithm, introduced next, offers one such approach, as does the fast Fourier transform (FFT), discussed later.

For convenience, we define the N th principle root of unity as

$$W_N = e^{j2\pi/N} = e^{j\Omega_0}. \quad (9.41)$$

Using this notation, the DFT analysis equation is

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{-kn}, \quad 0 \leq k \leq N - 1, \quad (9.42)$$

and the DFT synthesis equation is

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{kn}, \quad 0 \leq n \leq N - 1. \quad (9.43)$$

Noting that $W_N^{kN} = e^{j2\pi k} = 1$, we can write Eq. (9.42) as[†]

$$X[k] = W_N^{kN} \sum_{m=0}^{N-1} x[m] W_N^{-km} = \sum_{m=0}^{N-1} x[m] W_N^{k(N-m)}.$$

Written in this way, $X[k]$ resembles the convolution sum of Eq. (5.23). Let us capitalize on this fact. Define

$$x_N[n] = \begin{cases} x[n] & 0 \leq n \leq N - 1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$h_k[n] = W_N^{kn} u[n] = \begin{cases} W_N^{kn} & n \geq 0 \\ 0 & n < 0 \end{cases}.$$

For $n \geq 0$, next define

$$y_k[n] = x_N[n] * h_k[n] = \sum_{m=0}^n x_N[m] W_N^{k(n-m)}.$$

We can view $y_k[n]$ as the output of a causal system with complex-valued impulse response $h_k[n]$ to the finite-duration input $x_N[n]$. Recalling that $x_N[n] = x[n]$ for $0 \leq n \leq N - 1$ and $x_N[N] = 0$, we see that

$$y_k[N] = \sum_{m=0}^N x_N[m] W_N^{k(N-m)} = \sum_{m=0}^{N-1} x[m] W_N^{k(N-m)} = X[k].$$

[†]Here, we denote the sum variable as m rather than n to avoid later confusion.

This provides a filtering perspective of the DFT. That is, $X[k]$ can be found from our filter's output at $n = N$ as[†]

$$X[k] = y_k[n]|_{n=N}. \quad (9.44)$$

To learn the structure of this filter, we take the z -transform of $h_k[n]$ as

$$H_k(z) = \mathcal{Z}\{h_k[n]\} = \mathcal{Z}\{W_N^{kn}u[n]\} = \frac{1}{1 - W_N^k z^{-1}}.$$

Replacing $H_k(z)$ with $\frac{Y_k(z)}{X_N(z)}$ (see Eq. (7.27)) yields

$$\frac{Y_k(z)}{X_N(z)} = \frac{1}{1 - W_N^k z^{-1}}.$$

Cross-multiplying and taking the inverse z -transform yield

$$y_k[n] - W_N^k y_k[n-1] = x_N[n].$$

This recursive equation describes the first-order IIR filter shown in Fig. 9.24. Even though the filter coefficient is complex valued, the system is quite simple to implement.

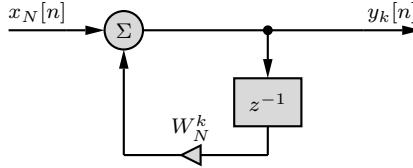


Figure 9.24: First-order complex IIR filter to compute $X[k]$.

To compute $X[k]$ using the filter of Fig. 9.24 requires N recursions, each of which requires one complex multiplication and one complex addition. Computing $X[k]$ for all N frequencies, therefore, requires N^2 complex multiplications and N^2 complex additions, or $4N^2$ real multiplications and $4N^2$ real additions. This makes the structure of Fig. 9.24, elegant looking as it may be, no more efficient than direct calculation of the DFT in terms of multiplications or additions. By engaging in a little mathematical trickery, however, Goertzel's algorithm improves the efficiency of computing $X[k]$.

To this end, let us multiply the numerator and denominator of $H_k(z)$ by $1 - W_N^{-k} z^{-1}$ to obtain the somewhat different but mathematically equivalent form of

$$\begin{aligned} H_k(z) &= \frac{1}{1 - W_N^k z^{-1}} \left(\frac{1 - W_N^{-k} z^{-1}}{1 - W_N^{-k} z^{-1}} \right) \\ &= \frac{1 - W_N^{-k} z^{-1}}{1 - (W_N^k + W_N^{-k}) z^{-1} + z^{-2}} \\ &= \frac{1 - W_N^{-k} z^{-1}}{1 - 2 \cos(2\pi k/N) z^{-1} + z^{-2}}. \end{aligned}$$

This expression, which is the basis of Goertzel's algorithm, suggests a second-order IIR structure, shown using direct form II in Fig. 9.25.

[†]For Eq. (9.44) to hold, the system must have relaxed initial condition $y_k[-1] = 0$.

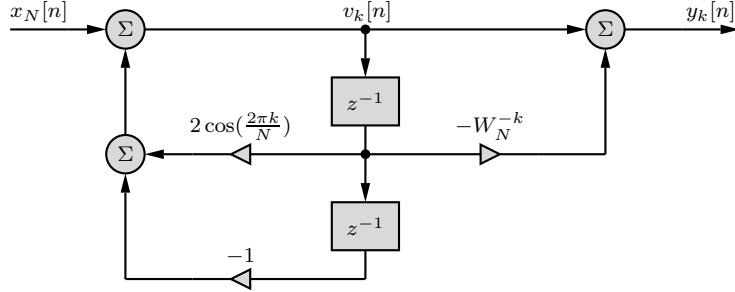


Figure 9.25: Goertzel's algorithm: second-order complex IIR filter to compute $X[k]$.

A Ludicrous Proposition?

It may seem a ludicrous proposition that the second-order implementation of Fig. 9.25 can be more efficient than the first-order implementation of Fig. 9.24, but this is precisely the case. The primary key to realizing the computational savings of Goertzel's algorithm is recognizing that we only need the output $y_k[n]$ at $n = N$ to determine $X[k]$. All other values of $y_k[n]$ are unimportant. Thus, we compute the intermediate variable $v_k[n]$ according to

$$v_k[n] = x_N[n] + 2 \cos\left(\frac{2\pi k}{N}\right) v_k[n-1] - v_k[n-2], \quad 0 \leq n \leq N. \quad (9.45)$$

Only at $n = N$ does the last feed-forward step of Fig. 9.25 need to be computed,

$$X[k] = y_k[n]|_{n=N} = v_k[N] - W_N^{-k} v_k[N-1].$$

Over $0 \leq n \leq N-1$, the feed-forward paths and computation of $y_k[n]$ are ignored completely.

Continuing our original assumption that input $x_N[n]$ is complex, Eq. (9.45) requires $2N$ real multiplies and $4N$ real additions. The final feed-forward step adds another 4 real additions and 4 real multiplications to the computation of $X[k]$. To compute all N values of $X[k]$ thus requires $(2N+4)N$ real multiplications and $(4N+4)N$ real additions. Despite having twice the order, the structure of Fig. 9.25 requires approximately half the number of real multiplications as does the structure of Fig. 9.24.

To realize a further reduction in computational complexity, notice that $h_{N-k}[n] = W_N^{(N-k)n} u[n]$ and

$$\begin{aligned} H_{N-k}(z) &= \frac{1}{1 - W_N^{(N-k)} z^{-1}} \left(\frac{1 - W_N^{-(N-k)} z^{-1}}{1 - W_N^{-(N-k)} z^{-1}} \right) \\ &= \frac{1 - W_N^{-(N-k)} z^{-1}}{1 - (W_N^{(N-k)} + W_N^{-(N-k)}) z^{-1} + z^{-2}} \\ &= \frac{1 - W_N^{-(N-k)} z^{-1}}{1 - 2 \cos(2\pi(N-k)/N) z^{-1} + z^{-2}} \\ &= \frac{1 - W_N^{-(N-k)} z^{-1}}{1 - 2 \cos(2\pi k/N) z^{-1} + z^{-2}}. \end{aligned}$$

This filter, which is used to compute $X[N-k]$, uses the same feedback paths as the filter used to compute $X[k]$. Thus, the computations of Eq. (9.45) used to compute $X[k]$ serve double duty to also compute $X[N-k]$. Only the final feed-forward step is different:

$$X[N-k] = v_k[N] - W_N^{-(N-k)} v_k[N-1].$$

This further reduces the computational demands of both multiplication and addition by half. Thus, Goertzel's algorithm computes $X[k]$ using approximately N^2 real multiplications and $2N^2$ real additions. Compared with a direct DFT computation using Eq. (9.2), Goertzel's algorithm uses approximately one-fourth the number of real multiplications and one-half the number of real additions.

9.7 The Fast Fourier Transform

While Goertzel's algorithm reduces the number of additions and multiplications to compute the DFT, the computational complexity of Goertzel's algorithm remains $\mathcal{O}(N^2)$. Thus, the benefits of Goertzel's algorithm are quickly overcome as N increases. The only real way to dramatically improve the situation is to reduce the *order* of computational complexity. This is precisely what is done by an algorithm developed by Tukey and Cooley in 1965 [1]. This algorithm, known as the *fast Fourier transform* (FFT), reduces the number of computations from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log_2 N)$. Particularly for large N , the computational savings are substantial. The FFT algorithm is largely what has made the Fourier transform accessible for digital signal processing.[†]

Secrets of the FFT

It is easy to understand the magic of the FFT. The secret is in the linearity of the Fourier transform as well as the DFT. Because of linearity, we can compute the Fourier transform of a signal $x(t)$ as a sum of the Fourier transforms of segments of $x(t)$ of shorter duration. The same principle applies to computation of the DFT. Consider a sequence of length $N = 16$ samples. As seen earlier, the direct DFT computation of this sequence requires $N^2 = 256$ multiplications and $N(N-1) = 240$ additions. However, we can split this sequence into two shorter sequences, each of length 8. To compute the DFT of each of these segments, we need 64 multiplications and 56 additions. Thus, we need a total of 128 multiplications and 112 additions. Suppose that we split the original sequence into four segments of length 4 each. To compute the DFT of each segment, we require 16 multiplications and 12 additions. Hence, we need a total of 64 multiplications and 48 additions. If we split the sequence into eight segments of length 2 each, we need 4 multiplications and 2 additions for each segment, resulting in a total of 32 multiplications and 8 additions. Thus, we have been able to reduce the number of multiplications from 256 to 32 and the number of additions from 240 to 8.[‡] Moreover, some of these multiplications turn out to be multiplications by 1 or -1 . All this fantastic economy in the number of computations is realized by the FFT without any approximation. The values obtained by the FFT are identical to those obtained by the DFT.

In the preceding example, we considered the relatively small value of $N = 16$. The reduction in the number of computations is much more dramatic for higher values of N . Take, for example, a DFT of size $N = 1024$. Direct computation using Eq. (9.2) requires on order of approximately 1×10^6 operations. Goertzel's algorithms reduces the number of additions to one-half and the number of multiplications to one-quarter. The FFT, however, requires on order of approximately 1×10^4 operations, a 100-fold improvement over direct computation and a 25-fold improvement over Goertzel's algorithm. The amount of improvement only continues to increase with N . Although there are many variations of the Tukey-Cooley algorithm, these can be grouped into two basic types: *decimation in time* (DIT) and *decimation in frequency* (DIF).

[†]It is worth noting that although it is most efficient to use the FFT to compute a complete spectrum, Goertzel's algorithm can be more efficient than the FFT when a limited number of frequency components are required. See Prob. 9.6-4.

[‡]For clarity's sake, we omit the few multiplications and additions required to combine the smaller DFTs together. Including these operations does not change the general computational savings (order of complexity) of the approach.

9.7.1 Decimation-in-Time Algorithm

The FFT algorithm is simplified if we choose N to be a power of 2, although such a choice is not essential. Thus, let us consider a sequence $x[n]$ with length N that is a power of 2. Next, we divide this N -point data sequence $x[n]$ into two $(\frac{N}{2})$ -point sequences $x_0[n]$ and $x_1[n]$ that consist of the even- and odd-numbered samples,[†] That is,

$$\{x_0[n]\}_{n=0}^{N/2-1} = \{x[0], x[2], \dots, x[N-2]\}$$

and

$$\{x_1[n]\}_{n=0}^{N/2-1} = \{x[1], x[3], \dots, x[N-1]\}.$$

These sequences are just $x[n]$ and $x[n+1]$ downsampled by a factor 2. That is, $x_0[n] = x[2n]$, and $x_1[n] = x[2n+1]$. Letting $W_N = e^{j2\pi/N} = e^{j\Omega_0}$, we next represent the DFT of $x[n]$ as

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} x[2n]W_N^{-2nk} + \sum_{n=0}^{\frac{N}{2}-1} x[2n+1]W_N^{-(2n+1)k}.$$

Replacing $x[2n]$ with $x_0[n]$, $x[2n+1]$ with $x_1[n]$, and using the fact that $W_{\frac{N}{2}} = W_N^2$, we have

$$\begin{aligned} X[k] &= \sum_{n=0}^{\frac{N}{2}-1} x_0[n]W_{\frac{N}{2}}^{-nk} + W_N^{-k} \sum_{n=0}^{\frac{N}{2}-1} x_1[n]W_{\frac{N}{2}}^{-nk} \\ &= X_0[k] + W_N^{-k} X_1[k], \quad 0 \leq k \leq \frac{N}{2} - 1. \end{aligned} \quad (9.46)$$

Here, $X_0[k]$ and $X_1[k]$ are the $(\frac{N}{2})$ -point DFTs of $x_0[n]$ and $x_1[n]$, respectively. Since $X_0[k]$ and $X_1[k]$ are $(\frac{N}{2})$ -point DFTs, they are themselves $(\frac{N}{2})$ -periodic. Hence,

$$X_0[k + \frac{N}{2}] = X_0[k] \quad \text{and} \quad X_1[k + \frac{N}{2}] = X_1[k]. \quad (9.47)$$

Moreover,

$$W_N^{-(k+\frac{N}{2})} = W_N^{-\frac{N}{2}} W_N^{-k} = e^{-j\pi} W_N^{-k} = -W_N^{-k}. \quad (9.48)$$

From Eqs. (9.46), (9.47), and (9.48), we obtain

$$X[k + \frac{N}{2}] = X_0[k] - W_N^{-k} X_1[k], \quad 0 \leq k \leq \frac{N}{2} - 1. \quad (9.49)$$

We can compute the first $\frac{N}{2}$ points of $X[k]$ using Eq. (9.46) and the last $\frac{N}{2}$ points using Eq. (9.49). That is, over $0 \leq k \leq \frac{N}{2} - 1$,

$$X[k] = X_0[k] + W_N^{-k} X_1[k] \quad \text{and} \quad X[k + \frac{N}{2}] = X_0[k] - W_N^{-k} X_1[k]. \quad (9.50)$$

From Eq. (9.50), we see that an N -point DFT can be computed by combining two $(\frac{N}{2})$ -point DFTs. These equations can be represented conveniently by the signal flow graph depicted in Fig. 9.26. This structure is known as a *butterfly*. The scale factor W_N^{-k} that appears in these computations is called the *twiddle factor*.

The next step is to compute the $(\frac{N}{2})$ -point DFTs $X_0[k]$ and $X_1[k]$. The computation of these DFTs is simplified by computing each as a combination of two $(\frac{N}{4})$ -point DFTs. To this end, we use downsampling to divide $x_0[n]$ and $x_1[n]$ each into two sequences. That is, we divide $x_0[n]$ into $x_{00}[n] = x_0[2n]$ and $x_{01}[n] = x_0[2n+1]$, and we divide $x_1[n]$ into $x_{10}[n] = x_1[2n]$ and $x_{11}[n] = x_1[2n+1]$. This process of subdividing each DFT into two smaller DFTs continues until we reach

[†]This is known as the *two-band polyphase decomposition* of $x[n]$. See [3] for a general treatment of this topic.

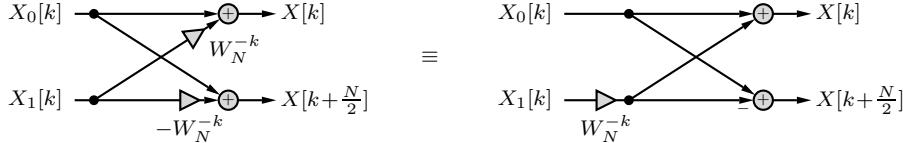


Figure 9.26: Example butterfly and its multiplication-efficient equivalent.

two-point DFTs (which require no multiplication at all). Let us demonstrate the procedure in its entirety for a length-8 sequence $x[n]$; that is,

$$\{x[n]\}_{n=0}^7 = \{x[0], x[1], x[2], x[3], x[4], x[5], x[6], x[7]\} \longleftrightarrow X[k], \quad 0 \leq k \leq 7.$$

We divide this length-8 sequence into two length-4 sequences $x_0[n]$ and $x_1[n]$, consisting of even- and odd-numbered samples, respectively, as

$$\{x_0[n]\}_{n=0}^3 = \{x[2n]\}_{n=0}^3 = \{x[0], x[2], x[4], x[6]\} \longleftrightarrow X_0[k], \quad 0 \leq k \leq 3$$

and

$$\{x_1[n]\}_{n=0}^3 = \{x[2n+1]\}_{n=0}^3 = \{x[1], x[3], x[5], x[7]\} \longleftrightarrow X_1[k], \quad 0 \leq k \leq 3.$$

We can determine $X[k]$ from $X_0[k]$ and $X_1[k]$ according to Eq. (9.50) as

$$X[k] = X_0[k] + W_8^{-k} X_1[k] \quad \text{and} \quad X[k+4] = X_0[k] - W_8^{-k} X_1[k].$$

Substituting $k = 0, 1, 2$, and 3 yields

$$\begin{aligned} X[0] &= X_0[0] + W_8^{-0} X_1[0], & X[4] &= X_0[0] - W_8^{-0} X_1[0], \\ X[1] &= X_0[1] + W_8^{-1} X_1[1], & X[5] &= X_0[1] - W_8^{-1} X_1[1], \\ X[2] &= X_0[2] + W_8^{-2} X_1[2], & X[6] &= X_0[2] - W_8^{-2} X_1[2], \\ X[3] &= X_0[3] + W_8^{-3} X_1[3], & X[7] &= X_0[3] - W_8^{-3} X_1[3]. \end{aligned}$$

At this point, we have divided our 8-point DFT into two 4-point DFTs, as illustrated in Fig. 9.27.

Continuing our decomposition, we next divide the 4-point DFTs $X_0[k]$ and $X_1[k]$ each into two 2-point DFTs. Let us begin with $X_0[k]$. First, we divide $x_0[n]$ into two length-2 sequences consisting of its even- and odd-numbered samples

$$\{x_{00}[n]\}_{n=0}^1 = \{x_0[2n]\}_{n=0}^1 = \{x[4n]\}_{n=0}^1 = \{x[0], x[4]\} \longleftrightarrow X_{00}[k], \quad k = 0, 1$$

and

$$\{x_{01}[n]\}_{n=0}^1 = \{x_0[2n+1]\}_{n=0}^1 = \{x[4n+2]\}_{n=0}^1 = \{x[2], x[6]\} \longleftrightarrow X_{01}[k], \quad k = 0, 1.$$

Similar to Eq. (9.50), we express the length-4 DFT $X_0[k]$ in terms of the two length-2 DFTs $X_{00}[k]$ and $X_{01}[k]$ as

$$X_0[k] = X_{00}[k] + W_4^{-k} X_{01}[k] \quad \text{and} \quad X_0[k+2] = X_{00}[k] - W_4^{-k} X_{01}[k].$$

Substituting $k = 0$ and 1 and using the fact that $W_4^1 = W_8^2$ yield

$$\begin{aligned} X_0[0] &= X_{00}[0] + W_8^{-0} X_{01}[0], & X_0[2] &= X_{00}[0] - W_8^{-0} X_{01}[0], \\ X_0[1] &= X_{00}[1] + W_8^{-2} X_{01}[1], & X_0[3] &= X_{00}[1] - W_8^{-2} X_{01}[1]. \end{aligned}$$

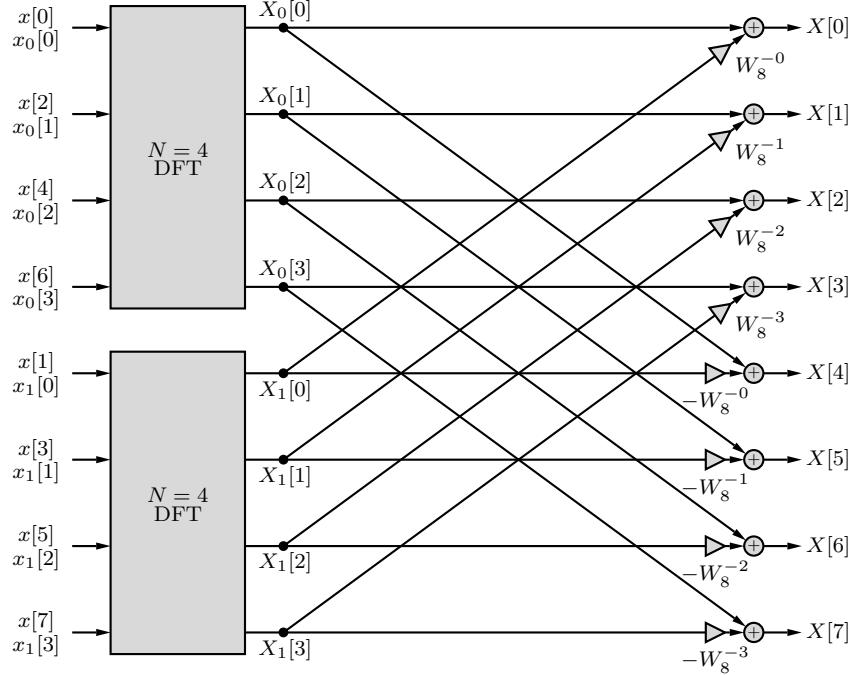


Figure 9.27: Dividing an 8-point DFT into two 4-point DFTs.

Using a similar procedure, $X_1[k]$ is expressed as[†]

$$\begin{aligned} X_1[0] &= X_{10}[0] + W_8^{-0}X_{11}[0], & X_1[2] &= X_{10}[0] - W_8^{-0}X_{11}[0], \\ X_1[1] &= X_{10}[1] + W_8^{-2}X_{11}[1], & X_1[3] &= X_{10}[1] - W_8^{-2}X_{11}[1]. \end{aligned}$$

Figure 9.28 shows this stage of our decomposition, where our 8-point DFT is divided into four 2-point DFTs.

The final 2-point DFTs are straightforward to compute using a simple butterfly. The 2-point DFT $X_{00}[k]$, for example, is given by (see Prob. 9.7-2)

$$X_{00}[0] = x[0] + W_8^{-0}x[4] \quad \text{and} \quad X_{00}[1] = x[0] - W_8^{-0}x[4]. \quad (9.51)$$

Similarly,

$$\begin{aligned} X_{01}[0] &= x[2] + W_8^{-0}x[6], & X_{01}[1] &= x[2] - W_8^{-0}x[6], \\ X_{10}[0] &= x[1] + W_8^{-0}x[5], & X_{10}[1] &= x[1] - W_8^{-0}x[5], \\ X_{11}[0] &= x[3] + W_8^{-0}x[7], & X_{11}[1] &= x[3] - W_8^{-0}x[7]. \end{aligned}$$

Substituting these equations into the structure of Fig. 9.28 results in the final 8-point decimation-in-time FFT shown in Fig. 9.29. This example is solved in three stages. Generally, there are $\log_2 N$ stages in an FFT computation. Each stage uses butterfly structures, as given in Eq. (9.50) and also Fig. 9.26, with each succeeding stage using half the points but twice as many variables. Thus the number of computations in each stage remains the same.

[†] First, we divide $x_1[n]$ into two length-2 sequences consisting of even- and odd-numbered samples,

$$\{x_{10}[n]\}_{n=0}^1 = \{x_1[2n]\}_{n=0}^1 = \{x[4n+1]\}_{n=0}^1 = \{x[1], x[5]\} \longleftrightarrow X_{10}[k], \quad k = 0, 1$$

and

$$\{x_{11}[n]\}_{n=0}^1 = \{x_1[2n+1]\}_{n=0}^1 = \{x[4n+3]\}_{n=0}^1 = \{x[3], x[7]\} \longleftrightarrow X_{11}[k], \quad k = 0, 1.$$

Next, we represent the length-4 DFT $X_1[k]$ in terms of the two length-2 DFTs $X_{10}[k]$ and $X_{11}[k]$ as

$$X_1[k] = X_{10}[k] + W_4^{-k}X_{11}[k] \quad \text{and} \quad X_1[k+2] = X_{10}[k] - W_4^{-k}X_{11}[k].$$

Substituting $k = 0$ and 1 and using the fact that $W_4^1 = W_8^2$ yield the desired result.

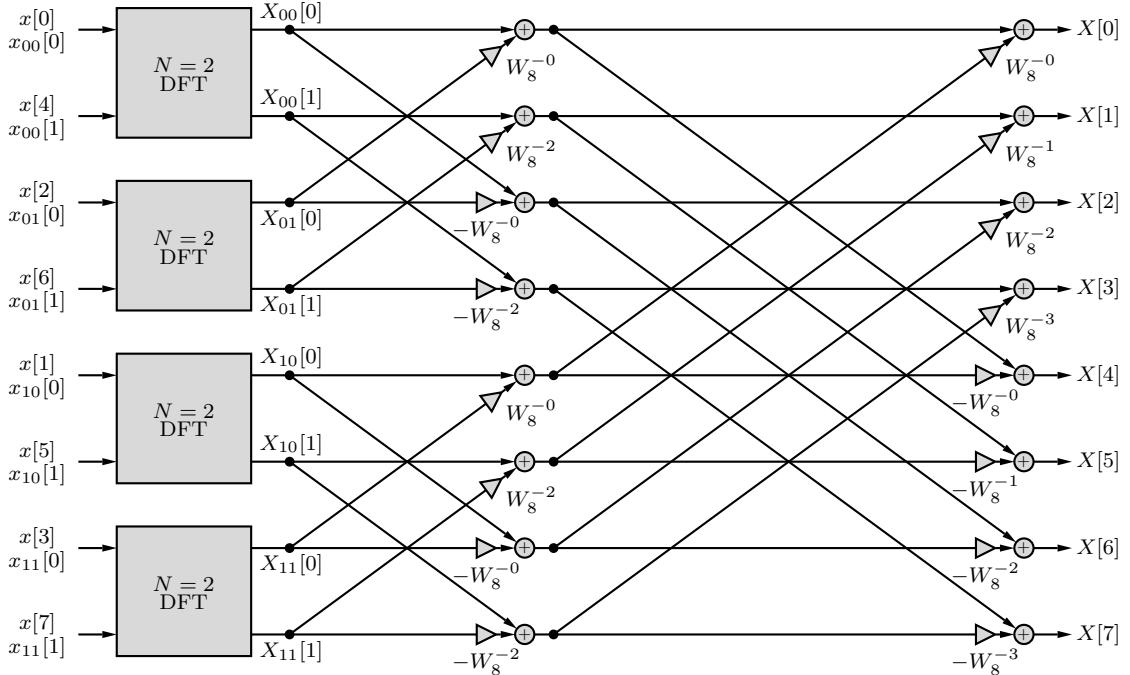


Figure 9.28: Dividing an 8-point DFT into four 2-point DFTs.

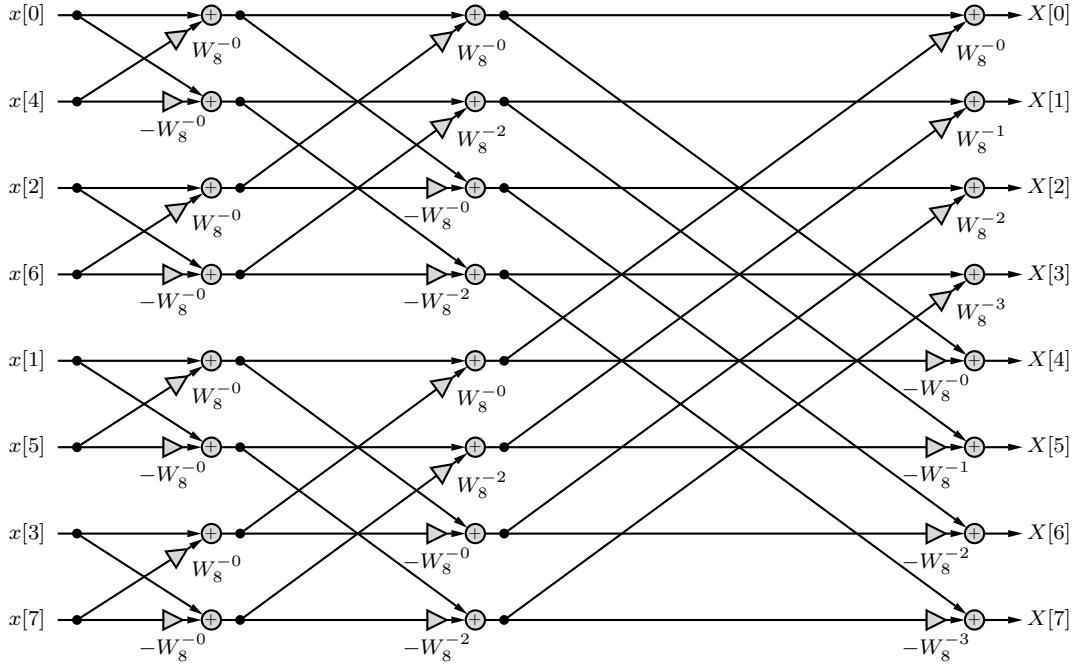


Figure 9.29: An 8-point decimation-in-time FFT.

To count the number of computations in the FFT, we observe that each stage requires an identical number of operations: N complex additions and N complex multiplications. Since there are $\log_2 N$ stages, the FFT algorithm requires, conservatively, a total of $N \log_2 N$ complex additions and $N \log_2 N$ complex multiplications to compute an N -point DFT. By using the multiplication-efficient form for each butterfly (see Fig. 9.26), we can reduce the number of complex multiplications by

half to $\frac{N}{2} \log_2 N$. Furthermore, many scale factors are 1 or -1 , which requires no multiplication at all. Figure 9.30 shows a multiplication-efficient form of the 8-point decimation-in-time FFT. This structure computes an 8-point DFT using 24 complex additions and only 5 complex multiplications. In fact, in this case, we can further reduce the number of complex multiplications to only 2 since the scale factor $W_8^{-2} = -j$ can, with a little clever programming, be implemented without multiplication.

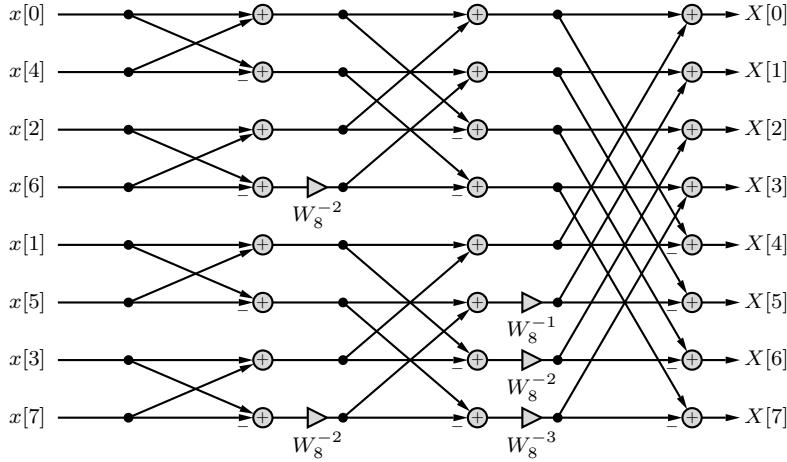


Figure 9.30: A multiplication-efficient 8-point decimation-in-time FFT.

A glance at Fig. 9.30 shows that although the output variables $X[0]$ through $X[7]$ appear in normal order, the input variables $x[0]$ through $x[7]$ are scrambled. This scrambling can be simply described by *bit-reversed addressing*. We express the index of each sample in binary form and then reverse the order of the binary digits (bit reversal) to find the actual sample ordering. For example, the index of sample $x[4]$ is 4, which is 100 in binary form. Bit reversing the binary digits yields 001, which is 1 in decimal. Thus, counting from 0, $x[4]$ appears in the second position at the input. Table 9.2 shows the bit-reversal process to determine sample ordering for $N = 8$. To accelerate FFT operations, some DSP processors even offer hardware bit-reversed addressing.

Normal Order	Binary Equivalent	Bit-Reversed Binary	Bit-Reversed Order
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Table 9.2: Using bit reversal to determine sample ordering for an $N = 8$ FFT.

▷ Drill 9.17 (Determining DIT FFT Input Order)

From Fig. 9.30, we see that $x[6]$ is the fourth input to an 8-point DIT FFT. What is the fourth input to a 128-point DIT FFT? What is the 100th input to a 128-point DIT FFT?



9.7.2 Decimation-in-Frequency Algorithm

In the *decimation-in-frequency* (DIF) FFT algorithm, instead of dividing the input $x[n]$ into two sequences of even- and odd-numbered samples, we divide $x[n]$ into two sequences formed by the first $\frac{N}{2}$ and the last $\frac{N}{2}$ samples, proceeding in the same way until a series of 2-point DFTs is reached in $\log_2 N$ stages. The resulting DIF FFT structure, shown in Fig. 9.31 for $N = 8$, has the input $x[n]$ in normal order and the output $X[k]$ in bit-reversed order. The total number of computations in this algorithm is the same as that in the decimation-in-time algorithm.

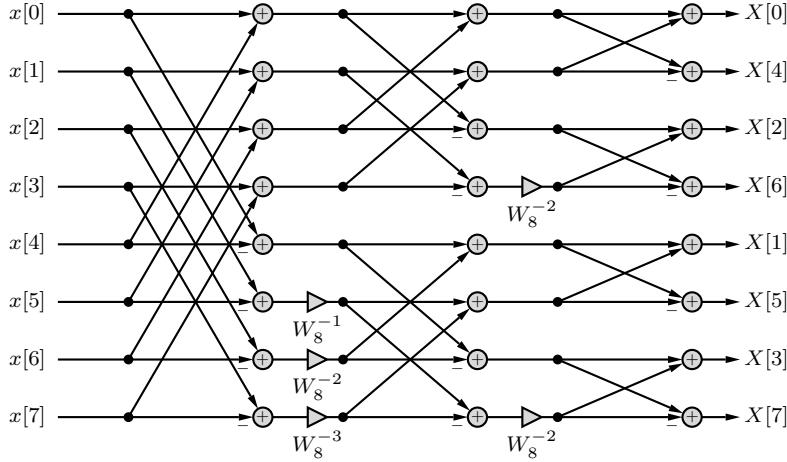


Figure 9.31: A multiplication-efficient 8-point decimation-in-frequency FFT.

We can also easily obtain a normal input and bit-reversed output structure from a standard decimation-in-time structure. All that needs to be done is to exchange various rows of the structure until the input is in normal order. Since the interconnections are not changed, these row exchanges do not affect the functionality of the structure. Take, for example, the 8-point DIT FFT of Fig. 9.30. By exchanging the $x[4]$ and $x[1]$ rows and the $x[6]$ and $x[3]$ rows, we obtain the structure of Fig. 9.32. Aside from the different positions of the twiddle factors, this structure is identical to the DIT FFT structure of Fig. 9.31.

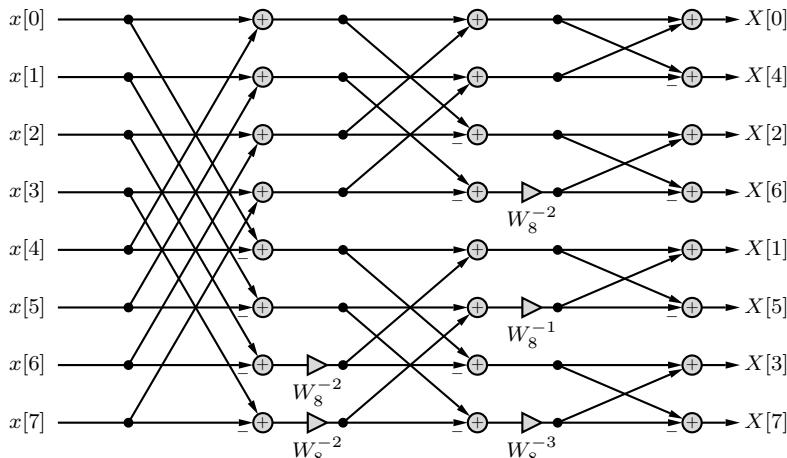


Figure 9.32: A multiplication-efficient 8-point decimation-in-time FFT with input in normal order.

Other FFT Algorithms

In our discussion so far, we have used the strategy of dividing the data into 2 at each stage. Clearly, this approach requires the sequence length to be a power of 2. In case the data does not satisfy this condition, we can zero pad the input to render the length a power of 2. Most FFTs use this algorithm, known as a *radix-2 FFT*. Both the DIT FFT of Fig. 9.30 and the DIF FFT of Fig. 9.31 are radix-2 FFTs.

There are also other algorithms that use successive divisions of the sequence by 4, 8, and so on. These approaches do save in number of computations, but the savings are not huge, structure complexity increases, and the input length is further restricted to be a power of 4, 8, and so on. Just as dividing the input into 2 at each stage results in a radix-2 FFT, dividing the data into 4 at each stage results in a radix-4 FFT, dividing into 8 at each stage results in a radix-8 FFT, and so on.

There are also algorithms that do not require the input length to be a power of 2 at all. Usually, such algorithms factor the input length and then divide the input by these factors, one factor for each stage. For example, a length $N = 6$ factors into 3×2 . Thus, an $N = 6$ DFT can be broken into two stages corresponding to the factors 3 and 2. When different stages utilize different factors, the result is known as a *mixed-radix FFT*. Such algorithms, which are substantially more complicated than radix-2 algorithms, generally do not work well when the length has only a few factors.

Estimating the CTFT with the DFT

Before leaving the DFT and the computationally efficient FFT algorithm, let us briefly discuss using the DFT to estimate the CTFT. This topic is of particular importance since we would like to use the computational simplicity and efficiency of the FFT to help us analyze signals that are continuous-time in origin. From the end of Sec. 6.4, we know that we can use the DTFT to exactly compute the CTFT as long as the CT signal is bandlimited. From this chapter, we further know that we can exactly compute the DTFT using the DFT as long as the DT signal has finite duration. Therefore, it would seem that there is hope to compute the CTFT exactly using the DFT.

Unfortunately, the conditions of finite duration and finite bandwidth are mutually exclusive, and some signals are neither! Thus, it is not possible to exactly compute the CTFT using the DFT. Before giving up in despair, however, we should note that all practical CT signals have finite energy and are *effectively bandlimited and effectively timelimited*. That is, any practical CT signal can be represented, with a controllable level of error, using some finite signal bandwidth B and some finite signal duration T_0 . To estimate the CTFT with the DFT, then, requires two basic conditions. First, we must sample the CT signal with a sampling frequency F_s that exceeds twice its effective bandwidth B (Nyquist criterion). Second, we must collect enough samples N so as to include the signal over its effective duration. Let us demonstrate these ideas through an example.

▷ Example 9.15 (Estimating the CTFT with the DFT)

Estimate the CTFT $X_c(\omega)$ of the CT signal $x_c(t) = e^{-t/10}u(t)$ using the DFT. Graphically compare the true and estimated spectra.

In this case, $x_c(t)$ is neither timelimited nor bandlimited, so the DFT can only approximate the CTFT. From entry 1 of Table 1.1 (page 48), signal $x_c(t) = e^{-t/10}u(t)$ has CTFT given by

$$X_c(\omega) = \frac{1}{j\omega + 0.1}.$$

This spectrum is lowpass in nature with peak magnitude $|X_c(0)| = 10$. To determine the effective bandwidth B , we use a 1% criterion,

$$0.01(10) = \left| \frac{1}{jB + 0.1} \right| \implies B = 0.1\sqrt{9999} \approx 10.$$

Using this effective bandwidth, the Nyquist sampling rate is $F_s = \frac{2B}{2\pi} = \frac{10}{\pi}$, and the sampling interval is $T = \frac{1}{F_s} = \frac{\pi}{10}$.

The signal $x_c(t)$ possesses an exponentially decaying envelope. To determine the effective duration T_0 , we again use a 1% criterion,

$$0.01(1) = e^{-T_0/10} \implies T_0 = -10 \ln(0.01).$$

Using the sampling interval $T = \pi/10$, the required number of DFT points is thus

$$N - 1 = \lceil \frac{T_0}{T} \rceil = \lceil -\frac{100}{\pi} \ln(0.01) \rceil = 147 \implies N = 148.$$

With these calculations complete, we are now prepared to estimate the CTFT using an N -point DFT. Combining Eq. (6.49), which connects the CTFT to the DTFT, and Eq. (9.1), which connects the DTFT to the DFT, a DFT-based estimate of the CTFT is

$$\hat{X}_c\left(\frac{k\Omega_0}{T}\right) = TX[k], \quad \left|\frac{k\Omega_0}{T}\right| \leq \frac{\pi}{T}. \quad (9.52)$$

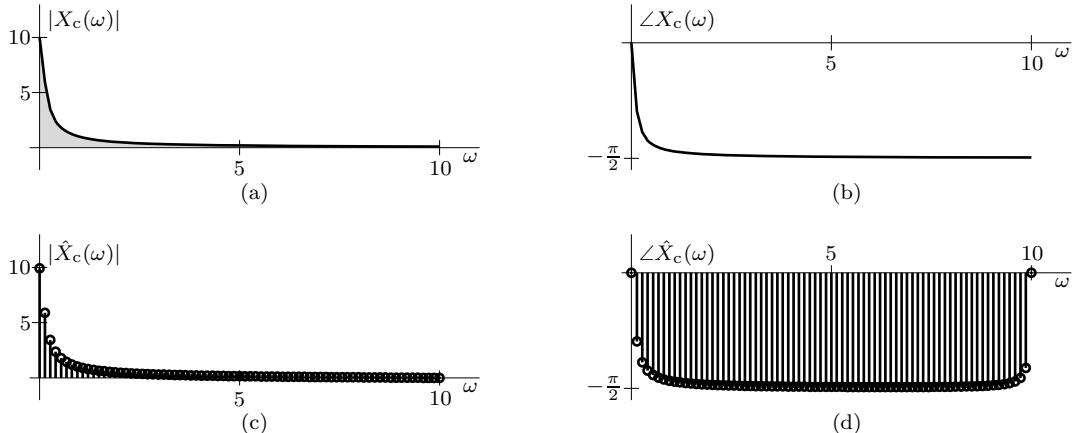


Figure 9.33: Estimating the CTFT with the DFT: (a) $|X_c(\omega)|$, (b) $\angle X_c(\omega)$, (c) $|\hat{X}_c(\omega)|$, and (d) $\angle \hat{X}_c(\omega)$.

Figure 9.33 compares the true and estimated spectra, computed using MATLAB.

```

01 xc = @(t) exp(-t/10).*(t>0)+0.5*(t==0); Xc = @(omega) 1./(1j*omega+0.1);
02 B = 10; Fs = 2*B/(2*pi); T = 1/Fs; N = ceil(-100/pi*log(0.01))+1;
03 n = 0:N-1; x = xc(n*T); Xhat = T*fft(x);
04 k = 0:ceil(N/2); Omega0 = 2*pi/N; omega = Omega0*k/T;
05 subplot(221); plot(omega,abs(Xc(omega)));
06 subplot(222); plot(omega,angle(Xc(omega)));
07 subplot(223); stem(omega,abs(Xhat(1:length(k)))));
08 subplot(224); stem(omega,angle(Xhat(1:length(k))));
```

Much like the impulse invariance method in Ch. 8, line 01 defines the CT unit step so that $u(0) = 0.5$. Lines 02–03 estimate the CTFT using the DFT, where for convenience we use MATLAB's built-in `fft` function to compute the DFT. Line 04 restricts our attention to those DFT points that correspond to the frequency range $0 \leq \omega \leq \frac{\pi}{T} = 10$. This is to avoid the (unnecessary) inconvenience of dealing with the second half of the DFT, which actually corresponds to negative frequencies. Lines

05–08 plot the true and estimated magnitude and phase spectra. Except for high frequencies, the DFT estimate is almost visually indistinguishable from the true CTFT. If more points are needed in the estimate, we simply zero pad the sampled signal before taking the DFT. Furthermore, estimate accuracy can be improved as needed by increasing (improving) our estimates of signal bandwidth and duration (B and T_0).

Example 9.15 \triangleleft

9.8 The Discrete-Time Fourier Series

Equation (9.17) shows that the DFT equations are periodic. Thus, although we started out with length- N sequences $x[n]$ and $X[k]$, the DFT equations themselves are periodic. In other words, using Eq. (9.5), if we compute $x[n]$ for values of n outside the range $0 \leq n \leq N - 1$, we find that $x[n]$ is periodic. The same is the case with $X[k]$. This observation allows us to write equations for the discrete-time Fourier series for an N -periodic signal.

Consider a length- N sequence $x[n]$ that is zero outside the range $0 \leq n \leq N - 1$. Let $\tilde{x}[n]$ be the N -periodic extension of $x[n]$, obtained by repeating $x[n]$ periodically as in Eq. (9.11). Similarly, let $\tilde{X}[k]$ be the N -periodic extension of $X[k]$, the length- N DFT of $x[n]$. Using $\Omega_0 = 2\pi/N$ and the periodicity property in Eq. (9.17), we have

$$\begin{aligned} \tilde{x}[n] &= \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}[k] e^{j\Omega_0 kn} \\ \text{and } \tilde{X}[k] &= \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\Omega_0 kn} = \sum_{n=0}^{N-1} x[n] e^{-j\Omega_0 kn}. \end{aligned}$$

Letting $\mathcal{D}[k] = \frac{1}{N} \tilde{X}[k]$, these two equations can be expressed as

$$\tilde{x}[n] = \sum_{k=0}^{N-1} \mathcal{D}[k] e^{j\Omega_0 kn} \quad (9.53)$$

$$\text{and } \mathcal{D}[k] = \frac{1}{N} \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\Omega_0 kn} = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\Omega_0 kn}. \quad (9.54)$$

Equation (9.53) is the synthesis equation of the discrete-time Fourier series (DTFS), and it expresses the N -periodic signal $\tilde{x}[n]$ as a sum of N exponentials consisting of $e^{j\Omega_0 n}$ and its harmonics. To find the DTFS of signal $\tilde{x}[n]$ is to represent it using Eq. (9.53). Unlike the continuous-time case, the discrete-time case has only N harmonics. This is because the discrete-time signals are bandlimited to 2π rad/sample, and the fundamental frequency is $\Omega_0 = 2\pi/N$. The analysis equation of the discrete-time Fourier series is given by Eq. (9.54), which expresses the N -periodic signal $\mathcal{D}[k]$ as a sum of N exponentials consisting of $e^{j\Omega_0 k}$ and its harmonics. Within the scale factor $\frac{1}{N}$, we see that the DTFS analysis and synthesis equations are identical to the DFT analysis and synthesis equations. Thus, the DTFS shares the same properties and insights of the DFT.

Periodic Extension of the Fourier Spectrum

Note that if $\phi[k]$ is an N -periodic function of k , then

$$\sum_{k=0}^{N-1} \phi[k] = \sum_{\{k\}_N} \phi[k], \quad (9.55)$$

where $\{k\}_N$ indicates any N consecutive values of k . This result follows because the right-hand side of Eq. (9.55) is the sum of N consecutive values of $\phi[k]$. Because $\phi[k]$ is periodic, each sample repeats with period N . Therefore, the same set of samples is present in any group of N consecutive samples, although not necessarily in the same order. Hence, the sum of any N consecutive samples of an N -periodic function $\phi[k]$ must be identical regardless of where we start the first term.

With these observations, let us return to the DTFS. Now $e^{-j\Omega_0 kn}$ is N -periodic because

$$e^{-j\Omega_0(k+N)n} = e^{-j\Omega_0 kn}e^{-j2\pi n} = e^{-jn\Omega_0 k}.$$

Therefore, if $\tilde{x}[n]$ is N -periodic, $\tilde{x}[n]e^{-j\Omega_0 kn}$ is also N -periodic. Hence, from Eq. (9.54), it follows that $\mathcal{D}[k]$ is also N -periodic, as is $\mathcal{D}[k]e^{j\Omega_0 kn}$. Now, because of Eq. (9.55), we can express Eqs. (9.53) and (9.54) as

$$\tilde{x}[n] = \sum_{\{k\}_N} \mathcal{D}[k]e^{j\Omega_0 kn} \quad (9.56)$$

and

$$\mathcal{D}[k] = \sum_{\{n\}_N} \tilde{x}[n]e^{-j\Omega_0 kn}. \quad (9.57)$$

If we plot $\mathcal{D}[k]$ for all values of k (rather than only $0 \leq k \leq N - 1$), then we see that the spectrum $\mathcal{D}[k]$ is N -periodic. Moreover, Eq. (9.55) shows that $\tilde{x}[n]$ can be synthesized by not only the N exponentials corresponding to $0 \leq k \leq N - 1$ but by any successive N exponentials in this spectrum, starting at any value of k (positive or negative). For this reason, it is customary to show the spectrum $\mathcal{D}[k]$ for all values of k and not just over the interval $0 \leq k \leq N - 1$. Yet, to synthesize $\tilde{x}[n]$ from this spectrum, we need to add only N consecutive components.

Adjacent spectral components of $\mathcal{D}[k]$ are separated by the frequency interval $\Omega_0 = \frac{2\pi}{N}$, and there are a total of N components repeating periodically along the $\Omega = k\Omega_0$ axis. Thus, $\mathcal{D}[k]$ repeats every 2π on the frequency scale Ω . Equations (9.55), (9.56), and (9.57) show that both $\tilde{x}[n]$ and its spectrum $\mathcal{D}[k]$ are periodic, and both have exactly N components over one period. The period of $\tilde{x}[n]$ is N , and that of $\mathcal{D}[k]$ is 2π radians.

Equation (9.57) shows that $\mathcal{D}[k]$ is complex in general, and $\mathcal{D}[-k]$ is the conjugate of $\mathcal{D}[k]$ if $\tilde{x}[n]$ is real. Thus, for real $\tilde{x}[n]$,

$$|\mathcal{D}[k]| = |\mathcal{D}[-k]| \quad \text{and} \quad \angle \mathcal{D}[k] = -\angle \mathcal{D}[-k].$$

Clearly, the magnitude spectrum $|\mathcal{D}[k]|$ is an even function of k , and the phase spectrum $\angle \mathcal{D}[k]$ is an odd function of k (or Ω). All these concepts are clarified by the examples to follow.

▷ Example 9.16 (Discrete-Time Fourier Series of a Sinusoid)

Find the discrete-time Fourier series (DTFS) for the periodic signal $\tilde{x}[n] = \sin(0.1\pi n)$, shown in Fig. 9.34. Sketch the magnitude and phase spectra $|\mathcal{D}[k]|$ and $\angle \mathcal{D}[k]$.

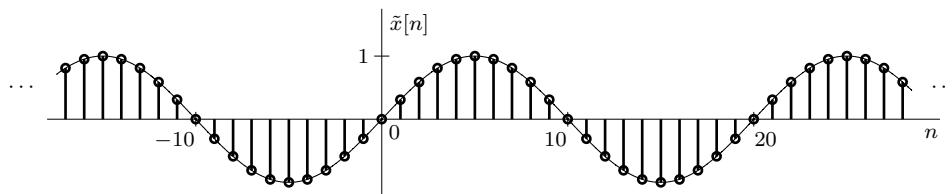


Figure 9.34: Periodic signal $\tilde{x}[n] = \sin(0.1\pi n)$.

In this case, the sinusoid $\tilde{x}[n] = \sin(0.1\pi n)$ is periodic because $\frac{\Omega}{2\pi} = \frac{1}{20}$ is a rational number, and the period N is (see Eq. (4.31))

$$N = m \left(\frac{2\pi}{\Omega} \right) = m \left(\frac{2\pi}{0.1\pi} \right) = 20m.$$

The smallest value of m that makes $20m$ an integer is $m = 1$. Therefore, the period $N = 20$, and $\Omega_0 = \frac{2\pi}{N} = 0.1\pi$. From Eq. (9.55), the DTFS for $\tilde{x}[n]$ can be expressed as

$$\tilde{x}[n] = \sum_{\{k\}_{20}} \mathcal{D}[k] e^{j0.1\pi kn},$$

where the sum is performed over any 20 consecutive values of k . We shall select the range $-10 \leq k < 10$ (values of k from -10 to 9). This choice corresponds to synthesizing $\tilde{x}[n]$ using the spectral components in the fundamental frequency range ($-\pi \leq \Omega < \pi$). Thus,

$$\tilde{x}[n] = \sum_{k=-10}^9 \mathcal{D}[k] e^{j0.1\pi kn},$$

where, according to Eq. (9.57),

$$\mathcal{D}[k] = \frac{1}{20} \sum_{n=-10}^9 \sin(0.1\pi n) e^{-j0.1\pi kn} = \frac{1}{40j} \left[\sum_{n=-10}^9 e^{j0.1\pi(1-k)n} - \sum_{n=-10}^9 e^{-j0.1\pi(1+k)n} \right].$$

Although this expression for $\mathcal{D}[k]$ is valid for all k , let us focus on a single period of $\mathcal{D}[k]$ where k takes on all values between -10 and 9 (the fundamental band). Over this range, the first sum on the right-hand side is zero for all values of k except $k = 1$, when the sum is equal to $N = 20$.[†] Similarly, the second sum is zero for all $-10 \leq k \leq 9$ except $k = -1$, when it is equal to $N = 20$. Therefore,

$$\mathcal{D}[1] = \frac{1}{2j} = \frac{1}{2} e^{-j\pi/2} \quad \text{and} \quad \mathcal{D}[-1] = -\frac{1}{2j} = \frac{1}{2} e^{j\pi/2}.$$

All the remaining coefficients $\mathcal{D}[k]$ over $-10 \leq k \leq 9$ are zero. The corresponding discrete-time Fourier series is given by

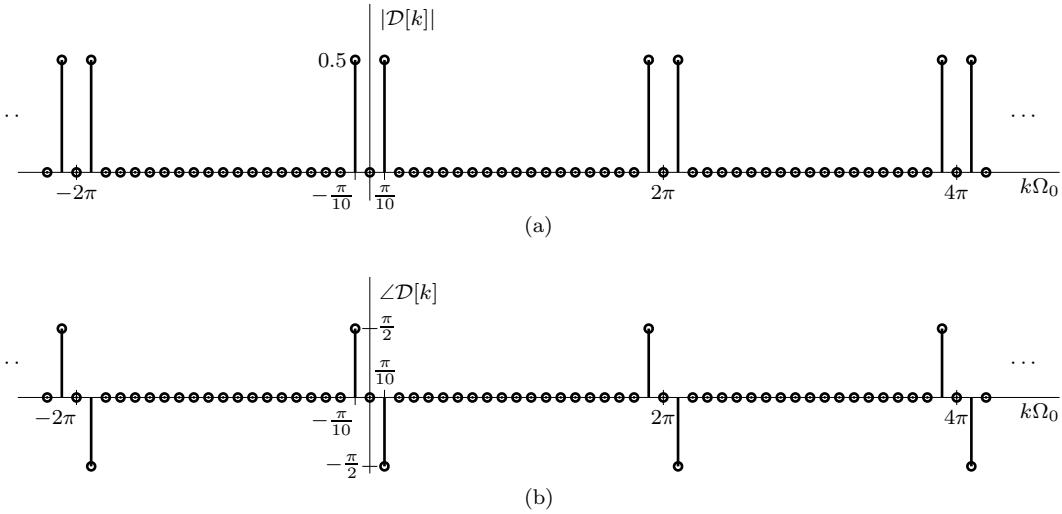
$$\tilde{x}[n] = \frac{1}{2j} (e^{j0.1\pi n} - e^{-j0.1\pi n}) = \sin(0.1\pi n). \quad (9.58)$$

Here, the fundamental frequency $\Omega_0 = 0.1\pi$.

Figures 9.35a and 9.35b show the respective magnitude and phase spectra, plotted as a function of frequency $\Omega = k\Omega_0$. Because of the periodicity property discussed earlier, both the magnitude and phase spectra are periodic functions of k with period $N = 20$. Nonetheless, it is sufficient to specify the spectra over any single period, such as the fundamental band $-\pi \leq \Omega < \pi$ ($-10 \leq k < 10$). According to Eq. (9.58), there are only two nonzero spectral components to $\tilde{x}[n]$, corresponding to $\mathcal{D}[1]$ and $\mathcal{D}[-1]$. The remaining 18 fundamental-band coefficients are zero. We can synthesize $\tilde{x}[n]$ by adding these 20 spectral components. Observe that the magnitude spectrum is an even function and the angle or phase spectrum is an odd function of k (or Ω) as expected.

The result of Eq. (9.58) is a trigonometric identity, and it could have been obtained immediately without the formality of finding the Fourier coefficients. We have intentionally chosen this trivial example to introduce the reader gently to the concept of the discrete-time Fourier series and its periodic nature. The DTFS is a way of expressing a periodic signal $\tilde{x}[n]$ in terms of exponentials of the form $e^{j\Omega_0 kn}$ and their harmonics. The result in Eq. (9.58) is merely a statement of the (obvious) fact that $\sin(0.1\pi n)$ can be expressed as a sum of the two exponentials $e^{j0.1\pi n}$ and $e^{-j0.1\pi n}$.

[†]To see this fact, we can simply evaluate the sum as a geometric progression or, more simply, apply an identity similar to Eq. (9.4).

Figure 9.35: Spectrum of $\tilde{x}[n] = \sin(0.1\pi n)$: (a) $|\mathcal{D}[k]|$ and (b) $\angle \mathcal{D}[k]$.

Because of the periodicity property of the discrete-time exponentials $e^{j\Omega_0 kn}$, the discrete-time Fourier series components can be selected using any $N = 20$ consecutive values of k (or within any 2π interval of frequencies Ω). For example, if we select the range $0 \leq k < 20$ (or $0 \leq \Omega < 2\pi$), we obtain the Fourier series as

$$\tilde{x}[n] = \sin(0.1\pi n) = \frac{1}{2j} (e^{j0.1\pi n} - e^{-j0.1\pi n}).$$

This series is equivalent to that in Eq. (9.58) because, as seen in Sec. 4.3, the two exponentials $e^{j1.9\pi n}$ and $e^{-j0.1\pi n}$ are identical.

Example 9.16 ◀

▷ **Example 9.17 (DTFS Analysis of a Periodic Gate Function)**

Find the discrete-time Fourier series spectrum $\mathcal{D}[k]$ for the periodic sampled gate function shown in Fig. 9.36.

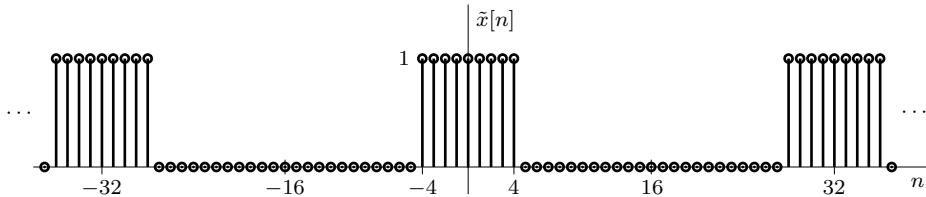


Figure 9.36: Periodic sampled gate function.

In this case, $N = 32$ and $\Omega_0 = \frac{2\pi}{32} = \frac{\pi}{16}$. Therefore,

$$\tilde{x}[n] = \sum_{k=\{32\}} \mathcal{D}[k] e^{j\frac{\pi}{16} kn},$$

where

$$\mathcal{D}[k] = \frac{1}{32} \sum_{n=\{32\}} \tilde{x}[n] e^{-j\frac{\pi}{16}kn}.$$

For our convenience, we shall choose the interval $-16 \leq n \leq 15$ to compute $\mathcal{D}[k]$, although any other interval of 32 points gives the same result. Over this interval, $\tilde{x}[n] = 1$ for $-4 \leq n \leq 4$ and is zero for all the remaining values of n . Thus,

$$\mathcal{D}[k] = \frac{1}{32} \sum_{n=-16}^{15} \tilde{x}[n] e^{-j\frac{\pi}{16}kn} = \frac{1}{32} \sum_{n=-4}^4 e^{-j\frac{\pi}{16}kn}.$$

This is a geometric progression with a common ratio $e^{-j\frac{\pi}{16}k}$. Using Table 5.1 on page 290,

$$\mathcal{D}[k] = \frac{1}{32} \left[\frac{e^{j\frac{4\pi k}{16}} - e^{-j\frac{5\pi k}{16}}}{1 - e^{-j\frac{\pi k}{16}}} \right] = \left(\frac{1}{32} \right) \frac{e^{-j\frac{0.5\pi k}{16}} \left[e^{j\frac{4.5\pi k}{16}} - e^{-j\frac{4.5\pi k}{16}} \right]}{e^{-j\frac{0.5\pi k}{16}} \left[e^{j\frac{0.5\pi k}{16}} - e^{-j\frac{0.5\pi k}{16}} \right]}.$$

Using Euler's identities and substituting Ω_0 for $\frac{\pi}{16}$, we obtain the DTFS spectrum as

$$\mathcal{D}[k] = \left(\frac{1}{32} \right) \frac{\sin\left(\frac{4.5\pi k}{16}\right)}{\sin\left(\frac{0.5\pi k}{16}\right)} = \left(\frac{1}{32} \right) \frac{\sin(4.5k\Omega_0)}{\sin(0.5k\Omega_0)}.$$

This spectrum, shown over more than one period, is depicted in Fig. 9.37.

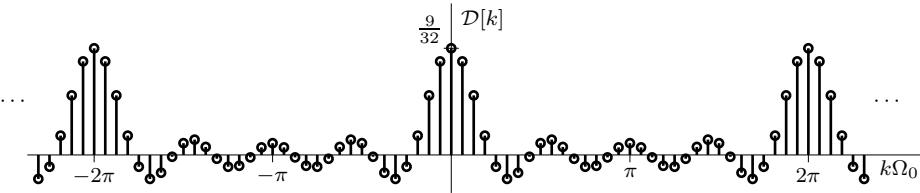


Figure 9.37: Fourier spectrum $\mathcal{D}[k]$ of a periodic sampled gate function.

Example 9.17 ◀

By restricting the range of a signal $x[n]$ to $0 \leq n \leq N - 1$, its DFT $X[k]$ equals the uniform samples of its DTFT $X(\Omega)$. The DTFS, on the other hand, is usually derived assuming that the signal $x[n]$ is periodic. As we have seen, however, the DFT and the DTFS are, within a scale factor, mathematically identical to one another. This fact once again reinforces the periodic perspective of the DFT. The DFT is a capable multipurpose tool that handles both the DTFT and the DTFS.

▷ Drill 9.18 (DTFS Using an Alternate Frequency Interval)

Determine the DTFS of the periodic signal $\tilde{x}[n]$ using the DTFS spectrum of Fig. 9.35 over the interval $2\pi < \Omega \leq 4\pi$. Show that this synthesis is equivalent to that in Eq. (9.58).

◀

▷ Drill 9.19 (Discrete-Time Fourier Series of a Sum of Two Sinusoids)

Find the period N and the DTFS for $\tilde{x}[n] = 4 \cos(0.2\pi n) + 6 \sin(0.5\pi n)$. Express the DTFS using $\mathcal{D}[k]$ computed over $0 \leq \Omega < 2\pi$.

◀

▷ **Drill 9.20 (Computing the DTFS Spectrum Using the DFT)**

Using MATLAB and the matrix form of the DFT, compute and plot the DTFS spectrum $\mathcal{D}[k]$ from Ex. 9.17.

△

9.9 Summary

The discrete Fourier transform (DFT) may be viewed as an economy-class DTFT, applicable when $x[n]$ is of finite length. The DFT is one of the most important tools for digital signal processing, especially when we implement it using the efficient fast Fourier transform (FFT) algorithm. Computed using the FFT algorithm, the DFT is truly the workhorse of modern digital signal processing.

The DFT $X[k]$ of an N -point signal $x[n]$ starting at $n = 0$ represents N uniform samples taken over the frequency interval $0 \leq \Omega < 2\pi$ of $X(\Omega)$, the DTFT of $x[n]$. Because DFT values are samples of the DTFT, a plot of the DFT $X[k]$ is like viewing the DTFT $X(\Omega)$ through a picket fence. If N is large enough, $X[k]$ will include enough samples to give a reasonably good picture of $X(\Omega)$. If N is not large enough, we can artificially increase N by padding zeros to the end of $x[n]$. This gives a corresponding increase in the size of the DFT, which is to say that the number of samples of the DTFT also increases. Both the DFT and its inverse are conveniently computed using simple matrix representations. Furthermore, we can reconstruct $X(\Omega)$ from the DFT using the interpolation formula. Taken together, we see that the DFT proves extremely useful in DTFT computations.

To ensure a unique one-to-one relationship between an N -point signal $x[n]$ and its DFT $X[k]$, it is customary to restrict $x[n]$ to the range $0 \leq n \leq N - 1$. Without such a restriction, the uniqueness of the DFT is lost. The reason for this nonuniqueness is that the contribution to the DFT from any sample of $x[n]$ at $n = r$ is the same as that of the same sample at the location $n = r + mN$, where m is an arbitrary integer. These observations also lead to a periodic view or circular representation of the signal $x[n]$.

The circular (or periodic) representation of a length- N sequence proves very useful in DFT studies. A modulo- N shift, for example, is conveniently interpreted as a circular shift. Many of the properties of the DFT are best viewed in the context of a circular representation. A circular representation is particularly useful in the graphical interpretation of circular convolution.

By properly zero padding two finite-duration sequences, circular convolution produces a result that is identical to linear convolution. To this end, both sequences are zero padded to have the same length as the linear convolution result. If a sufficient number of zeros are not padded, circular convolution yields erroneous results because of aliasing. However, even under such aliasing, it is possible that some of the samples of circular convolution are equal to the samples of linear convolution.

Circular convolution can be accomplished, often with a savings in overall computations, by using the DFT and its inverse. If the FFT algorithm is used, this process is known as fast convolution. With the potential for significant computational savings, fast convolution is attractive for certain discrete-time filtering applications. When a large, possibly infinite-duration sequence $x[n]$ is to be convolved with a finite-duration sequence $h[n]$, it is often desirable to partition $x[n]$ in smaller blocks, convolve each of these blocks with $h[n]$, and then combine the outputs corresponding to all the blocks. Two methods, overlap and add and overlap and save, are discussed. In the latter method, we use circular convolution with aliasing. Both methods require about the same amount of computations.

Ordinarily, direct computation of an N -point DFT requires N^2 complex multiplications and about N^2 complex additions. By exploiting symmetry and other characteristics, DFT computation efficiency is improved. Goertzel's algorithm is particularly simple and can reduce by half the number

of additions and reduce by three-quarters the number of multiplications. More dramatic improvements in computational efficiency are achieved by using the fast Fourier transform (FFT) algorithm. The FFT algorithm computes the DFT using only about $N \log_2 N$ complex additions and $N \log_2 N$ complex multiplications. For large N , the FFT yields dramatic benefits.

There is a close connection between the DFT of an N -point sequence $x[n]$ and the discrete-time Fourier series of $\tilde{x}[n]$, an N -periodic signal whose first period over $0 \leq n \leq N - 1$ is $x[n]$. Equations for computing the discrete-time Fourier series and its inverse are scaled versions of the DFT equations.

References

1. Cooley, J. W., and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series," *Mathematics of Computation*, Vol. 19, April 1965, pp. 297-301.
2. Lathi, B. P., *Linear Systems and Signals*, 2nd Ed., Oxford University Press, New York, 2005.
3. Mitra, S. K., *Digital Signal Processing: A Computer-Based Approach*, 3rd Ed., McGraw-Hill, New York, 2006.
4. Oppenheim, A. V., Schafer, R. W., and Buck, R. J., *Discrete-Time Signal Processing*, 2nd Ed., Prentice-Hall, Upper Saddle River, NJ, 1999.

Problems

9.1-1 For this problem, interpret the N -point DFT as an N -periodic function of k . Answer yes or no if the following frequency-domain signals are valid DFTs. For each valid DFT, determine the size N of the DFT and whether the time-domain signal is real.

- (a) $X_a[k] = j - \pi$
- (b) $X_b[k] = \sin(k/10)$
- (c) $X_c[k] = \sin(\pi k/10)$
- (d) $X_d[k] = \left(\frac{1+j}{\sqrt{2}}\right)^k$
- (e) $X_e[k] = \langle k + \pi \rangle_{10}$

9.1-2 Find the 4-point DFT of the signal

$$x[n] = 3\delta[n] - 2\delta[n-2] + 2\delta[n-3].$$

Compare the DFT spectrum $X[k]$ with the corresponding DTFT spectrum $X(\Omega)$.

9.1-3 Repeat Prob. 9.1-2 using the complex signal

$$x[n] = 3\delta[n] - 2j\delta[n-1] + \delta[n-2] + 2j\delta[n-3].$$

9.1-4 Consider the signal

$$x[n] = \delta[n] - 2\delta[n-1] + 2\delta[n-2] - \delta[n-3].$$

- (a) Compute and plot the 4-point DFT $X[k]$.
- (b) Using the fewest number of zeros possible, zero pad $x[n]$ so that the DFT frequency resolution is at least $\Omega_0 = 0.1$. Compute and plot the corresponding DFT $X_{\text{pad}}[k]$.

9.1-5 The 8-point DFT of a real signal $x[n]$ is known over $0 \leq k \leq 4$ to be

$$X[k] = 1 + 2j\delta[k-2] + \delta[k-4].$$

- (a) Determine $X[k]$ for $5 \leq k \leq 7$.
- (b) Determine and plot the corresponding time-domain signal $x[n]$.

9.1-6 Consider a complex signal comprised of two closely spaced complex exponentials

$$x[n] = e^{j2\pi 30n/100} + e^{j2\pi 33n/100}.$$

Plot each N -point DFT as a function of frequency $f_k = k/N$.

(a) Compute and plot the 10-point DFT of $x[n]$ using 10 samples ($0 \leq n \leq 9$). From the plot, can both exponentials be identified? Explain.

(b) Zero pad the signal from (a) with 490 zeros, and then compute and plot the 500-point DFT. Does this improve the picture of the DFT? Explain.

(c) Compute and plot the 100-point DFT of $x[n]$ using 100 samples ($0 \leq n \leq 99$). From the plot, can both exponentials be identified? Explain.

(d) Zero pad the signal from (c) with 400 zeros, and then compute and plot the 500-point DFT. Does this improve the picture of the DFT? Explain.

9.1-7 Repeat Prob. 9.1-6 using the complex signal

$$x[n] = e^{j2\pi 30n/100} + e^{j2\pi 31.5n/100}.$$

9.1-8 Consider a complex signal composed of a dc term and two complex exponentials

$$x[n] = 1 + e^{j2\pi 30n/100} + 0.5e^{j2\pi 43n/100}.$$

Plot each N -point DFT as a function of frequency $f_k = k/N$.

(a) Compute and plot the DFT of $x[n]$ using 20 samples ($0 \leq n \leq 19$). From the plot, can the two non-dc exponentials be identified? Given the amplitude relation between the two, the lower-frequency peak should be twice as large as the higher-frequency peak. Is this the case? Explain.

(b) Zero pad the signal from (a) to a total length of 500. Does this improve locating the two non-dc exponential components? Is the lower-frequency peak twice as large as the higher-frequency peak? Explain.

(c) Generate a length-20 Hann window (see Table 8.5) and apply it to $x[n]$. Using this windowed function, repeat parts (a) and (b). Comment on whether the window function helps or hinders the analysis.

9.1-9 Repeat Prob. 9.1-8 using the complex signal

$$x[n] = 1 + e^{j2\pi 30n/100} + 0.5e^{j2\pi 38n/100}.$$

9.1-10 Determine the 4-point DFT matrix \mathbf{D}_4 and the corresponding inverse DFT matrix \mathbf{D}_4^{-1} .

9.1-11 This problem investigates zero padding applied in the frequency domain. Plot each N -point DFT as a function of frequency $f_k = k/N$.

- (a) In MATLAB, create a vector \mathbf{x} that contains one period of the sinusoid $x[n] = \cos(\frac{\pi}{2}n)$. Plot the result. How “sinusoidal” does the signal appear?
- (b) Using the `fft` command, compute the DFT \mathbf{X} of vector \mathbf{x} . Plot the magnitude of the DFT coefficients. Do they make sense?
- (c) Zero pad the DFT vector to a total length of 100 by inserting the appropriate number of zeros in the middle of the vector \mathbf{X} . Call this zero-padded DFT sequence \mathbf{Y} . Why are zeros inserted in the middle rather than at the end? Take the inverse DFT of \mathbf{Y} and plot the result. What similarities exist between the new signal \mathbf{y} and the original signal \mathbf{x} ? What are the differences between \mathbf{x} and \mathbf{y} ? What is the effect of zero padding in the frequency domain? How is this type of zero padding similar to zero padding in the time domain?
- (d) Derive a general modification to the procedure of zero padding in the frequency domain to ensure that the amplitude of the resulting time-domain signal is left unchanged.
- (e) Consider one period of a square wave described by the length-8 vector $[1, 1, 1, 1, -1, -1, -1, -1]$. Zero pad the DFT of this vector to a length of 100, and call the result \mathbf{S} . Scale \mathbf{S} according to (d), take the inverse DFT, and plot the result. Does the new time-domain signal $s[n]$ look like a square wave? Explain.

9.1-12 Consider a length-4 unit amplitude pulse $x[n] = u[n] - u[n-4]$.

- (a) Compute and plot the 4-point DFT $X[k]$ of signal $x[n]$.
- (b) Use the DFT interpolation formula to compute the DTFT $X(\Omega)$ from the DFT $X[k]$. Plot $X(\Omega)$.
- (c) Sample $X(\Omega)$ over $0 \leq \Omega < 2\pi$ using 10 equally spaced points to create a signal $X_{10}[k]$. Compute and plot the IDFT of $X_{10}[k]$. Comment on your results.

9.2-1 MATLAB’s `fft` command computes the DFT of a vector \mathbf{x} assuming that the first sample occurs at time $n = 0$. Given that $\mathbf{X} = \text{fft}(\mathbf{x})$ has already been computed, derive a method to correct \mathbf{X} to reflect an arbitrary starting time $n = n_0$.

9.2-2 Determine the following:

- (a) $\langle n \rangle_5$ for $-5 \leq n \leq 10$
- (b) $\langle n \rangle_6$ for $-5 \leq n \leq 10$
- (c) $\langle -n \rangle_5$ for $-5 \leq n \leq 10$
- (d) $\langle n+3 \rangle_6$ for $-5 \leq n \leq 10$

9.2-3 Determine $\langle 3 - \langle n+2 \rangle_6 \rangle_5$ for $-5 \leq n \leq 10$.

9.2-4 Determine three different 4-point signals $x_a[n]$, $x_b[n]$, and $x_c[n]$ that, when modulo-4 shifted, all equal the 4-point signal

$$x[n] = 3\delta[n] - 2\delta[n-2] + 2\delta[n-3].$$

Using $\Omega_0 = \pi/2$, express the DTFT samples $X_a(k\Omega_0)$, $X_b(k\Omega_0)$, and $X_c(k\Omega_0)$ in terms of the DFT $X[k]$ of $x[n]$.

9.2-5 Consider the 4-point signal

$$x[n] = 3\delta[n] - 2\delta[n-2] + 2\delta[n-3].$$

- (a) Determine the circular representation of $x[n]$.
- (b) Determine the circular representation of modulo-4 reflection $x[\langle -n \rangle_4]$.
- (c) Determine the circular representation of modulo-4 shift $x[\langle n-1 \rangle_4]$.
- (d) Determine the circular representation of $x[\langle 3-n \rangle_4]$.

9.2-6 Repeat Prob. 9.2-5 using the complex signal

$$x[n] = 3\delta[n] - 2j\delta[n-1] + \delta[n-2] + 2j\delta[n-3].$$

9.3-1 A designer accidentally takes the N -point inverse DFT of a real signal instead of the N -point DFT. Devise a simple and efficient method to correct the result without re-computing the DFT.

9.3-2 If $y[n]$ is the real portion of $x[n]$, show that

$$Y[k] = \frac{1}{2}(X[k] + X^*[\langle -k \rangle_N]).$$

9.3-3 If $y[n]$ is the imaginary portion of $x[n]$, show that

$$Y[k] = \frac{1}{2j}(X[k] - X^*[\langle -k \rangle_N]).$$

9.3-4 If DFT $Y[k]$ is the real portion of DFT $X[k]$, determine an expression for $y[n]$ in terms of $x[n]$.

9.3-5 If DFT $Y[k]$ is the imaginary portion of DFT $X[k]$, determine an expression for $y[n]$ in terms of $x[n]$.

9.3-6 Using only $\delta[n] \longleftrightarrow 1$ and properties of the DFT, determine the 4-point DFTs of the following 4-point sequences:

- (a) $x_a = [0, 2, 0, 0]$
- (b) $x_b = [0, 0, 0, 4]$
- (c) $x_c = [0, 0, j, 0]$
- (d) $x_d = [0, 4, -j, 2]$

9.3-7 If the 4-point signal $[1, 2, 3, 4]$ has DFT $X[k]$, use DFT properties to determine the time-domain signals that correspond to the following spectra:

- (a) $X_a[k] = X^*[k]e^{j2\pi k/N}$
- (b) $X_b[k] = X[k] + X^*[k]$
- (c) $X_c[k] = X[\langle -k - 1 \rangle_4]$
- (d) $X_d[k] = X[k](-1)^k$
- (e) $X_e[k] = X^2[k]$

9.4-1 Use the graphical method to compute the circular convolution of

$$[1, -2, 3, -4]$$

and

$$[1, -1, -2, 2].$$

9.4-2 Use the graphical method to compute the circular convolution of

$$[1, -2, 3, -4, 0, 0]$$

and

$$[0, 0, 1, -1, -2, 2].$$

9.4-3 Use the graphical method to compute the circular convolution of

$$[1, j, -1, -j]$$

and

$$[1, j, -j, 1].$$

9.4-4 Use circular convolution to compute the linear convolution of

$$[1, 2, 3, 4]$$

and

$$[1, -1, 1].$$

9.4-5 Use circular convolution to compute the linear convolution of

$$[1, -2, 3, -4]$$

and

$$[4, -3, 2, -1].$$

9.4-6 Determine both the linear and circular convolutions of

$$[1, -2, 2, -1]$$

and

$$[1, -1, -2, 0].$$

Using the idea of time-domain aliasing, show how to obtain the circular convolution from the linear convolution.

9.5-1 Using DFT-based convolution, determine the output $y[n]$ of an LTID filter with impulse response $h[n] = \delta[n] + 2\delta[n-1] + \delta[n-2]$ to the input $x[n] = u[n] - u[n-4]$.

9.5-2 Repeat Prob. 9.5-1 for the input $x[n] = u[n-4] - 2u[n-8] + u[n+12]$.

9.5-3 An input $x[n] = (0.5)^n u[n]$ is applied to an LTID system with impulse response $h[n] = \delta[n] - \delta[n-1]$. Use the overlap-and-add method with $N_x = 3$ to find the output $y[n]$ over $0 \leq n \leq 19$.

9.5-4 Repeat Prob. 9.5-3 for the input $x[n] = \cos(2\pi n/6)u[n]$.

9.5-5 Repeat Prob. 9.5-3 for an LTID system with impulse response $h[n] = \delta[n] - \delta[n-2]$.

9.5-6 An input $x[n] = (0.5)^n u[n]$ is applied to an LTID system with impulse response $h[n] = \delta[n] - \delta[n-1]$. By initially partitioning the input $x[n]$ into nonoverlapping blocks of length 3, use the overlap-and-save method to find the output $y[n]$ over $0 \leq n \leq 19$.

9.5-7 Repeat Prob. 9.5-6 for the input $x[n] = \cos(2\pi n/6)u[n]$.

9.5-8 Repeat Prob. 9.5-6 for an LTID system with impulse response $h[n] = \delta[n] - \delta[n-2]$.

9.6-1 Show how to use a first-order filter to compute $X[3]$ of an ($N = 10$)-point DFT. If the system operates at $F_s = 1$ kHz, what are the digital and analog frequencies (Ω and ω) corresponding to $X[3]$?

9.6-2 We are interested in using a DSP operating at $F_s = 1000$ Hz to compute the frequency content of an input signal by using the appropriate DFT frequency bin.

- (a) Devise a Goertzel structure like Fig. 9.25 to compute as rapidly as possible (minimum N) the 50-Hz content of an input $x[n]$. What are the corresponding values of N and k ?
- (b) Simulate your structure in MATLAB, and compute the outputs in response to inputs $x_1[n] = \cos(2\pi 50n/1000)$ and $x_2[n] = \cos(2\pi 75n/1000)$. Comment on the results.

9.6-3 Repeat Prob. 9.6-2, but use N that is twice the minimum value needed to ensure that 50 Hz is a DFT frequency bin. How does this larger N change the system behavior and results? What happens if N is selected three times the minimum value needed to ensure that 50 Hz is a DFT frequency bin? Does increasing N always improve the results of the system?

9.6-4 Suppose that we are only interested in K of the N frequency bins of an N -point DFT. Devise a rule to determine when Goertzel's algorithm is more efficient than the FFT to compute the K desired DFT values.

9.6-5 A *dual-tone multiple-frequency* (DTMF) telephone keypad, divided into three columns and four rows, generates a different two-tone signal for each key. The frequency of one tone is determined by row as 697, 770, 852, or 941 Hz. The frequency of the other tone is determined by column as 1209, 1336, or 1477 Hz. To decode a DTMF signal requires the ability to detect each of these seven frequencies.

- (a) Determine a suitable sampling rate F_s and DFT size N to decode a DTMF signal $x[n]$. Assume that a key is pressed for a duration of at least 0.1 s. What seven values of k ($0 \leq k \leq N-1$) correspond to the desired DTMF frequencies?
- (b) Show an efficient Goertzel structure to compute $X[k]$, the DFT of DTMF signal $x[n]$.

9.6-6 As shown in Fig. 9.24, Goertzel's algorithm computes a value $X[k]$ using a first-order IIR filter structure. While this filter is not used in the typical way (only one value of the output is really used), it is still informative to investigate the frequency response for this structure. Using $N = 10$, plot the magnitude responses for each of the first-order complex IIR filters used to compute $X[k]$ ($0 \leq k \leq 9$). Explain how these responses are consistent with the primary function of this filter, namely, to compute different values of $X[k]$.

9.7-1 Derive an FFT-like structure similar to Fig. 9.30 that efficiently computes an 8-point inverse DFT.

9.7-2 Show that a 2-point FFT can be represented using a simple butterfly operation, as in Eq. (9.51).

9.7-3 From Fig. 9.30, we see that $x[6]$ is the fourth input to an 8-point DIT FFT.

- (a) What is the fourth input to a 256-point DIT FFT?
- (b) What is the 253rd input to a 256-point DIT FFT?
- (c) What input position (1st to 256th) should $x[28]$ take on a 256-point DIT FFT?

- (d) What input position (1st to 256th) should $x[103]$ take on a 256-point DIT FFT?

9.7-4 From Fig. 9.31, we see that $X[6]$ is the fourth output of an 8-point DIF FFT.

- (a) What is the fifth output to a 256-point DIF FFT?
- (b) What is the 223rd output to a 256-point DIF FFT?
- (c) What output position (1st to 256th) should $X[12]$ take on a 256-point DIF FFT?
- (d) What output position (1st to 256th) should $X[122]$ take on a 256-point DIF FFT?

9.7-5 Show how to construct a 16-point DIT FFT using the 8-point DIT FFT structure of Fig. 9.30, scalar multipliers, and summing nodes.

9.7-6 Suppose that signal $x[n]$ has finite length equal to 100. Describe in detail how this signal can be analyzed using a computationally efficient FFT structure.

9.7-7 Suppose that we are interested in computing the FFTs $X[k]$ and $Y[k]$ of two N -point real signals $x[n]$ and $y[n]$. By exploiting the FFT's natural ability to accommodate complex signals, this problem demonstrates that both N -point FFTs can be computed using a single N -point FFT. To this end, define the complex signal $v[n] = x[n] + jy[n]$ with FFT $V[k]$.

- (a) Show that

$$X[k] = \frac{1}{2} (V[k] + V^*[-k]),$$

- (b) Show that

$$Y[k] = \frac{1}{2j} (V[k] - V^*[-k]).$$

- (c) Determine the 4-point real signals $x[n]$ and $y[n]$ if

$$V[k] = [5 + 3j, -2 - j, -1 + 3j, 2].$$

9.7-8 Suppose that we are interested in computing the FFT $X[k]$ of a $2N$ -point real signal

$g[n]$. By exploiting the FFT's natural ability to accommodate complex signals, this problem demonstrates that the desired $2N$ -point FFT can be computed from a single N -point FFT. To this end, define the N -point complex signal $v[n] = x[n] + jy[n]$ with FFT $V[k]$, where $x[n] = g[2n]$ and $y[n] = g[2n+1]$. From Prob. 9.7-7, we know that

$$X[k] = \frac{1}{2} (V[k] + V^*[-k]),$$

and

$$Y[k] = \frac{1}{2j} (V[k] - V^*[-k]).$$

- (a) Show that

$$G[k] = X[k] + W_{2N}^{-k} Y[k],$$

$$\text{where } W_{2N} = e^{j2\pi/2N}.$$

- (b) Find the 8-point real signal $g[n]$ if

$$V[k] = [5 + 3j, -2 - j, -1 + 3j, 2].$$

9.7-9 A permutation matrix \mathbf{P} has a single 1 in each row and column with the remaining elements all 0. Permutation matrices are useful for reordering the elements of a vector; the operation $\mathbf{P}\mathbf{x}$ reorders the elements of a column vector \mathbf{x} based on the form of \mathbf{P} .

- (a) Fully describe an N -by- N permutation matrix named \mathbf{P}_r that performs a modulo- N reflection of \mathbf{x} .
- (b) Given DFT matrix \mathbf{D}_N , verify that $(\mathbf{D}_N)(\mathbf{D}_N) = \mathbf{D}_N^2$ produces a scaled permutation matrix. How does $\mathbf{D}_N^2 \mathbf{x}$ reorder the elements of \mathbf{x} ?
- (c) What is the result of $(\mathbf{D}_N^2)(\mathbf{D}_N^2)\mathbf{x} = \mathbf{D}_N^4 \mathbf{x}$?
- (d) Determine an 8-by-8 permutation matrix that bit-reverse orders vector \mathbf{x} for the DIT FFT algorithm of Fig. 9.30.

9.8-1 The DTFS duality property states that if

$$\tilde{x}[n] \longleftrightarrow \mathcal{D}[k],$$

then

$$\mathcal{D}[n] \longleftrightarrow \frac{1}{N} \tilde{x}[-k].$$

- (a) Prove the duality property of the DTFS. To do so, express the DTFS of $\frac{1}{N}\tilde{x}[-n]$, interchange the roles of n and k , and then compare the result to Eq. (9.54).
- (b) Follow a similar approach to part (a) to prove the DFT duality property of Eq. (9.20).

9.8-2 Determine the period N and the discrete-time Fourier series for the signal $\tilde{x}[n] = 2 \cos(0.1\pi n) + 3 \sin(0.3\pi n)$. Sketch both the magnitude and phase spectra $|\mathcal{D}[k]|$ and $\angle \mathcal{D}[k]$.

9.8-3 Determine and sketch the DTFS spectrum $\mathcal{D}[k]$ of an 8-periodic sawtooth signal $\tilde{x}[n]$ that over $0 \leq n \leq 7$ is given by

$$[0, 1, 2, 3, 4, 5, 6, 7].$$

How does the spectrum $\mathcal{D}[k]$ change if $\tilde{x}[n]$ is time reversed?

9.8-4 Derive an FFT-like structure similar to Fig. 9.30 that efficiently computes the 8-point DTFS. Recall that the DTFS computes a time-domain signal $\tilde{x}[n]$ from the spectrum $\mathcal{D}[k]$. Design your structure so that the output $\tilde{x}[n]$ is in normal order.

9.8-5 Create a 64-periodic signal $\tilde{y}[n]$ by negating every other gate of $\tilde{x}[n]$ in Fig. 9.36, beginning with the gate centered at $n = 0$. Determine and sketch the discrete-time Fourier series spectrum $\mathcal{D}[k]$ for $\tilde{y}[n]$.

Appendix A

MATLAB

MATLAB (a registered trademark of MathWorks, Inc.) is a language and interactive environment for numeric computation, algorithm development, data analysis, and data visualization. It is particularly well suited for digital signal processing applications. While MATLAB is relatively simple to use, it is a sophisticated package that can be a bit intimidating to new users. Fortunately, there are many excellent resources on MATLAB and its use, including MATLAB's own built-in documentation. This appendix provides a brief introduction to MATLAB that complements the MATLAB material found throughout this book.

Scripts and Help

When MATLAB is launched, a command window appears. Users issue commands at the command prompt (`>>`), and MATLAB responds, generally with the creation or modification of workspace objects. MATLAB supports different types of workspace objects, such as functions and strings, but usually objects are just data. The workspace window summarizes the names and important characteristics of currently available objects.

While users can directly input sequences of commands at the command prompt, it is generally preferable to instead use a MATLAB script file (M-file). A MATLAB script file is simply a text file (.m extension) that contains a collection of MATLAB statements. Comments are inserted by preceding text with a percent sign (%). Any text editor can be used to create an M-file, but MATLAB's built-in editor provides added functionality such as color-coded text, breakpoints, and various other features. M-files are easy to modify and facilitate rapid algorithm development. M-files are executed directly from MATLAB's editor or by typing the M-file name (without the .m extension) at the command prompt.

MATLAB includes a large number of built-in functions. These functions are easily combined to create new functions, which makes MATLAB a highly extensible language. Many MATLAB functions are named and used in obvious ways. For example, the cosine of a number is computed using the `cos` command. To obtain information about a function and its use, a user can simply type `help` followed by the command name. For example, typing `help cos` provides information on the cosine function, such as the requirement that its argument be in radians. MATLAB help also identifies related functions, which provides users a good way to learn new commands. Again using `help cos` as an example, MATLAB identifies the related functions `acos` (arc-cosine) and `cosd` (cosine with argument in degrees).

MATLAB also has a comprehensive help browser, which is accessed by typing `doc` at the command prompt, pressing the “Help” button in the MATLAB toolbar, or selecting “Product Help” under the “Help” drop-down menu. The help browser includes alphabetized and categorical function lists, full text-searching capabilities, product demos, and tutorial videos. These resources are high quality and make it easy to find information and learn about MATLAB.

Calculator Operations

MATLAB functions well as a scientific calculator. Addition, subtraction, multiplication, division, and exponentiation are performed using the traditional operator symbols `+`, `-`, `*`, `/`, and `^`. To subtract 2 from 5, for example, type `5-2` at the command prompt.[†]

```
01 5-2
ans = 3
```

Since the operation `5-2` in line 01 is not explicitly assigned to a variable, MATLAB assigns the result to a variable named `ans`.

Let us next compute $(8/4)^2$. MATLAB follows standard precedence rules for mathematical operators, so typing `8/4^2` yields an incorrect result of $8/16 = 0.5$ as exponentiation has precedence over division. Ordinary parentheses are used to overcome this problem and establish the desired order of operations.

```
02 (8/4)^2
ans = 4
```

Since the operation is not explicitly assigned to a variable, MATLAB again assigns the result to a variable named `ans`. Consequently, the result of line 01 is overwritten and lost once line 02 executes.

To retain a calculation for later use, we assign the result to some variable: simply precede an expression or number with a variable name and an equal sign. For example, we can assign variable `x` with the result of `5-2` by typing `x = 5-2`.

```
03 x = 5-2
x = 3
```

In similar fashion, we can assign a different variable `y` with the result of $(8/4)^2$.

```
04 y = (8/4)^2
y = 4
```

By utilizing unique variable names, results are individually retained.

MATLAB works as easily with complex numbers as it does with real numbers. MATLAB designates $\sqrt{-1}$ with `1j` or `1i`.[‡] Let us use this notation to construct a complex variable $w = x + jy = 3 + j4$, which corresponds to a 3-4-5 triangle.

```
05 w = x+1j*y
w = 3+4i
```

By default, MATLAB displays complex numbers in rectangular form.

MATLAB provides a variety of built-in calculator-like functions that are useful in working with complex numbers. The real and imaginary parts of a complex number are extracted using the `real` and `imag` commands, the modulus of a number is computed using the `abs` command, and the radian angle of a number is computed using the `angle` command. The argument of a function is passed parenthetically following the function name. For example, we can use `abs(w)` and `angle(w)` to determine the length and angle of w , respectively.

```
06 r = abs(w)
r = 5
07 theta = angle(w)
theta = 0.9273
```

Using these results, we see that $w = 3 + j4$ has polar form $w = re^{j\theta} = 5e^{j0.9273}$. Using MATLAB's `exp` function, we can readily verify that the polar form of w correctly evaluates to the rectangular form of $w = 3 + j4$.

[†]In this book, line numbers are provided for reference and are not part of the MATLAB code.

[‡]MATLAB also predefines variables `j` and `i` as $\sqrt{-1}$. It is best to avoid using `j` or `i` as $\sqrt{-1}$ since these values are easily overwritten. It is preferable to use `1j` or `1i` as these quantities cannot be overwritten and always equal $\sqrt{-1}$.

```
08 r*exp(1j*theta)
ans = 3+4i
```

To convert a radian angle to degrees, the radian angle must be multiplied by $180/\pi$. This is easily accomplished using the variable `pi`, which MATLAB predefines as π .

```
09 theta*180/pi
ans = 53.1301
```

Using this result, we see that we can represent w in phasor notation as $5\angle 53.1301^\circ$.

Like most scientific calculators, MATLAB provides a full selection of trigonometric functions: standard trigonometric functions `cos`, `sin`, `tan`; reciprocal trigonometric functions `sec`, `csc`, `cot`; inverse trigonometric functions `acos`, `asin`, `atan`, `asec`, `acsc`, `acot`; and hyperbolic variations `cosh`, `sinh`, `tanh`, `sech`, `csch`, `coth`, `acosh`, `asinh`, `atanh`, `asech`, `acsch`, `acoth`. As with the `angle` command, MATLAB trigonometric functions utilize units of radians. Unlike many scientific calculators, MATLAB supports complex arguments for any trigonometric function. For example, MATLAB easily confirms that $\cos(j) = \frac{1}{2}(e^{j(j)} + e^{-j(j)}) = \frac{1}{2}(e^{-1} + e^1) = 1.5431$.

```
09 cos(1j)
ans = 1.5431
```

Vectors and Matrices

Although MATLAB operates well as a simple calculator, its real power is in its ability to operate on vector and matrix data. By using vector or matrix arguments, many values are computed using a single expression. Let us begin by discussing the creation and manipulation of vector objects.

A vector of evenly spaced, real elements is obtained using the notation `a:b:c`, where `a` is the initial value, `b` is the step size, and `c` is the termination value. For example, `0:2:11` creates a length-6 row vector of even-valued integers ranging from 0 to 10.

```
01 0:2:11
ans = 0 2 4 6 8 10
```

Notice that the termination value might not appear in the final vector. Non-integer and negative step sizes are also permissible in the `a:b:c` notation. Furthermore, an apostrophe ('), which designates a complex-conjugate transpose operation in MATLAB, provides a convenient way to create a column vector from a row vector and vice versa. For example, `(11:-2.5:0)'` produces a length-5 row vector of values starting at 11 and decreasing by steps of 2.5 until the termination value of 0 is reached.

```
02 (11:-2.5:0)'
ans = 11.0000
8.5000
6.0000
3.5000
1.0000
```

It is often undesirable to display all the elements of a vector to the computer screen. Ending a MATLAB command with a semicolon (;) suppresses the display of data to the screen. For example, suppose that we wish to create a vector of integers between 0 and 99 and then determine the sum of those values.

```
03 k = 0:1:99;
04 sum(k)
ans = 4950
```

Line 03 creates the desired row vector `k` but, since the command is terminated with a semicolon, does not display the resulting 100 values to the screen. Line 04 uses MATLAB's `sum` command to compute the sum of the 100 elements comprising `k`, which evaluates to 4950 in this case. The `a:b:c`

notation can be simplified as `a:c` when the step size is one. Thus, typing `k = 0:99` produces a result that is identical to that of line 03.

To select particular elements within a vector (or matrix), we need to specify the indices of the desired vector elements. In MATLAB, indexing begins at 1 and increments integer-wise to the length of the vector.[†] Various elements of a vector (or matrix) are selected by parenthetically supplying the desired indices to the vector name. For example, we use indices from 1 to 10 to select the first 10 elements of the length-100 vector `k`.

```
05 k(1:10)
ans = 0 1 2 3 4 5 6 7 8 9
```

In line 05, the notation `1:10` produces a length-10 vector ranging from 1 to 10. This vector is passed parenthetically as indices to the variable `k` to select the corresponding elements, which in this case are the integers between 0 and 9 inclusive. In the context of indexing, the `end` command automatically references the final index of a vector and is particularly useful in accessing the last values of a vector of unknown or changing length. To provide an example, let us use the `end` command to extract the final 10 values of `k`.

```
06 k(end-9:end)
ans = 90 91 92 93 94 95 96 97 98 99
```

Many MATLAB functions accept vector arguments, for which they produce a vector of results. This makes it as easy to tell MATLAB to compute one or one million values of some function. For example, suppose that we want to compute $x(t) = \cos(t)$ over $0 \leq t \leq 2\pi$ using a time spacing of $\pi/4$.

```
07 t = 0:pi/4:2*pi;
08 x = cos(t)
x = 1.0000 0.7071 0.0000 -0.7071 -1.0000 -0.7071 -0.0000 0.7071 1.0000
```

Line 07 creates the desired length-9 vector of time values $[0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}, 2\pi]$. Line 08 evaluates the cosine function for each of the nine values of vector `t`. If we want to know the fourth element computed, we simply pass an index of 4 to vector `x`.

```
09 x(4)
ans = -0.7071
```

It is important to notice that `x(4)` in line 09 means the fourth element of vector `x`, not the function $x(t) = \cos(t)$ evaluated at $t = 4$. Parentheses have different meanings based on context: parentheses pass arguments to functions (such as `cos`) and pass indices to variables. If this distinction is forgotten, the results are going to be wrong.

In some cases, we need to create vectors that are not based on a set of equally spaced real numbers (the `a:b:c` notation). MATLAB uses brackets `[]` to concatenate objects, a construct that allows for the creation of arbitrary vectors. For example, suppose that we rolled a six-sided die six times and observed the sequence 3, 5, 1, 3, 6, and then 5. We can use brackets to create a vector `set1` that contains these observations.

```
10 set1 = [3,5,1,3,6,5]
set1 = 3 5 1 3 6 5
```

By separating elements with commas (or just blanks), concatenation occurs horizontally. Alternatively, concatenation occurs vertically when elements are separated with semicolons. Let us construct a column vector that contains the six additional observations of 2, 1, 5, 3, 3, and 1.

```
11 set2 = [2;1;5;3;3;1]
set2 =
1
5
3
3
1
```

[†]Other languages, such as C, begin indexing at 0. Careful attention must be paid to such differences.

A vector constructed using brackets functions the same as one constructed using the `a:b:c` notation. For example, an index of 4 always extracts the fourth element of a vector, regardless of the method of vector construction.

```
12 set1(4)
ans = 3
```

Brackets can also be used to concatenate vector objects. For example, let us combine `set1` and `set2` into a side-by-side pair of column vectors.

```
13 set3 = [set1',set2]
set3 =
 5  1
 1  5
 3  3
 6  3
 5  1
```

In line 13, we use an apostrophe to transpose row vector `set1` into a column vector and then use brackets to horizontally concatenate the result with the vector `set2`. The result is the matrix `set3`. The same result can also be achieved by using brackets and a combination of commas or blanks (to separate row elements) and semicolons (to separate rows).

```
14 set3 = [3,2;5,1;1,5;3,3;6,3;5,1]
set3 =
 5  1
 1  5
 3  3
 6  3
 5  1
```

Matrices are just two-dimensional vectors.[†] By convention, the first dimension of a matrix is the number of rows, and the second dimension is the number of columns. Thus, `set3` is a 6-by-2 matrix since it has six rows and two columns. Indexing matrices is identical to indexing vectors, except that two indices are utilized: one for the row and another for the column.[‡] For example, `set3(5,1)` selects the element in the fifth row and first column of matrix `set3`.

```
15 set3(5,1)
ans = 6
```

By supplying ranges of row and column indices, it is possible to extract multiple elements from a matrix. For example, suppose that we want to extract the upper half of matrix `set3`.

```
16 set3(1:3,:)
ans =
 3  2
 5  1
 1  5
```

In line 15, `1:3` specifies the first three rows be selected, and the isolated colon (`:`) indicates that all columns be selected.

In general, MATLAB assumes linear algebra rules when performing mathematical operations between two or more vectors or matrices. For example, if we multiply the 1-by-6 row vector `set1` and the 6-by-1 column vector `set2`, we obtain the inner product of these two vectors.

```
17 set1*set2
ans = 48
```

[†]Alternately, vectors are one-dimensional matrices. A length- N column vector is just an N -by-1 matrix, whereas a length- M row vector is just a 1-by- M matrix.

[‡]It is also possible to access all elements of a matrix using a single but less intuitive index, a topic we shall not cover. To learn more, consult MATLAB help.

If we multiply the 6-by-1 column vector `set2` and the 1-by-6 row vector `set1`, we obtain the outer product of these two vectors.

```
18 set2*set1
ans = 6 10 2 6 12 10
      3 5 1 3 6 5
     15 25 5 15 30 25
      9 15 3 9 18 15
      9 15 3 9 18 15
      3 5 1 3 6 5
```

The multiplications found in lines 17 and 18 both work since the two vectors are *conformable*, which is to say that the number of columns of the multiplier equals the number of rows of the multiplicand.

One of the most common sources of error in MATLAB programming is trying to mathematically manipulate vector or matrix objects in ways that are not compatible with their dimensions. For example, in algebra, we know that x^2 is just x times x . We might be tempted to square the elements of vector `set1` by multiplying it by itself.

```
19 set1*set1
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```

Since `set1` is not conformable with itself (its number of rows does not equal its number of columns), the requested multiplication makes no sense from a linear algebra perspective, and MATLAB complains with an error. Trying to raise x to a power of 2 does no better.

```
20 set1^2
??? Error using ==> mpower
Inputs must be a scalar and a square matrix.
```

The problem now is that MATLAB is trying to apply rules of matrix exponentiation to a vector, which does not work. The key here is to realize that what is needed is an element-by-element multiplication or exponentiation, not a matrix multiplication or exponentiation.

In element-by-element operations, a desired mathematical operation is performed between corresponding elements of two vectors (or matrices). Sometimes, such as in the case of addition or subtraction, the operation is naturally element by element, and no special action is needed in MATLAB. Other times, such as in the cases of multiplication, division, and exponentiation, MATLAB needs to be instructed that the element-by-element version of the operation is desired. This is readily accomplished by preceding the operator (`*`, `\`, or `^`) with a period (`.*`, `.\`, or `.^`). Element-by-element multiplication or exponentiation is just what is needed to square the elements of `set1`.

```
21 set1.*set1
ans = 9 25 1 9 36 25
22 set1.^2
ans = 9 25 1 9 36 25
```

Even though operations such as addition and subtraction are naturally element by element, this does not mean that errors are not possible. Suppose, for example, that we want to add the observations of `set1` with those of `set2`. Simply trying to add these two sets produces an error.

```
23 set1+set2
??? Error using ==> plus
Matrix dimensions must agree.
```

The problem is that while the two sets have the same number of elements, they do not have the same orientation. MATLAB might guess that an element-by-element operation is desired, but it has no way of knowing whether the result should be represented as a column or row vector. By transposing `set2` into a row vector, it can be added to `set1` to produce a row vector result.

```
24 set1+set2'
ans = 5 6 6 6 9 6
```

The same attention to vector orientation is also needed when using element-by-element multiplication. For example, suppose that we wish to element-by-element multiply `set1` and `set2`.

```
25 set1.*set2
??? Error using ==> times
Matrix dimensions must agree.
```

Here, an error is produced because `set1` is a row vector and `set2` is a column vector. Even though the two vectors have the same number of elements, MATLAB cannot element-by-element multiply the two together unless they also have the same orientation (both column vectors or both row vectors). Following the strategy used in line 24, we can transpose `set2` into a row vector and then perform element-by-element multiplication to obtain a row vector result.

```
26 set1.*set2'
ans = 6 5 5 9 18 5
```

We emphasize these ideas for a simple reason: improper dimension, orientation, or form of operation are among the most common MATLAB programming errors.

Plots

MATLAB makes it simple to visualize data with a wide variety of plotting functions. Here, we discuss the `plot` and `stem` commands, which are the primary commands to graph continuous-time and discrete-time functions, respectively. There are many other plotting functions, however, such as `semilogx`, `semilogy`, and `loglog` for logarithmic plots; `bar` and `pie` for bar and pie charts; `image` to display pictures; `contour` to generate contour plots; `contour3`, `plot3`, `mesh`, and `surf` for visualizing data in three dimensions; and others.

To begin, let us use the `plot` command to graph the continuous-time sinusoid $x(t) = \cos(2\pi t)$ over the interval $-1 \leq t \leq 1$. First, we construct a time vector `t`, and then we use that time vector to create a vector `x`.

```
01 t = (-1:0.001:1);
02 x = cos(2*pi*t);
```

Notice that `t` is a length-2001 vector of time values between -1 and 1 using a step size of 0.001 , and `x` is a length-2001 vector of the function $\cos(2\pi t)$ evaluated at those same time instances. We can plot `x` by simply typing `plot(x)`.

```
03 plot(x);
```

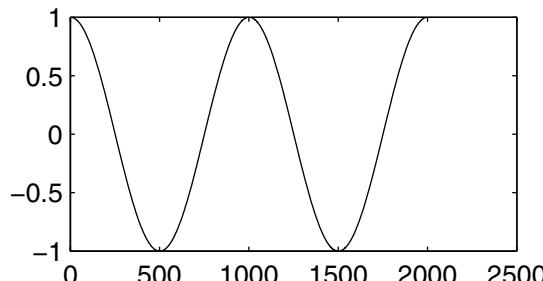


Figure A.1: MATLAB plot of $x(t) = \cos(2\pi t)$ over $-1 \leq t \leq 1$.

The result of line 03, shown in Fig. A.1, is unsatisfactory for a variety of reasons. Since we did not provide the plot command with the time vector t , MATLAB plots x against its index values, which range from 1 to 2001. In this example, MATLAB also extends the horizontal axis to a value of 2500, which is well past the computed values of the function. Thus, the graph's horizontal axis gives the impression that the plot covers times ranging from 0 to 2500 rather than from -1 to 1. The bounding box of the plot crowds the function peaks, a problem that obscures data and makes it difficult to tell if the function ends at index 2000 or remains unity over from 2000 to 2500. Furthermore, the graph lacks axis labels and therefore does not clearly indicate what is actually being plotted.

These deficiencies are easily remedied by passing both vectors t and x to the `plot` command, using the `xlabel` and `ylabel` commands to specify axis labels, and using the `axis` command to specify the lower and upper limits of the horizontal and vertical axes, respectively. The much improved result is shown in Fig. A.2. If desired, grid lines are easily added with the `grid` command.

```
04 plot(t,x);
05 xlabel('t');
06 ylabel('x(t)');
07 axis([-1 1 -1.25 1.25]);
```

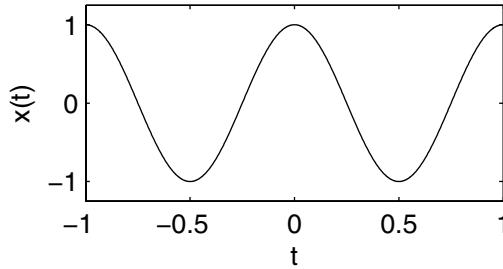


Figure A.2: Improved MATLAB plot of $x(t) = \cos(2\pi t)$ over $-1 \leq t \leq 1$.

Let us next consider the task of comparing our plot of $x(t) = \cos(2\pi t)$ with a plot of $y(t) = \sin(2\pi t)$ taken over the same time range. We can produce a plot of $\sin(2\pi t)$ by changing line 02 to read $y = \sin(2\pi t)$ and making appropriate changes of x to y in lines 04 through 07. Although this approach produces a plot of $\sin(2\pi t)$, it erases the plot of $\cos(2\pi t)$ in the process. Even if steps are taken to preserve the original plot, it is somewhat inconvenient to compare two separate plots in two separate windows. One solution to this dilemma is to use MATLAB's `subplot` function to tile multiple plots on a single figure window. The first argument of the `subplot` function specifies the number of subplot rows, the second argument specifies the number of subplot columns, and the third argument specifies the particular subplot of interest (numbered row-wise). Lines 08 through 18 combine the methods of lines 04 through 07 with the `subplot` command to produce a side-by-side presentation of $x(t) = \cos(2\pi t)$ and $y(t) = \sin(2\pi t)$, the result of which is shown in Fig. A.3.

```
08 subplot(1,2,1);
09 plot(t,x);
10 xlabel('t');
11 ylabel('x(t)');
12 axis([-1 1 -1.25 1.25]);
13 y = sin(2*pi*t);
14 subplot(1,2,2);
15 plot(t,y);
16 xlabel('t');
17 ylabel('y(t)');
18 axis([-1 1 -1.25 1.25]);
```

Although the plots of Fig. A.3 allow a side-by-side comparison of $\cos(2\pi t)$ and $\sin(2\pi t)$, an even more effective presentation method is to graph the two functions on a single plot. MATLAB

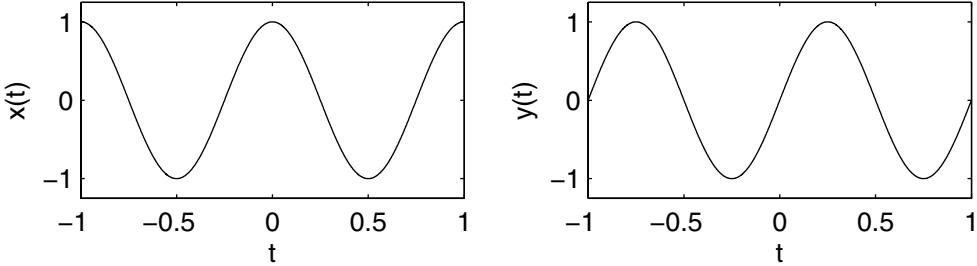


Figure A.3: MATLAB subplots of $x(t) = \cos(2\pi t)$ and $y(t) = \sin(2\pi t)$.

accommodates this common task by allowing data for multiple curves to be passed to a single `plot` command.

```

19 subplot(1,1,1);
20 plot(t,x,'k-',t,y,'k--');
21 xlabel('t');
22 ylabel('Amplitude');
23 axis([-1 1 -1.25 1.25]);
24 legend('x(t)', 'y(t)', 'location', 'EastOutside');

```

The `subplot(1,1,1)` command of line 19 returns the figure window from the previous pair of subplots (lines 08–18) to a single plot configuration.[†] To plot both $x(t)$ and $y(t)$ on the same graph, we type `plot(t,x,t,y)`. The additional arguments `'k-'` and `'k--'` shown in line 20 tell MATLAB to use a black solid line for the first curve $x(t)$ and to use a black dashed line for the second curve $y(t)$. MATLAB’s help resources provide details on these and other plot options. Line 24 uses the `legend` command to identify each curve. The result, shown in Figure A.4, is a quality plot.

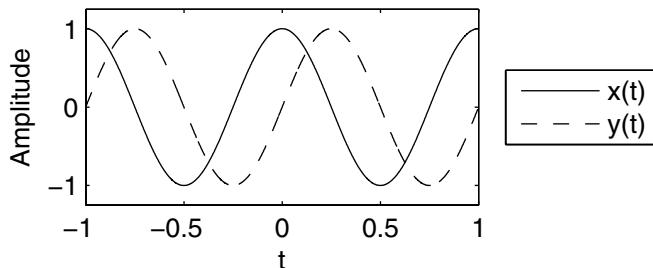


Figure A.4: MATLAB plot of $x(t) = \cos(2\pi t)$ (solid) and $y(t) = \sin(2\pi t)$ (dashed).

To simplify plotting discrete-time data, MATLAB offers the `stem` command, which operates much like the `plot` command, although multiple overlapping curves are not allowed. To provide an example, consider plotting the discrete-time sinusoid $x[n] = \cos(2\pi n/10)$ over the two-period time interval $-10 \leq n < 9$. Aside from using `stem` rather than `plot`, the code is nearly identical to our earlier example of plotting $x(t) = \cos(2\pi t)$. The result is shown in Fig. A.5.

```

25 n = -10:9;
26 x = cos(2*pi*n/10);
27 stem(n,x);
28 xlabel('n');
29 ylabel('x[n]');
30 axis([-10.5 9.5 -1.25 1.25]);

```

[†]A similar result is obtained by typing `clf`, a command that clears the current figure window.

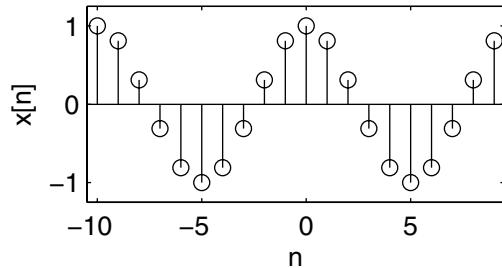


Figure A.5: MATLAB stem plot of $x[n] = \cos(2\pi n/10)$ over $-10 \leq n < 9$.

Relational and Logical Operators

In MATLAB, we can compare objects using any of six available relational operators: $>$, $<$, \geq , \leq , \neq , and \sim . If the comparison is true, a logical 1 is returned. If the comparison is false, a logical 0 is returned. When a relational operator acts on a vector or matrix, comparisons are made element-wise to produce an answer that has the same dimension as the input. Because they indicate when a condition holds true, relational operators are sometimes called *indicator functions*. To provide a simple example, let us construct a length-10 vector \mathbf{x} of integers ranging from 0 to 9 and then use a relational operator to test which elements of \mathbf{x} are greater than 5.

```
01 x = 0:9;
02 x>5
ans = 0 0 0 0 0 0 1 1 1 1
```

Since line 02 tests the length-10 vector \mathbf{x} , the result is a length-10 vector of logical 0s (when \mathbf{x} is not greater than 5) and logical 1s (when \mathbf{x} is greater than 5). As expected, we see that the last four elements of vector \mathbf{x} satisfy the condition that $\mathbf{x}>5$. To provide another example, let us next test which elements of \mathbf{x} are less than or equal to 8.

```
03 x<=8
ans = 1 1 1 1 1 1 1 1 1 0
```

In this case, we see that all but the last element of \mathbf{x} satisfy the condition $\mathbf{x}\leq 8$.

MATLAB also provides logical and, or, and not operators: $\&$, $|$, and \sim , respectively. Like relational operators, logical operators function in an element-wise fashion. In combination with relational operators, logical operators are useful in many tasks. For example, we can determine if $5 < x \leq 8$ by logically anding the result of $\mathbf{x}>5$ and $\mathbf{x}\leq 8$.

```
04 (x>5)&(x<=8)
ans = 0 0 0 0 0 0 1 1 1 0
```

Only three elements of the length-10 vector \mathbf{x} satisfy the condition $5 < x \leq 8$.

One powerful use of logical data is in logical indexing. In logical indexing, a logical vector is parenthetically passed to a variable to select only those elements where the logical vector is true (logical 1). In other words, when a logical data type is passed as an index to a variable, it positionally indicates (by 1s) which elements to select. To provide an example, let us use logical indexing to extract the elements of vector \mathbf{x} that satisfy $5 < x \leq 8$.

```
05 x((x>5)&(x<=8))
ans = 6 7 8
```

In line 05, the combined relational and logical operations of $(\mathbf{x}>5)\&(\mathbf{x}\leq 8)$ produce the length-10 logical result 0 0 0 0 0 0 1 1 1 0. Used as a logical index, this result selects the seventh, eighth, and ninth elements of \mathbf{x} , which have values 6, 7, and 8, respectively. We can obtain the same result through ordinary (linear) indexing by explicitly specifying that we need elements 7, 8, and 9 of vector \mathbf{x} .

```
06 x(7:9)
ans = 6 7 8
```

Logical indexing is often simpler and more intuitive than linear indexing.

It is worth pointing out that logical indexing requires the index to have a logical data type. It is not enough to specify a vector of 1s and 0s; a *logical* vector of 1s and 0s is required. For example, it might appear that `x([0 0 0 0 0 1 1 1 0])` would behave the same as line 05, but it produces an error instead.

```
07 x([0 0 0 0 0 0 1 1 1 0])
??? Subscript indices must either be real positive integers or logicals.
```

Despite appearances, `[0 0 0 0 0 0 1 1 1 0]` is a double, not logical, data type and therefore cannot directly serve in logical indexing. The error in line 07 can be avoided by converting `[0 0 0 0 0 1 1 1 0]` to a logical data type using MATLAB's `logical` command.

```
08 x(logical([0 0 0 0 0 0 1 1 1 0]))
ans = 6 7 8
```

This is not to say that MATLAB's bracket notation cannot be directly used for indexing. Bracket notation is fully capable of indexing as long as the elements concatenated together are valid indices. In fact, brackets allow for very flexible indexing that would otherwise be difficult to achieve, such as reordering or repeating elements. To provide two examples, `x([1:2:end,2:2:end])` reorders `x` so that the even elements precede the odd elements, and `x([5 5 5 4 4])` repeats the fifth element of `x` three times and the fourth element twice.

```
09 x([1:2:end,2:2:end])
ans = 0 2 4 6 8 1 3 5 7 9
10 x([5 5 5 4 4])
ans = 4 4 4 3 3
```

As we shall see in the next section, relational and logical operators are also very useful in constructing functions.

Functions and Program Control

Like most programming languages, MATLAB provides several options to define and evaluate custom functions. Furthermore, MATLAB offers various program control mechanisms such as if statements and loops. These features greatly enhance MATLAB's capabilities. To begin, we introduce anonymous functions. Then we cover several program control statements and briefly introduce script (M-file) functions.

Anonymous function are simple to create. The basic syntax to create an anonymous function is

```
FunctionName = @(argument list) expression
```

To provide an example, let us create an anonymous function `u` to represent the continuous-time unit step function

$$u(t) = \begin{cases} 1 & t > 0 \\ \frac{1}{2} & t = 0 \\ 0 & t < 0 \end{cases} .$$

Argument names can be arbitrarily selected since they are internal (local) to an anonymous function and do not correspond to desktop variables. Still, it is good practice to follow sensible naming conventions. For our unit step function, we name the argument `t` to correspond with time and then use relational operators to represent the piecewise character of the function.

```
01 u = @(t) 1.0*(t>0)+0.5*(t==0);
```

Like other MATLAB functions, anonymous functions are used by parenthetically passing arguments to the function name. For example, we type `u(-1:0.5:1)` to evaluate our unit step function at values $t = -1, -0.5, 0, 0.5$, and 1 .

```
02 u(-1:0.5:1)
ans = 0  0  0.5000  1.0000  1.0000
```

Anonymous functions can also be used in the construction of other anonymous functions. Suppose, for example, that we want to define the function $x(t) = e^{-t}u(t)$. We can create an anonymous function `x` that itself uses the anonymous function `u` previously defined.

```
03 x = @(t) exp(-t).*u(t);
```

Notice in line 03 that element-by-element multiplication is needed to properly multiply `exp(-t)` and `u(t)`. Let us evaluate and plot this function over the interval $-1 \leq t \leq 3$. First, we construct a time vector `t`, and then we plot the function `x` evaluated at these time instances. Figure A.6 shows the result, a causal signal that exponentially decays with time.

```
04 t = -1:0.0001:3;
05 plot(t,x(t))
06 axis([-1 3 -0.25 1.25]);
07 xlabel('t');
08 ylabel('x(t)');
```

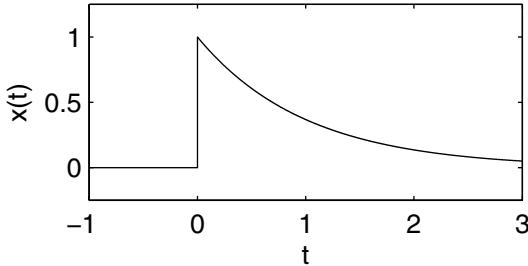


Figure A.6: MATALB plot of $x(t) = e^{-t}u(t)$ over $-1 \leq n \leq 3$.

We can also evaluate a function by passing it an expression. This makes it very simple to evaluate expressions such as $x(1-t)$, for example.

```
09 plot(t,x(1-t))
10 axis([-1 3 -0.25 1.25]);
11 xlabel('t');
12 ylabel('x(1-t)');
```

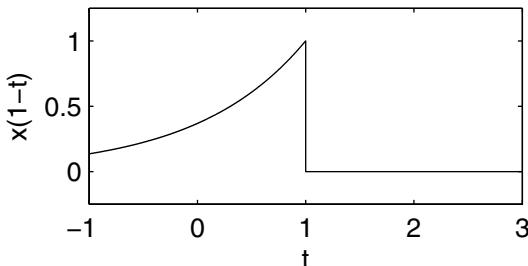


Figure A.7: MATALB plot of $x(1-t)$ over $-1 \leq n \leq 3$.

Comparing Figs. A.6 and A.7, we see that $x(1-t)$ is just a reflected and shifted version of $x(t)$, as expected.

Like anonymous functions, program control statements also enhance MATLAB functionality. Let us briefly consider three important types of program control: conditional execution, for loops, and while loops. MATLAB's `if`, `elseif`, and `else` statements allow conditional execution of code. The syntax is

```
if logicalexpression1
    statements1
elseif logicalexpression2
    statements2
else
    statements3
end
```

The `elseif` and `else` components can be removed or additional `elseif` statements can be added as circumstances warrant.

MATLAB's `for` statement allows repeated execution of a code segment. The syntax is

```
for LoopVariableName = values
    statements
end
```

The statements within a `for` loop execute once for each value taken by the loop variable. Loop variable values are usually specified as a vector using the `a:b:c` notation but can also be formed using brackets or mathematical expressions. For example, `(1:4)`, `[1 1 1 4]`, and `(1:4).^2` are all valid ways to specify four values of a loop variable.

MATLAB's `while` statement, which repeats execution of a code segment until some condition is met, has a syntax that is very similar to the `for` statement.

```
while logicalexpression
    statements
end
```

Of course, MATLAB allows nested `while`, `for`, and `if` statements.

To provide an example of using program control statements, let us write a program that determines the product of all prime numbers between 1 and 20.

```
13 result = 1;
14 for n = 1:20,
15     if isprime(n),
16         result = result*n;
17     end
18 end
19 result
result = 9699690
```

Line 13 initializes variable `result`, and line 14 establishes a `for` loop that sequences the loop variable `n` through the 20 integers ranging from 1 to 20. Line 15 uses an `if` statement and MATLAB's `isprime` command to check whether the loop variable `n` is prime. If the loop variable `n` is prime, line 16 updates the `result` variable. Line 17 concludes the `if` statement, and line 18 concludes the `for` loop. Line 19 provides the somewhat surprisingly large result of 9699690.

Often, loops and other program control statements can be avoided by using clever programming techniques. Consider the following alternate code that also computes the product of all prime numbers between 1 and 20:

```
20 n = 1:20; result = prod(n(isprime(n)))
result = 9699690
```

Here, the `prod` command computes the product of all values of vector `n` that, using logical indexing, are found to be prime. Clearly, line 20 is a more compact way to perform the tasks of lines 13–19. However, the program of lines 13–19 is probably more understandable, particularly to those who are not experts in MATLAB syntax.

Another option to consider in functions is the script, or M-file, function. Unlike anonymous functions, which do not require their own separate files, a script function requires a dedicated M-file. The syntax of an M-file function is

```
function [output list] = FunctionName(input list)
statements
```

The first line of a script function file must begin with the `function` keyword followed by a bracketed list of outputs, an equal sign, and then the function name with a parenthetical list of inputs. The remainder of the M-file contains the statements needed to define the function. The file should be named the same as the function and use a `.m` extension.

To provide an example, let us modify the code of line 20 to create a script function that computes the product of all primes from 1 to a user-defined end value of `N`.

```
function [result] = ProductOfPrimes(N)
n = 1:N; result = prod(n(isprime(n)));
```

Saving these two lines of code as a text file named `ProductOfPrimes.m` creates the desired script function. To test the function, we simply type the function name at the command prompt along with the desired input value. To provide two examples, consider computing the product of primes up to 20 and 100, respectively.

```
21 ProductOfPrimes(20)
ans = 9699690
22 ProductOfPrimes(100)
ans = 2.3056e+036
```

MATLAB Tips

We conclude this introduction to MATLAB with a few useful tips. Some tips involve advanced-level MATLAB commands not covered in this introduction; in these cases, MATLAB help resources should be consulted.

1. Use command-line help for unfamiliar functions. Command-line help provides basic function information and syntax and often concludes with a “See also” section that identifies related MATLAB commands; this is a great way to learn new MATLAB instructions. If the command-line help is insufficient, seek the browser help, which tends to be more complete.
2. Use the up and down arrow keys to speed up your command-line operations. Pressing the up or down arrow keys will scroll through the recent command history, allowing you to easily repeat commands without retyping them. By typing the first few characters of your desired sequence followed by the up or down arrow keys, MATLAB will only display commands in the history that begin with the same characters. Simultaneously press “Control” and “C” to break out of a never-ending program.
3. Remain aware of the difference between matrix and element-wise computations, and use the operation appropriate to your task. Often, matrix operators become element-wise operators when proceeded with a period (`.`). Element-wise operations require objects have the same size (dimensions); for simplicity, use the `(:)` notation to force all vectors into column vectors.

4. Except for the most trivial of computations, use an M-file for your work. M-files allow you to easily debug your program, reproduce your work, and modify programs to accommodate inevitable changes. Include comment lines at the beginning of your code that provide enough description so that you know what the program does. Use code cells, where each cell begins with the `%%` notation, to divide a large M-file into manageable parts.
5. Use anonymous or M-file functions when you need a result but do not really care about intermediate variables. Variables internal to functions are generally considered local and thus never appear on the desktop.
6. Label your figure axes. For most publications, do not use color. Rather, create black-and-white graphics that print clearly using black-and-white printers. When plotting graphics, use an appropriate resolution. If you plot too few points, your graph will be jagged and will not properly reflect the function. If you plot too many points, the graph will become excessively large, which causes excessive file sizes and can dramatically slow down document printing. Use the `print` command to convert figures to your preferred file format, and then embed them at a suitable scale into your report. If you don't need to produce a formal report, consolidate multiple graphs onto a single sheet using subplots.
7. Verify that your MATLAB results make sense. MATLAB has inherent limits in accuracy since it represents numbers with a finite number of bits. Certain operations compound these errors and can result in significant accuracy problems. An analytically correct expression is not always enough; use of numerically robust algorithms can help. Differences of large numbers, ratios of large factorials, rooting of high-order polynomials, and inverses of ill-conditioned matrices are all examples of situations where accuracy may become an issue.
8. Spend time to become proficient with MATLAB Handle Graphics, beginning with the `get` and `set` commands. Handle Graphics allows for the most customizable and professional plots. Perform all plot annotations and modifications using commands in your M-file, as opposed to using pull-down menus and mouse clicks. Only in this way will you be able to reliably, accurately, and quickly reproduce your plots, as well as tweak the look of your plot to meet your requirements. The more complicated the plot, the more important this advice becomes. While they may not look it, nearly every figure in this book has been created using MATLAB and its Handle Graphics commands.
9. Learn how to use MATLAB's low-level I/O commands, such as the `fprintf` command. Although they may be somewhat cumbersome to use, they are extremely powerful. They are effective at formatting data to more readable formats (thereby saving you lots of cutting, pasting, and typing); they are also effective in helping automate the collection and storage of large data sets.
10. When things go wrong (and they will), debug your MATLAB code just as you would any programming language: systematically verify your sequence of code. MATLAB has many useful features to assist with this task, including detailed error reporting, breakpoints, and other debugging features. When possible, try to divide complex code into blocks that can be individually checked.

Appendix B

Useful Tables

This appendix provides a collection of useful tables for quick reference. While most of these tables are found in identical form elsewhere in the book, a few are formed from multiple tables or are altogether new. The tables included in this appendix are summarized as follows:

1. Bilateral and unilateral Laplace transform properties (identical to Table 1.6 on page 72).
2. Bilateral and unilateral z -transform properties (identical to Table 7.2 on page 434).
3. A selection of useful sums (identical to Table 5.1 on page 290).
4. Fourier analysis and synthesis equations (new table).
5. Fundamental Fourier properties (combination of Table 1.2 on page 66 and Table 1.4 on page 68).
6. Fourier transform and Fourier series properties (identical to Table 1.3 on page 67).
7. Discrete-time Fourier transform and discrete Fourier transform properties (combination of Table 6.3 on page 356 and Table 9.1 on page 583).

Bilateral and Unilateral Laplace Transform Properties

Bilateral Laplace Transform	Unilateral Laplace Transform
<p>Synthesis: $x(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} X(s)e^{st} ds$</p> <p>Analysis: $X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt$, ROC: R_x</p> <p>Linearity: $ax(t) + by(t) \xrightleftharpoons{\mathcal{L}} aX(s) + bY(s)$, ROC: At least $R_x \cap R_y$</p> <p>Complex Conjugation: $x^*(t) \xrightleftharpoons{\mathcal{L}} X^*(s^*)$, ROC: R_x</p> <p>Scaling and Reversal: $x(at) \xrightleftharpoons{\mathcal{L}} \frac{1}{ a } X\left(\frac{s}{a}\right)$, ROC: R_x scaled by $1/a$ $x(-t) \xrightleftharpoons{\mathcal{L}} X(-s)$, ROC: R_x reflected</p> <p>Shifting: $x(t - t_0) \xrightleftharpoons{\mathcal{L}} X(s)e^{-st_0}$, ROC: R_x $x(t)e^{s_0 t} \xrightleftharpoons{\mathcal{L}} X(s - s_0)$, ROC: R_x shifted by $\text{Re}\{s_0\}$</p> <p>Differentiation: $\frac{d}{dt}x(t) \xrightleftharpoons{\mathcal{L}} sX(s)$, ROC: At least R_x $-tx(t) \xrightleftharpoons{\mathcal{L}} \frac{d}{ds}X(s)$, ROC: R_x</p> <p>Time Integration: $\int_{-\infty}^t x(\tau)d\tau \xrightleftharpoons{\mathcal{L}} \frac{1}{s}X(s)$, ROC: At least $R_x \cap (\text{Re}\{s\} > 0)$</p> <p>Convolution: $x(t) * y(t) \xrightleftharpoons{\mathcal{L}} X(s)Y(s)$, ROC: At least $R_x \cap R_y$</p>	<p>Synthesis: $x(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} X(s)e^{st} ds$</p> <p>Analysis: $X(s) = \int_{0-}^{\infty} x(t)e^{-st} dt$</p> <p>Linearity: $ax(t) + by(t) \xrightleftharpoons{\mathcal{L}_u} aX(s) + bY(s)$</p> <p>Complex Conjugation: $x^*(t) \xrightleftharpoons{\mathcal{L}_u} X^*(s^*)$</p> <p>Scaling and Reversal: If $a > 0$: $x(at) \xrightleftharpoons{\mathcal{L}_u} \frac{1}{a}X\left(\frac{s}{a}\right)$</p> <p>Shifting: If $t_0 > 0$: $x(t - t_0) \xrightleftharpoons{\mathcal{L}_u} X(s)e^{-st_0}$ $x(t)e^{s_0 t} \xrightleftharpoons{\mathcal{L}_u} X(s - s_0)$</p> <p>Differentiation: $\frac{d}{dt}x(t) \xrightleftharpoons{\mathcal{L}_u} sX(s) - x(0^-)$ (general case shown below) $-tx(t) \xrightleftharpoons{\mathcal{L}_u} \frac{d}{ds}X(s)$</p> <p>Time Integration: $\int_{0-}^t x(\tau)d\tau \xrightleftharpoons{\mathcal{L}_u} \frac{1}{s}X(s)$</p> <p>Convolution: $x(t) * y(t) \xrightleftharpoons{\mathcal{L}_u} X(s)Y(s)$</p>
Unilateral Laplace Transform Time Differentiation, General Case $x^{(k)}(t) = \frac{d^k}{dt^k}x(t) \xrightleftharpoons{\mathcal{L}_u} s^k X(s) - \sum_{i=0}^{k-1} s^{k-1-i} x^{(i)}(0^-)$	

Bilateral and Unilateral z -Transform Properties

Bilateral z -Transform	Unilateral z -Transform
<p>Synthesis: $x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz$</p> <p>Analysis: $X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$, ROC: R_x</p> <p>Linearity: $ax[n] + by[n] \xrightleftharpoons{Z} aX(z) + bY(z)$, ROC: At least $R_x \cap R_y$</p> <p>Complex Conjugation: $x^*[n] \xrightleftharpoons{Z} X^*(z^*)$, ROC: R_x</p> <p>Time Reversal: $x[-n] \xrightleftharpoons{Z} X(1/z)$, ROC: $1/R_x$</p> <p>Time Shifting: $x[n-m] \xrightleftharpoons{Z} z^{-m} X(z)$, ROC: Almost R_x</p> <p>z-Domain Scaling: $\gamma^n x[n] \xrightleftharpoons{Z} X(z/\gamma)$, ROC: γR_x</p> <p>z-Domain Differentiation: $nx[n] \xrightleftharpoons{Z} -z \frac{d}{dz} X(z)$, ROC: R_x</p> <p>Time Convolution: $x[n] * y[n] \xrightleftharpoons{Z} X(z)Y(z)$, ROC: At least $R_x \cap R_y$</p>	<p>Synthesis: $x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz$</p> <p>Analysis: $X(z) = \sum_{n=0}^{\infty} x[n] z^{-n}$</p> <p>Linearity: $ax[n] + by[n] \xrightleftharpoons{Z_u} aX(z) + bY(z)$</p> <p>Complex Conjugation: $x^*[n] \xrightleftharpoons{Z_u} X^*(z^*)$</p> <p>Time Reversal:</p> <p>Time Shifting: If $m > 0$: $x[n-m]u[n-m] \xrightleftharpoons{Z_u} z^{-m} X(z)$ (general case given below)</p> <p>z-Domain Scaling: $\gamma^n x[n] \xrightleftharpoons{Z_u} X(z/\gamma)$</p> <p>$z$-Domain Differentiation: $nx[n] \xrightleftharpoons{Z_u} -z \frac{d}{dz} X(z)$</p> <p>Time Convolution: $x[n] * y[n] \xrightleftharpoons{Z_u} X(z)Y(z)$</p>

Unilateral z -Transform Time Shifting, General Case

$$\text{If } m > 0: x[n-m]u[n] \xrightleftharpoons{Z_u} z^{-m} X(z) + z^{-m} \sum_{n=1}^m x[-n]z^n$$

$$\text{If } m < 0: x[n-m]u[n] \xrightleftharpoons{Z_u} z^{-m} X(z) - z^{-m} \sum_{n=0}^{-m-1} x[n]z^{-n}$$

A Selection of Useful Sums

- | | | |
|----|--|------------|
| 1. | $\sum_{m=p}^n r^m = \frac{r^p - r^{n+1}}{1-r}$ | $r \neq 1$ |
| 2. | $\sum_{m=0}^n m = \frac{n(n+1)}{2}$ | |
| 3. | $\sum_{m=0}^n m^2 = \frac{n(n+1)(2n+1)}{6}$ | |
| 4. | $\sum_{m=0}^n mr^m = \frac{r + [n(r-1)-1]r^{n+1}}{(r-1)^2}$ | $r \neq 1$ |
| 5. | $\sum_{m=0}^n m^2 r^m = \frac{r[(1+r)(1-r^n) - 2n(1-r)r^n - n^2(1-r)^2 r^n]}{(r-1)^3}$ | $r \neq 1$ |

Fourier Analysis and Synthesis Equations

Signal x is...	Aperiodic	Periodic
Continuous	FT (Analysis): $X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt$ IFT (Synthesis): $x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{j\omega t} d\omega$	FS (Analysis): $X_k = \frac{1}{T_0} \int_{T_0} x(t)e^{-jk\omega_0 t} dt$ FS (Synthesis): $x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}$
Discrete	DTFT (Analysis): $X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$ IDTFT (Synthesis): $x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega)e^{j\Omega n} d\Omega$	DFT (Analysis): $X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn},$ where $k = \{0 : N-1\}$ and $\Omega_0 = \frac{2\pi}{N}$ IDFT (Synthesis): $x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]e^{j\Omega_0 kn},$ where $n = \{0 : N-1\}$ and $\Omega_0 = \frac{2\pi}{N}$

While the discrete-time Fourier series (DTFS) and the discrete Fourier transform (DFT) both use $\Omega_0 = \frac{2\pi}{N}$, the DTFS differs from the DFT by a scale factor $\frac{1}{N}$ in the analysis equation and a scale factor N in the synthesis equation. The DTFS analysis and synthesis equations are thus

$$X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn} \quad \text{and} \quad x[n] = \sum_{k=0}^{N-1} X[k]e^{j\Omega_0 kn}.$$

Fundamental Fourier Properties

Time (or Frequency)	\longleftrightarrow	Frequency (or Time)
Real	\longleftrightarrow	Conjugate symmetric
Imaginary	\longleftrightarrow	Conjugate antisymmetric
Even	\longleftrightarrow	Even
Odd	\longleftrightarrow	Odd
Continuous	\longleftrightarrow	Aperiodic
Discrete	\longleftrightarrow	Periodic

Fourier Transform and Fourier Series Properties

Fourier Transform	Fourier Series
<p>Synthesis: $x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega$</p> <p>Analysis: $X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt$</p> <p>Duality: if $x(t) \iff X(\omega)$, then $X(t) \iff 2\pi x(-\omega)$</p> <p>Linearity: $ax(t) + by(t) \iff aX(\omega) + bY(\omega)$</p> <p>Complex Conjugation: $x^*(t) \iff X^*(-\omega)$</p> <p>Scaling and Reversal: $x(at) \iff \frac{1}{ a } X\left(\frac{\omega}{a}\right)$ $x(-t) \iff X(-\omega)$</p> <p>Shifting: $x(t - t_0) \iff X(\omega) e^{-j\omega t_0}$ $x(t)e^{j\omega_0 t} \iff X(\omega - \omega_0)$</p> <p>Differentiation: $\frac{d}{dt}x(t) \iff j\omega X(\omega)$ $-jtx(t) \iff \frac{d}{d\omega}X(\omega)$</p> <p>Time Integration: $\int_{-\infty}^t x(\tau)d\tau \iff \frac{X(\omega)}{j\omega} + \pi X(0)\delta(\omega)$</p> <p>Convolution: $x(t) * y(t) \iff X(\omega)Y(\omega)$ $x(t)y(t) \iff \frac{1}{2\pi} X(\omega) * Y(\omega)$</p> <p>Correlation: $\rho_{x,y}(\tau) = x(\tau) * y^*(-\tau) \iff X(\omega)Y^*(\omega)$</p> <p>Parseval's: $E_x = \int_{-\infty}^{\infty} x(t) ^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) ^2 d\omega$</p>	<p>Synthesis: $x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}$</p> <p>Analysis: $X_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk\omega_0 t} dt$</p> <p>Duality:</p> <p>Linearity: $ax(t) + by(t) \iff aX_k + bY_k$</p> <p>Complex Conjugation: $x^*(t) \iff X_{-k}^*$</p> <p>Scaling and Reversal: $x(-t) \iff X_{-k}$</p> <p>Shifting: $x(t - t_0) \iff X_k e^{-jk\omega_0 t_0}$ $x(t)e^{jk_0\omega_0 t} \iff X_{k-k_0}$</p> <p>Differentiation: $\frac{d}{dt}x(t) \iff jk\omega_0 X_k$</p> <p>Time Integration:</p> <p>Convolution: $\frac{1}{T_0} x(t) \circledast y(t) \iff X_k Y_k$ $x(t)y(t) \iff X_k * Y_k$</p> <p>Correlation: $\rho_{x,y}(\tau) = \frac{1}{T_0} x(\tau) \circledast y^*(-\tau) \iff X_k Y_k^*$</p> <p>Parseval's: $P_x = \frac{1}{T_0} \int_{T_0} x(t) ^2 dt = \sum_{k=-\infty}^{\infty} X_k ^2$</p>

Discrete-Time Fourier Transform and Discrete Fourier Transform Properties

Discrete-Time Fourier Transform	Discrete Fourier Transform
<p>Synthesis: $x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega$</p> <p>Analysis: $X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n}$</p> <p>Duality:</p> <p>Linearity: $ax[n] + by[n] \iff aX(\Omega) + bY(\Omega)$</p> <p>Complex Conjugation: $x^*[n] \iff X^*(-\Omega)$ $x^*[-n] \iff X^*(\Omega)$</p> <p>Scaling and Reversal: (see Sec. 6.6) $x[-n] \iff X(-\Omega)$</p> <p>Shifting: $x[n - m] \iff X(\Omega) e^{-j\Omega m}$ $x[n] e^{j\Omega_0 n} \iff X(\Omega - \Omega_0)$</p> <p>Differentiation: $-jnx[n] \iff \frac{d}{d\Omega} X(\Omega)$</p> <p>Convolution: $x[n] * y[n] \iff X(\Omega)Y(\Omega)$ $x[n]y[n] \iff \frac{1}{2\pi} X(\Omega) \circledast Y(\Omega)$</p> <p>Correlation: $\rho_{x,y}[l] = x[l] * y^*[-l] \iff X(\Omega)Y^*(\Omega)$</p> <p>Parseval's: $E_x = \sum_{n=-\infty}^{\infty} x[n] ^2 = \frac{1}{2\pi} \int_{2\pi} X(\Omega) ^2 d\Omega$</p>	<p>Synthesis: $x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\Omega_0 kn}, \quad \Omega_0 = \frac{2\pi}{N}$</p> <p>Analysis: $X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\Omega_0 kn}, \quad \Omega_0 = \frac{2\pi}{N}$</p> <p>Duality: if $x[n] \iff X[k]$, then $X[n] \iff Nx[\langle -k \rangle_N]$</p> <p>Linearity: $ax[n] + by[n] \iff aX[k] + bY[k]$</p> <p>Complex Conjugation: $x^*[n] \iff X^*[\langle -k \rangle_N]$ $x^*[\langle -n \rangle_N] \iff X^*[k]$</p> <p>Scaling and Reversal: $x[\langle -n \rangle_N] \iff X[\langle -k \rangle_N]$</p> <p>Shifting: $x[\langle n - m \rangle_N] \iff X[k] e^{-j\Omega_0 km}, \quad \Omega_0 = \frac{2\pi}{N}$ $x[n] e^{j\Omega_0 mn} \iff X[\langle k - m \rangle_N], \quad \Omega_0 = \frac{2\pi}{N}$</p> <p>Differentiation:</p> <p>Convolution: $x[n] \circledast y[n] \iff X[k]Y[k]$ $x[n]y[n] \iff \frac{1}{N} X[k] \circledast Y[k]$</p> <p>Correlation: $\rho_{x,y}[l] = x[l] \circledast y^*[\langle -l \rangle_N] \iff X[k]Y^*[k]$</p> <p>Parseval's: $E_x = \sum_{n=0}^{N-1} x[n] ^2 = \frac{1}{N} \sum_{k=0}^{N-1} X[k] ^2$</p>

Appendix C

Drill Solutions

Chapter 1

Drill 1.1 (CT Time Scaling)

Let $x(t) = \cos(\omega_0 t + \phi)$. Compressing $x(t)$ by $a > 1$ yields

$$x(at) = \cos(\omega_0(at) + \phi) = \cos(a\omega_0 t + \phi).$$

This sinusoid has frequency $a\omega_0$, an a -fold increase, and maintains the original phase ϕ .

Expanding $x(t)$ by $a > 1$ yields

$$x(t/a) = \cos(\omega_0(t/a) + \phi) = \cos\left(\frac{\omega_0}{a}t + \phi\right).$$

Here, the frequency is decreased a -fold to $\frac{\omega_0}{a}$, and the original phase ϕ remains unchanged.

These conclusions are verified in Fig. D1.1, which plots a sinusoid $x(t) = \sin(2t)$, its compressed version $x(3t)$, and an expanded version $x(t/2)$. Clearly, $x(3t)$ has three times the frequency of $x(t)$, and $x(t/2)$ has half the frequency of $x(t)$.

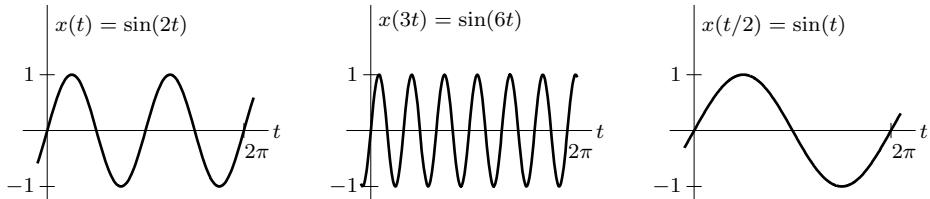


Figure D1.1

Drill 1.2 (Combined CT Operations)

Noting that $x(-3t - 4) = x(-3(t - \frac{-4}{3}))$, we see that this signal is a reflected, compressed (by 3), and shifted (by $-\frac{4}{3}$) version of $x(t)$, as shown in Fig. D1.2. We know that $x(-3t - 4)$ achieves a maximum when $-3t - 4 = T_1$ or at $t = \frac{T_1 + 4}{-3}$, as confirmed in Fig. D1.2.

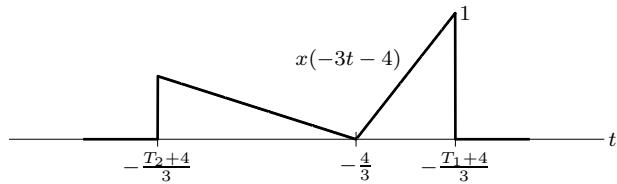


Figure D1.2

Drill 1.3 (CT Unit Gate Representations)

Using Eq. (1.2), we see that $\Pi(t) = u(t + b) - u(t - b)$ for $b = 1/2$. It is also possible to represent $\Pi(t)$ using reflected and time-shifted unit step functions as

$$\Pi(t) = u(-t + 1/2) - u(-t - 1/2).$$

Drill 1.4 (CT Unit Impulse Properties)

Using Eq. (1.5):

$$(a) (t^3 + 2t^2 + 3t + 4)\delta(t) = (t^3 + 2t^2 + 3t + 4)|_{t=0} \delta(t) = 4\delta(t)$$

$$(b) \delta(t) \sin(t^2 - \frac{\pi}{2}) = \sin(t^2 - \frac{\pi}{2})|_{t=0} \delta(t) = \sin(-\pi/2)\delta(t) = -\delta(t)$$

$$(c) e^{-2t}\delta(t + 1) = e^{-2t}|_{t=-1} \delta(t + 1) = e^2\delta(t + 1)$$

Using Eq. (1.6):

$$(d) \int_{-\infty}^{\infty} \delta(\tau) e^{-j\omega\tau} d\tau = e^{-j\omega\tau}|_{\tau=0} \int_{-\infty}^{\infty} \delta(\tau) d\tau = 1(1) = 1$$

$$(e) \int_{-\infty}^{\infty} \delta(\tau - 2) \cos(\frac{\pi\tau}{4}) d\tau = \cos(\frac{\pi\tau}{4})|_{\tau=2} \int_{-\infty}^{\infty} \delta(\tau - 2) d\tau = 0(1) = 0$$

$$(f) \int_{-\infty}^{\infty} e^{-2(t-\tau)}\delta(2 - \tau) d\tau = e^{-2(t-\tau)}|_{\tau=2} \int_{-\infty}^{\infty} \delta(2 - \tau) d\tau = e^{-2(t-2)}(1) = e^{-2(t-2)}$$

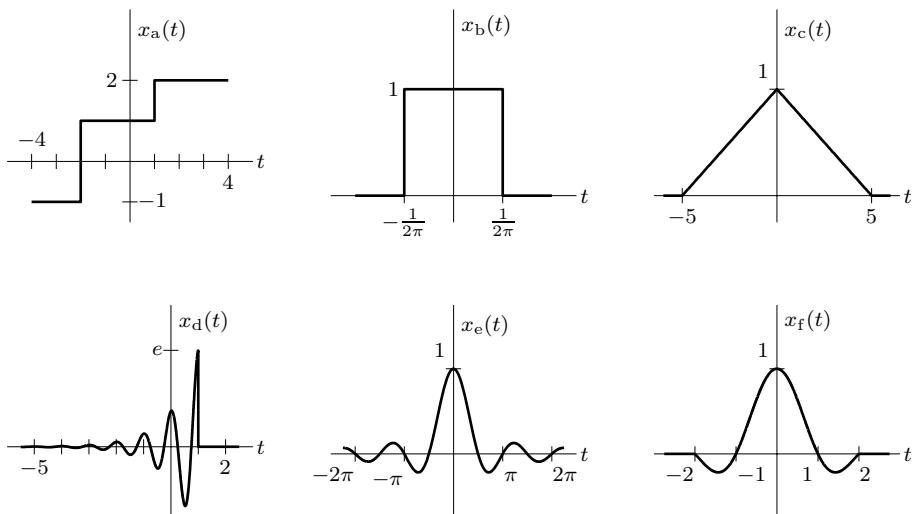
Drill 1.5 (Plotting CT Signal Models)

Figure D1.5

Drill 1.6 (Variations of Euler's Formula)

(a) Using Euler's formula and Eq. (1.18),

$$\cos(t) = \operatorname{Re} \{ \cos(t) + j \sin(t) \} = \operatorname{Re} \{ e^{jt} \} = \frac{e^{jt} + (e^{jt})^*}{2} = \frac{e^{jt} + e^{-jt}}{2}.$$

(b) Using Euler's formula and Eq. (1.19),

$$\sin(t) = \operatorname{Im} \{ \cos(t) + j \sin(t) \} = \operatorname{Im} \{ e^{jt} \} = \frac{e^{jt} - (e^{jt})^*}{2j} = \frac{e^{jt} - e^{-jt}}{2j}.$$

Drill 1.7 (The Imaginary Part Is Real)

Equation (1.19) yields $\operatorname{Im} \{ 1+j \} = \frac{1+j-(1+j)^*}{2j} = \frac{1+j-1-j}{2j} = 1$ and $\operatorname{Im} \{ j \} = \frac{j-j^*}{2j} = \frac{j+j}{2j} = 1$.

Drill 1.8 (Even and Odd Decompositions)

Applying Eq. (1.23) to $x(t) = e^{j\omega t}$ yields

$$x_e(t) = \frac{x(t) + x(-t)}{2} = \frac{e^{j\omega t} + e^{-j\omega t}}{2} = \cos(\omega t).$$

Similarly, applying Eq. (1.24) to $x(t) = e^{j\omega t}$ yields

$$x_o(t) = \frac{x(t) - x(-t)}{2} = \frac{e^{j\omega t} - e^{-j\omega t}}{2} = j \sin(\omega t).$$

The final step in both cases follows from Euler's formula (see Drill 1.6).

Drill 1.9 (Conjugate-Symmetric and Conjugate-Antisymmetric Decompositions)

Using Eqs. (1.28) and (1.29):

(a)

$$x_{a,cs}(t) = \frac{x_a(t) + x_a^*(-t)}{2} = \frac{e^{jt} + e^{-j(-t)}}{2} = e^{jt}$$

$$x_{a,ca}(t) = \frac{x_a(t) - x_a^*(-t)}{2} = \frac{e^{jt} - e^{-j(-t)}}{2} = 0$$

(b)

$$x_{b,cs}(t) = \frac{x_b(t) + x_b^*(-t)}{2} = \frac{je^{jt} + (-j)e^{-j(-t)}}{2} = 0$$

$$x_{b,ca}(t) = \frac{x_b(t) - x_b^*(-t)}{2} = \frac{je^{jt} - (-j)e^{-j(-t)}}{2} = je^{jt}$$

(c) Recognizing $x_c(t) = \sqrt{2}e^{j(t+\pi/4)} = (1+j)e^{jt}$ and using the previous two parts yield

$$x_{c,cs}(t) = e^{jt} \text{ and } x_{c,ca}(t) = je^{jt}.$$

Drill 1.10 (Signal Energy)

Figure D1.10 illustrates the signal $x(t) = \sin(2\pi t)\Pi(t-1/2)$. Energy is determined using Eq. (1.32):

$$E_x = \int_0^1 \sin^2(2\pi t) dt = \frac{1}{4\pi} [2\pi t + \sin(2\pi t) \cos(2\pi t)]|_{t=0}^1 = \frac{1}{4\pi} (2\pi) = \frac{1}{2}.$$

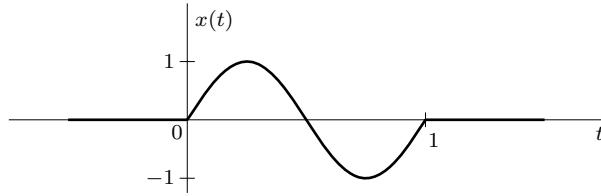


Figure D1.10

Drill 1.11 (Signal Power)

Using Eq. (1.33):

(a) $P_{x_a} = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |C|^2 dt = \lim_{T \rightarrow \infty} \frac{1}{T} |C|^2 \left(t \Big|_{t=-T/2}^{T/2} \right) = |C|^2$

(b) $P_{x_b} = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |u(t)|^2 dt = \lim_{T \rightarrow \infty} \frac{1}{T} \int_0^{T/2} 1 dt = \lim_{T \rightarrow \infty} \frac{1}{T} \left(t \Big|_{t=0}^{T/2} \right) = \frac{1}{2}$

Using Eq. (1.34) and $T_0 = \frac{2\pi}{\omega_0}$:

(c) $P_{x_c} = \frac{1}{T_0} \int_{T_0} |C e^{j\omega_0 t}|^2 dt = \frac{1}{T_0} \int_{T_0} |C|^2 |e^{j\omega_0 t}|^2 dt = \frac{1}{T_0} \int_{T_0} |C|^2 dt = |C|^2$

(d) $P_{x_d} = \frac{1}{T_0} \int_{T_0} |C \cos(\omega_0 t + \theta)|^2 dt = \frac{1}{T_0} \int_{T_0} |C \cos(\omega_0 t)|^2 dt = \frac{|C|^2}{T_0} \int_{T_0} \frac{e^{j2\omega_0 t} + 2 + e^{-j2\omega_0 t}}{4} dt = \frac{|C|^2}{T_0} \frac{0+2T_0+0}{4} = \frac{|C|^2}{2}.$

(e) For the last part, we begin with Eq. (1.33) and later use Eq. (1.34) as needed:

$$\begin{aligned}
 P_{x_e} &= \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |C_1 \cos(\omega_1 t + \theta_1) + C_2 \cos(\omega_2 t + \theta_2)|^2 dt \\
 &= \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[|C_1 \cos(\omega_1 t + \theta_1)|^2 + \right. \\
 &\quad \left. \frac{C_1 C_2^*}{2} [\cos(\omega_1 t + \theta_1 + \omega_2 t + \theta_2) + \cos(\omega_1 t + \theta_1 - \omega_2 t - \theta_2)] + \right. \\
 &\quad \left. \frac{C_1^* C_2}{2} [\cos(\omega_1 t + \theta_1 + \omega_2 t + \theta_2) + \cos(\omega_1 t + \theta_1 - \omega_2 t - \theta_2)] + \right. \\
 &\quad \left. |C_2 \cos(\omega_2 t + \theta_2)|^2 \right] dt
 \end{aligned}$$

When $0 \neq \omega_1 \neq \omega_2 \neq 0$, the middle two terms of the integral are simple sinusoids, which integrate to zero. Using the results of part (d), the final result simplifies to

$$P_{x_e} = \frac{|C_1|^2 + |C_2|^2}{2} \text{ for } 0 \neq \omega_1 \neq \omega_2 \neq 0.$$

When $\omega_1 = \omega_2 \neq 0$, the integral simplifies to

$$\begin{aligned}
 P_{x_e} &= \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[|C_1 \cos(\omega_1 t + \theta_1)|^2 + \right. \\
 &\quad \left. \frac{C_1 C_2^*}{2} [\cos(2\omega_1 t + \theta_1 + \theta_2) + \cos(\theta_1 - \theta_2)] + \right. \\
 &\quad \left. \frac{C_1^* C_2}{2} [\cos(2\omega_1 t + \theta_1 + \theta_2) + \cos(\theta_1 - \theta_2)] + \right. \\
 &\quad \left. |C_2 \cos(\omega_2 t + \theta_2)|^2 \right] dt.
 \end{aligned}$$

The $\cos(2\omega_1 t + \theta_1 + \theta_2)$ terms, being simple sinusoids, integrate to zero. This leaves

$$P_{x_e} = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[|C_1 \cos(\omega_1 t + \theta_1)|^2 + \frac{C_1 C_2^* + C_1^* C_2}{2} \cos(\theta_1 - \theta_2) + |C_2 \cos(\omega_2 t + \theta_2)|^2 \right] dt.$$

Simplifying with the results from part (d) yields

$$P_{x_e} = \frac{|C_1|^2 + (C_1 C_2^* + C_1^* C_2) \cos(\theta_1 - \theta_2) + |C_2|^2}{2} \text{ for } \omega_1 = \omega_2 \neq 0.$$

Drill 1.12 (Neither Energy nor Power)

$$P_x = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} |e^{-at}|^2 dt = \lim_{T \rightarrow \infty} \frac{1}{T} \frac{e^{-2at}}{-2a} \Big|_{t=-T/2}^{T/2} = \lim_{T \rightarrow \infty} \frac{1}{T} \frac{e^{-aT} - e^{aT}}{-2a}.$$

This limit is ∞/∞ indeterminate. Using L'Hôpital's rule, we obtain

$$P_x = \lim_{T \rightarrow \infty} \frac{e^{-aT} + a^{aT}}{2} = \lim_{T \rightarrow \infty} \cosh(aT).$$

Whether $a < 0$ or $a > 0$, $\lim_{T \rightarrow \infty} \cosh(aT) = \infty$. That is,

$$P_x = \infty \text{ for either } a < 0 \text{ or } a > 0.$$

Since $P_x = \infty$, $E_x = \infty$, and $x(t)$ is neither a power nor an energy signal. One can also reach this conclusions by realizing that regardless whether $a < 0$ or $a > 0$, $x(t)$ grows without bound in one direction as $|t| \rightarrow \infty$.

Drill 1.13 (Additivity Does Not Imply Homogeneity)

The superposition property of Eq. (1.35) requires that

$$H \left\{ \sum_{k=1}^2 c_k x_k(t) \right\} = \sum_{k=1}^2 c_k y_k(t).$$

For our system, $y(t) = H \{x(t)\} = \operatorname{Re} \{x(t)\}$, so

$$\begin{aligned} H \left\{ \sum_{k=1}^2 c_k x_k(t) \right\} &= \operatorname{Re} \left\{ \sum_{k=1}^2 c_k x_k(t) \right\} \\ &= \sum_{k=1}^2 \operatorname{Re} \{c_k x_k(t)\} \\ &= \sum_{k=1}^2 \operatorname{Re} \{c_k\} \operatorname{Re} \{x_k(t)\} - \operatorname{Im} \{c_k\} \operatorname{Im} \{x_k(t)\} \\ &= \sum_{k=1}^2 \operatorname{Re} \{c_k\} y_k(t) - \operatorname{Im} \{c_k\} \operatorname{Im} \{x_k(t)\} \\ &\neq \sum_{k=1}^2 c_k y_k(t). \end{aligned}$$

Since the superposition property does not hold, the system $y(t) = \operatorname{Re}\{x(t)\}$ is not linear.

Drill 1.14 (Testing LTIC Systems for BIBO Stability)

Eq. (1.41) states that the output of an LTIC system is given by

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau) d\tau.$$

Taking the absolute value of each side yields

$$|y(t)| = \left| \int_{-\infty}^{\infty} x(\tau)h(t - \tau) d\tau \right| < \int_{-\infty}^{\infty} |x(\tau)h(t - \tau)| d\tau.$$

Using Eq. (1.44), we can establish BIBO stability if $|y(t)|$ is finite for all bounded input $|x(t)| \leq K_x < \infty$:

$$|y(t)| < \int_{-\infty}^{\infty} |x(\tau)| |h(t - \tau)| d\tau < K_x \int_{-\infty}^{\infty} |h(t - \tau)| d\tau < \infty.$$

Dividing the right-hand inequality by K_x and using a change of variable yield the desired test for BIBO stability

$$\int_{-\infty}^{\infty} |h(\tau)| d\tau \leq K_h < \infty.$$

In other words, an LTIC system is BIBO stable if its impulse response is absolutely integrable.

Drill 1.15 (Computing the Transfer Functions of Common Systems)

- (a) An ideal delay of T has impulse response $h(t) = \delta(t - T)$. System behavior is easily verified using Eqs. (1.41) and (1.9):

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau) d\tau = \int_{-\infty}^{\infty} x(\tau)\delta(t - T - \tau) d\tau = x(t - T).$$

The corresponding system transfer function is directly computed using Eq. (1.46) as

$$H(s) = \int_{-\infty}^{\infty} h(\tau)e^{-s\tau} d\tau = \int_{-\infty}^{\infty} \delta(\tau - T)e^{-s\tau} d\tau = e^{-sT}.$$

This result is in agreement with the result in Ex. 1.4a.

- (b) An ideal differentiator is given as $y(t) = \frac{d}{dt}x(t)$, which is also obtained from Eq. (1.37) using $a_0 = b_1 = 1$ and all other coefficients zero. Thus, using Eq. (1.51), we find the transfer function of an ideal differentiator is given by

$$H(s) = \frac{b_1 s + b_0}{a_1 s + a_0} = \frac{s}{1} = s.$$

This result is identical to the result obtained in Ex. 1.4b.

An ideal integrator is given as $y(t) = \int_{-\infty}^t x(\tau) d\tau$. Differentiating this expression once yields the equivalent system $\frac{d}{dt}y(t) = x(t)$, which is also obtained from Eq. (1.37) using $a_1 = b_0 = 1$ and all other coefficients zero. Using Eq. (1.51), we find the transfer function of an ideal integrator is

$$H(s) = \frac{b_1 s + b_0}{a_1 s + a_0} = \frac{1}{s},$$

which agrees with the result found in Ex. 1.4c.

Drill 1.16 (Applying the Duality Property)

- (a) Pair 3 of Table 1.1 states that for $\operatorname{Re}\{\lambda\} < 0$,

$$\underbrace{e^{\lambda|t|}}_{x(t)} \iff \underbrace{\frac{-2\lambda}{\omega^2 + \lambda^2}}_{X(\omega)}.$$

The duality property, given in Eq. (1.80), therefore yields

$$\underbrace{\frac{-2\lambda}{t^2 + \lambda^2}}_{X(t)} \iff \underbrace{2\pi e^{\lambda|-\omega|}}_{2\pi x(-\omega)} = 2\pi e^{\lambda|\omega|}.$$

- (b) Multiplying pair 17 of Table 1.1 by $1/\pi$ and then setting $\omega_0 = t_0$ yield

$$\underbrace{\frac{1}{\pi} \cos(\omega_0 t)}_{x(t)} \iff \underbrace{[\delta(\omega - t_0) + \delta(\omega + t_0)]}_{X(\omega)}.$$

The duality property of Eq. (1.80) yields

$$\underbrace{[\delta(t - t_0) + \delta(t + t_0)]}_{X(t)} \iff \underbrace{2\pi \frac{1}{\pi} \cos(-\omega_0 t)}_{2\pi x(-\omega)} = 2 \cos(\omega_0 t).$$

Drill 1.17 (Combined Complex Conjugation and Time Reversal)

- (a) If $x(t)$ is conjugate symmetric, then

$$x(t) = x^*(-t) \iff X(\omega) = X^*(-(-\omega)) = X^*(\omega).$$

As shown in Eq. (1.16), $X(\omega) = X^*(\omega)$ implies that $X(\omega)$ is real. Thus, a conjugate-symmetric time-domain signal has a real Fourier transform.

- (b) If $x(t)$ is conjugate antisymmetric, then

$$x(t) = -x^*(-t) \iff X(\omega) = -X^*(-(-\omega)) = -X^*(\omega).$$

As shown in Eq. (1.17), $X(\omega) = -X^*(\omega)$ implies that $X(\omega)$ is imaginary. Thus, a conjugate-antisymmetric time-domain signal has a purely imaginary Fourier transform.

- (c) Since $y(t) = x_{\text{cs}}(t) = \frac{x(t) + x^*(-t)}{2}$, then, using properties and Eq. (1.18),

$$Y(\omega) = \frac{X(\omega) + X^*(\omega)}{2} = \operatorname{Re}\{X(\omega)\}.$$

- (d) Since $y(t) = x_{\text{ca}}(t) = \frac{x(t) - x^*(-t)}{2}$, then, using properties and Eq. (1.19),

$$Y(\omega) = \frac{X(\omega) - X^*(\omega)}{2} = j \frac{X(\omega) - X^*(\omega)}{2j} = j \operatorname{Im}\{X(\omega)\}.$$

Drill 1.18 (Using the Time-Shifting Property)

Multiplying pair 8 in Table 1.1 by $\frac{\pi}{B}$ and setting $\frac{B}{\pi} = \omega_0$ yield

$$x(t) = \text{sinc}(\omega_0 t) \iff X(\omega) = \frac{1}{\omega_0} \Pi\left(\frac{\omega}{2\pi\omega_0}\right).$$

Further, the time-shifting property tells us that $x(t - t_0) \iff X(\omega)e^{-j\omega t_0}$. Thus,

$$x(t - t_0) = \text{sinc}(\omega_0[t - t_0]) \iff X(\omega)e^{-j\omega t_0} = \frac{1}{\omega_0} \Pi\left(\frac{\omega}{2\pi\omega_0}\right) e^{-j\omega t_0}.$$

Figure D1.18 shows the corresponding rectangular magnitude and linear phase spectra.

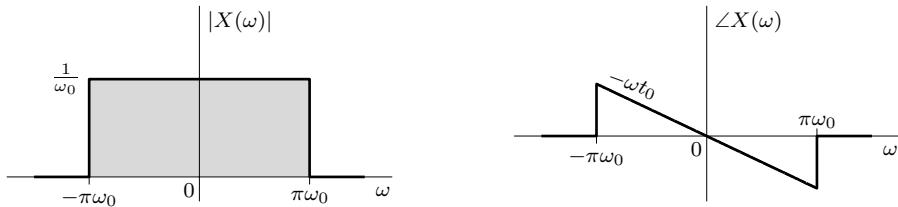


Figure D1.18

Drill 1.19 (Using the Modulation Property)

Figure D1.19 plots the signal $x(t) = \Pi\left(\frac{t}{4\pi}\right)$ and the corresponding modulated signal $x(t)\cos(5t)$. Using pair 7 in Table 1.1 yields

$$x(t) = \Pi\left(\frac{t}{4\pi}\right) \iff X(\omega) = 4\pi \text{sinc}(2\omega).$$

Combining this with the modulation property of Eq. (1.90) yields

$$\begin{aligned} x(t)\cos(\omega_0 t) &\iff \frac{1}{2}X(\omega - \omega_0) + \frac{1}{2}X(\omega + \omega_0) \\ \Pi\left(\frac{t}{4\pi}\right)\cos(5t) &\iff 2\pi \text{sinc}(2[\omega - 5]) + 2\pi \text{sinc}(2[\omega + 5]) \end{aligned}.$$

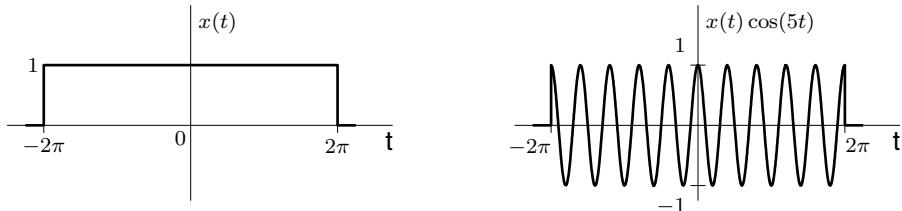


Figure D1.19

Drill 1.20 (Proving the Time-Integration Property with Convolution)

The time integration of a signal is represented using convolution as

$$y(t) = \int_{-\infty}^t x(\tau) d\tau = x(t) * u(t) \iff Y(\omega) = X(\omega)U(\omega).$$

Using pair 14 in Table 1.1, we know $u(t) \iff \pi\delta(\omega) + \frac{1}{j\omega}$. Thus, with the help of Eq. (1.5),

$$\int_{-\infty}^t x(\tau) d\tau \iff X(\omega) \left(\pi\delta(\omega) + \frac{1}{j\omega} \right) = \pi X(0)\delta(\omega) + \frac{X(\omega)}{j\omega}.$$

Drill 1.21 (Using the Time-Domain Convolution Property)

Using pair 1 in Table 1.1, we know $e^{\lambda t}u(t) \iff \frac{1}{j\omega - \lambda}$ if $\text{Re}\{\lambda\} < 0$. Using this with the convolution property of Eq. (1.95) yields

$$e^{\lambda_1 t}u(t) * e^{\lambda_2 t}u(t) \iff \frac{1}{(j\omega - \lambda_1)(j\omega - \lambda_2)}.$$

Computing the partial fraction expansion yields

$$\frac{1}{(j\omega - \lambda_1)(j\omega - \lambda_2)} = \frac{(\lambda_1 - \lambda_2)^{-1}}{j\omega - \lambda_1} + \frac{(\lambda_2 - \lambda_1)^{-1}}{j\omega - \lambda_2}.$$

This is inverted using pair 1 in Table 1.1 to yield the result

$$e^{\lambda_1 t}u(t) * e^{\lambda_2 t}u(t) = \frac{1}{\lambda_1 - \lambda_2}e^{\lambda_1 t}u(t) + \frac{1}{\lambda_2 - \lambda_1}e^{\lambda_2 t}u(t) = \frac{e^{\lambda_1 t} - e^{\lambda_2 t}}{\lambda_1 - \lambda_2}u(t),$$

provided that $\text{Re}\{\lambda_1\} < 0$ and $\text{Re}\{\lambda_2\} < 0$.

Drill 1.22 (Using the Frequency-Domain Convolution Property)

Manipulating pair 7 in Table 1.1 yields $\text{sinc}(\tau\omega) \iff \frac{1}{2\pi\tau} \Pi\left(\frac{t}{2\pi\tau}\right)$. Combining with the frequency-domain convolution property yields

$$\text{sinc}(\tau_1\omega) * \text{sinc}(\tau_2\omega) \iff 2\pi \left[\frac{1}{2\pi\tau_1} \Pi\left(\frac{t}{2\pi\tau_1}\right) \frac{1}{2\pi\tau_2} \Pi\left(\frac{t}{2\pi\tau_2}\right) \right].$$

The product of two unit gate functions is just the smaller of the two unit gate functions. Thus,

$$\text{sinc}(\tau_1\omega) * \text{sinc}(\tau_2\omega) \iff \begin{cases} \frac{1}{\tau_2} \left[\frac{1}{2\pi\tau_1} \Pi\left(\frac{t}{2\pi\tau_1}\right) \right] & \tau_1 < \tau_2 \\ \frac{1}{\tau_1} \left[\frac{1}{2\pi\tau_2} \Pi\left(\frac{t}{2\pi\tau_2}\right) \right] & \tau_1 > \tau_2 \end{cases}.$$

Inverting the right-hand side of the expression yields the final result of

$$\text{sinc}(\tau_1\omega) * \text{sinc}(\tau_2\omega) = \begin{cases} \frac{1}{\tau_2} \text{sinc}(\tau_1\omega) & \tau_1 < \tau_2 \\ \frac{1}{\tau_1} \text{sinc}(\tau_2\omega) & \tau_1 > \tau_2 \end{cases}.$$

Drill 1.23 (Correlation Order Is Important)

From Eqs. (1.97) and (1.98), we know that

$$\rho_{x,y}(\tau) = \int_{-\infty}^{\infty} x(t + \tau)y^*(t) dt = x(\tau) * y^*(-\tau).$$

Thus,

$$\rho_{y,x}(\tau) = \int_{-\infty}^{\infty} y(t + \tau)x^*(t) dt = y(\tau) * x^*(-\tau).$$

Since $x(\tau) * y^*(-\tau) \neq y(\tau) * x^*(-\tau)$ in general, it is clear that

$$\rho_{x,y}(\tau) \neq \rho_{y,x}(\tau).$$

To relate $\rho_{y,x}(\tau)$ to $\rho_{x,y}(\tau)$, we simplify $\rho_{y,x}^*(-\tau)$ as

$$\rho_{y,x}^*(-\tau) = \left(\int_{-\infty}^{\infty} y(t-\tau)x^*(t) dt \right)^* = \int_{-\infty}^{\infty} y^*(t-\tau)x(t) dt.$$

Performing a change of variable ($t - \tau \rightarrow t$) yields

$$\rho_{y,x}^*(-\tau) = \int_{-\infty}^{\infty} y^*(t)x(t+\tau) dt.$$

Comparing this result with the definition of $\rho_{x,y}(\tau)$ confirms that

$$\rho_{x,y}(\tau) = \rho_{y,x}^*(-\tau).$$

Drill 1.24 (Using Parseval's Theorem)

Pair 3 of Table 1.1 states that for $\operatorname{Re}\{\lambda\} < 0$,

$$\underbrace{e^{\lambda|t|}}_{x(t)} \iff \underbrace{\frac{-2\lambda}{\omega^2 + \lambda^2}}_{X(\omega)}.$$

Setting $\lambda = -a$, the duality property of Eq. (1.80) yields, for $a > 0$,

$$\underbrace{\frac{2a}{t^2 + a^2}}_{X(t)} \iff \underbrace{\frac{2\pi e^{-a|\omega|}}{2\pi x(-\omega)}}_{2\pi x(-\omega)} = 2\pi e^{-a|\omega|}.$$

The energy E_y of $y(t) = \frac{2a}{t^2 + a^2}$ is computed in the frequency domain using Parseval's theorem as

$$E_y = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left| 2\pi e^{-a|\omega|} \right|^2 d\omega = 2 \frac{4\pi^2}{2\pi} \int_0^{\infty} e^{-2a\omega} d\omega = 4\pi \frac{1}{-2a} e^{-2a\omega} \Big|_{\omega=0}^{\infty} = \frac{2\pi}{-a} (0 - 1) = \frac{2\pi}{a}.$$

Chapter 2

Drill 2.1 (Characterizing Behavior of a Real LTIC System)

Taking the Laplace transform of the system differential equation yields $(s^2 + 3s + 2)Y(s) = (s + 5)X(s)$. Taking the ratio of $Y(s)/X(s)$ and factoring the denominator yield the system transfer function

$$H(s) = \frac{s + 5}{(s + 2)(s + 1)}.$$

There are two poles ($p_1 = -2$ and $p_2 = -1$) and one finite zero ($z_1 = -5$), as shown in Fig. D2.1a. There is also one zero at infinity. The poles are nearest to the origin, which will boost the low-frequency response, and the zero at infinity ensures that high frequencies are eliminated. Thus, we expect the response to be lowpass, with dc gain of $H(0) = 5/2 = 2.5$. This prediction is verified with MATLAB-generated magnitude and phase response plots, as shown in Figs. D2.1b and D2.1c.

```
01 H = @(s) (s+5)./(s.^2+3*s+2); omega = linspace(-10,10,501);
02 subplot(211); plot(omega,abs(H(1j*omega)));
03 xlabel('omega'); ylabel('|H(j\omega)|');
04 subplot(212); plot(omega,angle(H(1j*omega)));
05 xlabel('omega'); ylabel('angle H(j\omega)');
```

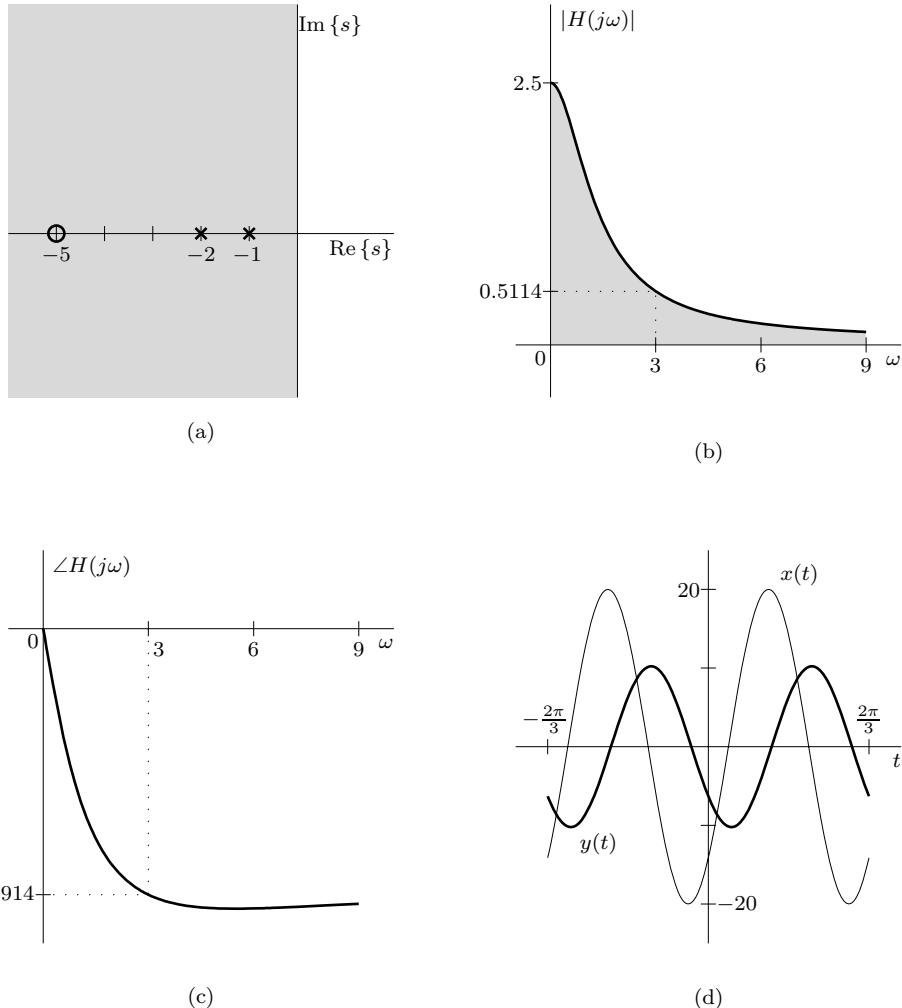


Figure D2.1

The input $x(t) = 20 \sin(3t - \pi/4)$ has radian frequency 3 rad/s. Consulting the magnitude and phase plots at this frequency, the response $y(t)$ to input $x(t)$ is thus

$$\begin{aligned} y(t) &= |H(j3)|20 \sin[3t - \pi/4 + \angle H(j3)] = 0.5114(20) \sin(3t - \pi/4 - 1.6914) \\ &= 10.2282 \sin(3t - 2.4768). \end{aligned}$$

The output is shown in Fig. D2.1d, which also plots the input for reference.

Drill 2.2 (Characterizing Behavior of a Complex LTIC System)

Similar to Ex. 2.2, MATLAB computes and plots the system zeros and poles, which are shown in Fig. D2.2a.

```
01 B = [1 2*j]; A = [1 2-4*j -3-4*j]; z=roots(B), p = roots(A)
02 subplot(131); plot(real(z),imag(z),'o',real(p),imag(p),'x');
z = -2i
p = -1+2i -1+2i
```

No conjugate poles accompany the repeated pole at $s = -1 + 2j$, which confirms that the system must be complex. Since the poles are nearest to frequency $\omega = 2$, we expect strong gain near this

(positive) frequency. The zero at $s = -2j$ ensures that the response is zero at the negative frequency $\omega = -2$. Additionally, there is a zero at infinity. Thus, we expect a response that peaks near $\omega = 2$, is zero at $\omega = -2$, and decays to zero as $\omega \rightarrow \pm\infty$.

Our predictions about system behavior are verified with MATLAB-generated magnitude and phase response plots, as shown in Figs. D2.2b and D2.2c. Neither response has the symmetry necessary for a real system.

```
03 omega = linspace(-6,6,501); H = polyval(B,1j*omega)./polyval(A,1j*omega);
04 subplot(132); plot(omega,abs(H)); subplot(133); plot(omega,unwrap(angle(H)));
```

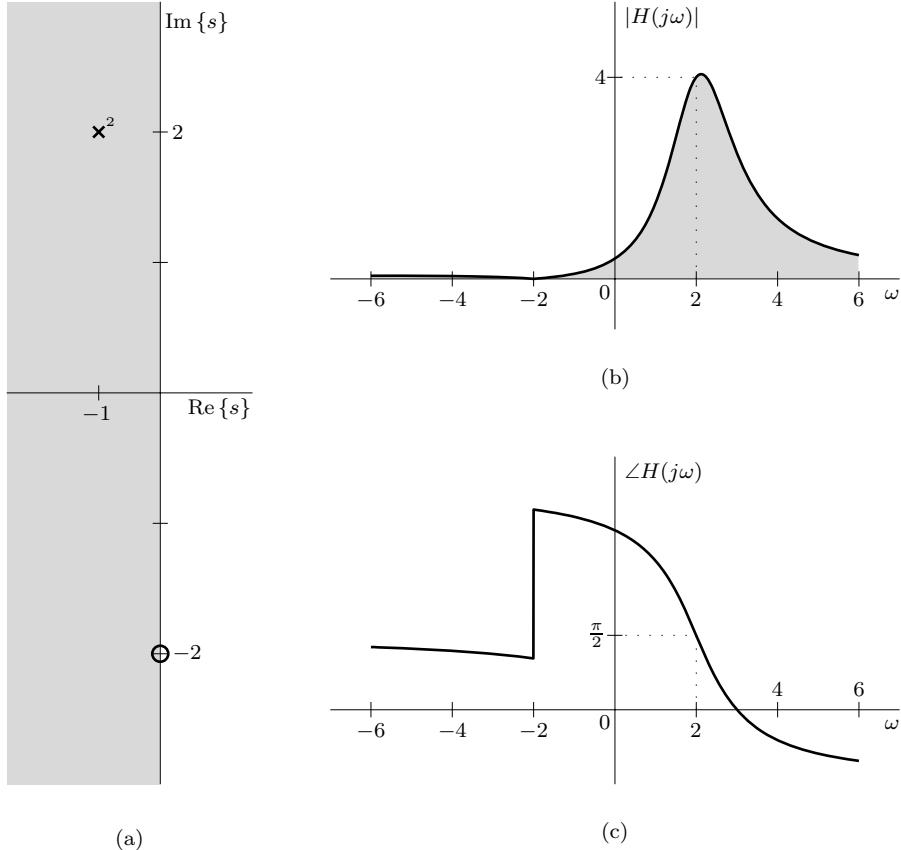


Figure D2.2

Since the system is complex, Eq. (2.5) cannot be used to determine the output $y(t)$ in response to input $x(t)$. Rather, we expand $x(t)$ using Euler's formula as

$$x(t) = 10 \sin(2t + \pi/4) = \frac{10e^{j(2t+\pi/4)} - 10e^{-j(2t+\pi/4)}}{2j}.$$

Due to the system zero at $s = -2j$, the component at $\omega = -2$ is completely removed. This leaves a single complex exponential component to the input, the output of which is computed using Eq. (2.1). Consulting the magnitude and phase plots at this frequency, the response $y(t)$ to input $x(t)$ is thus

$$\begin{aligned} y(t) &= |H(j2)| \frac{10}{2j} e^{j[2t+\pi/4+\angle H(j2)]} = (4)5e^{-j\pi/2} e^{j(2t+\pi/4+\pi/2)} \\ &= 20e^{j(2t+\pi/4)}. \end{aligned}$$

Drill 2.3 (System Analysis Using the Fourier Transform)

Using pair 2 of Table 1.1 and Eq. (2.11), the frequency-domain representation of the system output is

$$Y(\omega) = X(\omega)H(\omega) = \left(\frac{-1}{j\omega - 1} \right) \left(\frac{1}{j\omega + 2} \right).$$

Using partial fraction expansions, we express the output as

$$Y(\omega) = \frac{-1/3}{j\omega - 1} + \frac{1/3}{j\omega + 2}.$$

Again using Table 1.1, the time-domain result is thus

$$y(t) = \frac{1}{3}e^t u(-t) + \frac{1}{3}e^{-2t} u(t).$$

Drill 2.4 (Relating Magnitude Response and Filter Characteristics)

Figure 2.14a verifies the various relations between magnitude response and filter characteristics.

- (a) The magnitude response of an ideal filter can only take two possible values: zero or one. That is, $|H(\omega)| = 0$ or $|H(\omega)| = 1$ for all ω . As a consequence, ideal filters instantaneously transition from passband to stopband (and vice versa).
- (b) The magnitude response of a real filter must display even symmetry. That is, $|H(\omega)| = |H(-\omega)|$.
- (c) The magnitude response of a continuous-time filter must be aperiodic, as summarized in Table 1.4. The fact that $H(\omega)$ is a function of the continuous variable ω is not relevant; the magnitude responses of continuous-time and discrete-time filters are both functions of continuous frequency variables.
- (d) The magnitude response of a lowpass filter has a passband (unit or near-unit gain) for frequencies $|\omega| < \omega_p$ and a stopband (zero or near-zero gain) for frequencies $|\omega| > \omega_s$. That is, the filter passes low frequencies and attenuates high frequencies. In Fig. 2.14a, $\omega_p = \omega_s = \omega_1$.

Drill 2.5 (A Gaussian Frequency Response Is Unrealizable)

- (a) Using pair 11 of Table 1.1, we know that a Gaussian transforms into a Gaussian as

$$H(\omega) = e^{-\alpha\omega^2} \iff h(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{-t^2/(4\alpha)}.$$

Since this everlasting Gaussian impulse response is noncausal ($h(t) \neq 0$ for $t < 0$), the system is unrealizable.

- (b) Using Eq. (2.19), we evaluate the Paley-Wiener criterion according to

$$\int_{-\infty}^{\infty} \frac{|\ln|H(\omega)||}{1+\omega^2} d\omega = \int_{-\infty}^{\infty} \frac{\alpha\omega^2}{1+\omega^2} d\omega = \alpha [\omega - \tan^{-1}(\omega)] \Big|_{\omega=-\infty}^{\infty} \rightarrow \infty.$$

Since this integral is not finite, the criterion is violated, and the system is unrealizable.

Drill 2.6 (Computing and Plotting Kaiser Windows)

To plot and compute the effective widths of Kaiser windows, MATLAB is utilized.

```

01 tbyT = -.75:.001:.75; wrec = @(tbyT) 1.0*(abs(tbyT)<=0.5);
02 wkai = @(tbyT,alpha) besseli(0,alpha*sqrt(1-4*tbyT.^2))/besseli(0,alpha).*wrec(tbyT);
03 for alpha = [0,1,2,3,5,10,20],
04     line(tbyT,wkai(tbyT,alpha));
05     wkai2 = @(tbyT) wkai(tbyT,alpha).^2;
06     DeltaEwkai = @(Wby2) quad(wkai2,-Wby2,Wby2);
07     ObjFun = @(Wby2) abs(.95*Ewkai(.5)-Ewkai(Wby2)); W = 2*fminbnd(ObjFun,0,.5);
08 end

```

Line 01 defines a normalized time vector (t/T) and defines the rectangular window. Following Table 2.1, line 02 defines the Kaiser window. Line 03 begins a for loop to treat each value of α . Line 04 plots the Kaiser window for the current α . Effective window width is computed in exactly the same manner as effective bandwidth, treated in Ch. 1. In fact, lines 05–07 are basically identical to lines 01–03 in Ex. 1.14, which are explained on page 64. The only significant difference is that line 07 uses the `fminbnd` function rather than the `fminsearch` function; this function allows a bounded search just over the finite duration of the window function. Figure D2.6 plots the resulting Kaiser windows and summarizes normalized window energy E_w/T and effective window width W/T for the different α . The price of ever-smoother windows is an ever-smaller effective window width.

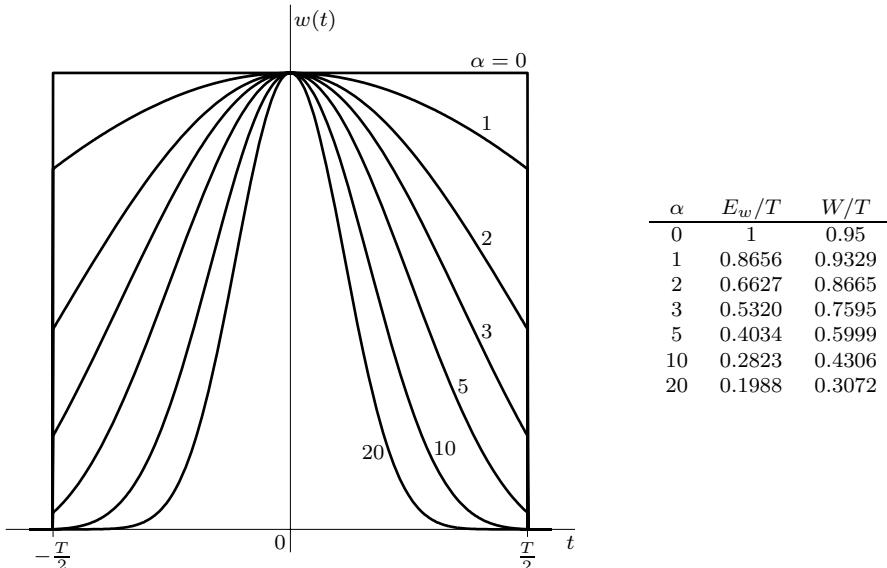


Figure D2.6

Drill 2.7 (Lowpass-to-Lowpass Transformation)

First, we determine the half-power (3-dB) frequency of the lowpass prototype by solving

$$|H(j\omega_0)|^2 = \frac{4}{\omega_0^2 + 4} = \frac{1}{2} \Rightarrow \omega_0 = 2.$$

Using Eq. (2.25), the lowpass-to-lowpass transformation rule and transformed filter are thus

$$s \rightarrow \frac{\omega_0 s}{\omega_1} = \frac{2s}{3} \quad \text{and} \quad H(s) = H_{lp}(2s/3) = \frac{2}{2s/3 + 2} = \frac{3}{s + 3}.$$

The magnitude response of the transformed filter, shown in Fig. D2.7, confirms that the 3-dB frequency is correctly located at 3 rad/s.

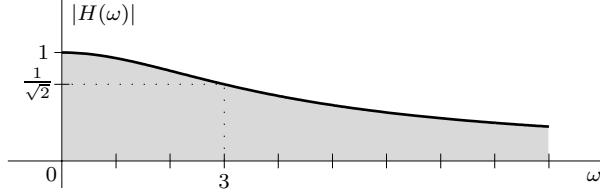


Figure D2.7

Drill 2.8 (Lowpass-to-Bandpass Transformation)

First, we determine the half-power (3-dB) frequency of the lowpass prototype by solving

$$|H(j\omega_0)|^2 = \frac{4}{\omega_0^2 + 4} = \frac{1}{2} \Rightarrow \omega_0 = 2.$$

Using Eq. (2.29), the lowpass-to-bandpass transformation rule is

$$s \rightarrow \omega_0 \frac{s^2 + \omega_1 \omega_2}{s(\omega_2 - \omega_1)} = 2 \frac{s^2 + 1(3)}{s(3 - 1)} = \frac{s^2 + 3}{s}.$$

The resulting bandpass filter is thus

$$H(s) = H_{lp} \left(\frac{s^2 + 3}{s} \right) = \frac{2}{(s^2 + 3)/s + 2} = \frac{2s}{s^2 + 2s + 3}.$$

The magnitude response of the transformed filter, shown in Fig. D2.8, confirms that the 3-dB frequencies are correctly located at 1 and 3 rad/s.

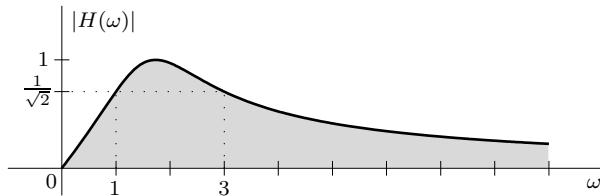


Figure D2.8

Drill 2.9 (Determining Butterworth Lowpass Filter Order)

To begin, we convert δ_p and δ_s into the corresponding attenuation parameters using Eq. (2.24). Then, as in Ex. 2.11, we use Eq. (2.38) to determine filter order K .

```
01 omegap = 100; deltap = 0.05; omegas = 200; deltas = 0.01;
02 alphap = -20*log10(1-deltap); alphas = -20*log10(deltas);
03 K = ceil(log((10^(alphas/10)-1)/(10^(alphap/10)-1))/(2*log(omegas/omegap)))
K = 9
```

Thus, a ninth-order Butterworth lowpass filter is required to meet the given specifications.

Drill 2.10 (Determining Butterworth Bandpass Filter Order)

The necessary Butterworth bandpass filter order is computed in the same fashion as in Ex. 2.12. Following that example, we first relate the bandpass characteristics to the stopband frequency (ω_s) of a normalized ($\omega_p = 1$) Butterworth filter.

```
01 omegap1 = 100; omegap2 = 200; omegas1 = 50; omegas2 = 250; omegap = 1;
02 omegas = abs([omegap*(omegas1^2-omegap1*omegap2)/(omegas1*(omegap2-omegap1)),...
03           omegap*(omegas2^2-omegap1*omegap2)/(omegas2*(omegap2-omegap1))])
omegas = 3.5000 1.7000
```

The value corresponding to the upper transition band is the most restrictive and dictates the lowpass prototype order.

```
04 omegas = min(omegas); alphap = 0.5; alphas = 40;
05 K = ceil(log((10^(alphas/10)-1)/(10^(alphap/10)-1))/(2*log(omegas/omegap)))
K = 11
```

A lowpass-to-bandpass transformation doubles the order of the lowpass prototype. Thus, the necessary Butterworth bandpass filter order is $K = 22$.

Drill 2.11 (Comparing Chebyshev and Butterworth Filter Orders)

Using Eqs. (2.46) and (2.38), the orders for Chebyshev and Butterworth filters are easily compared.

```
01 alphap = 0.5; alphas = 20; wsbywp = [1.01; 1.1; 2; 5; 10];
02 Kc = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))./acosh(wsbywp));
03 Kb = ceil(log(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))./log(wsbywp));
04 [wsbywp, Kc, Kb]
```

The results are summarized as

	ω_s/ω_p	1.01	1.1	2	5	10	.
Chebyshev order		29	10	4	2	2	.
Butterworth order		337	36	5	3	2	.

For more restrictive specifications, Chebyshev filter order can be an impressive order of magnitude smaller than the Butterworth case. Further, a Chebyshev filter never has higher order than a comparable Butterworth filter.

Drill 2.12 (Effect of Passband Ripple on Chebyshev Pole Locations)

To locate the Chebyshev poles from Butterworth circles and poles, the Butterworth circle radii and pole locations are first computed using Eqs. (2.47) and (2.35).

```
01 K = 3; k = [1:K]; alphap = 3; omegap = 1; epsilon = sqrt(10^(alphap/10)-1);
02 Bradius1 = omegap*sinh(asinh(1/epsilon)/K); B1pk = Bradius1*exp(1j*pi/(2*K)*(2*k-1+K));
03 Bradius2 = omegap*cosh(asinh(1/epsilon)/K); B2pk = Bradius2*exp(1j*pi/(2*K)*(2*k-1+K));
04 Cpk = real(B1pk)+1j*imag(B2pk);
```

To provide reference, the Butterworth circles and Chebyshev ellipse are next found.

```
05 omega = linspace(0, 2*pi*3/4, 1001);
06 Bcircle1 = Bradius1*exp(1j*omega); Bcircle2 = Bradius2*exp(1j*omega);
07 Cellipse = Bradius1*cos(omega)+1j*Bradius2*sin(omega);
```

Lastly, the results are plotted.

```
08 plot(real(Bcircle1), imag(Bcircle1), '--', real(B1pk), imag(B1pk), 'x', ...
09      real(Bcircle2), imag(Bcircle2), '--', real(B2pk), imag(B2pk), 'x', ...
10      real(Cellipse), imag(Cellipse), '--', real(Cpk), imag(Cpk), 'x')
```

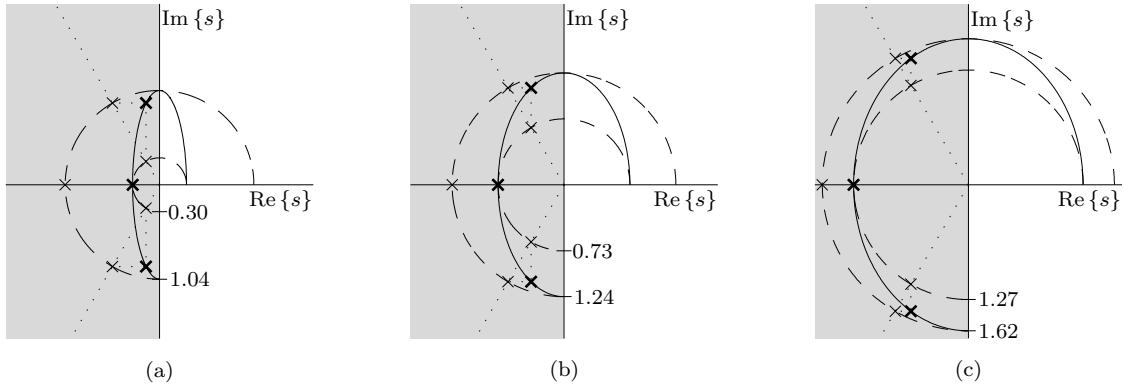


Figure D2.12

The results, using $\alpha_p = 3$ dB, are shown in Fig. D2.12a.

The $\alpha_p = 0.3$ dB and $\alpha_p = 0.03$ dB cases, shown in Figs. D2.12b and D2.12c, respectively, are produced with simple modification to line 01. As Fig. D2.12 demonstrates, the Chebyshev ellipse expands and converges toward a circle as α_p is decreased. That is, as we eliminate our tolerance for passband ripple, our Chebyshev filter looks more and more like a Butterworth response.

Drill 2.13 (Chebyshev Lowpass Filter Design)

Although the problem specifies an order of $K = 6$, we begin by computing the minimum order to meet the given specifications.

```

01 omegap = 10; omegas = 30; deltap = 0.1; deltas = 0.01;
02 alphap = -20*log10(1-deltap), alphas = -20*log10(deltas),
03 K = ceil(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/acosh(omegas/omegap))
K = 4

```

Since the minimum required order is four, our sixth-order solution will likely exceed specifications by significant margins. If we proceed as in Ex. 2.13, we can construct passband and stopband buffer zones, but our passband ripple will meet specifications exactly without margin for error (due to the nature of the Chebyshev response, stopband behavior will exceed specifications everywhere). Thus, it is worthwhile to strengthen our passband ripple constraint. Consider what happens when the passband ripple requirement is reduced by a factor of 10 so that $\delta_p \leq 0.01$. Making the appropriate change to line 01 and re-executing lines 01–03 yield the result $K = 5$, which is still below the specified order of six.

Using the strengthened passband ripple requirement, we can design a $K = 6$ filter whose response nicely exceeds all specifications. As in Ex. 2.13, we choose ω_p as the mean of the two limiting extremes.

```

04 K = 6;
05 omegap = [omegap,omegas/cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K)];
06 omegap = mean(omegap)
omegap = 13.2306

```

Next, we use Eqs. (2.44), (2.45), (2.47), and (2.48) to determine the desired transfer function coefficients.

```

05 epsilon = sqrt(10^(alphap/10)-1); k = 1:K;
06 H0 = (mod(K,2)==1)+(mod(K,2)==0)/sqrt(1+epsilon^2);
07 pk = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
        1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));

```

```
09 B = H0*prod(-pk), A = poly(pk)
B = 1176347.05
A = 1.00 23.29 533.73 6724.15 65864.89 388261.43 1188229.35
```

Thus,

$$H(s) = \frac{1176347.05}{s^6 + 23.29s^5 + 533.73s^4 + 6724.15s^3 + 65864.89s^2 + 388261.43s + 1188229.35}.$$

The corresponding magnitude response, plotted using MATLAB, is shown in Fig. D2.13a. Given the stringent passband ripple requirement, the zoom plot of Fig. D2.13b more clearly shows that passband requirements are met, and the dB plot of Fig. D2.13c better illustrates that stopband requirements are met.

```
10 omega = linspace(0,35,1001); H = B./(polyval(A,1j*omega));
11 subplot(311); plot(omega,abs(H));
12 subplot(312); plot(omega,abs(H)); axis([0 35 .98 1.01]);
13 subplot(313); plot(omega,20*log10(abs(H))); axis([0 35 -60 10]);
```

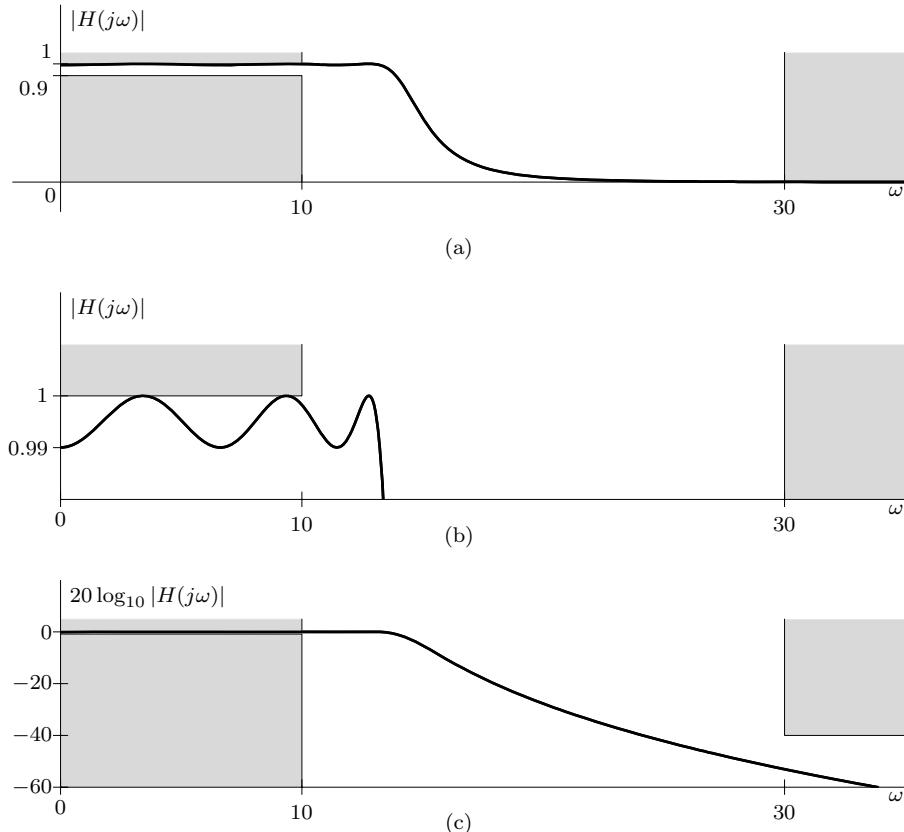


Figure D2.13

Although the designed filter easily meets all specifications, the solution is not unique. An infinite number of possible filters exist that satisfy all the constraints of the problem.

Drill 2.14 (Investigating Bessel-Thomson Filter Characteristics)

Using Table 2.5, a fifth-order normalized ($\omega_1 = 1$) Bessel-Thomson prototype transfer function is

given as

$$H_p(s) = \frac{\mathcal{B}_K(0)}{\mathcal{B}_K(s)} = \frac{945}{s^5 + 15s^4 + 105s^3 + 420s^2 + 945s + 945}.$$

A lowpass-to-lowpass transformation allows us to obtain the general response of Eq. (2.59) as

$$H(s) = H_p\left(\frac{s}{\omega_1}\right) = \frac{945\omega_1^5}{s^5 + 15\omega_1 s^4 + 105\omega_1^2 s^3 + 420\omega_1^3 s^2 + 945\omega_1^4 s + 945\omega_1^5},$$

which has low-frequency delay response of $1/\omega_1$. To set our low-frequency delay response equal to $100 \mu\text{s}$ requires $\omega_1 = 1/0.0001 = 10^4$, which yields a final transfer function of

$$H(s) = \frac{945(10^{20})}{s^5 + 15(10^4)s^4 + 105(10^8)s^3 + 420(10^{12})s^2 + 945(10^{16})s + 945(10^{20})}.$$

The corresponding magnitude and delay responses, shown in Fig. D2.14, are computed using MATLAB. The delay response calculation of line 04 is a numerical approximation based on a simple first-order difference approximation to the derivative $-\frac{d}{d\omega}\angle H(j\omega)$.

```

01 B = 945*10^20; A = [1 15*10^4 105*10^8 420*10^12 945*10^16 945*10^20];
02 omega = linspace(0,50000,1001); H = polyval(B,1j*omega)./polyval(A,1j*omega);
03 subplot(211); plot(omega,abs(H));
04 DelayResp = -diff(unwrap(angle(H)))/(omega(2)-omega(1));
05 subplot(212); plot(omega(1:end-1),DelayResp);

```

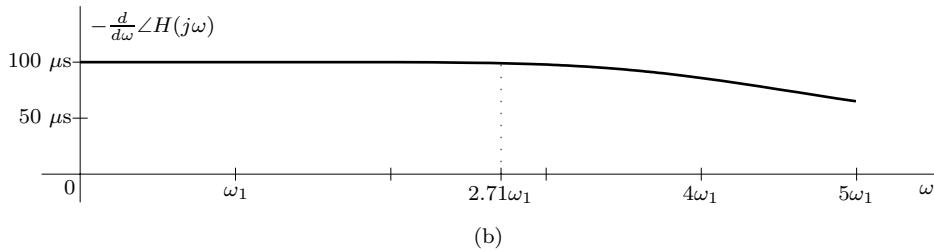
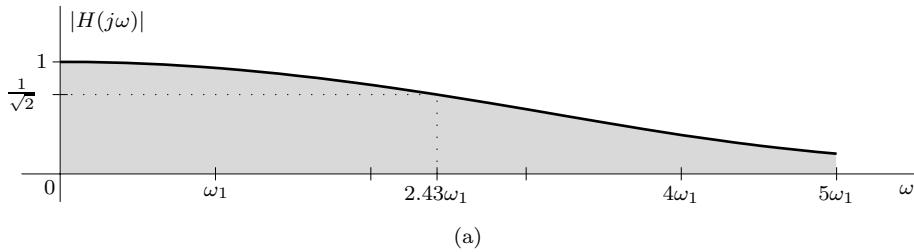


Figure D2.14

Using Fig. D2.14a, the 3-dB frequency is found to be approximately $2.43\omega_1$, or 24300 rad/s. To determine the bandwidth over which the delay response of Fig. D2.14b is approximately constant requires us to quantify what is meant by “approximately constant.” Using a tolerance of 1%, we find that bandwidth of constant delay is approximately $2.71\omega_1$, or 27100 rad/s.

Chapter 3

Drill 3.1 (Determining the Nyquist Rate and the Nyquist Interval)

From pair 8 of Table 1.1, we know $\frac{B}{\pi} \text{sinc}\left(\frac{Bt}{\pi}\right) \iff \Pi\left(\frac{\omega}{2B}\right)$. Thus, $\text{sinc}(100t)$ has spectral width of

50 Hz, and $\text{sinc}(50t)$ has spectral width of 25 Hz.

- (a) For $x_a(t) = \text{sinc}(100t)$, the Nyquist rate is $F_s = 2(50) = 100$ Hz, and the Nyquist interval is $1/F_s = 10$ ms.
- (b) For $x_b(t) = \text{sinc}(50t) + \text{sinc}(100t)$, the spectral width remains 50 Hz. Thus, the Nyquist rate is $F_s = 2(50) = 100$ Hz, and the Nyquist interval is $1/F_s = 10$ ms.
- (c) Using the frequency-domain convolution property, we compute the spectral width of $x_c(t) = \text{sinc}(50t) \text{sinc}(100t)$ as $25 + 50 = 75$ Hz. Thus, the Nyquist rate is $F_s = 2(75) = 150$ Hz, and the Nyquist interval is $1/F_s = 6\frac{2}{3}$ ms.
- (d) Using the time-domain convolution property, we compute the spectral width of $x_d(t) = \text{sinc}(50t) * \text{sinc}(100t)$ as 25 Hz, which is the smaller of the two individual spectral widths. Thus, the Nyquist rate is $F_s = 2(25) = 50$ Hz, and the Nyquist interval is $1/F_s = 20$ ms.

Drill 3.2 (Not All Pulses Are Appropriate for Pulse Sampling)

If the pulse $p(t)$ has zero area, then $\tilde{p}(t) = \sum_{n=-\infty}^{\infty} p(t-nT)$ must also have zero area. Consequently, $\tilde{P}_0 = 0$, and the $k = 0$ replicate $X(\omega - k\omega_s) = X(\omega)$, which is centered at $\omega = 0$, is destroyed. Without this copy of $X(\omega)$ in $X_{\tilde{p}}(\omega)$, it is impossible to recover $X(\omega)$ through simple lowpass filtering.

Drill 3.3 (Interpolating a Discrete Rectangular Pulse)

We use MATLAB to compute and plot the interpolation of the discrete rectangular pulse.

```

01 xn = @(n) 1.0*(abs(n)<=5); N = 19; tbyT = linspace(-20,20,1001);
02 xhat = 0; for n=-N:N,
03     xhat = xhat + xn(n)*sinc(tbyT-n);
04 end
05 plot(tbyT,xhat,-N:N,xn(-N:N),'o');

```

Line 01 define the samples $x[n]$ of the rectangular pulse, the limits $\pm N$ for plotting, and the normalized time vector t/T . Lines 02–04 estimate the corresponding continuous-time, bandlimited signal $x(t)$ using the interpolation formula of Eq. (3.13). Lastly, line 05 plots the result, shown in Fig. D3.3, including the original samples. Clearly, $\hat{x}(t)$ is not a continuous-time rectangular pulse. Rather, it is a *bandlimited* pulse whose samples, properly aligned, are a rectangular pulse. The interpolation formula, which necessarily produces bandlimited signals, should not produce a true continuous-time rectangular pulse, which has infinite bandwidth.

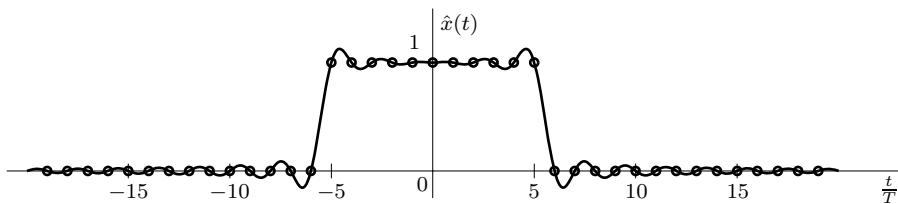


Figure D3.3

Drill 3.4 (Graphical Verification of Aliasing in Sinusoids)

Figure D3.4a shows the spectrum of signal $x(t) = \cos[(\omega_s/2 + \omega_1)t]$. When $x(t)$ is sampled to produce $x_{\tilde{\delta}}(t)$, the spectrum $X(\omega)$ is replicated every ω_s , as shown in Fig. D3.4b. Over $-\omega_s \leq \omega \leq \omega_s$, we see the original spectrum ($k = 0$ replicate) as well as portions of two other replicates ($k = -1$ and $k = 1$). Figure D3.4c shows the spectrum of signal $y(t) = \cos[(\omega_s/2 - \omega_1)t]$. When $y(t)$ is sampled to produce $y_{\tilde{\delta}}(t)$, the spectrum $Y(\omega)$ is replicated every ω_s , as shown in Fig. D3.4d. Although the

impulses correspond to different replicates, it is clear that the two sampled signal spectra are equal; that is, $Y_{\tilde{\delta}}(\omega) = X_{\tilde{\delta}}(\omega)$. The result also holds true for $\omega_1 > \omega_s/2$, although the labeling of replicates will likely change somewhat.

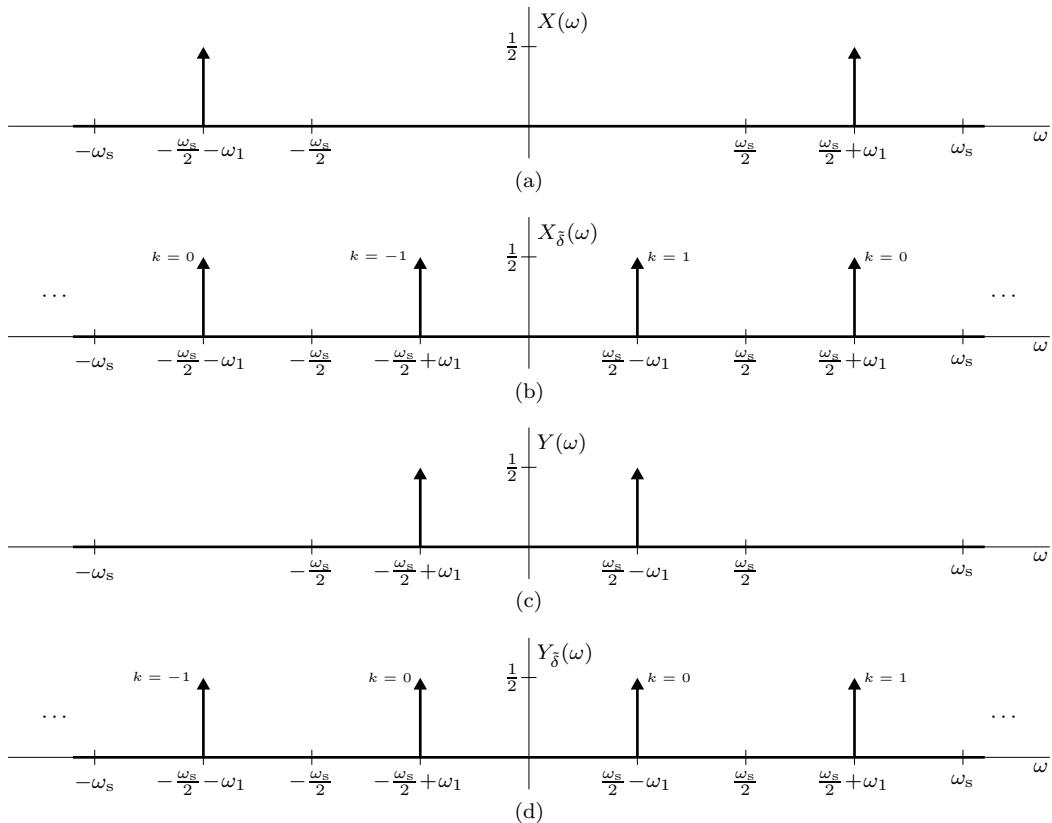


Figure D3.4

Drill 3.5 (Anti-Aliasing Filter Type Is Important)

Recall that when a signal is sampled, all frequency content beyond $F_s/2$ aliases back to the fundamental band $[-F_s/2, F_s/2]$. If the sampled signal has broadband noise or broadband content, the fundamental-band accumulation of multiple aliases can be significant. A good anti-aliasing filter needs to remove the high-frequency components so that the fundamental-band accumulation of aliased components is limited and, hopefully, insignificant.

A lowpass Butterworth filter has a monotonically decreasing frequency response. What this means is that components at ever-higher frequencies are increasingly attenuated. Even the lowest Butterworth roll-off of -20 dB/dec ensures that there is a limit to the total accumulation of aliased content from higher frequencies. Each decade increase in frequency is accompanied by at least a tenfold reduction in gain. Aliased content is only significant for a finite number of foldings, which ensures that the total accumulation is likewise limited.

The roll-off of an elliptic filter, on the other hand, is at best -20 dB/dec (odd-order filters) and at worst 0 dB/dec (even-order filters). In the case of an even-order elliptic filter, the combination of a fixed attenuation with infinite aliases translates to a potentially devastating distortion of the fundamental band. Even a minuscule stopband gain, when compounded with an infinite number of foldings, can produce unacceptable results.

Drill 3.6 (Investigating Equivalence of Aliased Sinusoids)

Let $x_1(t) = \cos(2\pi 90t)$ and $x_2(t) = \cos(2\pi 110t)$. Sampling signal $x_1(t)$ at $F_s = 200$ Hz ($T = 1/200$) produces $x_1(nT) = \cos(2\pi \frac{90}{200}n) = \cos(2\pi \frac{9}{20}n)$. Similarly, sampling signal $x_2(t)$ at $F_s = 200$ Hz ($T = 1/200$) produces $x_2(nT) = \cos(2\pi \frac{110}{200}n) = \cos(2\pi \frac{11}{20}n)$. A sinusoid is unchanged by adding any integer multiple of 2π , so $x_2(nT) = \cos(2\pi \frac{11}{20}n - 2\pi \frac{20}{20}n) = \cos(-2\pi \frac{9}{20}n)$. Since cos is an even function, it follows that $x_2(nT) = \cos(2\pi \frac{9}{20}n) = x_1(nT)$.

Had the signals $x_1(t)$ and $x_2(t)$ been each offset in phase by some factor θ , their sampled versions, while identical in $|f_a|$, would differ by a sign factor in the phase term. To understand why, notice that the apparent frequency of $x_1(t)$ is $\langle f_0 + F_s/2 \rangle_{F_s} - F_s/2 = \langle 90 + 100 \rangle_{200} - 100 = 90$ Hz, whereas the apparent frequency of $x_2(t)$ is $\langle 110 + 100 \rangle_{200} - 100 = -90$. Since the two apparent frequencies differ by a sign, the phase of the corresponding apparent sinusoids will also differ by a sign factor, as explained in the text.

Drill 3.7 (An Alternate Form for Apparent Sinusoids)

Using Eq. (3.14) to compute apparent frequency f_a , we know that signal $\sin(2\pi f_0 t + \theta)$ will appear as $\sin(2\pi f_a t + \theta)$. If $f_a < 0$, we can rewrite the apparent sinusoid as $\sin(-2\pi |f_a| t + \theta)$. Since $\sin(-x) = -\sin(x)$, the resulting apparent sinusoid is represented as

$$-\sin(2\pi |f_a| t - \theta) \quad \text{for } f_a < 0.$$

Thus, when the original signal is in the form of sine rather than cosine, the apparent sinusoid includes both a phase reversal and an amplitude sign change.

Drill 3.8 (Computing Apparent Frequency)

From Eq. (3.14), we compute apparent frequency as $f_a = \langle f_0 + F_s/2 \rangle_{F_s} - F_s/2$. In each case, the sampling frequency remains constant at $F_s = 100$ Hz.

- (a) For $f_0 = 40$ Hz, $f_a = \langle 40 + 50 \rangle_{100} - 50 = 40$, and $|f_a| = 40$.
- (b) For $f_0 = 60$ Hz, $f_a = \langle 60 + 50 \rangle_{100} - 50 = -40$, and $|f_a| = 40$.
- (c) For $f_0 = 140$ Hz, $f_a = \langle 140 + 50 \rangle_{100} - 50 = 40$, and $|f_a| = 40$.
- (d) For $f_0 = 160$ Hz, $f_a = \langle 160 + 50 \rangle_{100} - 50 = -40$, and $|f_a| = 40$.

Interestingly, each of the four cases has, in absolute value, the same apparent frequency.

Drill 3.9 (Sampling a Bandpass Signal)

Using Eq. (3.16), we determine that the maximum possible number of replicates in band 0 to f_1 is

$$K_{\max} = \left\lfloor \frac{f_2}{B} \right\rfloor - 1.$$

Simplifying Eq. (3.15), we compute permissible sampling rates according to

$$\frac{2}{K+1}f_2 < F_s < \frac{2}{K}f_1.$$

In the present case, $f_1 = 5$ kHz, and $f_2 = 7$ kHz. Thus,

$$K_{\max} = \left\lfloor \frac{7}{2} \right\rfloor - 1 = 2.$$

The permissible sampling rates include

$$\begin{array}{ll} F_s > 14 \text{ kHz} & (K = 0 \text{ case}) \\ 7 \text{ kHz} < F_s < 10 \text{ kHz} & (K = 1 \text{ case}) \\ 14/3 \text{ kHz} < F_s < 5 \text{ kHz} & (K = 2 \text{ case}) \\ \text{no solutions} & (\text{all } K > 2) \end{array}.$$

From these calculations, the lowest possible sampling rate is clearly bound by $F_s = 14/3 \text{ kHz}$.

Drill 3.10 (Sampling a Complex Bandpass Signal)

Figure D3.10 illustrates the spectrum $X(\omega) = \frac{\omega-1}{2}\Pi\left(\frac{\omega-2}{2}\right)$ of complex signal $x(t)$. This signal, which is comprised solely of positive-frequency components, has a bandwidth of 2 rad/s. Further, this bandwidth constitutes the entire spectral duration of the signal. Consequently, $x(t)$ can be recovered from samples taken at any rate that exceeds this width,

$$\omega_s > 2 \text{ rad/s} \quad \text{or} \quad F_s > \frac{1}{\pi} \text{ Hz.}$$

The Nyquist rate is twice the highest (absolute) frequency of $x(t)$,

$$\omega_{\text{Nyq}} > 6 \text{ rad/s} \quad \text{or} \quad F_{\text{Nyq}} > \frac{3}{\pi} \text{ Hz.}$$

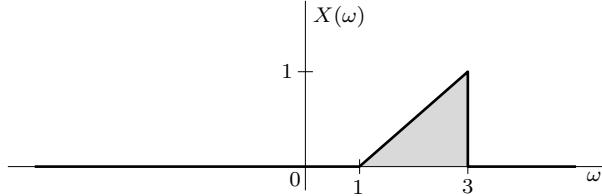


Figure D3.10

Drill 3.11 (Computing the Number of Bits and Baud Rate)

The number of bits needed to encode the 128 possible states (characters) is

$$B = \log_2(128) = 7.$$

Thus, to transmit 100000 characters per second requires a baud rate of

$$7 \times 100000 = 700000 \text{ bits/s.}$$

Drill 3.12 (Quantization with M -ary Symbols)

Using M -ary numbers, a total of B digits produces $L = M^B$ distinct states (or levels). Thus, to represent L levels requires $B = \lceil \log_M(L) \rceil$ digits. The ceiling operator is used to ensure an integer number of digits B .

Using these relations, we compute the number of digits required to achieve $L = 64$ levels for the cases $M = 2, 3, 4, 5, 6, 7$, and 8 .

M	2	3	4	5	6	7	8
$\log_M(L)$	6.0000	3.7856	3.0000	2.5841	2.3211	2.1372	2.0000
B	6	4	3	3	3	3	2

Drill 3.13 (Identifying Quantization Method)

Since the quantization boundaries of Fig. 3.22, reproduced in Fig. D3.13, are uniformly spaced, we know that the converter is linear rather than nonlinear. Since 0 is not a quantization level, we know that the converter is symmetric rather than asymmetric. Lastly, since quantized samples lie in the middle of each quantization bin rather than at the edge, we know that the converter is rounding instead of truncating. Thus, the converter is

linear, symmetric, and rounding.

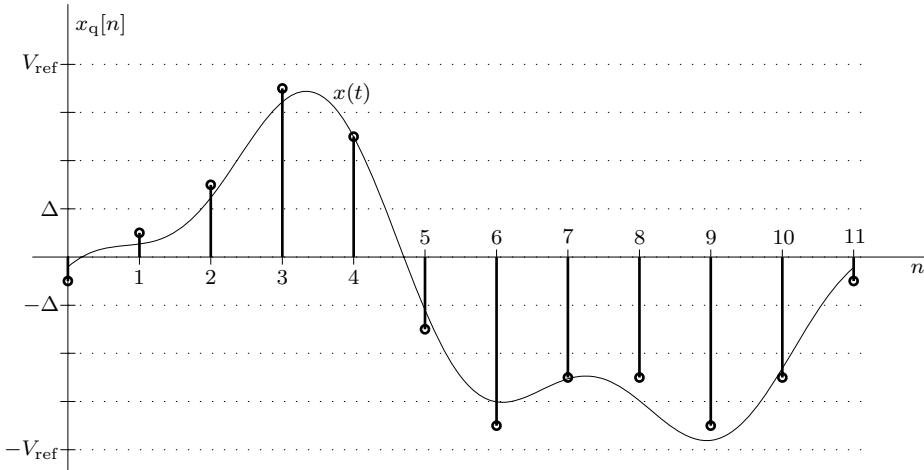


Figure D3.13

Drill 3.14 (Quantization Can Amplify Noise)

Whenever a signal stays for an extended period of time at a level that is also a quantization boundary, any amount of noise, no matter how small, will cause the sample to jitter in random fashion from one side to the other. In essence, the quantization process acts as a low-noise amplifier whenever the nominal state of the signal is also a quantization boundary.

Such is the situation for an asymmetric truncating converter when $x(t)$ is nominally zero. An asymmetric rounding converter, on the other hand, does not include 0 as a quantization boundary and will thus tend to suppress low-level noise when the desired signal $x(t)$ is zero. Using similar reasoning, a truncating symmetric converter is preferred in this situation over a rounding symmetric converter since the former does not include 0 as a quantization boundary.

Drill 3.15 (ADC Errors Reduce Dynamic Range)

For a 16-bit converter, each quantization level is $1/2^{16}$ of full scale, which provides a dynamic range of $20 \log_{10}(2^{16}) = 96.3296$ dB. For a 16-bit converter with an ENOB of 12.25, each quantization level is effectively $1/2^{12.25}$ of full scale, which reduces the dynamic range to $20 \log_{10}(2^{12.25}) = 73.7523$ dB. Thus, a 16-bit converter with an ENOB of 12.25 loses $96.3296 - 73.7523 = 22.5772$ dB of dynamic range.

Drill 3.16 (Flash Converters)

Since the converter is asymmetric, zero must be a quantization level. Further, since the reference voltages are balanced ($\pm V_{\text{ref}}$) and there are an even number ($L = 2^B$) of equal-valued resistors R connected in series, the midpoint of the resistor chain, and thus one of the $L - 1$ quantization boundaries, is 0 volts. Of the two types of asymmetric converters, only the truncating converter

uses 0 as a quantization boundary. Thus, this flash converter is

asymmetric and truncating.

By changing R_1 to $\frac{3}{2}R$ and R_L to $\frac{1}{2}R$, all the quantization boundaries are effectively shifted by $\Delta/2$. Assuming that zero is still a quantization level, this has the net effect of changing the converter from a truncating type to a rounding type. A similar result is obtained, albeit with more effort and cost, by adjusting the upper and lower reference voltages to $(1 - 2^{-B})V_{\text{ref}}$ and $-(1 + 2^{-B})V_{\text{ref}}$, respectively.

Chapter 4

Drill 4.1 (DT Time Shifting)

To begin, we note that $x[n]$ in Fig. 4.3a is represented as

$$x[n] = (0.9)^n(u[n-3] - u[n-11]).$$

To left shift (advance) this signal by 3 units requires

$$y[n] = x[n+3] = (0.9)^{n+3}(u[n+3-3] - u[n+3-11]) = 0.729(0.9)^n(u[n] - u[n-8]).$$

The original signal $x[n]$ and its shift $y[n] = x[n+3]$ are shown in Fig. D4.1.

```
01 u = @(n) (n>=0); x = @(n) (0.9).^(n.*(u(n-3)-u(n-11))); y = @(n) x(n+3);
02 n = (-5:15); subplot(121); stem(n,x(n)); subplot(122); stem(n,y(n));
```

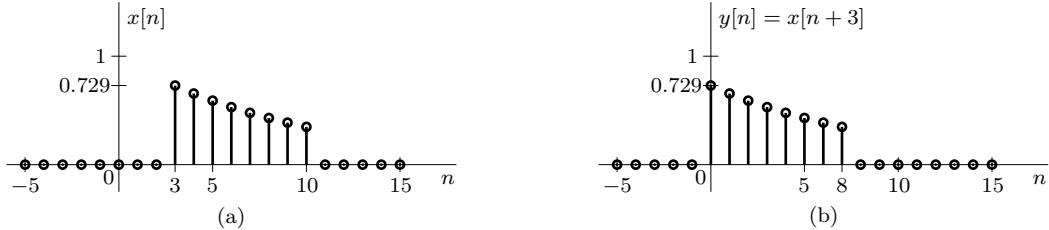


Figure D4.1

Drill 4.2 (DT Time Reversal)

In this case, $x[n] = e^{-n/2}(u[n+3] - u[n-3])$ and $y[n] = x[-n]$. Created using MATLAB, Fig. D4.2 illustrates both signals.

```
01 u = @(n) (n>=0); x = @(n) exp(-n/2).* (u(n+3)-u(n-3)); y = @(n) x(-n);
02 n = (-7:7); subplot(121); stem(n,x(n)); subplot(122); stem(n,y(n));
```

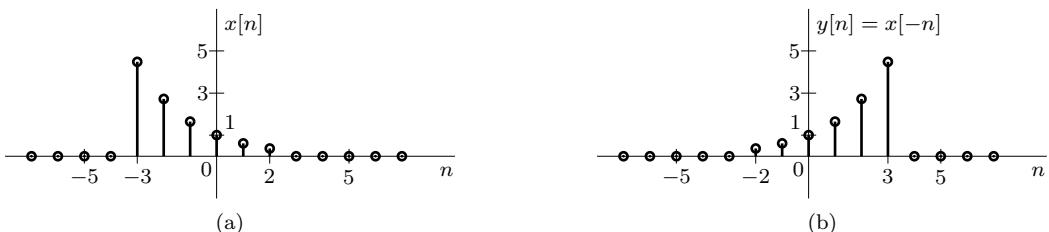


Figure D4.2

Drill 4.3 (Combined DT Time Shifting and Reversal)

To right shift (delay) the signal $x[n]$ by m units, we substitute n with $n - m$ as

$$x[n] \rightarrow x[n - m].$$

To time reverse this result, we substitute n with $-n$ as

$$x[n - m] \rightarrow x[-n - m].$$

Thus, $x[-n - m]$ can be obtained through a right shift by m followed by time reversal.

It is also possible to obtain the same result in a different manner. First, we time reverse the signal $x[n]$ as

$$x[n] \rightarrow x[-n].$$

Next, we left shift this result by m , replacing n with $n + m$, as

$$x[-n] \rightarrow x[-(n + m)] = x[-n - m].$$

Thus, $x[-n - m]$ can also be obtained through time reversal followed by a left shift by m .

Drill 4.4 (Preserving Odd-Numbered Samples during Compression)

Discrete-time signals, such as $x[n]$ and $y[n]$, are defined for all integer n . If we let $y[n] = x[2n]$, we see that

$$\dots, \underbrace{y[-1] = x[-2]}_{n=-1}, \underbrace{y[0] = x[0]}_{n=0}, \underbrace{y[1] = x[2]}_{n=1}, \underbrace{y[2] = x[4]}_{n=2}, \dots$$

In other words, $y[n] = x[2n]$ is just the even-numbered samples of $x[n]$.

Similarly, if we let $y[n] = x[2n + 1]$, we see that

$$\dots, \underbrace{y[-1] = x[-1]}_{n=-1}, \underbrace{y[0] = x[1]}_{n=0}, \underbrace{y[1] = x[3]}_{n=1}, \underbrace{y[2] = x[5]}_{n=2}, \dots$$

Clearly, $y[n] = x[2n + 1]$ is just the odd-numbered samples of $x[n]$.

Notice that the sequences $x[2n]$ and $x[2n + 1]$ contain all the original values of $x[n]$. Such a decomposition of $x[n]$ into its even- and odd-numbered samples is called a *two-band polyphase decomposition*.

Drill 4.5 (Expansion and Interpolation)

MATLAB provides a convenient tool to both expand and interpolate the signal $x[n] = \sin(2\pi n/4)$. To ensure that the expansion is performed correctly, that is to insert zeros for non-integer arguments of $x[n]$, notice the term `mod(n,1)==0` in line 01. This multiplier basically serves as an indicator of whether or not n is an integer. If n is not an integer, the expression evaluates to zero, as desired.

```
01 n = -10:10; x = @(n) sin(2*pi*n/4).*((mod(n,1)==0));
02 xup = @(n) x(n/2); xi = @(n) xup(n-1)/2+xup(n)+xup(n+1)/2;
03 subplot(311); stem(n,x(n)); subplot(312); stem(n,xup(n)); subplot(313); stem(n,xi(n));
```

The original signal $x[n]$ is shown in Fig. D4.5a, and the twofold expansion $x_\uparrow[n] = x[2n]$ is shown in Fig. D4.5b. There are many ways to interpolate an expanded signal. The approach of Fig. D4.5c is a *linear*, or straight-line, interpolation. While this method follows the underlying waveform better than the upsampled signal $x_\uparrow[n]$, we see that the interpolation is not perfect; in this case, a triangle wave is produced rather than the expected sinusoid.

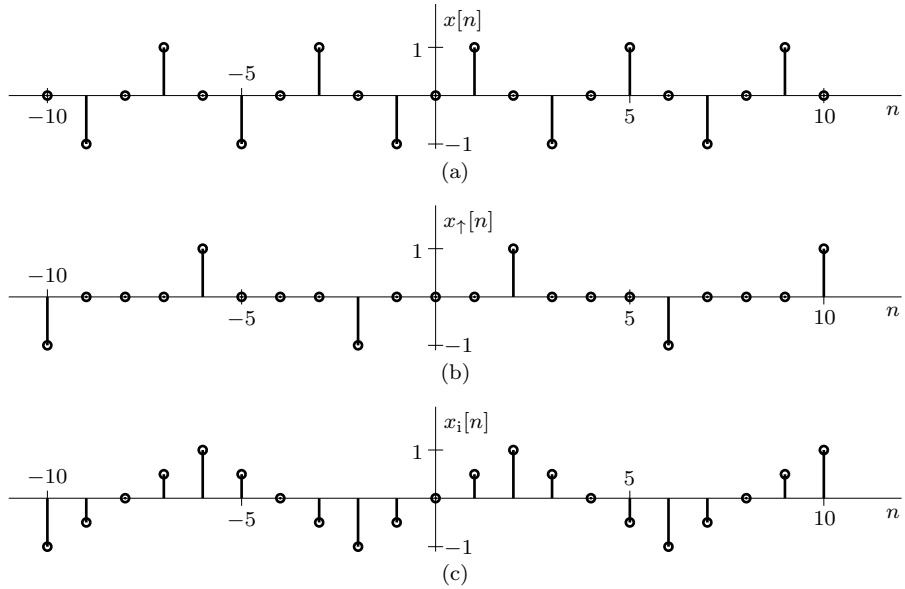


Figure D4.5

Drill 4.6 (Using Kronecker Delta Functions to Represent Expansion)

The expansion of signal $x[n]$ by factor L using Kronecker delta functions is stated as

$$x_{\uparrow}[n] = \sum_{m=-\infty}^{\infty} x[m] \delta[n - Lm].$$

Due to the $\delta[n - Lm]$ term, this sum has, at most, one nonzero term for a given n . A nonzero term only occurs when $n - Lm = 0$, or when n is an integer multiple of L . That is, when $n - Lm = 0$ or $m = n/L$, $\delta[n - Lm] = 1$, and the sum evaluates to $x[m] = x[n/L]$. Taken together, we see that

$$x_{\uparrow}[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{otherwise} \end{cases},$$

which is precisely the definition of expansion given in Eq. (4.6).

For the signal $x[n] = u[n + 4] - u[n - 5]$, which is 1 for $-4 \leq n \leq 4$ and 0 otherwise, we see from Eq. (4.13) that an $L = 4$ expansion is given as

$$x_{\uparrow}[n] = \sum_{m=-4}^4 \delta[n - 4m].$$

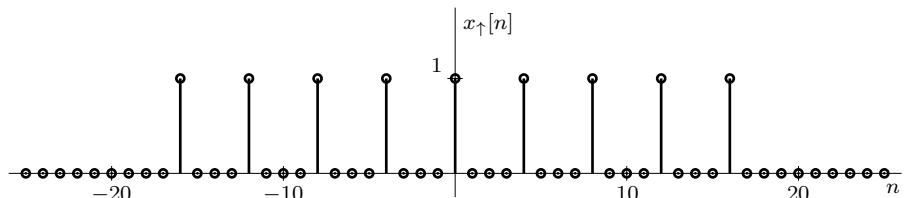


Figure D4.6

Although this signal is easy to sketch by hand, MATLAB also plots the result with very little effort (see Fig. D4.6).

```
01 delta = @n (n==0); xup = @n 0;
02 for m = -4:4, xup = @n xup(n) + delta(n-4*m); end
03 n = -25:25; stem(n,xup(n));
```

Drill 4.7 (Relating CT and DT Exponentials)

Sampling the CT exponentials using a sample interval of $T = 1$ yields

- | | | | |
|------------------|--------------|--------------------|------------------------|
| (a) $e^{0n} = 1$ | (b) e^n | (c) $e^{-0.6931n}$ | (d) $(-e^{-0.6931})^n$ |
| (e) 2^n | (f) 2^{-n} | (g) $e^{-n/4}$ | (h) $e^{j\pi n}$ |

Converting these DT exponentials into the form z^n yields

- | | | | |
|---------------------|-----------------------|------------------------|------------------------|
| (a) $z_a^n = 1^n$ | (b) $z_b^n = e^n$ | (c) $z_c^n = (0.5)^n$ | (d) $z_d^n = (-0.5)^n$ |
| (e) $z_e^n = (2)^n$ | (f) $z_f^n = (0.5)^n$ | (g) $z_g^n = (0.78)^n$ | (h) $z_h^n = (-1)^n$ |

Notice that cases (c) and (f) are identical.

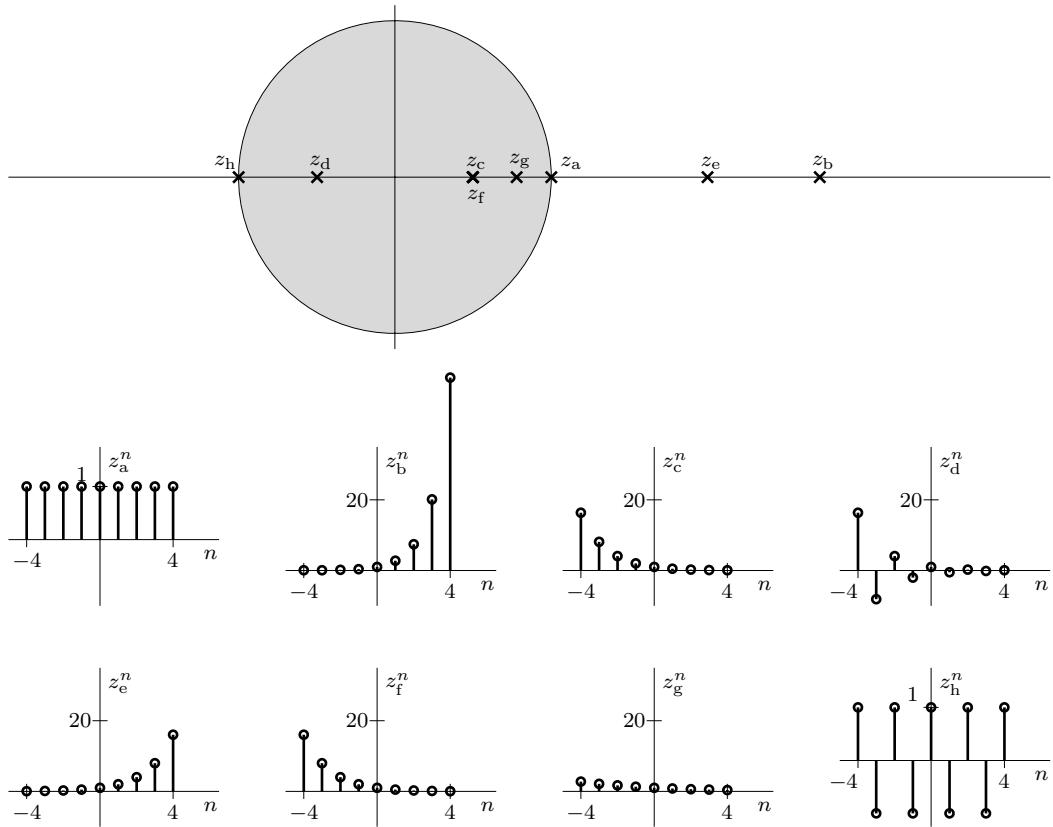


Figure D4.7

Figure D4.7 locates each constant z and plots the corresponding DT exponential. Cases (c), (d), (f), and (g) are located inside the unit circle and correspond to decaying exponentials. Cases (b) and (e) are located outside the unit circle and correspond to growing exponentials. Cases (a) and (h) are located on the unit circle and correspond to exponentials with constant envelopes. The

plots also confirm that the decay or growth rate of a DT exponential z^n is related to the distance of z from the unit circle. Values of z close to the unit circle decay or grow more slowly than those that are further away.

Drill 4.8 (Apparent Frequency of DT Sinusoids)

Using Eq. (4.17), we find the apparent frequencies of $\Omega = [2\pi, 3\pi, 5\pi, 3.2\pi, 22.1327, \pi + 2]$ as

$$\Omega_a = [0, -\pi, -\pi, -0.8\pi, -3, -\pi + 2].$$

Taking the absolute value yields the desired result of

$$|\Omega_a| = [0, \pi, \pi, 0.8\pi, 3, \pi - 2].$$

We know that a DT sinusoid of frequency Ω can be expressed in terms of $|\Omega_a|$ as

$$\cos(\Omega n + \theta) = \begin{cases} \cos(|\Omega_a|n + \theta) & \Omega_a \geq 0 \\ \cos(|\Omega_a|n - \theta) & \Omega_a < 0 \end{cases}.$$

Since $\Omega_a < 0$ in all but the first case, the phase changes sign for all but the first case. Interestingly, the cases with $\Omega_a = -\pi$ can also be represented as $\Omega_a = \pi$, which requires no sign change in phase (see Prob. 4.2-10).

Drill 4.9 (Exponentially-Varying DT Sinusoids)

Since $\langle 49\pi/6 + \pi \rangle_{2\pi} - \pi = \pi/6$, both $x_a[n]$ and $x_b[n]$ have the same apparent frequency of $\Omega_a = \pi/6$. Further, note that

$$x_a[n] = (0.9)^n \cos\left(\frac{\pi}{6}n - \frac{\pi}{3}\right) = 0.5e^{-j\pi/3}(0.9e^{j\pi/6})^n + 0.5e^{j\pi/3}(0.9e^{-j\pi/6})^n,$$

and

$$x_b[n] = (1.1)^n \cos\left(\frac{49\pi}{6}n - \frac{\pi}{3}\right) = 0.5e^{-j\pi/3}(1.1e^{j\pi/6})^n + 0.5e^{j\pi/3}(1.1e^{-j\pi/6})^n.$$

Thus, $x_a[n]$ requires two exponentials z_1^n and z_2^n , where

$$z_1 = 0.9e^{j\pi/6} \quad \text{and} \quad z_2 = 0.9e^{-j\pi/6}.$$

Similarly, $x_b[n]$ requires two exponentials z_1^n and z_2^n , where

$$z_1 = 1.1e^{j\pi/6} \quad \text{and} \quad z_2 = 1.1e^{-j\pi/6}.$$

Figure D4.9, generated in MATLAB, plots both signals.

```
01 xa = @(n) (0.9).^(n).*cos(pi*n/6-pi/3); xb = @(n) (1.1).^(n).*cos(49*pi*n/6-pi/3);
02 n = (-10:10); subplot(121); stem(n,xa(n)); subplot(122); stem(n,xb(n));
```

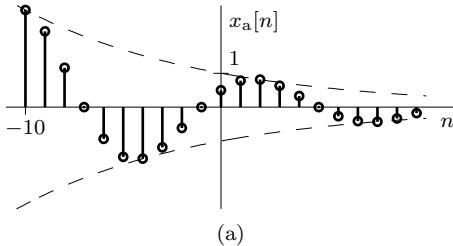
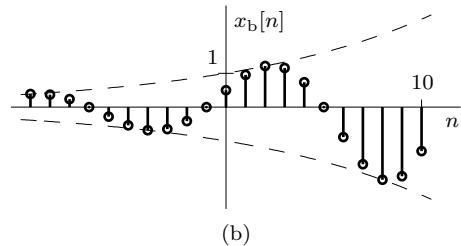


Figure D4.9



Drill 4.10 (Establishing the Periodicity of DT Sinusoids)

- (a) In this case, the sinusoid $\cos(\frac{3\pi}{7}n)$ is periodic since its frequency $\Omega = 3\pi/7 = 2\pi 3/14 = 2\pi m/N_0$ is a rational multiple of 2π . Thus,

$$N_0 = 14 \quad \text{and} \quad \Omega = 2\pi \frac{3}{14} \neq 2\pi \frac{1}{14} = \Omega_0.$$

- (b) The sinusoid $\cos(\frac{10}{7}n)$ is aperiodic since its frequency $\Omega = 10/7$ is not a rational multiple of 2π .
- (c) The sinusoid $\cos(\sqrt{\pi}n)$ is also aperiodic since its frequency $\Omega = \sqrt{\pi}$ is not a rational multiple of 2π .
- (d) In the final case, the sinusoid $\sin(2.35\pi n + 1)$ is periodic since its frequency $\Omega = 2.35\pi = 2\pi 47/40 = 2\pi m/N_0$ is a rational multiple of 2π . Thus,

$$N_0 = 40 \quad \text{and} \quad \Omega = 2\pi \frac{47}{40} \neq 2\pi \frac{1}{40} = \Omega_0.$$

Drill 4.11 (Energy and Power of a Causal DT Exponential)

For $|z| < 1$, $x[n] = z^n u[n]$ is a decaying causal exponential. The energy is computed as

$$E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 = \sum_{n=0}^{\infty} |z|^{2n} = \frac{1}{1 - |z|^2}.$$

Since $|z| < 1$, E_x is finite, and $x[n]$ is an energy signal ($P_x = 0$).

For $|z| = 1$, $x[n] = u[n]$ is a constant 1 for all $n \geq 0$. The power is computed as

$$P_x = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N |x[n]|^2 = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=0}^N 1 = \lim_{N \rightarrow \infty} \frac{N+1}{2N+1} = \frac{1}{2}.$$

Since the power is finite, $x[n]$ is a power signal ($E_x = \infty$).

For $|z| > 1$, $x[n] = z^n u[n]$ is a growing causal exponential. The power is computed as

$$P_x = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N |x[n]|^2 = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=0}^N |z|^{2n} = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \frac{1 - |z|^{N+1}}{1 - |z|^2} \rightarrow \infty.$$

Since the power is infinite, $x[n]$ is neither an energy signal nor a power signal.

Drill 4.12 (Approximating Derivatives)

We can approximate a first derivative $dy(t)/dt$ using forward differences as

$$\left. \frac{d}{dt} y(t) \right|_{t=nT} = \lim_{T \rightarrow 0} \frac{y([n+1]T) - y(nT)}{T} \approx \frac{1}{T} (y[n+1] - y[n]).$$

Thus, we can approximate a second derivative using forward differences by observing that

$$\begin{aligned} \left. \frac{d^2}{dt^2} y(t) \right|_{t=nT} &= \lim_{T \rightarrow 0} \frac{1}{T} \left(\left. \frac{d}{dt} y(t) \right|_{t=(n+1)T} - \left. \frac{d}{dt} y(t) \right|_{t=nT} \right) \\ &\approx \frac{1}{T} \left(\frac{y[n+2] - y[n+1]}{T} - \frac{y[n+1] - y[n]}{T} \right) \\ &= \frac{1}{T^2} (y[n+2] - 2y[n+1] + y[n]). \end{aligned}$$

This result is the same as Eq. (4.51) except for a shift by 2.

Drill 4.13 (Modeling Differential Equations with Difference Equations)

Using Eq. (4.52) to approximate each derivative, we approximate $\frac{d^2}{dt^2}y(t) + 3\frac{d}{dt}y(t) + 2y(t) = 1000x(t)$ as

$$\frac{1}{T^2}(y[n] - 2y[n-1] + y[n-2]) + \frac{3}{T}(y[n] - y[n-1]) + 2y[n] = x[n].$$

Substituting $T = 0.05$ yields

$$400y[n] - 800y[n-1] + 400y[n-2] + 60y[n] - 60y[n-1] + 2y[n] = 1000x[n].$$

Collecting terms and simplifying produce

$$y[n] - \frac{860}{462}y[n-1] + \frac{400}{462}y[n-2] = \frac{1000}{462}x[n].$$

For the initial conditions, we note that $y(0) = 0 = y[0]$ and $\dot{y}(0) = 3 \approx \frac{1}{T}y[0] - y[-1] = -20y[-1]$. Thus, the initial conditions are

$$y[0] = 0 \quad \text{and} \quad y[-1] = -3/20.$$

Drill 4.14 (Linear and Time-Invariant Systems)

Since the accumulator, the first-order backward difference, and the first-order forward difference are all special cases of the first-order difference equation $a_0y[n+1] + a_1y[n] = b_0x[n+1] + b_1x[n]$, it is sufficient to demonstrate the linearity and time invariance of this general equation.

To demonstrate linearity, notice that this case is nearly identical to the first-order difference equation treated in Ex. 4.13; the only difference is that the coefficient to $y[n+1]$ is now a_0 rather than 1. The additional coefficient a_0 is irrelevant to the proof of linearity given in Ex. 4.13, so we know that these first-order systems are linear.

To demonstrate time invariance, notice that the system operation $x[n] \rightarrow y[n]$ is represented by

$$a_0y[n+1] + a_1y[n] = b_0x[n+1] + b_1x[n].$$

Shifting this result by m yields

$$a_0y[n-m+1] + a_1y[n-m] = b_0x[n-m+1] + b_1x[n-m].$$

This also represents how the system responds to a time-shifted input; that is, $x[n-m] \rightarrow y[n-m]$. Since the system and the time-shifting operation commute, the system is time invariant. It is also possible to establish that the system is time invariant by noting that the system parameters (a_0 , a_1 , b_0 , and b_1) are not functions of time.

Drill 4.15 (Linear or Time-Invariant Systems)

- (a) Let $y_1[n] = nx_1[n]$ and $y_2[n] = nx_2[n]$. Multiplying these two equations by c_1 and c_2 , respectively, and then adding yield

$$c_1y_1[n] + c_2y_2[n] = c_1nx_1[n] + c_2nx_2[n] = n(c_1x_1[n] + c_2x_2[n]).$$

This equation also represents how the system responds to a sum of scaled inputs; that is, $c_1x_1[n] + c_2x_2[n] \rightarrow c_1y_1[n] + c_2y_2[n]$. Thus, the superposition property holds, and the system is linear.

Next, we shift $y[n] = nx[n]$ to produce $y[n-m] = (n-m)x[n-m]$. This result is not the same as $y[n-m] = nx[n-m]$, which reflects $x[n-m] \rightarrow y[n-m]$. Thus, the system is not time invariant. It is also possible to establish that the system is time variant by noting that the coefficient (system parameter) that multiplies $x[n]$ is, and therefore depends on, the time variable n .

- (b) Let $y_1[n] = |x_1[n]|^2$ and $y_2[n] = |x_2[n]|^2$. Multiplying these two equations by c_1 and c_2 , respectively, and then adding yield

$$c_1y_1[n] + c_2y_2[n] = c_1|x_1[n]|^2 + c_2|x_2[n]|^2.$$

This result is not the same as $c_1y_1[n] + c_2y_2[n] = |c_1x_1[n] + c_2x_2[n]|^2$, so the system is not linear.

Next, we shift $y[n] = |x[n]|^2$ to produce $y[n - m] = |x[n - m]|^2$. This result also represents how the system responds to a time-shifted input; that is, $x[n - m] \rightarrow y[n - m]$. Thus, the system is time invariant. Thought of another way, the system operation does not depend on the independent variable n .

Drill 4.16 (Causality of Backward and Forward Difference Systems)

Since both systems are LTI, causality is readily established from their respective impulse response functions. The first-order backward difference is given as $y[n] = x[n] - x[n - 1]$, so its impulse response is $h[n] = \delta[n] - \delta[n - 1]$. Since $h[n] = 0$ for $n < 0$, the first-order backward difference is causal.

The first-order forward difference is given as $y[n] = x[n + 1] - x[n]$, so its impulse response is $h[n] = \delta[n + 1] - \delta[n]$. Since $h[-1] = 1 \neq 0$, the first-order forward difference is not causal.

Drill 4.17 (Properties of a Time Reversal System)

Let $y_1[n] = x_1[-n]$ and $y_2[n] = x_2[-n]$. Multiplying these two equations by c_1 and c_2 , respectively, and then adding yield

$$c_1y_1[n] + c_2y_2[n] = c_1x_1[-n] + c_2x_2[-n].$$

This equation also represents how the system responds to a sum of scaled inputs; that is, $c_1x_1[n] + c_2x_2[n] \rightarrow c_1y_1[n] + c_2y_2[n]$. Thus, the superposition property holds, and the system is linear.

Next, consider the input $x[n] = u[n]$. The system output is $y[n] = u[-n]$. Clearly, the output precedes the input. Thus, the system is not causal.

Notice that the impulse response of this system is $h[n] = \delta[-n] = \delta[n]$. Even though $h[n] = 0$ for $n < 0$, we cannot conclude that the system is causal. To understand why, realize that we can test causality using $h[n]$ only for linear and time-invariant systems, and this system is not time invariant.

Drill 4.18 (Digital Differentiator Stability)

The digital differentiator of Ex. 4.9 is represented as $y[n] = \frac{1}{T}(x[n] - x[n - 1])$. If $|x[n]| \leq K_x < \infty$, then

$$|y[n]| = \left| \frac{1}{T}(x[n] - x[n - 1]) \right| \leq \frac{2}{T}K_x < \infty.$$

As long as $T > 0$, a bounded input is guaranteed to yield a bounded output. Therefore, the digital differentiator is BIBO stable.

The response of a DT differentiator to the unit step $x[n] = u[n]$ is $y[n] = \frac{1}{T}\delta[n]$ (see Eq. (4.12)). This is a bounded output. The response of a CT differentiator to $x(t) = u(t)$ is similar: $y(t) = \dot{x}(t) = \delta(t)$ (see Eq. (1.8)). However, this is not a bounded output. Although a digital differentiator is BIBO stable, a CT differentiator is not BIBO stable.

Drill 4.19 (Static and Dynamic Systems)

- (a) The output at time $n + 1$ depends on the input one unit in the past at time n , which requires that the system possess memory. Further, the output does not depend on future inputs. Thus, the system $y[n + 1] = x[n]$ is dynamic and causal.

- (b) The output at time n depends on the input at the same time n , which indicates an instantaneous system. Thus, the system $y[n] = (n + 1)x[n]$ is static and causal.
- (c) The output at time $n + 1$ depends on the input at the same time $n + 1$, which indicates an instantaneous system. Thus, the system $y[n + 1] = x[n + 1]$ is static and causal.
- (d) The output at time $n - 1$ depends on the input at the future time n , which requires that the system possess memory. Thus, the system $y[n - 1] = x[n]$ is dynamic and noncausal.

Drill 4.20 (Fractional Sampling Rate Conversion)

The structure of Fig. 4.34a expands an input by L and then compresses the result by M . This changes the input sampling rate of F_s to $\frac{L}{M}F_s$. Choosing L and M to be coprime, the desired rate $0.6F_s$ requires $L = 3$ and $M = 5$. Figure D4.20a shows the input signal $x[n] = \cos(\Omega n)$, where $\Omega = 4\pi/30$. The $L = 3$ expander triples the number of points, shown as $x_{\uparrow}[n]$ in Fig. D4.20b. Lastly, an $M = 5$ compressor produces the waveform $x_{\uparrow\downarrow}[n]$, shown in Fig. D4.20c, which has 60% as many points as the original.

```

01 x = @(n) cos(4*pi*n/30).*mod(n,1)==0;
02 L = 3; M = 5; xup = @(n) x(n/L); xupdown = @(n) xup(n*M);
03 n = 0:30; subplot(311); stem(n,x(n));
04 n = 0:30*L; subplot(312); stem(n,xup(n));
05 n = 0:30*L/M; subplot(313); stem(n,xupdown(n));

```

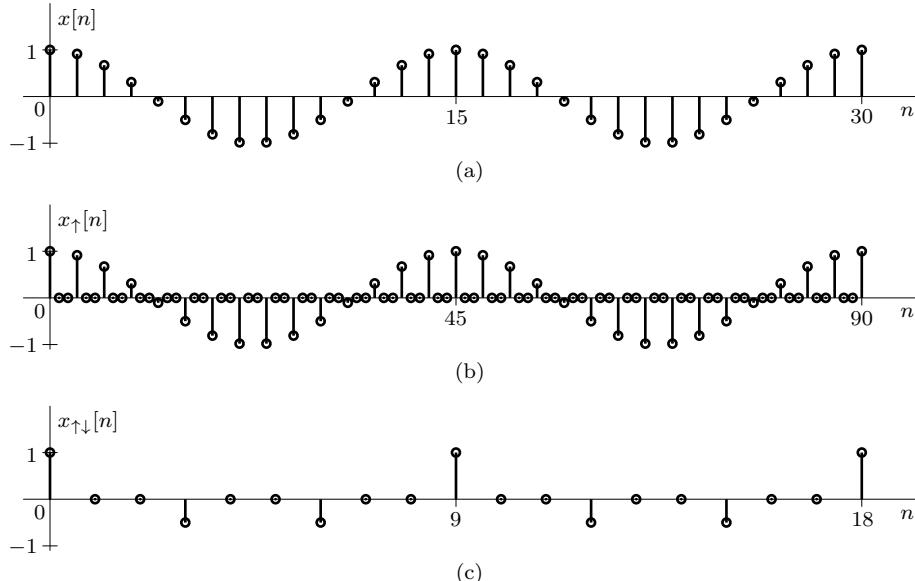


Figure D4.20

Chapter 5

Drill 5.1 (Iterative Solution to a Difference Equation)

Following the form of Eq. (5.1), the system is expressed as

$$y[n] = 2y[n - 1] + x[n - 1].$$

To begin, we set $n = 0$ in this equation and use $y[-1] = 10$ and $x[0] = 2u[-1] = 0$ to obtain

$$y[0] = 2(10) + 0 = 20.$$

Next, we set $n = 1$ and use the values $y[0] = 20$ and $x[1] = 2u[0] = 2$ to obtain

$$y[1] = 2(20) + 2 = 42.$$

Proceeding in this way iteratively, we obtain

$$\begin{aligned} y[2] &= 2(42) + 2 = 86 \\ y[3] &= 2(86) + 2 = 174 \end{aligned}$$

$$\vdots$$

Drill 5.2 (Solution to a Nonrecursive Difference Equation)

Since the system is nonrecursive, the outputs $y[0]$ and $y[1234]$ are computed through direct substitution. Thus,

$$\begin{aligned} y[0] &= \frac{1}{5} (\cos(\pi)u[2] + \cos(\pi/2)u[1] + \cos(0)u[0] + \cos(-\pi/2)u[-1] + \cos(-\pi)u[-2]) \\ &= \frac{1}{5}(-1 + 0 + 1 + 0 + 0) = 0, \end{aligned}$$

and

$$\begin{aligned} y[1234] &= \frac{1}{5} (\cos(\pi 1236/2)u[1236] + \cos(\pi 1235/2)u[1235] + \cos(\pi 1234/2)u[1234] + \\ &\quad + \cos(-\pi 1233/2)u[1233] + \cos(-\pi 1232/2)u[1232]) \\ &= \frac{1}{5}(1 + 0 - 1 + 0 + 1) = \frac{1}{5}. \end{aligned}$$

Drill 5.3 (Computing the Zero-Input Response)

- (a) Here, the characteristic equation is $\gamma - 0.8 = 0$, so $\gamma = 0.8$ is a characteristic root. Thus, the zero-input response is of the form

$$y[n] = c(0.8)^n.$$

Using $y[-1] = 10$, we find that $10 = c/(0.8)$ or that $c = 8$. Thus, the ZIR is

$$y[n] = 8(0.8)^n.$$

This result, shown in Fig. D5.3a, is confirmed iteratively using MATLAB.

```
01 n = (0:10); stem(n,8*(0.8).^n);
02 n = (-1:2); y = zeros(size(n)); y(n<0) = 10;
03 for ind = find(n>=0), y(ind) = 0.8*y(ind-1); end; y(n>=0)
ans = 8.0000 6.4000 5.1200
```

- (b) Here, the characteristic equation is $\gamma + 0.8 = 0$, so $\gamma = -0.8$ is a characteristic root. Thus, the zero-input response is of the form

$$y[n] = c(-0.8)^n.$$

Using $y[-1] = 10$, we find that $10 = c/(-0.8)$ or that $c = -8$. Thus, the ZIR is

$$y[n] = -8(-0.8)^n.$$

This result, shown in Fig. D5.3b, is confirmed iteratively using MATLAB.

```

01 n = (0:10); stem(n,-8*(-0.8).^n);
02 n = (-1:2); y = zeros(size(n)); y(n<0) = 10;
03 for ind = find(n>=0), y(ind) = -0.8*y(ind-1); end; y(n>=0)
ans = -8.0000  6.4000 -5.1200

```

- (c) Here, the characteristic equation is $\gamma^2 + 0.3\gamma - 0.1 = (\gamma + 0.5)(\gamma - 0.2)$. Thus, $\gamma_1 = -0.5$ and $\gamma_2 = 0.2$ are the characteristic roots, and the zero-input response is of the form

$$y[n] = c_1(-0.5)^n + c_2(0.2)^n.$$

Using $y[-1] = 1$ and $y[-2] = 33$, we see that

$$\begin{bmatrix} -2 & 5 \\ 4 & 25 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 33 \end{bmatrix}.$$

```

01 c = inv([-2 5;4 25])*(1;33)
c =
1

```

Thus, the zero-input response is

$$y[n] = 2(-0.5)^n + (0.2)^n.$$

This result, shown in Fig. D5.3c, is confirmed iteratively using MATLAB.

```

02 n = (0:10); stem(n,2*(-0.5).^n+(0.2).^n);
03 n = (-2:2); y = zeros(size(n)); y(n<0) = [33 1];
04 for ind = find(n>=0), y(ind) = -0.3*y(ind-1)+0.1*y(ind-2); end; y(n>=0)
ans = 3.0000 -0.8000 0.5400

```

- (d) Here, the characteristic equation is $\gamma^2 + 4 = (\gamma - 2j)(\gamma + 2j)$. Thus, $\gamma_1 = 2j = 2e^{j\pi/2}$ and $\gamma_2 = -2j = 2e^{-j\pi/2}$ are the characteristic roots, and the zero-input response is of the form

$$y[n] = c_1(2j)^n + c_1^*(-2j)^n.$$

Using $y[-1] = -\frac{1}{2\sqrt{2}}$ and $y[-2] = \frac{1}{4\sqrt{2}}$, we see that

$$\begin{bmatrix} -j0.5 & j0.5 \\ -0.25 & -0.25 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2\sqrt{2}} \\ \frac{1}{4\sqrt{2}} \end{bmatrix}.$$

```

01 c = inv([-1j*0.5 1j*0.5;-0.25 -0.25])*([-1/(2*sqrt(2));1/(4*sqrt(2))])
c =
-0.3536-0.3536i
-0.3536+0.3536i

```

Put in polar form, we find $c_1 = 0.5e^{-3\pi/4} = c_1^*$. Thus, the zero-input response is

$$y[n] = 0.5e^{-3\pi/4}(2j)^n + 0.5e^{3\pi/4}(-2j)^n.$$

Converting to real form, we obtain

$$y[n] = (2)^n \cos(\pi n/2 - 3\pi/4).$$

This result, shown in Fig. D5.3d, is confirmed iteratively using MATLAB.

```

02 n = (-2:10); stem(n,2.^n.*cos(pi*n/2-3*pi/4));
03 n = (-2:2); y = zeros(size(n)); y(n<0) = [1/(4*sqrt(2)) -1/(2*sqrt(2))];
04 for ind = find(n>=0), y(ind) = -4*y(ind-2); end; y(n>=0)
ans = -0.7071 1.4142 2.8284

```

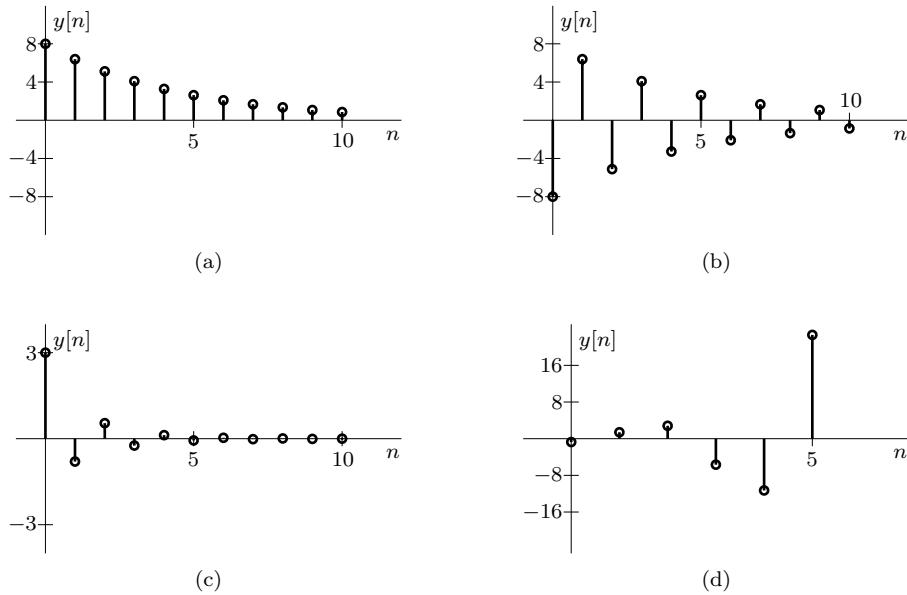


Figure D5.3

Drill 5.4 (Determining the Impulse Response)

- (a) In this case, the characteristic equation is $\gamma^2 - 5\gamma + 6 = (\gamma - 3)(\gamma - 2)$. Thus, the impulse response is of the form

$$h[n] = A_0\delta[n] + (c_1(3)^n + c_2(2)^n) u[n].$$

For this second-order ($K = 2$) system, $a_K = 6$ and $b_K = -19$. Thus, $A_0 = b_K/a_K = -\frac{19}{6}$. To find c_1 and c_2 , we first find $h[0]$ and $h[1]$ iteratively; that is,

$$h[0] = 5h[-1] - 6h[-2] + 8\delta[-1] - 19\delta[-2] = 0,$$

and

$$h[1] = 5h[0] - 6h[-1] + 8\delta[0] - 19\delta[-1] = 8.$$

Next, we substitute these values into the expression for $h[n]$.

$$\begin{aligned} -\frac{19}{6} + c_1 + c_2 &= 0 \\ 0 + 3c_1 + 2c_2 &= 8 \end{aligned} \quad \Rightarrow \quad \begin{bmatrix} 1 & 1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} \frac{19}{6} \\ 8 \end{bmatrix}.$$

Using the matrix representation, MATLAB easily computes the constants c_1 and c_2 .

```
01 c = inv([1 1;3 2])*[19/6;8]
c =
  1.6667
  1.5000
```

Thus,

$$h[n] = -\frac{19}{6}\delta[n] + \left(\frac{5}{3}(3)^n + \frac{3}{2}(2)^n\right) u[n].$$

- (b) In this case, the characteristic equation is $\gamma^2 - 4\gamma + 4 = (\gamma - 2)^2$. Thus, the impulse response is of the form

$$h[n] = A_0\delta[n] + (c_1(2)^n + c_2n(2)^n) u[n].$$

For this second-order ($K = 2$) system, $a_K = 4$ and $b_K = 0$. Thus, $A_0 = b_K/a_K = 0$. To find c_1 and c_2 , we first find $h[0]$ and $h[1]$ iteratively; that is,

$$h[0] = 4h[-1] - 4h[-2] + 2\delta[0] - 2\delta[-1] = 2,$$

and

$$h[1] = 4h[0] - 4h[-1] + 2\delta[1] - 2\delta[0] = 4(2) - 2 = 6.$$

Next, we substitute these values into the expression for $h[n]$.

$$\begin{aligned} c_1 + 0c_2 &= 2 \\ 2c_1 + 2c_2 &= 6 \end{aligned} \quad \Rightarrow \quad \begin{aligned} c_1 &= 2 \\ c_2 &= 1 \end{aligned}.$$

Thus,

$$h[n] = (2 + n)2^n u[n].$$

- (c) Since this is a nonrecursive system, the impulse response is directly as

$$h[n] = \delta[n] - 2\delta[n - 1].$$

Drill 5.5 (Analytic Computation of the Convolution Sum)

By definition, the convolution $y[n] = x[n] * h[n]$ is given as

$$\begin{aligned} y[n] &= \sum_{k=-\infty}^{\infty} x[k]h[n-k] \\ &= \sum_{k=-\infty}^{\infty} (0.8)^k u[k] u[n-k]. \end{aligned}$$

For $n < 0$, this sum evaluates to 0. For $n \geq 0$, we have

$$y[n] = \sum_{k=0}^n (0.8)^k = \frac{1 - (0.8)^{n+1}}{1 - 0.8}.$$

Thus,

$$y[n] = 5(1 - (0.8)^{n+1})u[n].$$

Changing $h[n]$ to $u[n - 1]$ rather than $u[n]$, we have

$$y[n] = \sum_{k=-\infty}^{\infty} (0.8)^k u[k] u[n-1-k].$$

For $n - 1 < 0$ or $n < 1$, this sum evaluates to 0. For $n \geq 1$, we have

$$y[n] = \sum_{k=0}^{n-1} (0.8)^k = \frac{1 - (0.8)^n}{1 - 0.8}$$

or

$$y[n] = 5(1 - (0.8)^n)u[n-1].$$

Thus, shifting $h[n]$ by one causes the output to shift by the same amount.

Drill 5.6 (Convolution Using Properties and Tables)

- (a) Noting that $(0.8)^{n+1}u[n] * u[n]$ is just $0.8(0.8)^n u[n] * u[n]$, scaling pair 5 from Table 5.2 yields the answer

$$0.8 \left(\frac{1 - (0.8)^{n+1}}{1 - 0.8} \right) u[n] = 4 (1 - (0.8)^{n+1}) u[n].$$

- (b) First, we consider the convolution $n(\frac{1}{3})^n u[n] * (\frac{1}{5})^n u[n]$. Using entry 9 of Table 5.2, this convolution is given as

$$\frac{\frac{1}{5}(\frac{1}{3})}{(\frac{1}{5} - \frac{1}{3})^2} \left((\frac{1}{3})^n + \frac{\frac{1}{3} - \frac{1}{5}}{\frac{1}{5}} n(\frac{1}{3})^n - (\frac{1}{5})^n \right) u[n]$$

or

$$\frac{15}{4} \left(3^{-n} - \frac{2}{3} n 3^{-n} - 5^{-n} \right) u[n].$$

Now, the desired convolution $n3^{-n}u[n] * (0.2)^n u[n-1]$ is just $n(\frac{1}{3})^n u[n] * \frac{1}{5}(\frac{1}{5})^{n-1} u[n-1]$, which is $n(\frac{1}{3})^n u[n] * (\frac{1}{5})^n u[n]$ scaled by $\frac{1}{5}$ and shifted by 1. Applying this scale factor and shift, the final result is thus

$$\frac{3}{4} \left(3^{-n+1} - \frac{2}{3} (n-1) 3^{-n+1} - 5^{-n+1} \right) u[n-1].$$

- (c) In this case, we first note from the distributive property that $(e^{-n}u[n] - j\delta[n+1]) * 2^{-n}u[n]$ is equal to $e^{-n}u[n] * 2^{-n}u[n] - j\delta[n+1] * 2^{-n}u[n]$. Using entries 8 and 1 of Table 5.2, the result is

$$2e \frac{e^{-n} - 2^{-n}}{2 - e} + j2^{-(n+1)} u[n+1].$$

Drill 5.7 (Graphical Procedure for the Convolution Sum)

To begin, we plot both $x[m]$ and $h[m]$, as shown in Figs. D5.7a and D5.7b. These plots are identical to plots of $x[n]$ and $h[n]$, except that m replaces n . It is simpler to invert and shift $h[n]$ rather than $x[n]$, so we choose to follow the procedure using $x[m]$ and $h[n-m]$. To obtain $h[n-m]$, we first reflect $h[m]$ to obtain $h[-m]$, as shown in Fig. D5.7c. Next, we shift $h[-m]$ by n to obtain $h[n-m]$, as shown in Fig. D5.7d.

Expressed mathematically, we see that

$$x[m] = (0.8)^m u[m-1] \quad \text{and} \quad h[n-m] = u[n+3-m].$$

For $n+3 < 1$ ($n < -2$), there is no overlap between $x[m]$ and $h[n-m]$, as shown in Fig. D5.7e. Thus,

$$y[n] = 0 \quad \text{for } n < -2.$$

Figure D5.7f shows the general situation for $n \geq -2$. The two functions $x[m]$ and $h[n-m]$ overlap over the interval $1 \leq m \leq n+3$. Therefore, for $n \geq -2$,

$$y[n] = \sum_{m=1}^{n+3} x[m]h[n-m] = \sum_{m=1}^{n+3} (0.8)^m.$$

Using entry 1 of Table 5.1,

$$y[n] = 5 (0.8 - (0.8)^{n+4}) \quad \text{for } n \geq -2.$$

Combining the results for $n < -2$ and $n \geq -2$ yields

$$y[n] = 4 (1 - (0.8)^{n+3}) u[n+2].$$

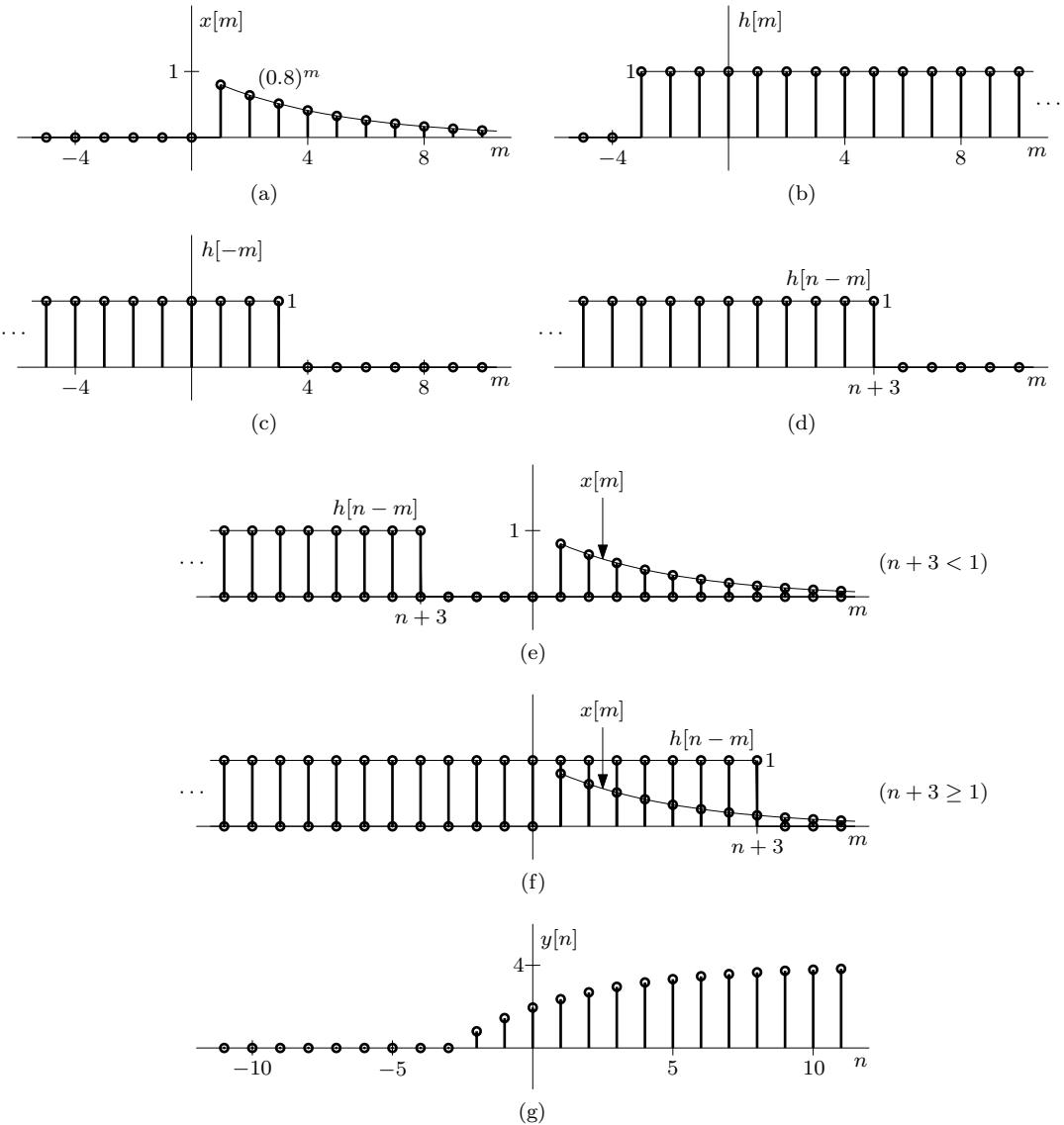


Figure D5.7

Drill 5.8 (Sliding-Tape Procedure for the Convolution Sum)

We begin by representing $x[m]$ and $h[m]$ as number sequences on tapes, as shown in Figs. D5.8a and D5.8b. Since the variable m does not appear on these tapes, we shade $m = 0$ for reference.

The $h[-m]$ tape of Fig. D5.8c is obtained by inverting the $h[m]$ tape about the origin ($m = 0$). Shifting the inverted tape by n slots yields $h[n-m]$, as shown in Fig. D5.8d.

To obtain $y[n] = x[n] * h[n]$, we slide the $h[n-m]$ tape along the $x[m]$ tape, multiply the values of adjacent slots, and add all the products. For $n+1 \leq -3$ ($n \leq -4$), the $x[m]$ and $h[n-m]$ tapes do not overlap, and $y[n]$ is therefore 0. Figure D5.8e illustrates the $n+1 = -3$ case, just before the $h[n-m]$ tape begins to overlap the $x[m]$ tape. At $n+1 = -2$, the two tapes begin to overlap (Fig. D5.8f). In this case,

$$y[-3] = (1 \times 1) = 1.$$

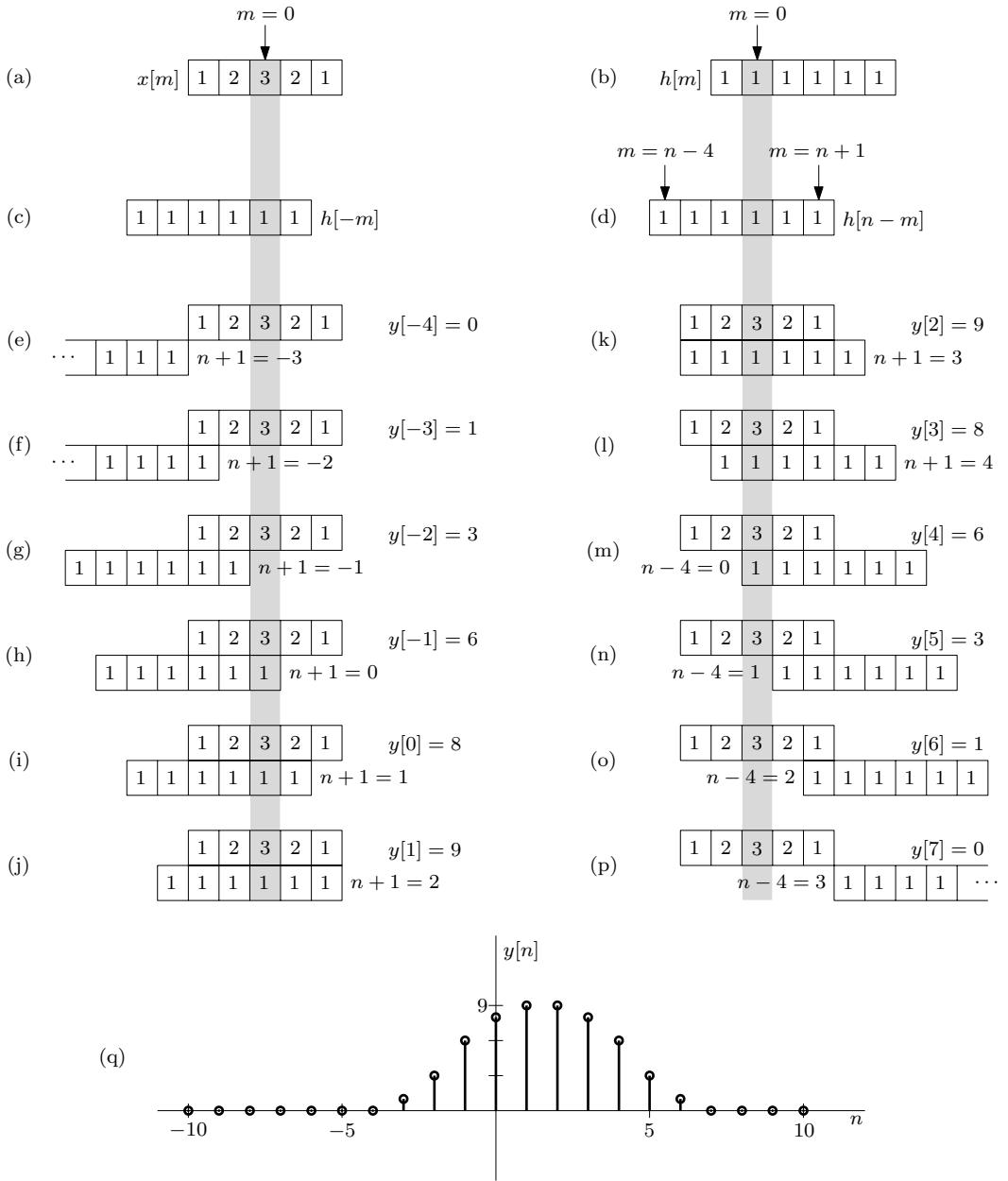


Figure D5.8

At $n+1 = -1$ (Fig. D5.8g), two slots overlap, and

$$y[-2] = (1 \times 1) + (2 \times 1) = 3.$$

Continuing this process (Figs. D5.8h through D5.8o), we obtain the values of $y[n]$ up to $n-4 = 2$ ($n = 6$). For $n-4 \geq 3$ ($n \geq 7$), the two tapes again have no overlap, and $y[n]$ again becomes zero. Viewed in sequence, Figs. D5.8e through D5.8p show $h[n-m]$ sliding forward, one slot at a time, to produce $y[n]$, one value of n at a time. The width of $x[n]$ is 4, and the width of $h[n]$ is 5. From the width property, the width of $y[n]$ is thus $4 + 5 = 9$, a fact that is confirmed by the Fig. D5.8q plot of $y[n]$.

Drill 5.9 (Polynomial Multiplication by Convolution)

The expression $(x^4 + 2x^2 + 3)^2(4x - 2 + 3x^{-1})$ is a product of three polynomial pieces, so its expansion requires two convolution operations. To avoid tedious hand calculations, we compute the convolutions using MATLAB's `conv` command.

```
01 conv(conv([1 0 2 0 3],[1 0 2 0 3]),[4 -2 3])
ans = 4 -2 19 -8 52 -20 78 -24 72 -18 27
```

Since the highest-power term in this expansion is x^9 , the final expansion is thus

$$4x^9 - 2x^8 + 19x^7 - 8x^6 + 52x^5 - 20x^4 + 78x^3 - 24x^2 + 72x - 18 + 27x^{-1}.$$

Drill 5.10 (Transfer Functions of LTID Systems)

- (a) In operator notation, the digital differentiator is

$$E\{y[n]\} = \left(\frac{1}{T}E - \frac{1}{T}\right)\{x[n]\}.$$

The transfer function $H(z)$ is thus

$$H(z) = \frac{Y(z)}{X(z)} = \frac{B(z)}{A(z)} = \frac{z-1}{Tz}.$$

- (b) Written in advance form, the unit delay system is

$$y[n+1] = x[n].$$

In operator notation, this becomes

$$E\{y[n]\} = x[n].$$

The transfer function $H(z)$ is thus

$$H(z) = \frac{B(z)}{A(z)} = \frac{1}{z}.$$

Drill 5.11 (Determining System Stability)

- (a) In this case, the characteristic polynomial is

$$A(\gamma) = (\gamma + 1)(\gamma^2 + 4\gamma + 5) = (\gamma + 1)(\gamma + 2 + j)(\gamma + 2 - j).$$

Figure D5.11a shows the three characteristic roots in the complex plane. Since the roots $\gamma = -2 \pm j$ are outside the unit circle, the system is both BIBO unstable and asymptotically unstable.

- (b) In this case, the characteristic polynomial is

$$A(\gamma) = (\gamma - 1)^2(\gamma + 0.5).$$

Figure D5.11b shows the three characteristic roots in the complex plane. Since a repeated root ($\gamma = 1$) exists on the unit circle, the system is both BIBO unstable and asymptotically unstable.

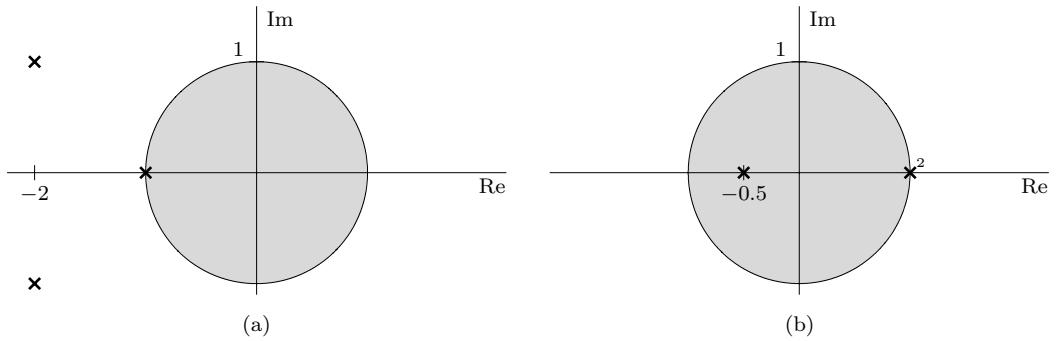


Figure D5.11

Drill 5.12 (Forced Response to a Sinusoid)

In this case,

$$H(\xi) = \frac{B(\xi)}{A(\xi)} = \frac{\xi + 0.32}{\xi^2 - \xi + 0.16}.$$

For the input $\cos(2n + \pi/3)u[n]$, the forced response is

$$y_\phi[n] = |H(e^{j2})| \cos[2n + \pi/3 + \angle H(e^{j2})] u[n],$$

where

$$H(e^{j2}) = \frac{e^{j2} + 0.32}{(e^{j2})^2 - e^{j2} + 0.16} = 0.548e^{-j2.990}.$$

Therefore,

$$y_\phi[n] = 0.548 \cos(2n + \pi/3 - 2.990)u[n] = 0.548 \cos(2n - 1.943)u[n].$$

Chapter 6

Drill 6.1 (Magnitude and Phase Spectra of a Two-Sided Exponential)

For $|\gamma| \geq 1$, the DTFT of $x[n]$ does not exist. For $|\gamma| < 1$, the DTFT of $x[n]$ is obtained using Eq. (6.1).

$$\begin{aligned} X(\Omega) &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} \\ &= \sum_{n=-\infty}^{-1} \gamma^{-n}e^{-j\Omega n} + \sum_{n=0}^{\infty} \gamma^n e^{-j\Omega n} \\ &= \sum_{n=-\infty}^{-1} (\gamma^{-1}e^{-j\Omega})^n + \sum_{n=0}^{\infty} (\gamma e^{-j\Omega})^n \\ &= \frac{0 - 1}{1 - \gamma^{-1}e^{-j\Omega}} + \frac{1 - 0}{1 - \gamma e^{-j\Omega}} \\ &= \frac{\gamma}{e^{-j\Omega} - \gamma} + \frac{e^{j\Omega}}{e^{j\Omega} - \gamma} \\ &= \frac{1 - \gamma^2}{1 - 2\gamma \cos(\Omega) + \gamma^2}, \quad |\gamma| < 1. \end{aligned}$$

Using $\gamma = 0.8$, Fig. D6.1 plots the resulting (real) spectrum $X(\Omega)$.

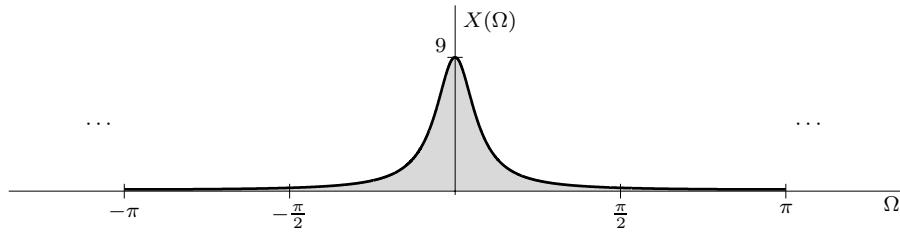


Figure D6.1

Drill 6.2 (Verification of DTFT Pairs)

To verify Eq. (6.11), we take the IDTFT (Eq. (6.2)) of $X(\Omega) = 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k)$ to obtain

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) e^{j\Omega n} d\Omega.$$

Within the region of integration ($-\pi$ to π), there is present only a single delta function, identified by some value $k = k_1$ (for $|\Omega_0| < \pi$, $k_1 = 0$). Thus, only one term in the summation is required, and our IDTFT expression simplifies to

$$x[n] = \int_{-\pi}^{\pi} \delta(\Omega - \Omega_0 - 2\pi k_1) e^{j\Omega n} d\Omega.$$

Using the sifting property of the impulse function (Eq. (1.6)), this integral evaluates to

$$x[n] = e^{j(\Omega_0 + 2\pi k_1)n} = e^{j\Omega_0 n} e^{j2\pi k_1 n} = e^{j\Omega_0 n}.$$

Thus, we have shown that the IDTFT of $2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k)$ is $e^{j\Omega_0 n}$, which confirms Eq. (6.11). To confirm Eq. (6.10), we simply repeat this procedure using $\Omega_0 = 0$.

Drill 6.3 (DTFT Pairs for Generalized Sinusoids)

Scaling pair 16 from CTFT Table 1.1 shows that $\frac{e^{j\theta}}{2} e^{j\omega_0 t}$ has a spectrum that is bandlimited to π if $\omega_0 \leq \pi$. This pair is given by

$$\frac{e^{j\theta}}{2} e^{j\omega_0 t} \iff e^{j\theta} \pi \delta(\omega - \omega_0).$$

Similarly,

$$\frac{e^{-j\theta}}{2} e^{-j\omega_0 t} \iff e^{-j\theta} \pi \delta(\omega + \omega_0).$$

Adding these two pairs together, we see that the CTFT of a generalized sinusoid is

$$\cos(\omega_0 t + \theta) \iff \pi (\delta(\omega - \omega_0) e^{j\theta} + \delta(\omega + \omega_0) e^{-j\theta}), \quad |\omega_0| < \pi.$$

We obtain the desired pair by substituting $t = n$ and $\omega = \Omega$ in this CTFT pair. As long as $|\Omega_0| < \pi$, this results in the DTFT pair

$$\cos(\Omega_0 n + \theta) \iff \pi (\delta(\Omega - \Omega_0) e^{j\theta} + \delta(\Omega + \Omega_0) e^{-j\theta}), \quad |\Omega_0| < \pi.$$

This describes the DTFT in the fundamental band only. Over the entire range of Ω , the DTFT pair is given by

$$\cos(\Omega_0 n + \theta) \iff \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) e^{j\theta} + \delta(\Omega + \Omega_0 - 2\pi k) e^{-j\theta}.$$

Using a similar procedure (or using the fact that $\sin(\Omega_0 n + \theta) = \cos(\Omega_0 n + \theta - \pi/2)$), we establish that

$$\sin(\Omega_0 n + \theta) \iff \frac{\pi}{j} \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) e^{j\theta} - \delta(\Omega + \Omega_0 - 2\pi k) e^{-j\theta}.$$

Drill 6.4 (Using the Time-Reversal Property)

From pair 15 of Table 6.2, we know that

$$\cos(\Omega_0 n) u[n] \iff \underbrace{\frac{e^{j2\Omega} - e^{j\Omega} \cos(\Omega_0)}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1}}_{A(\Omega)} + \frac{\pi}{2} [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)], \quad |\Omega| \leq \pi.$$

Using the time-reversal property, we know that

$$\cos(-\Omega_0 n) u[-n] \iff \underbrace{\frac{e^{-j2\Omega} - e^{-j\Omega} \cos(\Omega_0)}{e^{-j2\Omega} - 2 \cos(\Omega_0) e^{-j\Omega} + 1}}_{B(\Omega)} + \frac{\pi}{2} [\underbrace{\delta(-\Omega - \Omega_0)}_{\delta(\Omega + \Omega_0)} + \underbrace{\delta(-\Omega + \Omega_0)}_{\delta(\Omega - \Omega_0)}], \quad |\Omega| \leq \pi.$$

Further, we know that

$$-\delta[n] \iff -1.$$

Since $\cos(-\Omega_0 n) u[-n] = \cos(\Omega_0 n) u[-n]$, adding these three expressions yields

$$\cos(\Omega_0 n) \iff A(\Omega) + B(\Omega) - 1 + \pi [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)].$$

However, we note that

$$\begin{aligned} B(\Omega) - 1 &= \frac{e^{-j2\Omega} - e^{-j\Omega} \cos(\Omega_0)}{e^{-j2\Omega} - 2 \cos(\Omega_0) e^{-j\Omega} + 1} - 1 \\ &= \frac{e^{-j2\Omega} - e^{-j\Omega} \cos(\Omega_0) - (e^{-j2\Omega} - 2 \cos(\Omega_0) e^{-j\Omega} + 1)}{e^{-j2\Omega} - 2 \cos(\Omega_0) e^{-j\Omega} + 1} \\ &= \frac{e^{-j\Omega} \cos(\Omega_0) - 1}{e^{-j2\Omega} - 2 \cos(\Omega_0) e^{-j\Omega} + 1} \left(\frac{e^{j2\Omega}}{e^{j2\Omega}} \right) \\ &= \frac{e^{j\Omega} \cos(\Omega_0) - e^{j2\Omega}}{e^{j2\Omega} - 2 \cos(\Omega_0) e^{j\Omega} + 1} \\ &= -A(\Omega). \end{aligned}$$

Thus, $A(\Omega) + B(\Omega) - 1 = 0$, and

$$\cos(\Omega_0 n) \iff \pi [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)], \quad |\Omega| \leq \pi.$$

This is precisely pair 13 of Table 6.2.

Drill 6.5 (Using the Time-Shifting Property)

From pair 7 of Table 6.1, we know that

$$u[n] - u[n - L_x] \iff \frac{\sin(L_x \Omega / 2)}{\sin(\Omega / 2)} e^{-j\Omega(L_x - 1)/2}.$$

Assuming that L_x is odd and using the time-shifting property, left shifting this signal by $\frac{L_x - 1}{2}$ yields

$$u[n + (L_x - 1)/2] - u[n - L_x + (L_x - 1)/2] \iff \frac{\sin(L_x \Omega / 2)}{\sin(\Omega / 2)} e^{-j\Omega(L_x - 1)/2} e^{j\Omega(L_x - 1)/2}.$$

Simplifying, we obtain

$$u[n + (L_x - 1)/2] - u[n - (L_x + 1)/2] \iff \frac{\sin(L_x\Omega/2)}{\sin(\Omega/2)}.$$

As hoped, this is the DTFT pair found in Ex. 6.1.

Drill 6.6 (Using the Frequency-Shifting Property)

From pair 10 of Table 6.1, we know that

$$1 \iff 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\pi k).$$

From the frequency-shifting property, we know that replacing Ω with $\Omega - \Omega_0$ in the frequency domain causes multiplication by $e^{j\Omega_0 n}$ in the time domain. Applying the frequency-shifting property to pair 10 thus yields

$$e^{j\Omega_0 n} \iff 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k).$$

This is pair 12 of Table 6.1. Similarly, replacing Ω with $\Omega + \Omega_0$ in the frequency domain causes multiplication by $e^{-j\Omega_0 n}$ in the time domain. Thus,

$$e^{-j\Omega_0 n} \iff 2\pi \sum_{k=-\infty}^{\infty} \delta(\Omega + \Omega_0 - 2\pi k).$$

Adding these two expressions and dividing by 2 yield

$$\cos(\Omega_0 n) \iff \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k).$$

This is pair 13 of Table 6.1.

Drill 6.7 (Using the Frequency-Domain Convolution Property)

We solve this problem in the fundamental band first and then periodically extend the result. Assuming $|\Omega_0| < \pi$ (as given), pairs 11 and 13 in Table 6.2 are

$$u[n] \iff \frac{e^{j\Omega}}{e^{j\Omega} - 1} + \pi\delta(\Omega)$$

and

$$\cos(\Omega_0 n) \iff \pi [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)].$$

Multiplying the two time-domain expressions yields $x[n] = \cos(\Omega_0 n)u[n]$. Using the frequency-domain convolution property, the spectrum $X(\Omega)$ found through convolution is

$$X(\Omega) = \frac{1}{2\pi} \int_{2\pi} \pi [\delta(\Omega - \lambda - \Omega_0) + \delta(\Omega - \lambda + \Omega_0)] \left[\frac{e^{j\lambda}}{e^{j\lambda} - 1} + \pi\delta(\lambda) \right] d\lambda.$$

Using the sifting property of the delta function, this integral evaluates to

$$X(\Omega) = \frac{1}{2} \frac{e^{j(\Omega - \Omega_0)}}{e^{j(\Omega - \Omega_0)} - 1} + \frac{1}{2} \frac{e^{j(\Omega + \Omega_0)}}{e^{j(\Omega + \Omega_0)} - 1} + \frac{\pi}{2} [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)].$$

Combining the first two terms yields, after some simplification,

$$X(\Omega) = \frac{e^{j2\Omega} - e^{j\Omega} \cos(\Omega_0)}{e^{j2\Omega} - 2e^{j\Omega} \cos(\Omega_0) + 1} + \frac{\pi}{2} [\delta(\Omega - \Omega_0) + \delta(\Omega + \Omega_0)].$$

This is pair 15 in Table 6.2. Periodically extending this result with period 2π yields pair 15 in Table 6.1.

Drill 6.8 (Practical LPF Design)

The desired LPF is obtained with only minor adjustments to the code used to produce Fig. 6.16.

```

01 Omega1 = 2*pi/3; ng = 5; n = -5:15; Omega = linspace(-pi,pi,5001);
02 hhat = @(n) wc/pi*sinc(Omega1*(n-ng)/pi).*((n>=0)&(n<=2*ng));
03 Hhat = @(Omega) polyval(hhat(0:2*ng),exp(1j*Omega))./exp(1j*Omega*(2*ng));
04 subplot(311); stem(n,hhat(n));
05 subplot(312); plot(Omega,abs(Hhat(Omega)));
06 subplot(313); plot(Omega,unwrap(angle(Hhat(Omega))));
```

The result is shown in Fig. D6.8. Although this LPF does not have ideal characteristics, it is a realizable filter. Clearly, the LPF cutoff frequency is correctly located at $\Omega_1 = 2\pi/3$. Compared with the filter of Fig. 6.16, this filter has a shorter-duration impulse response and thus causes less delay in producing an output. However, the price of this shorter delay is a poorer approximation of the ideal filter response, particularly near the transition bands of the filter.

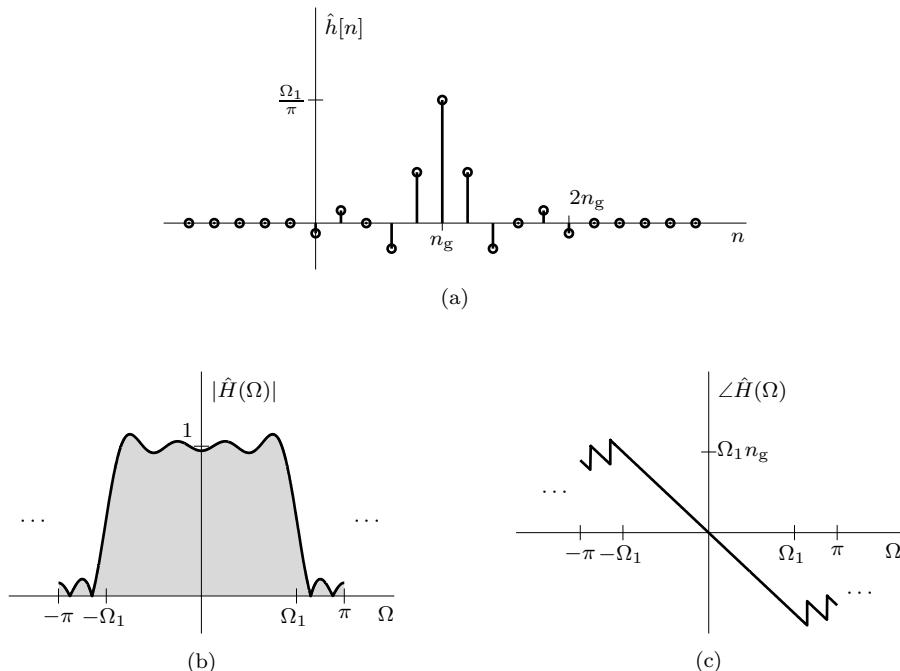


Figure D6.8

Drill 6.9 (Another Bandlimited Analog Delay)

Procedurally, the design of a bandlimited analog delay of $275 \mu\text{s}$ is identical to the $250\text{-}\mu\text{s}$ case of Ex. 6.20. Since both systems use inputs bandlimited to 10 kHz , the minimum possible sampling period is $T = 50 \mu\text{s}$ in both cases. Thus, in terms of sample periods, a delay of $275 \mu\text{s}$ equals $5.5T$ (a non-integer multiple of T). The impulse response of the DT filter is thus

$$h[n] = \text{sinc}[n - 5.5].$$

Unlike Ex. 6.20, this impulse response does not reduce to a simple delta function. Further, $h[n]$ is both everlasting and noncausal, which makes it impossible to realize the desired system in practice.

Truncation can render $h[n]$ causal, but it will necessarily distort the frequency response of the system to something other than a simple delay. Unless the desired delay is an integer multiple of the sampling frequency, the system of Fig. 6.21a cannot perfectly achieve a bandlimited analog delay.

It is also worth noting that there are differences in realizing different fractional delays. For example, a delay of $5.5T$ is more accurately realized than a delay of $0.5T$. The primary reason is that truncation of $h[n]$, which is needed to produce a causal system, eliminates more energy in the $0.5T$ case than in the $5.5T$ case. As more energy is eliminated, more distortion is introduced in the system response.

Drill 6.10 (Downsampling a Sinusoid)

Using Eq. (6.68), the spectrum of $x_{\downarrow}[n] = x[2n] = \cos(2\Omega_0 n)$ is given by

$$X_{\downarrow}(\Omega) = \frac{1}{2}X\left(\frac{\Omega}{2}\right) + \frac{1}{2}X\left(\frac{\Omega - 2\pi}{2}\right).$$

Using pair 13 of Table 6.1 for $X(\Omega)$, we obtain

$$\begin{aligned} X_{\downarrow}(\Omega) &= \frac{\pi}{2} \sum_{k=-\infty}^{\infty} \delta\left(\frac{\Omega}{2} - \Omega_0 - 2\pi k\right) + \delta\left(\frac{\Omega}{2} + \Omega_0 - 2\pi k\right) \\ &\quad + \delta\left(\frac{\Omega - 2\pi}{2} - \Omega_0 - 2\pi k\right) + \delta\left(\frac{\Omega - 2\pi}{2} + \Omega_0 - 2\pi k\right). \end{aligned}$$

Expressing each delta function with a common denominator of 2 yields

$$\begin{aligned} X_{\downarrow}(\Omega) &= \frac{\pi}{2} \sum_{k=-\infty}^{\infty} \delta\left(\frac{\Omega - 2\Omega_0 - 4\pi k}{2}\right) + \delta\left(\frac{\Omega + 2\Omega_0 - 4\pi k}{2}\right) \\ &\quad + \delta\left(\frac{\Omega - 2\Omega_0 - 4\pi k - 2\pi}{2}\right) + \delta\left(\frac{\Omega + 2\Omega_0 - 4\pi k - 2\pi}{2}\right). \end{aligned}$$

Using the hint that $\delta(ax) = \delta(x)/|a|$, we obtain

$$\begin{aligned} X_{\downarrow}(\Omega) &= \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\Omega_0 - 4\pi k) + \delta(\Omega + 2\Omega_0 - 4\pi k) \\ &\quad + \delta(\Omega - 2\Omega_0 - 4\pi k - 2\pi) + \delta(\Omega + 2\Omega_0 - 4\pi k - 2\pi). \end{aligned}$$

Combining terms, this reduces to

$$X_{\downarrow}(\Omega) = \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\Omega_0 - 2\pi k) + \delta(\Omega + 2\Omega_0 - 2\pi k).$$

Referring to pair 13 of Table 6.1, we recognize that this is the transform of $\cos(2\Omega_0 n)$, as expected. Lastly, we note that if $\Omega_0 > \pi/2$, aliasing occurs in the downsampled signal. In other words, if $\Omega_0 > \pi/2$, then the apparent frequency of $x_{\downarrow}[n] = x[2n] = \cos(2\Omega_0 n)$ will be something other than twice the frequency of $x[n]$.

Drill 6.11 (Interpolating a Sinusoid)

For convenience, we assume that $|2\Omega_0| < \pi$ ($|\Omega_0| < \pi/2$). Using pair 13 of Table 6.1, we know that

$$x[n] = \cos(2\Omega_0 n) \iff X(\Omega) = \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - 2\Omega_0 - 2\pi k) + \delta(\Omega + 2\Omega_0 - 2\pi k).$$

Next, we upsample this signal by factor $L = 2$. According to Eq. (6.74), the spectrum of the upsampled signal is

$$X_{\uparrow}(\Omega) = X(2\Omega) = \pi \sum_{k=-\infty}^{\infty} \delta(2\Omega - 2\Omega_0 - 2\pi k) + \delta(2\Omega + 2\Omega_0 - 2\pi k).$$

Noting that $\delta(ax) = \delta(x)/|a|$, this expression simplifies to

$$X_{\uparrow}(\Omega) = \frac{\pi}{2} \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - \pi k) + \delta(\Omega + \Omega_0 - \pi k).$$

We next represent this π -periodic expression as a sum of 2π -periodic components

$$X_{\uparrow}(\Omega) = \frac{\pi}{2} \sum_{k=-\infty}^{\infty} \underbrace{\delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k)}_{\text{desired spectrum}} + \underbrace{\delta(\Omega - \Omega_0 - \pi - 2\pi k) + \delta(\Omega + \Omega_0 - \pi - 2\pi k)}_{\text{undesired images}}.$$

Applying an ideal interpolation filter with gain $L = 2$ and cutoff frequency $\pi/2$ (Eq. (6.75)), the undesired images are removed to produce the final spectrum

$$X_i(\Omega) = X_{\uparrow}(\Omega)H_i(\Omega) = \pi \sum_{k=-\infty}^{\infty} \delta(\Omega - \Omega_0 - 2\pi k) + \delta(\Omega + \Omega_0 - 2\pi k).$$

This is identical to pair 13 of Table 6.1 for a sinusoid of frequency Ω_0 , as desired. Thus, $L = 2$ interpolation of $x[n] = \cos(2\Omega_0 n)$ results in $x_i[n] = \cos(\Omega_0 n)$.

Drill 6.12 (Frequency Response of a Linear Interpolator)

The impulse response of an ideal linear interpolator is given by Eq. (6.80) as

$$\hat{h}_i[n] = \Lambda\left(\frac{n}{2L}\right).$$

Defining $x[n] = u[n] - u[n - L]$, notice that

$$\hat{h}_i[n] = \frac{1}{L}x[n] * x[-n].$$

Using the convolution and time-reversal properties, the corresponding frequency response is

$$\hat{H}_i(\Omega) = \frac{1}{L}X(\Omega)X(-\Omega).$$

Using pair 7 of Table 6.1, we thus obtain

$$\hat{H}_i(\Omega) = \frac{1}{L} \frac{\sin(L\Omega/2)}{\sin(\Omega/2)} e^{-j\Omega(L-1)/2} \frac{\sin(-L\Omega/2)}{\sin(-\Omega/2)} e^{j\Omega(L-1)/2}.$$

Simplifying, we obtain the desired result of

$$\hat{H}_i(\Omega) = \frac{1}{L} \left[\frac{\sin(\Omega L/2)}{\sin(\Omega/2)} \right]^2.$$

Using $L = 4$, we use MATLAB to plot this frequency response and the frequency response for ideal interpolation. The results are shown in Fig. D6.12.

```
01 L = 4; Om = linspace(-2*pi,2*pi,5000);
02 Hihat = @(Om) ((sin(Om*L/2)./sin(Om/2)).^2)/L;
03 Hi = @(Om) L*((mod(Om,2*pi)<=pi/L)+(mod(Om,2*pi)>=2*pi-pi/L));
04 plot(Om,Hihat(Om),Om,Hi(Om))
```

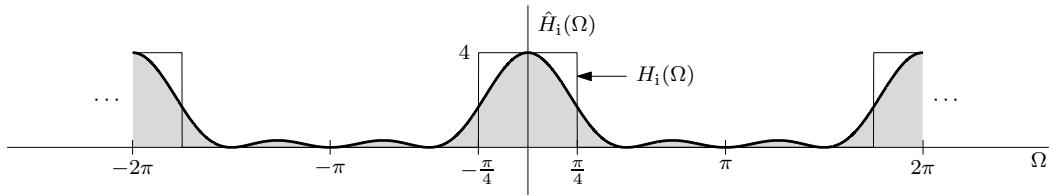


Figure D6.12

Drill 6.13 (Linear Interpolation)

To repeat Ex. 6.27 using the signal $x[n] = \cos(2\pi n/7)(u[n] - u[n - 7])$ rather than $x[n] = \sin(2\pi n/7)(u[n] - u[n - 7])$, we simply change line 01 in the MATLAB code for that example.

```

01 x = @n cos(2*pi*n/7).*(n>=0)&(n<7).*mod(n,1)==0; n = -2:8;
02 subplot(311); stem(n,x(n)); xlabel('n'); ylabel('x[n]');
03 xup = @n x(n/4); n2 = (n(1)*4:n(end)*4);
04 subplot(312); stem(n2,xup(n2)); xlabel('n'); ylabel('x_{\uparrow}[n]');
05 hi = [1 2 3 4 3 2 1]/4; n3 = (-3+n2(1):3+n2(end));
06 subplot(313); stem(n3,conv(xup(n2),hi)); xlabel('n'); ylabel('x_i[n]');

```

The results, shown in Fig. D6.13, are quite similar as those obtained in Fig. 6.34 for $x[n] = \sin(2\pi n/7)(u[n] - u[n - 7])$. The primary differences are seen near the boundaries of the waveforms. Since $x[n] = \sin(2\pi n/7)(u[n] - u[n - 7])$ is zero at both boundaries ($n = 0$ and $n = 7$), its interpolation turns out quite smooth, closely reflecting the waveform we expect. However, in the case of $x[n] = \cos(2\pi n/7)(u[n] - u[n - 7])$, the two boundary points (at $n = 0$ and $n = 7$) are nonzero. Consequently, transitions occur in the interpolation at these boundaries. These transitions, while necessary, do not seem quite so reflective of the expected (sinusoidal) waveform. To help minimize this behavior, signals are sometimes started and ended more gradually using some form of tapered window.

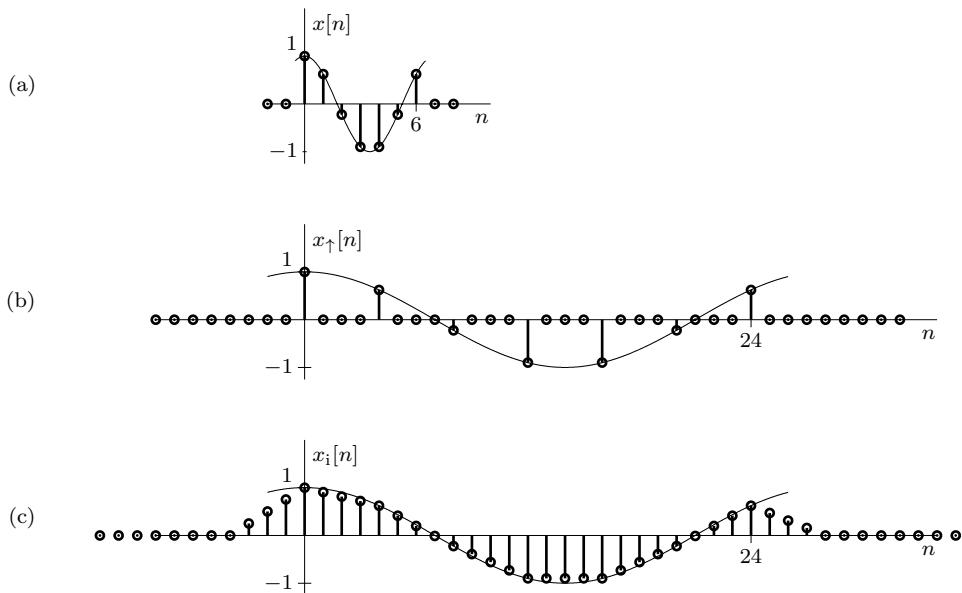


Figure D6.13

Chapter 7

Drill 7.1 (Unilateral z -Transforms)

- (a) Using Eq. (7.7), we see that

$$X_a(z) = \sum_{n=0}^{\infty} x[n]z^{-n} = \sum_{n=4}^9 z^{-n}.$$

Simplifying this geometric sum yields

$$X_a(z) = \frac{z^{-4} - z^{-10}}{1 - z^{-1}} = \frac{z}{z-1}(z^{-4} - z^{-10}).$$

Alternately, direct expansion yields

$$X_a(z) = \sum_{k=4}^9 z^{-k} = z^{-4} + z^{-5} + z^{-6} + z^{-7} + z^{-8} + z^{-9}.$$

Both expressions for $X_a(z)$ are mathematically equivalent. Since we are computing the unilateral z -transform, there is no requirement to specify that the ROC for $X_a(z)$ is $|z| > 0$.

- (b) Noting that $\gamma = \sqrt{2}$, $\beta = \pi/4$, and $\theta = -1.415$, pair 10 of Table 7.1 yields

$$X_b(z) = 20.65 \frac{z[z \cos(\theta) - |\gamma| \cos(\beta - \theta)]}{z^2 - 2|\gamma| \cos(\beta)z + |\gamma|^2} = \frac{z(3.2042z + 17.1957)}{z^2 - 2z + 2}.$$

Although unnecessary for this unilateral z -transform, the ROC of $X_b(z)$ is $|z| > \sqrt{2}$.

Drill 7.2 (Inverse Unilateral z -Transform by Partial Fraction Expansion)

- (a) Using MATLAB, we first expand $X_a(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([2 -1],poly([1 -0.5]))
r = 0.6667  1.3333
p = 1  -0.5
k = []
```

Thus,

$$\frac{X_a(z)}{z} = \frac{2z-1}{(z-1)(z+0.5)} = \frac{\frac{2}{3}}{z-1} + \frac{\frac{4}{3}}{z+0.5},$$

and

$$X_a(z) = \frac{\frac{2}{3}z}{z-1} + \frac{\frac{4}{3}z}{z+0.5}.$$

Using pair 3 of Table 7.1, we obtain

$$x_a[n] = \left[\frac{2}{3} + \frac{4}{3}(-0.5)^n \right] u[n].$$

- (b) Using MATLAB, we expand $X_b(z)$ into modified partial fractions.

```
02 [r,p,k] = residue([1],poly([1 -0.5 0]))
r = 0.6667  1.3333  -2.0000
p = 1  -0.5  0
k = []
```

Thus,

$$\frac{X_b(z)}{z} = \frac{1}{z(z-1)(z+0.5)} = \frac{\frac{2}{3}}{z-1} + \frac{\frac{4}{3}}{z+0.5} + \frac{-2}{z},$$

and

$$X_b(z) = \frac{\frac{2}{3}z}{z-1} + \frac{\frac{4}{3}z}{z+0.5} - 2.$$

Using pair 3 of Table 7.1, we obtain

$$x_b[n] = \left[\frac{2}{3} + \frac{4}{3}(-0.5)^n \right] u[n] - 2\delta[n].$$

(c) Using MATLAB, we expand $X_c(z)$ into modified partial fractions.

```
03 [r,p,k] = residue([9],poly([-2 0.5 0.5 0]))
r = -0.7200 -17.2800 7.200 18.0000
p = -2 0.5 0.5 0
k = []
```

Thus,

$$\frac{X_c(z)}{z} = \frac{9}{z(z+2)(z-0.5)^2} = \frac{-0.72}{z+2} + \frac{-17.28}{z-0.5} + \frac{7.2}{(z-0.5)^2} + \frac{18}{z},$$

and

$$X_c(z) = \frac{-0.72z}{z+2} + \frac{-17.28z}{z-0.5} + \frac{7.2z}{(z-0.5)^2} + 18.$$

Using pairs 3 and 5 of Table 7.1, we obtain

$$x_c[n] = [-0.72(-2)^n - 17.28(0.5)^n + 14.4n(0.5)^n] u[n] + 18\delta[n].$$

(d) Using MATLAB, we expand $X_d(z)$ into modified partial fractions.

```
04 [r,p,k] = residue([5 -5],[1 -1.6 0.8])
r = 2.5000 + 1.2500i
      2.5000 - 1.2500i
p = 0.8000 + 0.4000i
      0.8000 - 0.4000i
k = []
```

Thus,

$$\frac{X_d(z)}{z} = \frac{5(z-1)}{z^2 - 1.6z + 0.8} = \frac{2.5 + j1.25}{z - 0.8 - j0.4} + \frac{2.5 - j1.25}{z - 0.8 + j0.4},$$

and

$$X_d(z) = \frac{(2.5 + j1.25)z}{z - 0.8 - j0.4} + \frac{(2.5 - j1.25)z}{z - 0.8 + j0.4}.$$

Scaling pair 10 of Table 7.1 by $r = 2|2.5 + j1.25| = 5.5902$ and identifying $|\gamma| = |0.8 + j0.4| = 0.8944$, $\beta = \angle(0.8 + j0.4) = 0.4636$, and $\theta = \angle(2.5 + j1.25) = 0.4636$, we obtain

$$x_d[n] = 5.5902(0.8944)^n \cos(0.4636n + 0.4636)u[n].$$

Drill 7.3 (Impulse Response by Inverse z -Transform)

(a) Using Eq. (5.33), we determine that

$$H_a(z) = \frac{B(z)}{A(z)} = \frac{8z - 19}{z^2 - 5z + 6}.$$

Using MATLAB, we expand $H_a(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([8 -19],[1 -5 6 0])
r = 1.6667  1.5000  -3.1667
p = 3.0000  2.0000  0
k = []
```

Thus,

$$\frac{H_a(z)}{z} = \frac{8z - 19}{z(z^2 - 5z + 6)} = \frac{\frac{5}{3}}{z-3} + \frac{\frac{3}{2}}{z-2} + \frac{\frac{-19}{6}}{z},$$

and

$$H_a(z) = \frac{\frac{5}{3}z}{z-3} + \frac{\frac{3}{2}z}{z-2} - \frac{19}{6}.$$

Using pair 3 of Table 7.1, we obtain

$$h_a[n] = \left[\frac{5}{3}(3)^n + \frac{3}{2}(2)^n \right] u[n] - \frac{19}{6}\delta[n].$$

(b) Using Eq. (5.33), we determine that

$$H_b(z) = \frac{B(z)}{A(z)} = \frac{2z^2 - 2z}{z^2 - 4z + 4}.$$

Using MATLAB, we expand $H_b(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([8 -19],[1 -5 6 0])
r = 2  2  0
p = 2  2  0
k = []
```

Thus,

$$\frac{H_b(z)}{z} = \frac{2z^2 - 2z}{z^2 - 4z + 4} = \frac{2}{z-2} + \frac{2}{(z-2)^2},$$

and

$$H_b(z) = \frac{2z}{z-2} + \frac{2z}{(z-2)^2}.$$

Using pairs 3 and 5 of Table 7.1, we obtain

$$h_b[n] = [2+n](2)^n u[n].$$

(c) Using Eq. (5.33), we determine that

$$H_c(z) = 1 - 2z^{-1}.$$

Using pair 12 of Table 7.1, we obtain

$$h_c[n] = \delta[n] - 2\delta[n-1].$$

Drill 7.4 (Inverse Bilateral z -Transform by Partial Fraction Expansion)

Using MATLAB, we expand $X(z)$ into modified partial fractions.

```
01 [r,p,k] = residue(1,[1 5/6 1/6])
r = -6.0000  6.0000
p = -0.5000  -0.3333
k = []
```

Thus,

$$\frac{X(z)}{z} = \frac{z}{z^2 + \frac{5}{6}z + \frac{1}{6}} = \frac{-6}{z + \frac{1}{2}} + \frac{6}{z + \frac{1}{3}},$$

and

$$X(z) = \frac{-6z}{z + \frac{1}{2}} + \frac{6z}{z + \frac{1}{3}}.$$

Since the ROC is $\frac{1}{3} < |z| < \frac{1}{2}$, we know that the mode at $\frac{1}{2}$ corresponds to a left-sided signal and that the mode at $\frac{1}{3}$ corresponds to a right-sided signal. Using pairs 3 and 5 of Table 7.1, we thus obtain

$$x[n] = 6(-\frac{1}{2})^n u[-n-1] + 6(-\frac{1}{3})^n u[n].$$

Drill 7.5 (Inverse z -Transform by Power Series Expansion)

- (a) Since $|z| > 0.5$ corresponds to a causal signal, we first obtain a series expansion in powers of z^{-1} as

$$\begin{array}{r} 1 + 0.5z^{-1} + 0.25z^{-2} + 0.125z^{-3} + \dots \\ z - 0.5 \overline{) z} \\ z - 0.5 \\ \hline 0.5 \\ 0.5 - 0.25z^{-1} \\ \hline 0.25z^{-1} \\ 0.25z^{-1} - 0.125z^{-2} \\ \hline 0.125z^{-2} \\ \vdots \end{array}.$$

Thus,

$$X_a(z) = 1 + \frac{1}{2}z^{-1} + \frac{1}{4}z^{-2} + \frac{1}{8}z^{-3} + \dots.$$

Using pair 12 of Table 7.1, we obtain

$$x_a[n] = \delta[n] + \frac{1}{2}\delta[n-1] + \frac{1}{4}\delta[n-2] + \frac{1}{8}\delta[n-3] + \dots = (\frac{1}{2})^n u[n].$$

This matches the expected result of pair 3 of Table 7.1.

- (b) Since $|z| < 0.5$ corresponds to an anti-causal signal, we first obtain a series expansion in powers of z as

$$\begin{array}{r} -2z - 4z^2 - 8z^3 - 16z^4 - \dots \\ -0.5 + z \overline{) z} \\ z - 2z^2 \\ \hline 2z^2 \\ 2z^2 - 4z^3 \\ \hline 4z^3 \\ 4z^3 - 8z^4 \\ \hline 8z^4 \\ \vdots \end{array}.$$

Thus,

$$X_b(z) = -2z - 4z^2 - 8z^3 - 16z^4 - \dots.$$

Using pair 12 of Table 7.1, we obtain

$$x_b[n] = -2\delta[n+1] - 4\delta[n+2] - 8\delta[n+3] - 16\delta[n+4] - \dots = -(\frac{1}{2})^n u[-n-1].$$

This matches the expected result of pair 14 of Table 7.1.

Drill 7.6 (Time-Reversal Property)

Pair 2 of Table 7.1 indicates that

$$u[n] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-1}, \quad |z| > 1.$$

The time-reversal property states that changing n to $-n$ in the time domain requires changing z to $1/z$. Thus,

$$u[-n] \xleftrightarrow{\mathcal{Z}} \frac{\frac{1}{z}}{\frac{1}{z}-1}, \quad \left| \frac{1}{z} \right| > 1.$$

Simplifying, we obtain the desired result of

$$u[-n] \xleftrightarrow{\mathcal{Z}} \frac{-1}{z-1}, \quad |z| < 1.$$

Drill 7.7 (Using the Time-Shifting Property)

To begin, we represent the signal of interest using unit step functions as

$$x_a[n] = u[n-4] - u[n-10].$$

Using the right-shifting property of Eq. (7.21) and the fact that $u[n] \xleftrightarrow{\mathcal{Z}_u} \frac{z}{z-1}$, we see that

$$x_a[n] = u[n-4] - u[n-10] \xleftrightarrow{\mathcal{Z}_u} (z^{-4} - z^{-10}) \frac{z}{z-1} = X_a(z).$$

Drill 7.8 (Using the z -Domain Scaling Property)

Pair 2 of Table 7.1 states that

$$u[n] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-1}, \quad |z| > 1.$$

Using Eq. (7.24), we see that

$$\gamma^n u[n] \xleftrightarrow{\mathcal{Z}} \frac{\frac{z}{\gamma}}{\frac{z}{\gamma}-1}, \quad |z| > |\gamma|.$$

Simplifying, we obtain the desired result of pair 3 of Table 7.1,

$$\gamma^n u[n] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-\gamma}, \quad |z| > |\gamma|.$$

We similarly derive pair 14. To begin, pair 13 of Table 7.1 states that

$$-u[-n-1] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-1}, \quad |z| < 1.$$

Using Eq. (7.24), we see that

$$-\gamma^n u[-n-1] \xleftrightarrow{\mathcal{Z}} \frac{\frac{z}{\gamma}}{\frac{z}{\gamma}-1}, \quad |z| < |\gamma|.$$

Simplifying, we obtain the desired result of pair 14 of Table 7.1,

$$-\gamma^n u[-n-1] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-\gamma}, \quad |z| < |\gamma|.$$

Drill 7.9 (Using the z -Domain Differentiation Property)

Pair 3 of Table 7.1 states that

$$\gamma u[n] \xleftrightarrow{\mathcal{Z}} \frac{z}{z-\gamma}, \quad |z| > |\gamma|.$$

Using Eq. (7.25), we see that

$$n\gamma^n u[n] \xrightarrow{z} -z \frac{d}{dz} \left(\frac{z}{z-\gamma} \right) = -z \left(\frac{1}{z-\gamma} - \frac{z}{(z-\gamma)^2} \right), \quad |z| > |\gamma|.$$

Simplifying, we obtain the desired result of pair 5 of Table 7.1,

$$n\gamma^n u[n] \xrightarrow{z} \frac{\gamma z}{(z-\gamma)^2}, \quad |z| > |\gamma|.$$

Applying Eq. (7.25) to this result (pair 5), we see that

$$n^2\gamma^n u[n] \xrightarrow{z} -z \frac{d}{dz} \left(\frac{\gamma z}{(z-\gamma)^2} \right) = -z \left(\frac{\gamma}{(z-\gamma)^2} - \frac{2\gamma z}{(z-\gamma)^3} \right), \quad |z| > |\gamma|.$$

Simplifying, we obtain the desired result of pair 6 of Table 7.1,

$$n^2\gamma^n u[n] \xrightarrow{z} \frac{\gamma z(z+\gamma)}{(z-\gamma)^3}, \quad |z| > |\gamma|.$$

Applying Eq. (7.25) to this result (pair 6), we see that

$$n^3\gamma^n u[n] \xrightarrow{z} -z \frac{d}{dz} \left(\frac{\gamma z(z+\gamma)}{(z-\gamma)^3} \right) = -z \left(\frac{2\gamma z + \gamma^2}{(z-\gamma)^3} - \frac{3(\gamma z^2 + \gamma^2 z)}{(z-\gamma)^4} \right), \quad |z| > |\gamma|.$$

Simplifying, we obtain the desired result of

$$n^3\gamma^n u[n] \xrightarrow{z} \frac{\gamma z(z^2 + 4\gamma z + \gamma^2)}{(z-\gamma)^4}, \quad |z| > |\gamma|.$$

Drill 7.10 (Using the Convolution Property)

From Table 7.1, we know that

$$u[n] \xrightarrow{z} \frac{z}{z-1} \quad \text{and} \quad u[n-1] \xrightarrow{z} \frac{1}{z-1}, \quad \text{both with } |z| > 1.$$

Using the convolution property, we find that

$$u[n] * u[n-1] \xrightarrow{z} \left(\frac{z}{z-1} \right) \left(\frac{1}{z-1} \right) = \frac{z}{(z-1)^2}, \quad |z| > 1.$$

Using pair 5 of Table 7.1 with $\gamma = 1$, we see that the z -transform $\frac{z}{(z-1)^2}$ also corresponds to the signal $nu[n]$. Thus,

$$u[n] * u[n-1] = nu[n].$$

Drill 7.11 (z -Transform Solution of Linear Difference Equations)

Applying the unilateral z -transform to the delay form difference equation $y[n] - \frac{5}{6}y[n-1] + \frac{1}{6}y[n-2] = 5x[n-1] - x[n-2]$ yields

$$Y(z) - \frac{5}{6}(z^{-1}Y(z) + y[-1]) + \frac{1}{6}(z^{-2}Y(z) + y[-1]z^{-1} + y[-2]) = (5z^{-1} - z^{-2})X(z).$$

Since $x[n]$ is causal, any input terms such as $x[-1]$ and $x[-2]$ are necessarily zero. Substituting $y[-1] = 2$, $y[-2] = 0$, and $X(z) = \frac{z}{z-1}$, we obtain

$$Y(z) \left(1 - \frac{5}{6}z^{-1} + \frac{1}{6}z^{-2} \right) - \frac{5}{3} + \frac{1}{3}z^{-1} = \frac{5-z^{-1}}{z-1}.$$

Multiplying this expression by z^2 and rearranging yield

$$Y(z) \left(z^2 - \frac{5}{6}z + \frac{1}{6} \right) = \frac{5z^2 - z}{z - 1} + \frac{5}{3}z^2 - \frac{1}{3}z.$$

Solving for $Y(z)$ yields, after some simplification,

$$Y(z) = \frac{z \left(\frac{5}{3}z^2 + 3z - \frac{2}{3} \right)}{(z - \frac{1}{3})(z - \frac{1}{2})(z - 1)}.$$

Next, we use MATLAB to expand $Y(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([5/3 3 -2/3],poly([1/3,1/2,1]))
r = 12.0000 -15.0000 4.6667
p = 1.0000 0.5000 0.3333
k = []
```

Thus,

$$\frac{Y(z)}{z} = \frac{12}{z - 1} + \frac{-15}{z - \frac{1}{2}} + \frac{\frac{14}{3}}{z - \frac{1}{3}},$$

and

$$Y(z) = \frac{12z}{z - 1} + \frac{-15z}{z - \frac{1}{2}} + \frac{\frac{14}{3}z}{z - \frac{1}{3}}.$$

Inverting, the system output is therefore

$$y[n] = \left[12 - 15\left(\frac{1}{2}\right)^n + \frac{14}{3}\left(\frac{1}{3}\right)^n \right] u[n].$$

Drill 7.12 (z -Transform Solution Using Initial Conditions $y[0]$, $y[1]$, ..., $y[K - 1]$)

Since the initial conditions are given in terms of $y[0]$ and $y[1]$, it is most convenient to express the difference equation in advance form as $y[n + 2] + 3y[n + 1] + 2y[n] = x[n + 1] + 3x[n]$. Taking the unilateral z -transform yields

$$Y(z) (z^2 + 3z + 2) - z^2 y[0] - zy[1] - 3zy[0] = X(z) (z + 3) - zx[0].$$

Notice that since the advance form is used, input terms (such as $x[0]$) now come into play. Substituting $x[0] = 1$, $y[0] = 1$, $y[1] = 2$, and $X(z) = \frac{z}{z-1}$, we obtain

$$Y(z) (z^2 + 3z + 2) - z^2 - 5z = \frac{z(z+3)}{z-1} - z.$$

Solving for $Y(z)$ yields, after some simplification,

$$Y(z) = \frac{z(z^2 + 4z - 1)}{(z - 1)(z + 1)(z + 2)}.$$

Next, we use MATLAB to expand $Y(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([1 4 -1],poly([1,-1,-2]))
r = -1.6667 0.6667 2.0000
p = -2.0000 1.0000 -1.0000
k = []
```

Thus,

$$\frac{Y(z)}{z} = \frac{\frac{5}{3}}{z+2} + \frac{\frac{2}{3}}{z-1} + \frac{2}{z+1},$$

and

$$Y(z) = \frac{\frac{5}{3}z}{z+2} + \frac{\frac{2}{3}z}{z-1} + \frac{2z}{z+1}.$$

Inverting, the system output is therefore

$$y[n] = \left[\frac{5}{3}(-2)^n + \frac{2}{3} + 2(-1)^n \right] u[n].$$

Drill 7.13 (Finding the ZIR and ZSR by z -Transform)

This problem is identical to Drill 7.11 except that rather than the total solution, the ZIR and ZSR are required. From the solution to Drill 7.11,

$$Y(z) \left(z^2 - \frac{5}{6}z + \frac{1}{6} \right) = \underbrace{\frac{5z^2 - z}{z-1}}_{\text{input terms}} + \underbrace{\frac{5}{3}z^2 - \frac{1}{3}z}_{\text{IC terms}}.$$

Solving for $Y(z)$ but keeping the IC and input terms separated yield

$$Y(z) = \underbrace{\frac{z(5z-1)}{(z-\frac{1}{3})(z-\frac{1}{2})(z-1)}}_{\text{ZSR}} + \underbrace{\frac{z(\frac{5}{3}z-\frac{1}{3})}{(z-\frac{1}{3})(z-\frac{1}{2})}}_{\text{ZIR}}.$$

Now, we use MATLAB to expand the ZSR into modified partial fractions.

```
01 [r,p,k] = residue([5 -1],poly([1/3,1/2,1]))
r = 12.0000 -18.0000 6.0000
p = 1.0000 0.5000 0.3333
k = []
```

Thus,

$$Y_{\text{zsr}}(z) = \frac{12z}{z-1} + \frac{-18z}{z-\frac{1}{2}} + \frac{6z}{z-\frac{1}{3}},$$

and

$$y_{\text{zsr}}[n] = \left[12 - 18(\frac{1}{2})^n + 6(\frac{1}{3})^n \right] u[n].$$

Next, we use MATLAB to expand the ZIR into modified partial fractions.

```
01 [r,p,k] = residue([5/3 -1/3],poly([1/3,1/2]))
r = 3.0000 -1.3333
p = 0.5000 0.3333
k = []
```

Thus,

$$Y_{\text{zir}}(z) = \frac{3z}{z-\frac{1}{2}} + \frac{-\frac{4}{3}z}{z-\frac{1}{3}},$$

and

$$y_{\text{zir}}[n] = \left[3(\frac{1}{2})^n - \frac{4}{3}(\frac{1}{3})^n \right] u[n].$$

As required and expected, summing the ZSR and ZIR generates the result of Drill 7.11,

$$y[n] = y_{\text{zsr}}[n] + y_{\text{zir}}[n] = \left[12 - 15(\frac{1}{2})^n + \frac{14}{3}(\frac{1}{3})^n \right] u[n].$$

Drill 7.14 (Working with a System Transfer Function)

(a) To begin, we note that

$$x[n] = 3^{-(n+1)} u[n] \xrightarrow{Z_u} \frac{\frac{1}{3}z}{z - \frac{1}{3}}.$$

In the transform domain, the ZSR is just the product of the input and system transfer function

$$Y_{\text{zsr}}(z) = X(z)H(z) = \frac{\frac{1}{3}z}{z - \frac{1}{3}} \left(\frac{z - \frac{1}{2}}{(z + \frac{1}{2})(z - 1)} \right).$$

Next, we use MATLAB to expand $Y_{\text{zsr}}(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([1/3 -1/6],poly([1/3,-1/2,1]))
r = 0.1667 -0.2667 0.1000
p = 1.0000 -0.5000 0.3333
k = []
```

Thus,

$$Y_{\text{zsr}}(z) = \frac{\frac{1}{6}z}{z - 1} + \frac{-\frac{4}{15}z}{z + \frac{1}{2}} + \frac{\frac{1}{10}z}{z - \frac{1}{3}},$$

and

$$y_{\text{zsr}}[n] = \left[\frac{1}{6} - \frac{4}{15} \left(-\frac{1}{2} \right)^n + \frac{1}{10} \left(\frac{1}{3} \right)^n \right] u[n].$$

(b) To determine the difference equation that relates $y[n]$ to $x[n]$, we first note that

$$H(z) = \frac{Y(z)}{X(z)} = \frac{z - \frac{1}{2}}{z^2 - \frac{1}{2}z - \frac{1}{2}}.$$

Cross-multiplying, we obtain

$$\left(z^2 - \frac{1}{2}z - \frac{1}{2} \right) Y(z) = \left(z - \frac{1}{2} \right) X(z).$$

Since the system is assumed to be both controllable and observable, we need not worry about any potential pole-zero cancellations. Inverting the previous transform-domain expression, we thus obtain the desired system difference equation as

$$y[n+2] - \frac{1}{2}y[n+1] - \frac{1}{2}y[n] = x[n+1] - \frac{1}{2}x[n].$$

Drill 7.15 (ZSR by z -Transform for a Two-Sided Input)

To begin, we note that

$$x[n] = \left(\frac{1}{4} \right)^n u[n] + 5(3)^n u[-n-1] \xrightarrow{Z} \frac{z}{z - \frac{1}{4}} - \frac{5z}{z - 3} = \frac{-4z^2 - \frac{7}{4}z}{(z - \frac{1}{4})(z - 3)}, \quad \frac{1}{4} < |z| < 3.$$

In the transform domain, the ZSR is just the product of the input and system transfer function

$$Y_{\text{zsr}}(z) = H(z)X(z) = \frac{z}{z - \frac{1}{2}} \left(\frac{-4z^2 - \frac{7}{4}z}{(z - \frac{1}{4})(z - 3)} \right).$$

Since we later consider both causal and anti-causal systems, we ignore the ROC at this time. Next, we use MATLAB to expand $Y_{\text{zsr}}(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([-4 -7/4 0],poly([1/4,1/2,3]))
r = -6.0000 3.0000 -1.0000
p = 3.0000 0.5000 0.2500
k = []
```

Thus,

$$Y_{\text{zsr}}(z) = \frac{-6z}{z-3} + \frac{3z}{z-\frac{1}{2}} + \frac{-z}{z-\frac{1}{4}}.$$

- (a) In the case that the system is causal, we see that the ROC for $H(z)$ is $|z| > \frac{1}{2}$ and the ROC of the ZSR is therefore $\frac{1}{2} < |z| < 3$. Thus, the time-domain expression of the causal system ZSR is

$$y_a[n] = 6(3)^n u[-n-1] + [3(\frac{1}{2})^n - (\frac{1}{4})^n] u[n].$$

- (b) In the case that the system is anti-causal, we see that the ROC for $H(z)$ is $|z| < \frac{1}{2}$ and the ROC of the ZSR is therefore $\frac{1}{4} < |z| < \frac{1}{2}$. Thus, the time-domain expression of the anti-causal system ZSR is

$$y_b[n] = [6(3)^n - 3(\frac{1}{2})^n] u[-n-1] - (\frac{1}{4})^n u[n].$$

Drill 7.16 (Assessing System Stability)

- (a) An accumulator with impulse response $h_a[n] = u[n]$ has transfer function $H_a(z) = \frac{z}{z-1}$. The single pole of this system is on the unit circle, which makes the system marginally stable. To assess external (BIBO) stability, we determine whether or not the impulse response is absolutely summable. Since $\sum_{n=-\infty}^{\infty} |h_a[n]| = \sum_{n=0}^{\infty} 1 \rightarrow \infty$, $h_a[n]$ is not absolutely summable, and the system is therefore BIBO unstable.
- (b) The system $H_b(z) = \frac{z+1}{(z+0.5)(z-1)(z-2)}$ has three poles, located at $z = -\frac{1}{2}, 1$, and 2 , respectively. Further, the ROC of $H_b(z)$ is given as $0.5 < |z| < 1$. The annular shape of this ROC confirms that the system is noncausal. Since the ROC contacts (but does not include) the unit circle and there are no repeated roots on the unit circle, the system is marginally stable.

To assess BIBO stability, we first use MATLAB to expand $H_b(z)$ into modified partial fractions.

```
01 [r,p,k] = residue([1 1],poly([0,-.5,1,2]))
r = 0.6000 -1.3333 -0.2667 1.0000
p = 2.0000 1.0000 -0.5000 0
k = []
```

Thus,

$$H_b(z) = \frac{\frac{3}{5}z}{z-2} + \frac{-\frac{4}{3}z}{z-1} + \frac{-\frac{4}{15}z}{z+\frac{1}{2}} + 1.$$

For the ROC $0.5 < |z| < 1$, the impulse response is thus

$$h_b[n] = \left[-\frac{3}{5}(2)^n + \frac{4}{3} \right] u[-n-1] - \frac{4}{15}(\frac{1}{2})^n u[n].$$

Due to the $\frac{4}{3}u[-n-1]$ term, $\sum_{n=-\infty}^{\infty} |h_b[n]| \rightarrow \infty$, and the system is BIBO unstable.

Drill 7.17 (Inverse Systems)

The inverse system transfer function is just the reciprocal of the original transfer function; that is,

$$H_i(z) = \frac{1}{H(z)} = \frac{z-0.5}{z+2}.$$

Due to the pole $z = -2$, which is outside the unit circle, the inverse system is unstable and therefore not well behaved. This is a common problem with inverse systems, which swap the roles

of the original system's poles and zeros. Zeros whose locations in no way impact the stability of the original system become poles that, depending on location, can render the inverse system unstable.

Drill 7.18 (Canonical System Realizations)

To begin, it is useful to represent the transfer function using Eq. (7.45) in terms of z^{-1} as

$$H(z) = \frac{\frac{1}{16}z^{-1}}{1 - \frac{3}{2}z^{-1} + \frac{9}{8}z^{-2}}.$$

Following the structures of Figs. 7.17 and 7.19, the DFII and TDFII realizations of this system are shown in Figs. D7.18a and D7.18b, respectively.

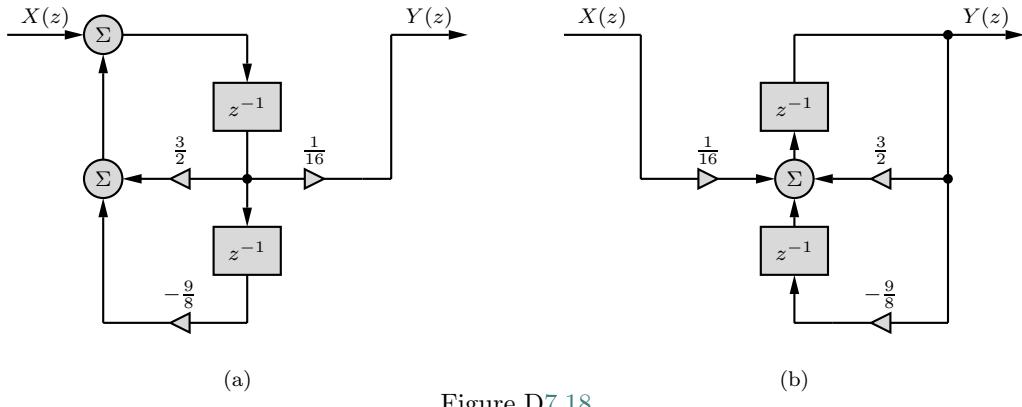


Figure D7.18

Drill 7.19 (The Unpopular Transposed Direct Form I)

Applying the transpose operation to the DFI structure of Fig. 7.14, we obtain the TDFI structure shown in Fig. D7.19. Not only is this structure non-canonical, but it also orders system poles before system zeros. For these two reasons, the TDFI is not a popular structure for system realization.

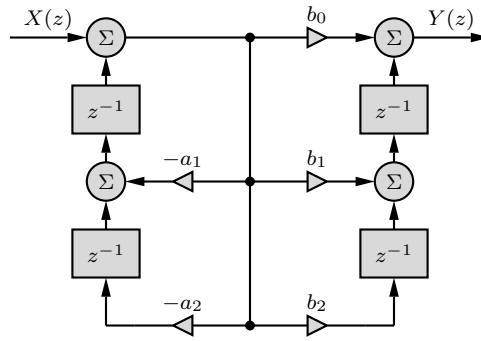


Figure D7.19

Drill 7.20 (A Non-Realizable Transfer Function)

In this case, we realize that

$$H(z) = \frac{Y(z)}{X(z)} = \frac{z^2 + \frac{1}{4}}{z} = \frac{1 + \frac{1}{4}z^{-2}}{z^{-1}}.$$

In delay form, the corresponding difference equation is

$$y[n-1] = x[n] + \frac{1}{4}x[n-2].$$

Clearly, this system is noncausal and, therefore, not realizable. No matter how we might try to shift this difference equation, the output always depends on a future value of the input. In fact, whenever a transfer function is not proper, the corresponding system is noncausal and, therefore, not realizable. This fact explains why practical systems have proper transfer functions.

Looked at a different way, we see that $H(z) = z + \frac{1}{4}z^{-1}$. Inverting, we find that the impulse response is $h[n] = \delta[n+1] + \frac{1}{4}\delta[n-1]$. The $\delta[n+1]$ term clearly renders the impulse response noncausal.

Drill 7.21 (Determining and Using Frequency Response)

In advance operator form, the system equation is expressed as

$$(E - 0.5) \{y[n]\} = x[n].$$

Therefore, the transfer function of the system is

$$H(z) = \frac{1}{z - 0.5}.$$

Letting $z = e^{j\Omega}$, the frequency response is

$$H(e^{j\Omega}) = \frac{1}{e^{j\Omega} - 0.5}.$$

The magnitude and phase responses are thus

$$|H(e^{j\Omega})| = \frac{1}{\sqrt{1.25 - \cos(\Omega)}} \quad \text{and} \quad \angle H(e^{j\Omega}) = -\tan^{-1} \left(\frac{\sin(\Omega)}{\cos(\Omega) - 0.5} \right).$$

MATLAB readily computes and graphs the magnitude and phase responses, which are shown in Figs. D7.21a and D7.21b, respectively.

```
01 Omega = linspace(-2*pi, 2*pi, 500); H = 1./(exp(j*Omega)-0.5);
02 subplot(121); plot(Omega,abs(H)); subplot(122); plot(Omega,angle(H));
```

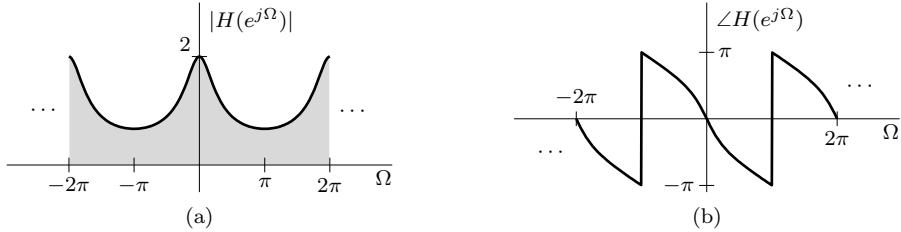


Figure D7.21

Sampling the CT input $x(t) = \cos(1000t - \frac{\pi}{3})$ with $T = 0.5$ ms produces the DT input $x[n] = \cos(0.5n - \frac{\pi}{3})$. Further, a sinusoidal input passes through an LTID system changed only in gain and phase. Since $|H(e^{j0.5})| = 1.6386$ and $\angle H(e^{j0.5}) = -0.9037$, the response to $x[n] = \cos(0.5n - \frac{\pi}{3})$ is therefore

$$y[n] = 1.6386 \cos(0.5n - \frac{\pi}{3} - 0.9037) = 1.6386 \cos(0.5n - 1.9509).$$

Drill 7.22 (Frequency Response of Unit Delay System)

The transfer function of the unit delay system $y[n] = x[n - 1]$ is $H(z) = z^{-1}$. Letting $z = e^{j\Omega}$, the frequency response is thus

$$H(z) = e^{-j\Omega}.$$

The magnitude and phase responses are

$$|H(e^{j\Omega})| = 1 \quad \text{and} \quad \angle H(e^{j\Omega}) = -\Omega.$$

Clearly, the unit delay system causes a linear phase shift ($-\Omega$) in an input sinusoid without changing the input sinusoid's magnitude.

Drill 7.23 (Complex Notch Filter)

The transfer function $H(z) = \frac{z-j}{2z}$ has a pole at the origin and a zero located at $z = j$ ($\Omega = \pi/2$). The zero at $z = j$ produces a stopband (notch) in the magnitude response that is centered at $\Omega = \pi/2$. Since this complex zero is not accompanied by its complex conjugate, the filter is necessarily complex, and the magnitude response is not symmetric in Ω .

MATLAB readily computes and graphs the magnitude and phase responses, which are shown in Figs. D7.23a and D7.23b, respectively.

```
01 Omega = linspace(-2*pi, 2*pi, 500); H = (exp(1j*Omega)-1j)./(2*exp(1j*Omega));
02 subplot(121); plot(Omega, abs(H)); subplot(122); plot(Omega, angle(H));
```

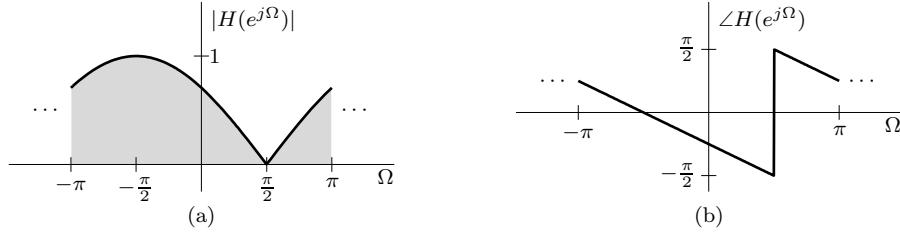


Figure D7.23

To determine the output of this system, notice that we can use Euler's formula to represent the input as

$$x[n] = \cos(\pi n/2) = \frac{1}{2}e^{j\pi n/2} + \frac{1}{2}e^{-j\pi n/2}.$$

The first component is completely removed by the zero at $z = j$. The second component is passed with unity gain (and zero phase shift). Thus, the (complex) response to $x[n]$ is

$$y[n] = \frac{1}{2}e^{-j\pi n/2}.$$

Chapter 8

Drill 8.1 (Impulse Invariance Method: Digital Butterworth LPF)

We follow Ex. 8.1 to complete this design, which shows that the transfer function of a first-order Butterworth lowpass filter with 3-dB cutoff frequency ω_c is

$$H_c(s) = \frac{\omega_c}{s + \omega_c}.$$

Setting $s = j\omega$ in $H_c(s)$, we have

$$|H_c(j\omega)| = \frac{\omega_c}{\sqrt{\omega^2 + \omega_c^2}}.$$

In this case, the maximum value of $|H_c(j\omega)|$ is 1, which occurs at $\omega = 0$. Applying a 10% criterion leads to $|H_c(j\pi/T)| = 0.1$. Observe that

$$|H_c(j\omega)| \approx \frac{\omega_c}{\omega} \quad \text{for } \omega \gg \omega_c.$$

Hence,

$$|H_c(j\pi/T)| \approx \frac{\omega_c}{\pi/T} = 0.1 \implies \pi/T = 10\omega_c = 10(2\pi 250) = 5000\pi.$$

Thus, a 10% criterion yields $T = 1/5000$ and $F_s = 5000$ Hz.

According to Eq. (8.3) (or pair 4 in Table 8.1), the transfer function $H(z)$ of the corresponding digital filter is

$$H(z) = \frac{\omega_c T}{2} \left(\frac{z + e^{-\omega_c T}}{z - e^{-\omega_c T}} \right).$$

Substituting $\omega_c = 500\pi$ and $T = 1/5000$ into this equation yields

$$H(z) = 0.1571 \left(\frac{z + 0.7304}{z - 0.7304} \right) = \frac{0.1571z + 0.1147}{z - 0.7304}.$$

Except that it operates at $F_s = 5000$ Hz rather than $10^6/\pi$, this digital filter is identical to that computed in Ex. 8.1; the TDFII realization is shown in Fig. 8.4, the magnitude response is shown in Fig. 8.5a, and the phase response is shown in Fig. 8.5a.

Changing the design criterion to a 1% rule yields a tenfold increase in the sampling frequency to $F_s = 50000$ Hz. Clearly, it is not possible to use a 1% rule and also satisfy the requirement that $F_s \leq 10000$ Hz. The highest allowable sampling rate $F_s = 10000$ Hz produces $|H_c(j\pi/T)| = \omega_c/(\pi/T) = 1/20$, which is consistent with a 5% rule.

Drill 8.2 (Bilinear Transform Using Discrete-Time Specifications)

To begin this design, we prewarp the digital passband frequency $\Omega_p = \pi/3$ to the analog passband frequency ω'_p as

$$\omega'_p = \tan\left(\frac{\Omega_p}{2}\right) = \tan\left(\frac{\pi}{6}\right) = 0.5774.$$

Next, we use MATLAB to design the prewarped analog lowpass filter. Line 02 uses Eq. (2.45) to determine ϵ , lines 03–04 use Eq. (2.47) to determine the Chebyshev poles p_k , line 05 uses Eq. (2.44) to determine the dc gain $H'_c(j0)$, and line 06 uses Eq. (2.48) to determine the coefficients of the prewarped analog filter transfer function $H'_c(s) = B'_c(s)/A'_c(s)$.

```

01 Omegap = pi/3; omegapp = tan(Omegap/2); K = 2; k = 1:K;
02 alphap = 1; epsilon = sqrt(10^(alphap/10)-1);
03 pk = -omegapp*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
04     1j*omegapp*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
05 Hcp0 = 1/sqrt(1+epsilon^2);
06 Bcp = abs(Hcp0)*prod(-pk), Acp = poly(pk)
    Bcp = 0.3275
    Acp = 1.0000  0.6338  0.3675

```

Thus, the prewarped analog transfer function is

$$H'_c(s) = \frac{B'_c(s)}{A'_c(s)} = \frac{0.3275}{s^2 + 0.6338s + 0.3675}.$$

Expanding Eq. (8.20), we obtain the coefficients of the desired digital filter.

```

07 B = Bcp/prod(1-pk)*poly([-1,-1]), A = poly((1+pk)./(1-pk))
    B = 0.1637  0.3273  0.1637
    A = 1.0000  -1.7241  2.7276

```

That is,

$$H(z) = H'_c(s)|_{s=\frac{z-1}{z+1}} = \frac{0.3275}{s^2 + 0.6338s + 0.3675} \Big|_{s=\frac{z-1}{z+1}} = \frac{0.1637(z^2 + 2z + 1)}{z^2 - 1.7241z + 2.7276}.$$

To verify that requirements are met, Fig. D8.2 plots the magnitude response $|H(e^{j\Omega})|$ versus Ω .

```

08 Omega = linspace(0,pi,1001);
09 H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
10 plot(Omega,abs(H),'k');

```

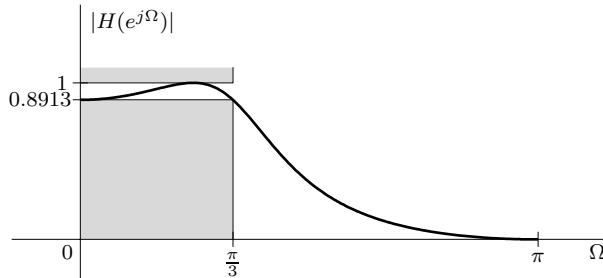


Figure D8.2

Drill 8.3 (Digital Bandstop IIR Filter Design)

To design a 12th-order digital inverse Chebyshev bandstop filter, we begin with a 6th-order Chebyshev lowpass prototype with normalized passband frequency $\omega_p = 1$.

```

01 alphap = 2; alphas = 40; K = 6; omegap = 1;
02 omegas = omegap*cosh(acosh(sqrt((10^(alphas/10)-1)/(10^(alphap/10)-1)))/K);
03 epsilon = 1/sqrt(10^(alphas/10)-1); k = 1:K;
04 P = -omegap*sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
05 1j*omegap*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));

```

Here, line 01 sets the initial parameters. Rearranging Eq. (2.46) (or Eq. (2.52)), line 02 determines the corresponding Chebyshev prototype stopband frequency ω_s . Using Eq. (2.53), line 03 computes ϵ , and lines 04–05 determine the prototype Chebyshev LPF poles using Eq. (2.47).

Next, we use Eqs. (2.55), (2.56), and (2.54) to determine the pole locations, zero locations, and gain factor of the inverse Chebyshev LPF. Notice that the inverse Chebyshev prototype also has normalized passband frequency.

```

06 P = omegap*omegas./P; Z = 1j*omegas.*sec(pi*(2*k-1)/(2*K));
07 bL = prod(P./Z); aK = 1; L = length(Z); K = length(P);

```

Using Eq. (8.21), we next prewarp our DT passband specifications to the corresponding CT passband specifications. Then, using Eqs. (8.24) and (8.25), we determine the desired digital inverse Chebyshev bandstop filter.

```

08 Omegap1 = pi/3; omegap1p = tan(Omegap1/2); Omegap2 = 2*pi/3; omegap2p = tan(Omegap2/2);
09 c1 = (omegap1p*omegap2p-1)/(omegap1p*omegap2p+1);
10 c2 = (omegap2p-omegap1p)/(omegap1p*omegap2p+1);
11 for i = 1:L, Zdig(i,:) = roots([1 2*c1*Z(i)./(Z(i)-c2) (Z(i)+c2)./(Z(i)-c2)]); end
12 for i = 1:K, Pdig(i,:) = roots([1 2*c1*P(i)./(P(i)-c2) (P(i)+c2)./(P(i)-c2)]); end
13 B = bL/aK*prod(c2-Z)/prod(c2-P)*poly(Zdig(:)'), A = poly(Pdig(:)'),
B = 0.2695 0 1.4280 0 3.3205 0 4.3230 0 3.3205 0 1.4280 0 0.2695
A = 1.0000 0 2.9430 0 4.2710 0 3.6337 0 1.8829 0 0.5559 0 0.0726

```

Thus, the digital filter transfer function is

$$H(z) = \frac{0.2695z^{12} + 1.4280z^{10} + 3.3205z^8 + 4.3230z^6 + 3.3205z^4 + 1.4280z^2 + 0.2695}{z^{12} + 2.9430z^{10} + 4.2710z^8 + 3.6337z^6 + 1.8829z^4 + 0.5559z^2 + 0.0726}.$$

From the lowpass-to-bandstop transformation of Eq. (2.31), ω_s maps to two stopband frequencies of the analog bandstop filter, which are determined by solving the quadratic

$$\omega_s \omega^2 + (\omega_{p_2} - \omega_{p_1})\omega - \omega_s \omega_{p_1} \omega_{p_2} = 0.$$

The bilinear transform then converts these values according to $\Omega = 2 \tan^{-1}(\omega)$.

```
14 Omegas = 2*atan(abs(roots([omegas omegap2p-omegap1p -omegas*omegap1p*omegap2p])));
Omegas = 1.9143 1.2273
```

Thus, the critical stopband frequencies are

$$\Omega_{s_1} = 1.2273 = 0.3907\pi \quad \text{and} \quad \Omega_{s_2} = 1.9143 = 0.6093\pi.$$

The magnitude response plot of Fig. D8.3 confirms these calculations as well as the overall behavior of the digital bandstop filter.

```
15 Omega = linspace(0,pi,1001); H = polyval(B,exp(1j*Omega))./polyval(A,exp(1j*Omega));
16 plot(Omega/pi,abs(H),'k-');
```

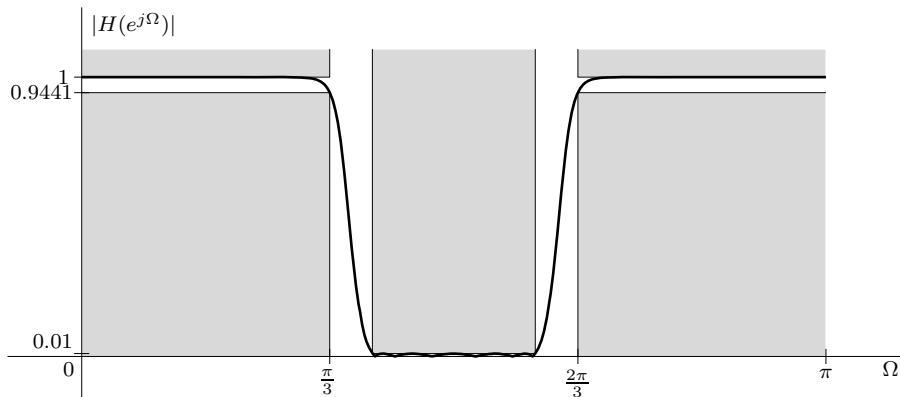


Figure D8.3

Drill 8.4 (Filter Realization Using a Cascade of Second-Order Sections)

To begin, we execute lines 01–09 in Ex. 8.8 to design the desired 10th-order digital Chebyshev bandstop filter.

```
01 alphap = 1; K = 5; k = 1:K; epsilon = sqrt(10^(alphap/10)-1); Z = [];
02 P = -sinh(asinh(1/epsilon)/K)*sin(pi*(2*k-1)/(2*K))+...
03 1j*cosh(asinh(1/epsilon)/K)*cos(pi*(2*k-1)/(2*K));
04 bL = 1*prod(-P); aK = 1; L = length(Z); K = length(P);
05 Omegap1 = pi/4; omegap1p = tan(Omegap1/2); Omegap2 = pi/2; omegap2p = tan(Omegap2/2);
06 c1 = (omegap1p*omegap2p-1)/(omegap1p*omegap2p+1);
07 c2 = (omegap2p-omegap1p)/(omegap1p*omegap2p+1);
08 Zdig = repmat(roots([1 2*c1 1]),K-L,1);
09 for i = 1:K, Pdig(i,:) = roots([1 2*c1*P(i)./(P(i)-c2) (P(i)+c2)./(P(i)-c2)]); end
```

The locations of the digital zeros and poles, as well as the distance of the poles from the origin, are easily displayed.

```
10 Zdig = Zdig(:, Pdig = Pdig(:, PDist = abs(Pdig(:))
Zdig = 0.4142+0.9102i      Pdig = 0.6894+0.6879i      PDist = 0.9739
      0.4142-0.9102i          0.6976+0.5355i          0.8795
      0.4142+0.9102i          0.6246+0.0000i          0.6246
      0.4142-0.9102i          0.6976-0.5355i          0.8795
      0.4142+0.9102i          0.6894-0.6879i          0.9739
      0.4142-0.9102i          -0.0010-0.9632i          0.9632
      0.4142+0.9102i          -0.1491-0.7997i          0.8135
      0.4142-0.9102i          -0.2838-0.0000i          0.2838
      0.4142+0.9102i          -0.1491+0.7997i          0.8135
      0.4142-0.9102i          -0.0010+0.9632i          0.9632
```

Since the conjugate zeros $0.4142 \pm j0.9102$ are repeated five times, no pole-zero pairing is required. Each of the five second-order stages possesses the same pair of conjugate zeros, realized as

$$B(z) = (z - 0.4142 - j0.9102)(z - 0.4142 + j0.9102) = z^2 - 0.8284z + 1.$$

To determine the poles for each of the five second-order stages, we follow the recommended (reverse-order) procedure on page 508, which orders the two real poles first and follows with the conjugate poles ordered beginning with those furthest from the unit circle. In this way, the poles for each of the five stages are realized as

$$A_1(z) = (z - 0.6246)(z + 0.2838) = z^2 - 0.3408z - 0.1772,$$

$$A_2(z) = (z + 0.1491 - j0.7997)(z + 0.1491 + j0.7997) = z^2 + 0.2982z + 0.6617,$$

$$A_3(z) = (z - 0.6976 - j0.5355)(z - 0.6976 + j0.5355) = z^2 - 1.3953z + 0.7735,$$

$$A_4(z) = (z + 0.0010 - j0.9632)(z + 0.0010 + j0.9632) = z^2 + 0.0020z + 0.9278,$$

and $A_5(z) = (z - 0.6894 - j0.6879)(z - 0.6894 + j0.6879) = z^2 - 1.3787z + 0.9484.$

Referencing Eq. (8.27), we compute the filter's overall gain factor as

```
10 bL/aK*prod(c2-Z)/prod(c2-P)
ans = 0.1779
```

Figure D8.4 shows the final realization, utilizing the gain factor of 0.1779 and a cascade of five second-order TDFII stages.

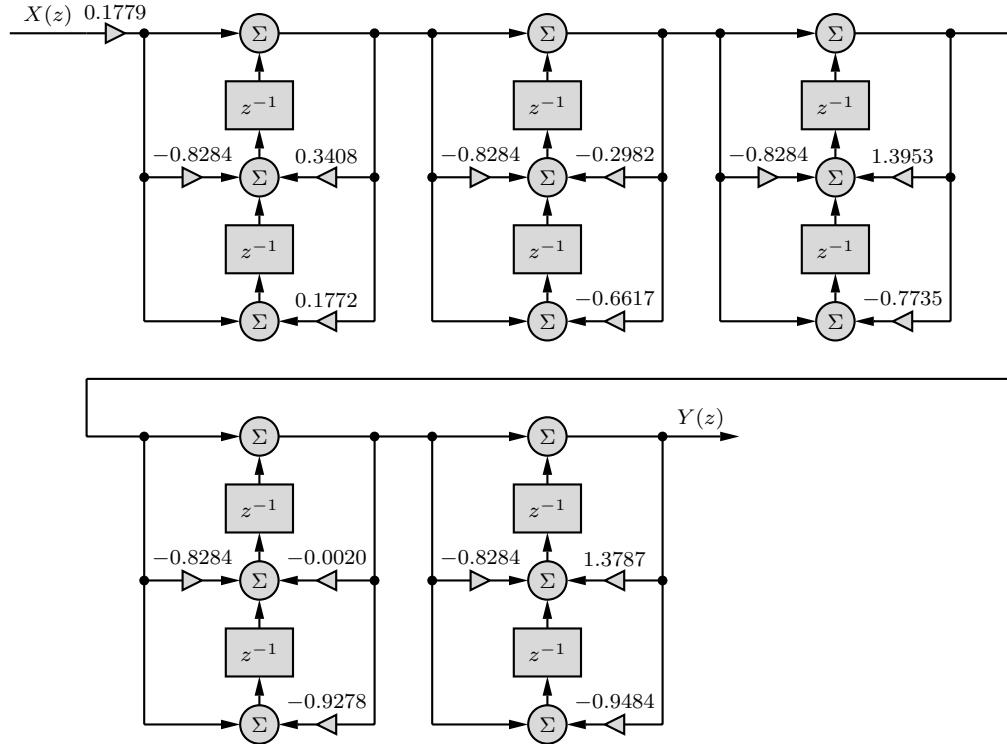


Figure D8.4

Drill 8.5 (Realization of Type III and Type IV FIR Filters)

Multiplier-efficient realizations of type III and type IV linear phase FIR filters are shown in Figs. D8.5a and D8.5b, respectively.

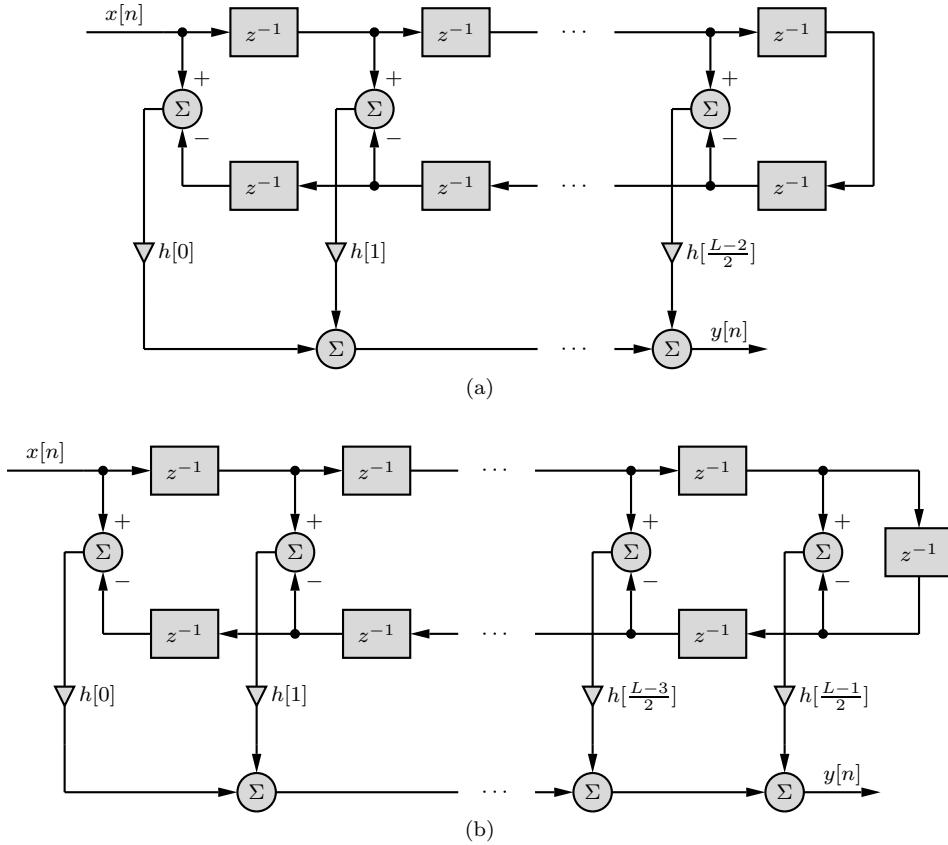


Figure D8.5

Drill 8.6 (Window Method Lowpass Filter Design)

The rectangular and Hamming lowpass filter designs are easily accomplished with minor modifications to the code of Ex. 8.11. Since order is one less than length, each order-98 filter possesses a length of $L_h = 99$. Due to this large length, we plot rather than print the impulse response coefficient values. Magnitude response plots help highlight the behavioral differences between the two filters.

```

01 Lh = 99; fc = 20000; T = 1/(4*fc); n = 0:Lh-1;
02 hc = @(t) 1/(2*T)*sinc(t/(2*T)); wrec = @(n) 1.0*((n>=0)&(n<=Lh-1));
03 hrec = T*hc(n*T-(Lh-1)/2*T).*wrec(n); subplot(221); stem(n,hrec);
04 wham = @(n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
05 hham = T*hc(n*T-(Lh-1)/2*T).*wham(n); subplot(222); stem(n,hham);
06 Omega = linspace(-pi,pi,1001);
07 Hrec = polyval(hrec,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
08 Hham = polyval(hham,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
09 subplot(223); plot(Omega,abs(Hrec)); subplot(224); plot(Omega,abs(Hham));

```

As shown in Figs D8.6a and D8.6b, the impulse responses for the rectangular and Hamming windows are quite similar. Owing to the taper of the Hamming window, the primary difference is that the Hamming window impulse response is smoother at its edges. Looking at the magnitude response plots of Figs D8.6c and D8.6d, we see that the rectangular window filter possesses a rapid transition band but rather significant passband and stopband ripples. The Hamming window filter, on the other hand, has markedly less passband and stopband ripple but a more gradual transition band.

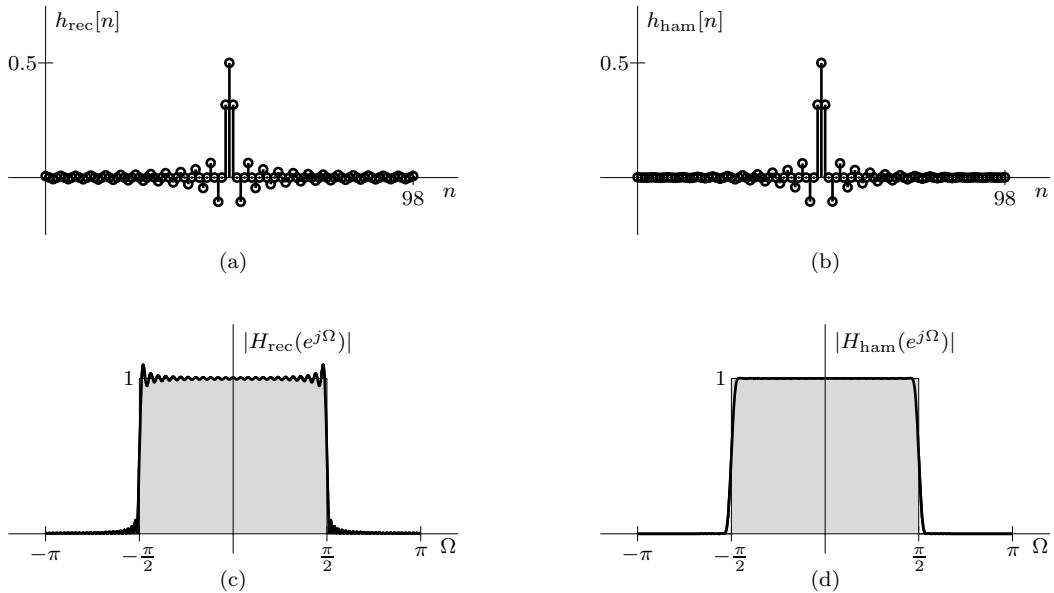


Figure D8.6

Drill 8.7 (Triangular Window Lowpass Filter Design)

To obtain the digital cutoff frequency of $\Omega_c = \frac{\pi}{2}$, we require the impulse response $h_c(t)$ of a suitable CT lowpass filter with cutoff frequency ω_c . Since no system sampling rate is specified, we can choose any ω_c and T such that $\Omega_c = \omega_c T$. Let us choose $T = 1$ and $\omega_c = \frac{\pi}{2}$. Using pair 8 of Table 1.1, the impulse response of the desired ideal (zero phase) lowpass filter is

$$h_c(t) = \frac{\omega_c}{\pi} \text{sinc}\left(\frac{\omega_c t}{\pi}\right).$$

Following the approach of Ex. 8.11, we use MATLAB to complete the design. Line 01 defines the CT filter. Line 02 defines the filter length and, using entry 2 of Table 8.5, the required triangular window. Using Eq. (8.36), line 03 computes the desired DT impulse response.

```
01 Omegac = pi/2; T = 1; omegac = Omegac/T; hc = @(t) omegac/(pi)*sinc(omegac*t/pi);
02 Lh = 9; n = 0:Lh-1; wbar = @(n) 1-abs(2*n-(Lh-1))/(Lh-1).*((n>=0)&(n<=Lh-1));
03 h = T*hc(n*T-(Lh-1)/2*T).*wbar(n)
h = 0 -0.0265 0 0.2387 0.5000 0.2387 0 -0.0265 0
```

Using these results with Eq. (8.30), we thus confirm that the DT filter transfer function is

$$H(z) = -0.0265z^{-1} + 0.2387z^{-3} + 0.5z^{-4} + 0.2387z^{-5} - 0.0265z^{-7}.$$

Notice that since the triangular window is defined to be zero at its end points, this $L_h = 9$ filter looks more like a length-7 filter. Many sources alternately define the triangular window so as to avoid this situation. Indeed, there is some advantage to realizing this filter instead as

$$H(z) = -0.0265 + 0.2387z^{-2} + 0.5z^{-3} + 0.2387z^{-4} - 0.0265z^{-6}.$$

To conclude, we lastly compute and plot the magnitude response of the filter. As shown in Fig. D8.7, the response of this low-order filter only roughly approximates the desired ideal.

```
04 Omega = linspace(-pi,pi,1001);
05 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega); plot(Omega,abs(H));
```

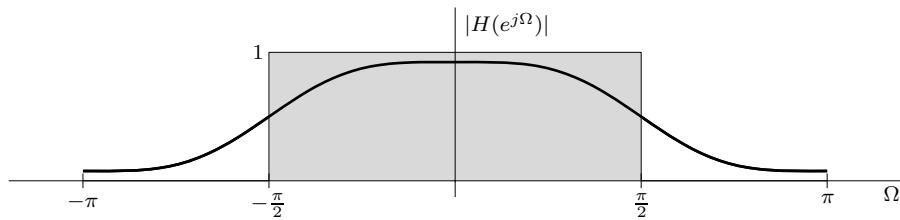


Figure D8.7

Drill 8.8 (Window Method Differentiator Design)

The desired Hann window differentiator is designed in nearly an identical fashion as the rectangular and Hamming window differentiators of Ex. 8.12. The only differences are changes in window type and length. Lines 01–06 design the $L_h = 49$ filter, whereas lines 07–11 design the $L_h = 50$ filter. The scaled impulse responses of the $L_h = 49$ and $L_h = 50$ filters are shown in Figs. D8.8a and D8.8c, respectively, and the corresponding scaled magnitude responses are shown in Figs. D8.8b and D8.8d. As in the case of the rectangular and Hamming window designs, the odd-length (type III) filter is not well behaved at $\Omega = \pm\pi$. The even-length (type IV) filter, however, closely approximates the desired differentiator behavior over the entire range $-\pi \leq \Omega \leq \pi$.

```

01 Lh = 49; n = 0:Lh-1; Omega = linspace(-pi,pi,1001);
02 whan = @(n) 0.5*(1-cos(2*pi*n/(Lh-1))).*(n>=0)&(n<=Lh-1));
03 Th49 = cos(pi*(n-(Lh-1)/2))./(n-(Lh-1)/2).*whan(n);
04 Th49(n==(Lh-1)/2) = 0;
05 TH49 = polyval(Th49,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
06 subplot(221); stem(n,Th49); subplot(222); plot(Omega,abs(TH49));
07 Lh = 50; n = 0:Lh-1;
08 whan = @(n) 0.5*(1-cos(2*pi*n/(Lh-1))).*(n>=0)&(n<=Lh-1));
09 Th50 = -sin(pi*(n-(Lh-1)/2))./(pi*(n-(Lh-1)/2).^2).*whan(n);
10 TH50 = polyval(Th50,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
11 subplot(223); stem(n,Th50); subplot(224); plot(Omega,abs(TH50));

```

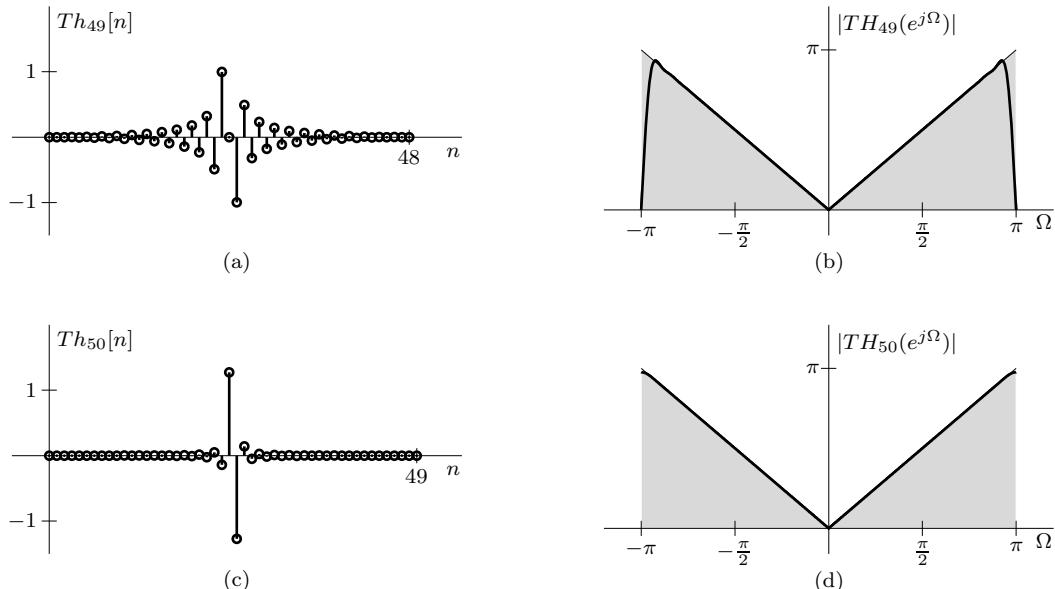


Figure D8.8

Drill 8.9 (Another Window Method Design)

In this design, we wish to design a length-7 FIR filter to realize $H_c(j\omega) = \Lambda(\omega T/\pi)$. Since $H_c(j\omega)$ is bandlimited to $B = \frac{\pi}{2T} < \frac{\pi}{T}$, the CT impulse response of Eq. (8.35) is computed using a simple inverse Fourier transform. Using pair 10 of Table 1.1, the CT impulse response is thus

$$h_c(t) = \frac{1}{4T} \text{sinc}^2\left(\frac{t}{4T}\right).$$

Using Eq. (8.36), the impulse response of the DT filter is

$$h[n] = Th_c([n - \frac{L_h-1}{2}]T)w_{\text{rec}}[n] = \frac{1}{4} \text{sinc}^2\left(\frac{1}{4}[n - \frac{L_h-1}{2}]\right) w_{\text{rec}}[n].$$

MATLAB allows us to readily compute and plot $h[n]$ and $|H(e^{j\Omega})|$.

```

01 Lh = 7; n = 0:Lh-1; Omega = linspace(-pi,pi,1001);
02 wrec = @ (n) 1.0*((n>=0)&(n<=Lh-1));
03 h = 0.25*(sinc((n-(Lh-1)/2)/4)).^2.*wrec(n)
    h = 0.0225 0.1013 0.2026 0.2500 0.2026 0.1013 0.0225
04 H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
05 subplot(121); stem(n,h); subplot(122); plot(Omega,abs(H));

```

Using the results of line 03 with Eq. (8.30), the DT filter transfer function is

$$H(z) = 0.0225 + 0.1023z^{-1} + 0.2026z^{-2} + 0.25z^{-3} + 0.2026z^{-4} + 0.1013z^{-5} + 0.0225z^{-6}.$$

Figures D8.9a and D8.9b show the impulse response $h[n]$ and magnitude response $|H(e^{j\Omega})|$, respectively. Despite its low order, this DT filter does a respectable job of approximating $H_c(j\omega) = \Lambda(\omega T/\pi)$. Increasing L_h improves the quality of the approximation.

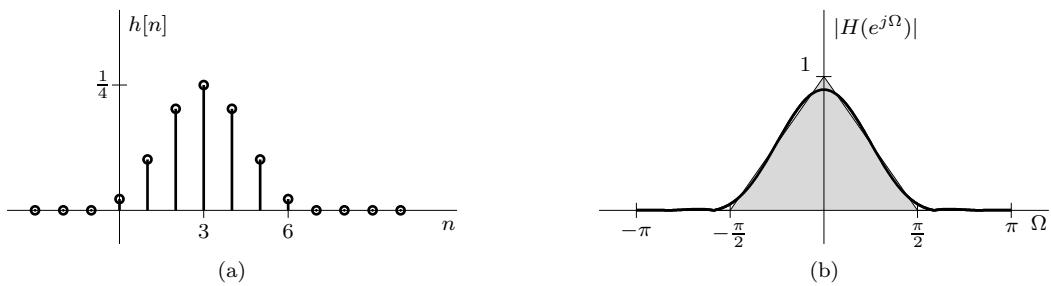


Figure D8.9

Drill 8.10 (Choosing between Small and Large Ripple Windows)

We can see from Table 8.6 that both the Hamming and Blackman windows should satisfy the design ripple requirements of $r_p \leq 0.1$ dB and $\alpha_s \geq 50$ dB. From the same table, we see that the given transition width of $\Delta\Omega = 0.05\pi$ suggests a filter length of $L_h = \frac{6.64\pi}{0.05\pi} + 1 = 134$ for the Hamming window and a length $L_h = \frac{11.13\pi}{0.05\pi} + 1 = 224$ for the Blackman window. Thus, while both filters can meet given requirements, the Hamming window produces a filter length 40% shorter than the Blackman window. Since filter length directly relates to filter complexity, the Hamming window is clearly superior to the Blackman window in this case.

Drill 8.11 (A Guaranteed Failure of the Window Method)

In this case, the difficulty is that the filter length is even yet the desired response needs to pass high frequencies ($\Omega = \pm\pi$). The frequency response $H_c(\omega)$ of the (bandlimited) highpass prototype is an even function, which means that the corresponding impulse response $h_c(t)$ is also even. Combined with a symmetric window, the window design method of Eq. (8.37) delivers a symmetric impulse

response $h[n]$. If $h[n]$ is odd length, the filter is necessarily type I. If $h[n]$ is even length, the filter is necessarily type II. From Table 8.4, we see that although a type I filter is suitable for all basic filter types, a type II filter is unsuitable for HP and BS filters since type II filters necessarily possess an odd number of zeros at $z = -1$. Thus, we see that the window design method using an even filter length cannot pass the desired frequencies $\Omega = \pm\pi$. Doubling the filter length does not help since that filter is also of even length. Increasing filter length by one, however, changes the filter from type II to type I and dramatically improves the response at high frequencies. Let us demonstrate these ideas with a simple example.

Consider a highpass filter with cutoff frequency $\Omega_c = \frac{\pi}{2}$ ($\omega_c = \frac{\pi}{2T}$). Using Eq. (8.35), the corresponding bandlimited CT impulse response is

$$h_c(t) = \frac{1}{2\pi} \left[\int_{-\frac{\pi}{T}}^{-\frac{\pi}{2T}} e^{j\omega t} d\omega + \int_{\frac{\pi}{2T}}^{\frac{\pi}{T}} e^{j\omega t} d\omega \right] = \frac{1}{T} \text{sinc}\left(\frac{t}{T}\right) - \frac{1}{2T} \text{sinc}\left(\frac{t}{2T}\right).$$

Using this result with Eq. (8.36), the desired DT filter impulse response is

$$h[n] = [\text{sinc}\left(n - \frac{L_h-1}{2}\right) - \frac{1}{2} \text{sinc}\left(\frac{n}{2} - \frac{L_h-1}{4}\right)] w[n].$$

Using a Hamming window for $w[n]$, let us investigate the $L_h = 20$ and $L_h = 21$ cases.

```

01 Lh = 20; n = 0:Lh-1;
02 wham = @(n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
03 h20 = (sinc(n-(Lh-1)/2)-0.5*sinc(n/2-(Lh-1)/4)).*wham(n);
04 Omega = linspace(-pi,pi,1001); H20 = polyval(h20,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
05 subplot(221); stem(n,h20); subplot(222); plot(Omega,abs(H20))
06 Lh = 21; n = 0:Lh-1;
07 wham = @(n) (0.54-0.46*cos(2*pi*n/(Lh-1))).*((n>=0)&(n<=Lh-1));
08 h21 = (sinc(n-(Lh-1)/2)-0.5*sinc(n/2-(Lh-1)/4)).*wham(n);
09 H21 = polyval(h21,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
10 subplot(223); stem(n,h21); subplot(224); plot(Omega,abs(H21))

```

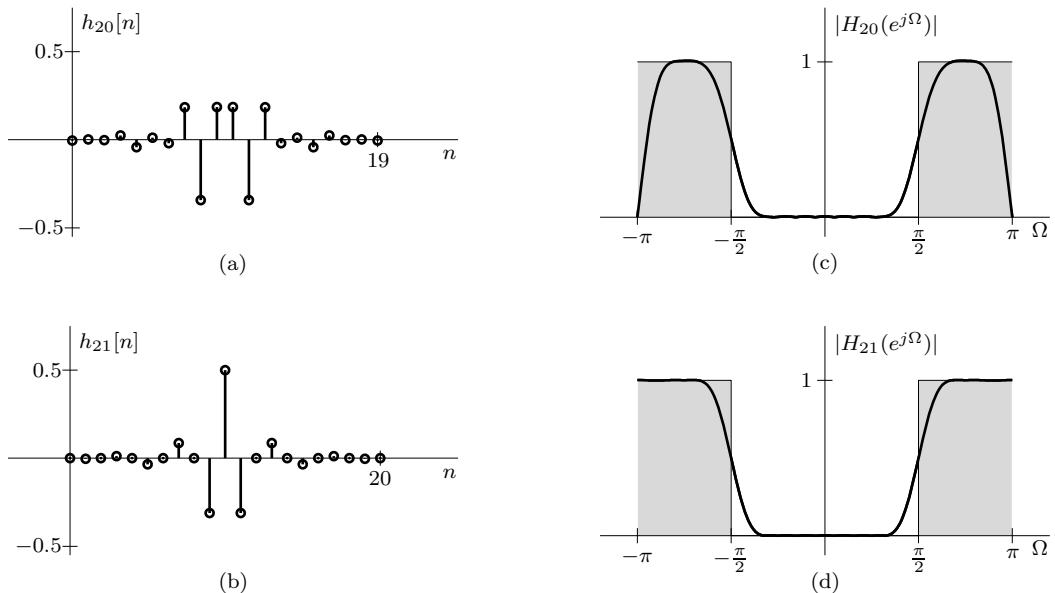


Figure D8.11

Figs. D8.11a and D8.11c show the impulse responses for $L_h = 20$ and $L_h = 21$, respectively. Both cases possess the expected midpoint symmetry. Figs. D8.11b and D8.11d show the corresponding

magnitude responses. The two cases produce nearly identical behavior at all but the highest frequencies. The $z = -1$ zero of the even-length (type II) filter, however, guarantees failure of this highpass filter's high-frequency response.

While this example concentrates on the highpass case, the same difficulties also occur with bandstop filters and multiband filters that attempt to pass $\Omega = \pm\pi$.

Drill 8.12 (Alternate Frequency Sampling Method Allpass Filter Designs)

To design an $L_h = 7$ type I allpass FIR filter, we need only change the frequency samples in the Ex. 8.18 MATLAB code from lowpass (ones and zeros) to allpass (all ones). In this case, we use four frequency samples beginning at $\Omega = 0$ and uniformly spaced by $2\pi/7$ to determine the four unknown coefficients of the impulse response; the remaining three values are obtained by symmetry. Figure D8.12a shows the resulting magnitude response and the frequency samples utilized in the design.

```
01 Lh = 7; n = (0:1:(Lh-1)/2); k = n'; Omega0 = 2*pi/Lh; Omegak = k*Omega0;
02 AI = 2*cos(Omegak*((Lh-1)/2-n(1:end-1))); AI = [AI,ones(length(Omegak),1)];
03 hI = inv(AI)*ones(size(Omegak)); hI = [hI;flipud(hI(1:end-1))]';
04 Omega = linspace(0,pi,1001); HI = polyval(hI,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
05 subplot(221); plot(Omega,abs(HI),Omegak,ones(size(Omegak)),',ko');
```

The design of an $L_h = 8$ type II allpass FIR filter is nearly identical to the design of an $L_h = 7$ type I allpass FIR filter. From Table 8.3, we infer that the only difference is in the final column of matrix \mathbf{A} : the type II case computes this column as the previous columns rather than using a column of ones. As in the $L_h = 7$ type I case, the $L_h = 8$ type II filter has four unknown coefficients of the impulse response (the remaining four values are obtained by symmetry). We obtain these values using four frequency samples beginning at $\Omega = 0$ and uniformly spaced by $2\pi/8$. Figure D8.12b shows the resulting magnitude response and the frequency samples utilized in the design. Although intended to be allpass, this type II filter is unable to pass high frequencies near $\Omega = \pi$, which is consistent with the restriction given in Table 8.4 that type II filters must have an odd number of zeros at $z = -1$.

```
06 Lh = 8; n = (0:1:(Lh-2)/2); k = n'; Omega0 = 2*pi/Lh; Omegak = k*Omega0;
07 AII = 2*cos(Omegak*((Lh-1)/2-n));
08 hII = inv(AII)* ones(size(Omegak)); hII = [hII;flipud(hII)]';
09 HII = polyval(hII,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
10 subplot(222); plot(Omega,abs(HII),Omegak,ones(size(Omegak)),',ko');
```

The design of an $L_h = 7$ type III allpass FIR filter is similar the previous type I and type II designs. From Table 8.3, we see that matrix \mathbf{A} is constructed using the sine function rather than the cosine function. Since the midpoint of a type III filter's impulse response is zero, an $L_h = 7$ type III filter has three unknown coefficients; the remaining coefficients are determined by antisymmetry. Since the sine of zero is always zero, we cannot use a frequency sample at $\Omega = 0$ in the design since it renders the matrix \mathbf{A} singular (non-invertible). Thus, in this case, we use three frequency samples beginning at $\Omega = 2\pi/7$ and uniformly spaced by $2\pi/7$. Figure D8.12c shows the resulting magnitude response and the frequency samples utilized in the design. Consistent with the Table 8.4 restrictions that type III filters possess an odd number of zeros at both $z = 1$ and $z = -1$, this filter can neither pass high frequencies nor low frequencies.

```
11 Lh = 7; n = (0:1:(Lh-3)/2); k = n'+1; Omega0 = 2*pi/Lh; Omegak = k*Omega0;
12 AIII = 2*sin(Omegak*((Lh-1)/2-n));
13 hIII = inv(AIII)*ones(size(Omegak)); hIII = [hIII;0;-flipud(hIII)]';
14 HIII = polyval(hIII,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
15 subplot(223); plot(Omega,abs(HIII),Omegak,ones(size(Omegak)),',ko');
```

The design of an $L_h = 8$ type IV allpass FIR filter is nearly identical to the design of an $L_h = 7$ type III allpass FIR filter. Since the midpoint of this filter is between samples, the impulse response does not require a value of zero in the middle, as in the case of a type III filter. Since an $L_h = 8$ type IV filter has four unknown coefficients (the remaining four values are obtained by antisymmetry), we

require four frequency samples, which in this case are taken beginning at $\Omega = 2\pi/8$ and uniformly spaced by $2\pi/8$. Figure D8.12d shows the resulting magnitude response and the frequency samples utilized in the design. Consistent with the Table 8.4 restriction that type IV filters possess an odd number of zeros at $z = 1$, this filter cannot pass low frequencies.

```

16 Lh = 8; n = (0:1:(Lh-2)/2); k = n'+1; Omega0 = 2*pi/Lh; Omegak = k*Omega0;
17 AIV = 2*sin(Omegak*((Lh-1)/2-n));
18 hIV = inv(AIV)*ones(size(Omegak)); hIV = [hIV;-flipud(hIV)]';
19 HIV = polyval(hIV,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
20 subplot(224); plot(Omega,abs(HIV),Omegak,ones(size(Omegak)), 'ko');

```

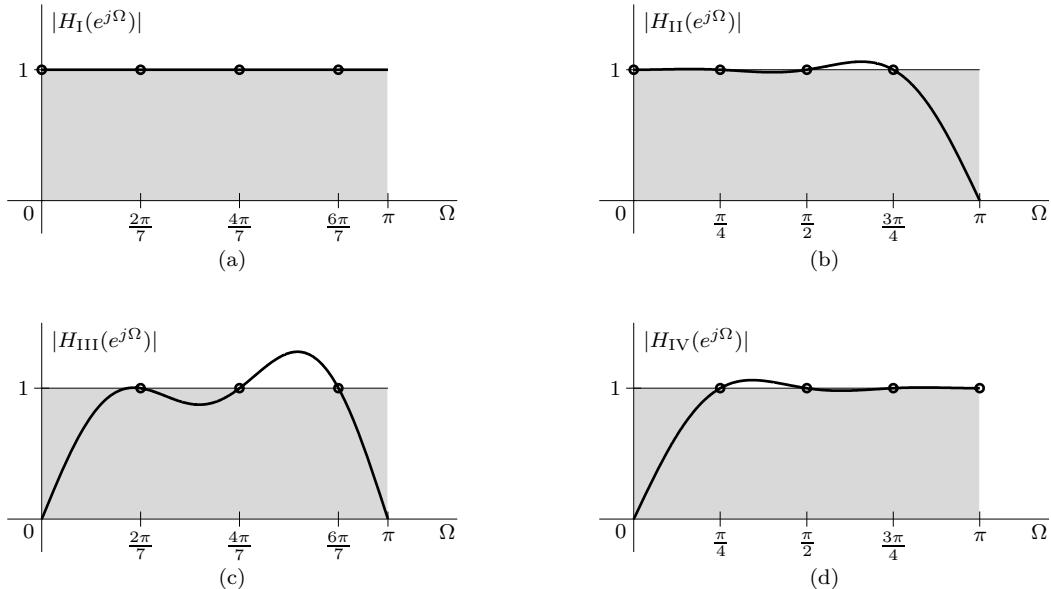


Figure D8.12

Although we utilize uniformly spaced frequency samples in these designs, this is not required. Different frequency samples yield different frequency responses, although the general limitations of the different filter types remain.

Drill 8.13 (Frequency-Weighted Least-Squares Highpass Filter Design)

The desired frequency-weighted least-squares (FWLS) highpass filter is designed using nearly identical MATLAB code to the FWLS LPF of Ex. 8.21. Aside from the obvious changes in parameter values Ω_p , Ω_s , and L_h , the main differences are the frequency-weighting function Q (line 03) and a highpass (rather than lowpass) desired frequency response H_d (line 05). The resulting impulse response $h[n]$ and dB magnitude response $20 \log_{10} |H(e^{j\Omega})|$ are shown in Figs. D8.13a and D8.13b, respectively. From line 12, we see that the actual passband ripple is $\delta_p = 1.4736(10)^{-4}$, which, using Eq. (8.40), corresponds to $r_p = 1.2799(10)^{-3}$ dB. This value is too small to visually discern in the magnitude response plot of Fig. D8.13b. From line 13, we see that the stopband ripple is $\delta_s = 1.6339(10)^{-5}$, which corresponds to a stopband attenuation of $\alpha_s = 95.7357$ dB. Notice that the stopband ripple is approximately ten times smaller than the passband ripple, which corresponds nicely to the frequency-weighting function Q being 10 times larger in the stopband as in the passband.

```

01 Omegap = pi/2; Omegas = 3*pi/8;
02 Lh = 101; K = 10*Lh; k = (0:K-1); Omegak = k*2*pi/K;
03 Q = 10.0*(Omegak<=Omegas)+1.0*(Omegak>=Omegap);

```

```

04 Q(fix(K/2)+2:end) = Q(round(K/2):-1:2);
05 Hd = 1.0*(Omegak>=0megap).*exp(-1j*k*pi*(Lh-1)/K);
06 Hd(fix(K/2)+2:end) = conj(Hd(round(K/2):-1:2));
07 l = (0:Lh-1)'; a = exp(1j*l*Omegak)*Q./K; b = exp(1j*l*Omegak)*(Hd.*Q/K).';
08 A = toeplitz(a); h = (A\b);
09 n = (0:Lh-1)'; subplot(211); stem(n,h);
10 Omega = linspace(0,pi,1001); H = polyval(h,exp(1j*Omega)).*exp(-1j*(Lh-1)*Omega);
11 subplot(212); plot(Omega/pi,20*log10(abs(H)));
12 deltap = 2*max(abs(abs(H(Omega>=0megap))-1))
deltap = 1.4736e-004
13 deltas = max(abs(H(Omega<=0megas)))
deltas = 1.6339e-005

```

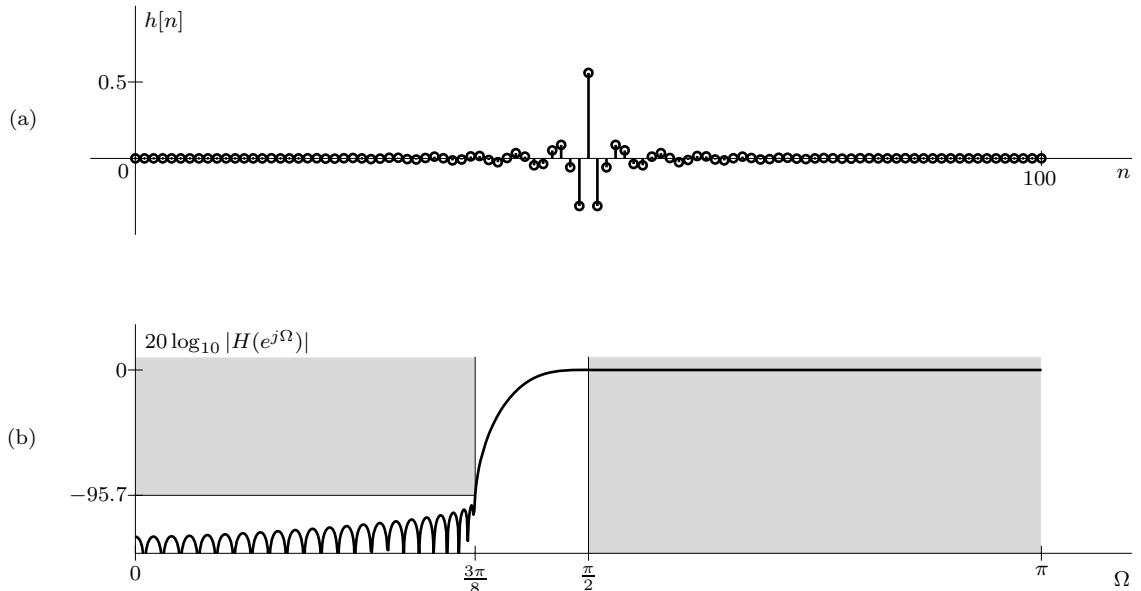


Figure D8.13

Chapter 9

Drill 9.1 (3-Point DFT of a Length-3 Signal)

Using Eq. (9.2), the DFT of $x[n] = 2\delta[n] + \delta[n - 1] + \delta[n - 2]$ is

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\Omega_0 kn} = 2 + e^{-j2\pi k/3} + e^{-j4\pi k/3} = 2 + 2 \cos(2\pi k/3).$$

Substituting $k = 0, 1$, and 2 yields

$$X[0] = 4, \quad X[1] = 1, \quad \text{and } X[2] = 1.$$

Using Eq. (6.1), the DTFT of $x[n]$ is

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} = 2 + e^{-j\Omega} + e^{-j2\Omega} = 2 + 2 \cos(\Omega/2)e^{-j3\Omega/2}.$$

Using MATLAB, we can readily verify that the DFT $X[k]$ is equal to samples of the DTFT $X(\Omega)$ at intervals of $\Omega_0 = 2\pi/N = 2\pi/3$.

```

01 N = 3; Omega0 = 2*pi/N; k = (0:N-1); Xk = 2+2*cos(2*pi*k/3)
    Xk = 4 1 1
02 Omega = linspace(0,2*pi,6001); XOmega = 2+2*cos(Omega/2).*exp(-1j*3*Omega/2);
03 subplot(121); stem(k*Omega0,abs(Xk)); line(Omega,abs(XOmega));
04 subplot(122); stem(k*Omega0,angle(Xk)); line(Omega,angle(XOmega));

```

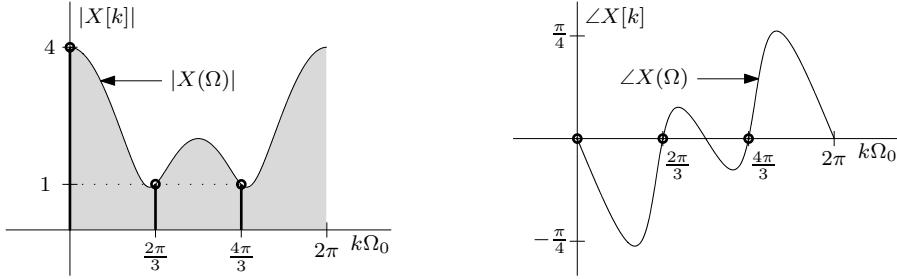


Figure D9.1

Drill 9.2 (3-Point and 8-Point DFTs of a Length-3 Signal)

- (a) Using Eq. (6.1), the DTFT of $x[n] = \delta[n] + \delta[n - 1] + \delta[n - 2]$ is

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} = 1 + e^{-j\Omega} + e^{-j2\Omega} = e^{-j\Omega}[1 + 2 \cos(\Omega)].$$

- (b) Using Eq. (9.2) and $\Omega_0 = 2\pi/3$, the 3-point DFT of $x[n]$ is

$$X[k] = \sum_{n=0}^2 x[n]e^{-j2\pi kn/3} = 1 + e^{-j2\pi k/3} + e^{-j4\pi k/3} = e^{-j2\pi k/3}[1 + 2 \cos(2\pi k/3)].$$

Substituting $k = 0, 1$, and 2 yields

$$X[0] = 3, \quad X[1] = 0, \quad \text{and } X[2] = 0.$$

Using MATLAB, we can readily verify that the DFT $X[k]$ is equal to samples of the DTFT $X(\Omega)$ at intervals of $\Omega_0 = 2\pi/3$, as shown in Fig. D9.2a. From the results of line 01, we notice that $X[2]$ and $X[3]$ are computed not quite exactly as zero. This computer rounding error causes the aberrant phase computations at $k = 2$ and $k = 3$.

```

01 N = 3; Omega0 = 2*pi/N; k = (0:N-1); Xk = exp(1j*2*pi*k/3).*(1+2*cos(2*pi*k/3))
    Xk = 3.00 -0.00+0.00i 0.00+0.00i
02 Omega = linspace(0,2*pi,6001); XOmega = exp(-1j*Omega).* (1+2*cos(Omega));
03 subplot(221); stem(k*Omega0,abs(Xk)); line(Omega,abs(XOmega));
04 subplot(222); stem(k*Omega0,angle(Xk)); line(Omega,angle(XOmega));

```

- (c) Using Eq. (9.2), the 8-point DFT of $x[n]$ is

$$X[k] = \sum_{n=0}^7 x[n]e^{-j\pi kn/4} = 1 + e^{-j\pi k/4} + e^{-j2\pi k/4} = e^{-j\pi k/4}[1 + 2 \cos(\pi k/4)].$$

Using MATLAB, we verify the conjugate symmetry of $X[k]$ about $k = N/2 = 4$, which is to say that $X[1] = X^*[7]$, $X[2] = X^*[6]$, and $X[3] = X^*[5]$. Figure D9.2b graphically shows that $X[k]$ is equal to samples of the DTFT $X(\Omega)$ at intervals of $\Omega_0 = \pi/4$.

```

05 N = 8; Omega0 = 2*pi/N; k = (0:N-1); Xk = exp(-1j*pi*k/4).*(1+2*cos(pi*k/4))
Xk = 3.00 1.71-1.71i -1.00i 0.29+0.29i 1.00 0.29-0.29i 1.00i 1.71+1.71i
06 subplot(223); stem(k*Omega0,abs(Xk)); line(Omega,abs(XOmega));
07 subplot(224); stem(k*Omega0,angle(Xk)); line(Omega,angle(XOmega));

```

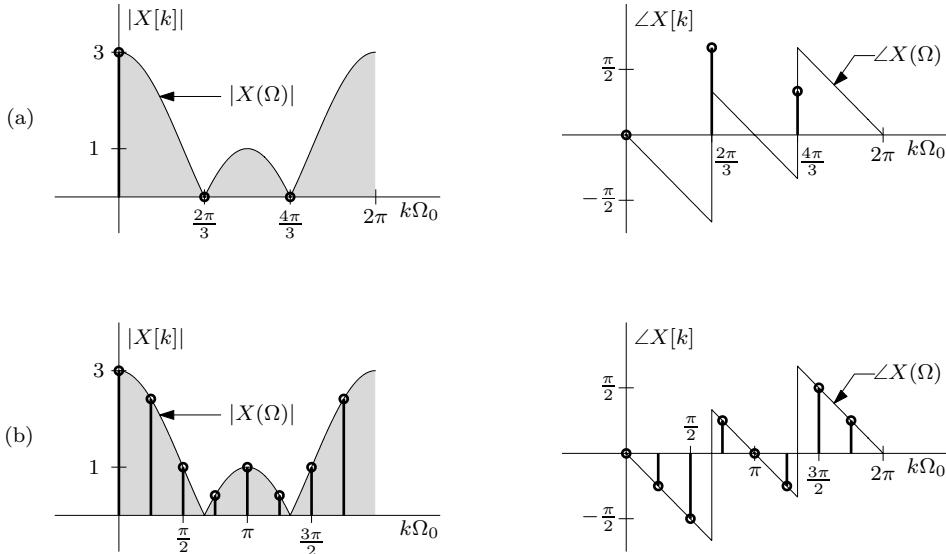


Figure D9.2

Drill 9.3 (Signal Length Reduction and Aliasing)

The 2-point DFT $\hat{X}[0] = 3$ and $\hat{X}[1] = 1$, obtained by undersampling the DTFT of the 3-point signal $x[n] = \delta[n] + \delta[n - 1] + \delta[n - 2]$, has IDFT

$$\hat{x}[n] = \frac{1}{2} \sum_{k=0}^1 \hat{X}[k] e^{j\pi kn} = \frac{3}{2} + \frac{1}{2} e^{j\pi n}.$$

Substituting $n = 0$ and 1 yields

$$\hat{x}[0] = 2 \quad \text{and} \quad \hat{x}[1] = 1.$$

By undersampling the original 3-point signal's DTFT, the duration of the corresponding time-domain signal $\hat{x}[n]$ is reduced. Viewed another way, undersampling in frequency produces aliasing in the time domain. In this case, the $n = 2$ sample of the length-3 original signal $x[n] = \delta[n] + \delta[n - 1] + \delta[n - 2]$ aliases to $n = 0$ to produce the length-2 signal $\hat{x}[n] = 2\delta[n] + \delta[n - 1]$.

The DTFT of $\hat{x}[n]$ is

$$\hat{X}(\Omega) = \sum_{n=0}^1 \hat{x}[n] e^{-j\Omega n} = 2 + e^{-j\Omega}.$$

This DTFT is clearly different from the original 3-point signal DTFT $X(\Omega) = e^{-j\Omega}[1 + 2 \cos(\Omega)]$, although both functions are equal at $\Omega = 0$ and $\Omega = \pi$.

Drill 9.4 (DFT Interpolation to Obtain the DTFT)

Due to the cumbersome nature of the Eq. (9.10) interpolation formula, we use Eq. (9.8) to interpolate

the 2-point DFT $X[0] = 3$ and $X[1] = 1$ as

$$\begin{aligned}
 X(\Omega) &= \frac{1}{2} \sum_{k=0}^1 X[k] \sum_{n=0}^1 e^{-j(\Omega-\pi k)n} \\
 &= \frac{1}{2} \sum_{k=0}^1 X[k] \left(1 + e^{-j(\Omega-\pi k)} \right) \\
 &= \frac{1}{2} \left[3 (1 + e^{-j\Omega}) + 1 (1 + e^{-j(\Omega-\pi)}) \right] \\
 &= \frac{1}{2} [3 + 3e^{-j\Omega} + 1 - e^{-j\Omega}] \\
 &= 2 + e^{-j\Omega}.
 \end{aligned} \tag{C.1}$$

Equation (9.10) yields, after somewhat tedious simplification, the same result.

Drill 9.5 (Nonuniqueness of DTFT Samples)

From Drill 9.1, we know that signal $x[n] = 2\delta[n] + \delta[n - 1] + \delta[n - 2]$ has DTFT given by $X(\Omega) = 2 + 2\cos(\Omega/2)e^{-j3\Omega/2}$ and DFT given by

$$X[0] = 4, \quad X[1] = 1, \quad \text{and } X[2] = 1.$$

The DTFT of signal $y[n] = \delta[n + 1] + 2\delta[n] + \delta[n - 1]$ is

$$Y(\Omega) = \sum_{n=-1}^1 y[n]e^{-j\Omega n} = e^{j\Omega} + 2 + e^{-j\Omega} = 2 + 2\cos(\Omega).$$

Sampling this DTFT at intervals of $\Omega_0 = 2\pi/3$ produces

$$Y(0\Omega_0) = 4, \quad Y(1\Omega_0) = 1, \quad \text{and } Y(2\Omega_0) = 1.$$

Thus, although $Y(\Omega) \neq X(\Omega)$, the samples $Y(k\Omega_0)$ exactly match the DFT samples $X[k]$.

Drill 9.6 (Modulo- N Shift and DTFT Sample Equivalence)

The length-3 signal $x[n]$, defined as $x[0] = 2$, $x[1] = 1$, and $x[2] = 1$, ranges over $0 \leq n \leq 2$. The length-3 signal $y[n]$, defined as $y[-1] = 1$, $y[0] = 2$, and $y[1] = 1$, ranges over $-1 \leq n \leq 1$. A modulo-3 shift of $y[n]$, which relocates $y[n]$ to the range $0 \leq n \leq 2$, only requires that the sample $y[-1]$ be moved to its modulo-3 location of $\langle -1 \rangle_3 = 2$. Since $y[0] = x[0]$, $y[1] = x[1]$ and $y[-1] = x[2]$, the modulo-3 shift of $y[n]$ equals $x[n]$.

As discussed in Sec. 9.2.1, if an N -point signal is the modulo- N shift of another signal, then the N DTFT samples of both signals exactly match. Since the three DTFT samples of signal $y[n]$ match the three DTFT samples of $x[n]$ (see Drill 9.5), it is not surprising that $y[n]$ is modulo-3 equivalent to $x[n]$.

There are an infinite number of signals that are modulo-3 shift equivalent to $x[n]$. Let us determine a 3-point signal $z[n]$ with an arbitrarily chosen range of $20 \leq n \leq 22$ that is modulo-3 shift equivalent to $x[n]$. Since $\langle 20 \rangle_3 = 2$, $\langle 21 \rangle_3 = 0$, and $\langle 22 \rangle_3 = 1$, we set

$$z[20] = x[2], \quad z[21] = x[0], \quad \text{and } z[22] = x[1].$$

Defined in this way, $z[n]$ is modulo-3 equivalent to $x[n]$, and the N DTFT samples of both signals are identical.

Drill 9.7 (Circular Representations of 3-Point Signals)

From Drill 9.6, we see that the 3-point signal $x[n]$ is defined as $x[0] = 2$, $x[1] = 1$, and $x[2] = 1$.

Similarly, 3-point signal $y[n]$ is defined as $y[-1] = 1$, $y[0] = 2$, and $y[1] = 1$. Figure D9.7a and D9.7b show the circular representations of $x[n]$ and $y[n]$, respectively. Since $y[n]$ is modulo-3 shift equivalent to $x[n]$ (see Drill 9.6), the circular representation of $x[n]$ is identical to the circular representation of $y[n]$.

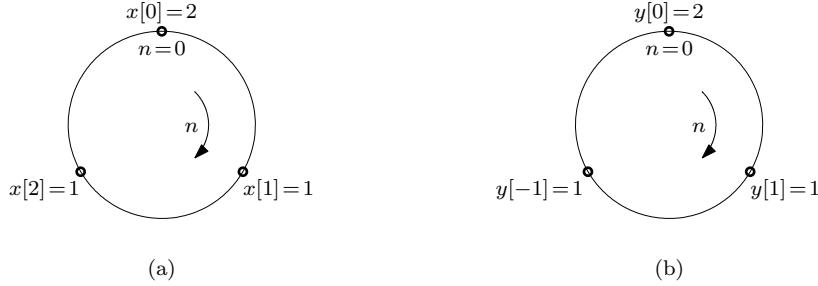


Figure D9.7

Drill 9.8 (Modulo- N Reflection)

To begin, Figure D9.8a shows the reflection $x[-n]$ of the 3-point signal $x[n]$ specified by $x[1] = 1$, $x[2] = 2$, and $x[3] = 3$. Over $0 \leq n \leq 2$ (heavy lines), Fig. D9.8b shows $x[\langle -n \rangle_N]$ obtained by transporting the samples of $x[-n]$ to their modulo-3 locations, as per Eq. (9.13). Since all three nonzero samples of $x[-n]$ are outside the range $0 \leq n \leq 2$, every sample needs to be relocated.

In the second method, we periodically replicate the Fig. D9.8a signal $x[-n]$. As shown in Fig. D9.8b (light lines), this periodic replication produces $x[\langle -n \rangle_N]$ over the principal range $0 \leq n \leq N - 1$.

In the third method, we use a circular representation of $x[-n]$, as depicted in Fig. D9.8c. This is achieved by placing the samples of $x[n]$ in a *counterclockwise direction* along a circular dial to represent $x[-n]$. We now read this diagram in a *clockwise direction* to obtain $x[\langle -n \rangle_N]$. We read the values at $n = 0, 1$, and 2 as $x[3], x[2]$, and $x[1]$, respectively. This matches Fig. D9.8b, the result of the first two methods.

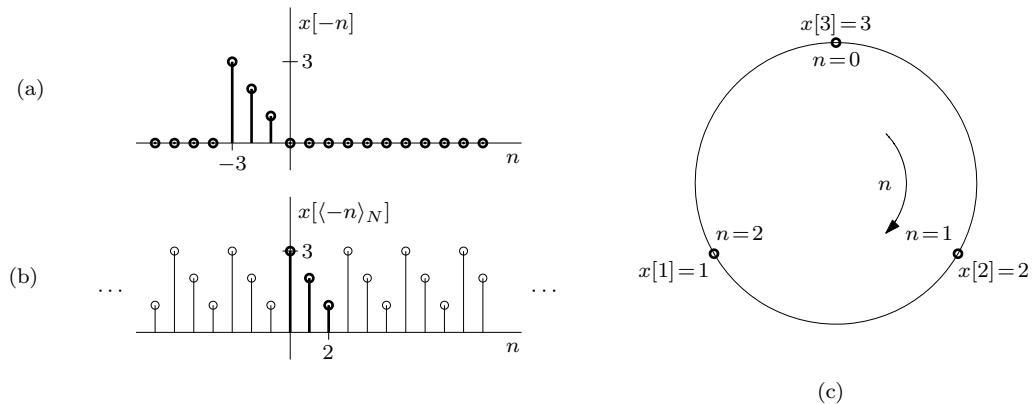


Figure D9.8

Drill 9.9 (Circular Shift)

Figure D9.9a shows the $m = 4$ shift $x[n - 4]$ of the 3-point signal $x[n]$ specified by $x[0] = 2$ and $x[1] = x[2] = 1$. Over $0 \leq n \leq 2$ (heavy lines), Fig. D9.9b shows $x[\langle n - 4 \rangle_N]$ obtained by transporting the samples of $x[n - 4]$ to their modulo-3 locations, as per Eq. (9.13).

In the second method, we periodically replicate the Fig. D9.9a signal $x[n - 4]$. As shown in Fig. D9.9b (light lines), this periodic replication produces $x[\langle n - 4 \rangle_N]$ over the principal range $0 \leq n \leq N - 1$ (heavy lines).

In the third method, we use a circular representation of $x[n - 4]$, as depicted in Fig. D9.9c. This is achieved by rotating the circular representation of $x[n]$ clockwise by $m = 4$ positions. This result is equivalent to Fig. D9.9b, the result of the first two methods.

Since $\langle -1019 \rangle_3 = \langle 4 \rangle_3 = 1$, a shift by $m = -1019$ is identical to the shift by $m = 4$.

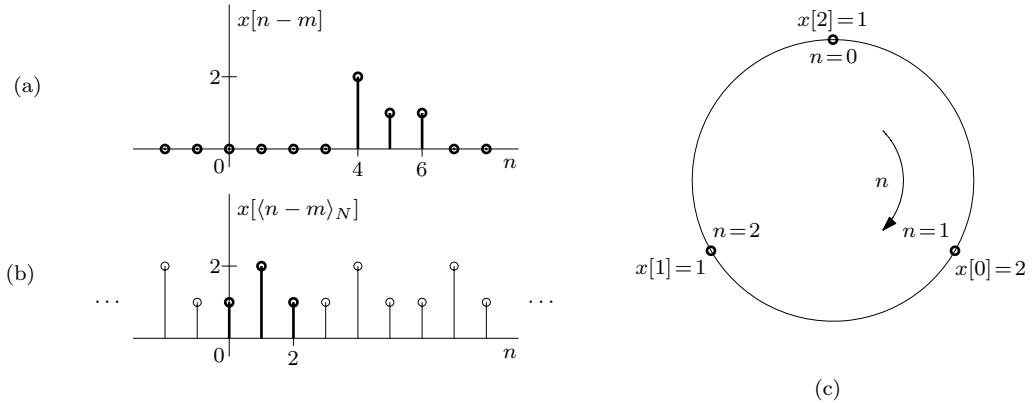


Figure D9.9

Drill 9.10 (Combined Modulo- N Reflection and Shift)

Let us use three methods to determine $x[\langle -n - 2 \rangle_N]$ of the 6-point signal $x[n]$ shown in Fig. 9.10a.

Figure D9.10a combines reflection and shift to produce $x[-n - 2]$. To obtain $x[\langle -n - 2 \rangle_N]$, we transport the samples of $x[-n - 2]$ to their modulo-6 locations, as shown in Fig. D9.10b (heavy lines).

In the second method, we periodically replicate the Fig. D9.10a signal $x[-n - 2]$. As shown in Fig. D9.10b (light lines), this periodic replication produces $x[\langle -n - 2 \rangle_N]$ over the principal range $0 \leq n \leq N - 1$ (heavy lines).

In the third method, we produce a circular representation of $x[-n - 2]$, as depicted in Fig. D9.10c. This is achieved by first rotating the circular representation of $x[n]$ clockwise by 2 positions (shift) and then reflecting this result about the vertical axis (reflection). This result is equivalent to the Fig. D9.9b result of the first two methods.

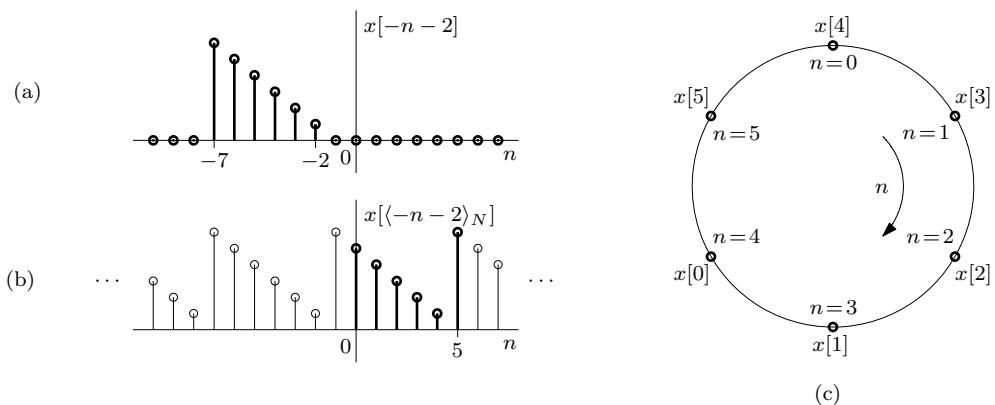


Figure D9.10

Drill 9.11 (Simplified Modulo- N Reflection)

Accepting that $x[0] = x[N]$, the modulo- N reflection is easily obtained from Eq. (9.18) as $x[\langle -n \rangle_N] = x[N-n]$. In the present case, we desire $x[\langle -n \rangle_N]$ for the length-3 signal $x[n]$ specified by $x[0] = 3$, $x[1] = 2$, and $x[2] = 1$. Thus,

$$\{x[\langle -n \rangle_N]\}_{n=0}^2 = \{x[N-n]\}_{n=0}^2 = \{x[3], x[2], x[1]\} = \{3, 1, 2\}.$$

Drill 9.12 (Conjugate-Antisymmetry Property)

From the time-domain complex conjugation property of Eq. (9.24), we know that the DFT of $x^*[n]$ is $X^*[N-k]$. Now, when signal $x[n]$ is purely imaginary, $x[n] = -x^*[n]$. Taking the DFT of this equation therefore yields the desired result of

$$X[k] = -X^*[N-k].$$

For an imaginary signal $x[n]$, the spectrum $X[k]$ is midpoint conjugate antisymmetric.

Drill 9.13 (Computing Circular Convolution Graphically)

Over $0 \leq n \leq 5$, the 6-point signals $x[n]$ and $h[n]$ are $[-1, 1, 2, -2, 1, 1]$ and $[1, 2, 3, 4, 5, 6]$, respectively. The 6-point circular convolution $y[n] = x[n] \circledast h[n]$ is expressed mathematically as

$$y[n] = \sum_{m=0}^5 x[m]h[\langle n-m \rangle_6].$$

Let us use the graphical method to compute the desired value $y[2]$. To do this, we represent $x[m]$ and $h[\langle 2-m \rangle_6]$ using two concentric circles. We take the inner circle as the (fixed) circular representation of $x[m]$, drawn in the regular clockwise manner. The outer circle, which represents $h[\langle 2-m \rangle_6]$, is obtained by first writing $h[m]$ in the counterclockwise direction to obtain the modulo- N reflection $h[\langle -m \rangle_6]$. This outer circle is then rotated clockwise by two positions to obtain $h[\langle 2-m \rangle_6]$. Both circular representations are shown in Fig. D9.13.

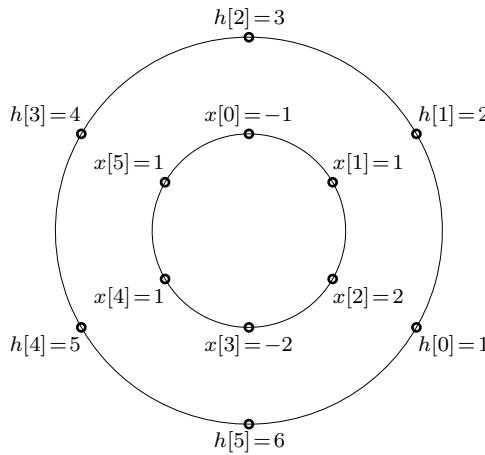


Figure D9.13

To compute $y[2]$, we multiply the corresponding points on our concentric circles and sum the results,

$$\begin{aligned} y[2] &= x[0]h[2] + x[1]h[1] + x[2]h[0] + x[3]h[5] + x[4]h[4] + x[5]h[3] \\ &= -1(3) + 1(2) + 2(1) - 2(6) + 1(5) + 1(4) = -2. \end{aligned}$$

To further verify that $y[2] = -2$, we compute the entire 6-point circular convolution in MATLAB as $y[n] = \text{IDFT}\{\text{DFT}(x[n])\text{DFT}(h[n])\}$, $0 \leq n \leq 5$.

```
01 x = [1;2;3;4;5;6]; h = [-1;1;2;-2;1;1]; N = length(x); n = (0:N-1);
02 k=(0:N-1); Omega0 = 2*pi/N; DN = exp(-1j*Omega0*k'*n);
03 X = DN*x; H = DN*h; y = conj(DN)/N*(X.*H)
y = 12 8 -2 12 8 4
```

The same result is also obtained using the built-in MATLAB functions `fft` and `ifft` to compute the DFT and IDFT, respectively.

```
04 y = ifft(fft([-1,1,2,-2,1,1]).*fft([1,2,3,4,5,6]))
y = 12 8 -2 12 8 4
```

Drill 9.14 (Aliasing in Circular Convolution)

We determine the desired 3-point circular convolution using the DFT and IDFT as

$$[3, 2, 3] \circledast [1, 2, 3] = \text{IDFT}\{\text{DFT}([3, 2, 3])\text{DFT}([1, 2, 3])\}.$$

This expression is conveniently computed using MATLAB.

```
01 N = 3; n = (0:N-1); k=(0:N-1); Omega0 = 2*pi/N; DN = exp(-1j*Omega0*k'*n);
02 conj(DN)/N*((DN*[3;2;3]).*(DN*[1;2;3]))
ans = 15 17 16
```

Perhaps more simply, the same result is also obtain using the built-in MATLAB functions `fft` and `ifft` to compute the DFT and IDFT, respectively.

```
03 ifft(fft([3;2;3]).*fft([1;2;3]))
ans = 15 17 16
```

We can verify this result graphically with two concentric circles. On the inner circle, we represent $[3, 2, 3]$ in the standard way, moving clockwise from a 12 o'clock starting position. On the outer circle, we represent $[1, 2, 3]$ beginning at the 12 o'clock position and then moving counterclockwise (modulo reflection of the signal). To compute the circular convolution for time n , the outer circle is rotated clockwise by n positions, corresponding points on the two circles are multiplied together, and then the results are summed. Using Fig. D9.14 as our guide, the 3-point circular convolution is thus computed as

$$[1, 2, 3] \circledast [3, 2, 3] = \begin{cases} 3(1) + 2(3) + 3(2) = 15 & n = 0 \\ 3(2) + 2(1) + 3(3) = 17 & n = 1 \\ 3(3) + 2(2) + 3(1) = 16 & n = 2 \end{cases}.$$

To compute the linear convolution $[3, 2, 3] * [1, 2, 3]$, we again turn to MATLAB.

```
02 conv([3,2,3],[1,2,3])
ans = 3 8 16 12 9
```

This result can also be obtained by graphical means such as the sliding-tape method.

The linear convolution of two signals can, through aliasing, be used to compute the circular convolution of those same two signals. To do this, we perform a modulo- N shift of the linear convolution and add together any overlapping points. In the present example, we modulo-3 shift the linear convolution result $[3, 8, 16, 12, 9]$ ($0 \leq n \leq 4$). The first three values are already in their modulo-3 positions, so they do not move. However, the $n = 3$ value of 12 needs to be moved (aliased) to $\langle 3 \rangle_3 = 0$, where it is added to the existing value of 3 to produce 15. Similarly, the $n = 4$ value of 9 aliases to $\langle 4 \rangle_3 = 1$, where it is added to the existing value of 8 to produce 17. The 3-point result $[15, 17, 16]$ exactly matches the original 3-point circular convolution.

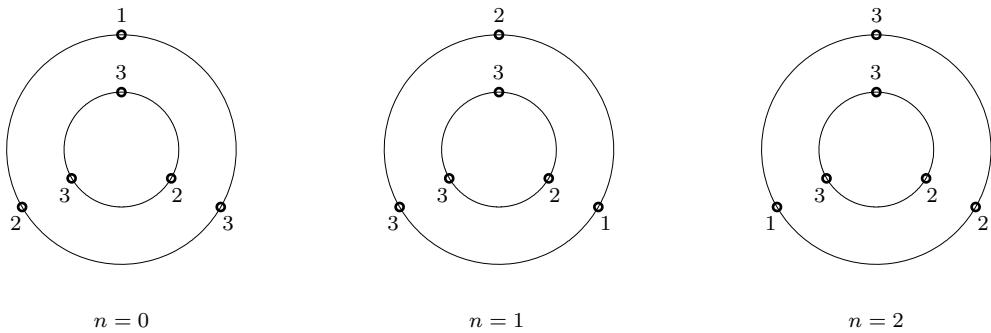


Figure D9.14

Drill 9.15 (Convolution Using the DFT)

We can compute the linear convolution $x[n] * h[n]$ using the DFT and IDFT as

$$x[n] * h[n] = \text{IDFT} \{ \text{DFT}(x_{\text{pad}}) \text{DFT}(h_{\text{pad}}) \}.$$

This expression is conveniently computed using MATLAB code similar to that in Ex. 9.11.

```
01 x = [1;1;1;1]; h = [1;2;1]; Nx = length(x); Nh = length(h);
02 xpad = [x;zeros(Nh-1,1)]; hpad = [h;zeros(Nx-1,1)];
03 N = Nx+Nh-1; n = (0:N-1); k=(0:N-1); Omega0 = 2*pi/N; DN = exp(-1j*Omega0*k'*n);
04 Xpad = DN*xpad; Hpad = DN*hpad; y = conj(DN)/N*(Xpad.*Hpad),
y = 1 3 4 4 3 1
```

To verify this result, we compute the convolution $x[n] * h[n]$ using the sliding-tape method, as illustrated in Fig. D9.15. As expected, the results of both methods are identical.

$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 0$	$y[0] = 1(1) = 1$
$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 1$	$y[1] = 1(2) + 1(1) = 3$
$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 2$	$y[2] = 1(1) + 1(2) + 1(1) = 4$
$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 3$	$y[3] = 1(1) + 1(2) + 1(1) = 4$
$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 4$	$y[4] = 1(1) + 1(2) = 3$
$\begin{array}{ c c c c }\hline & 1 & 1 & 1 & 1 \\ \hline 1 & 2 & 1 & \end{array} \rightarrow n = 5$	$y[5] = 1(1) = 1$

Figure D9.15

Drill 9.16 (Block Convolution)

In this problem, we use two methods of block convolution to determine the output of an LTID system

with FIR impulse response $h[n] = 3\delta[n] + 2\delta[n-1] + 3\delta[n-2]$ and input $\{x[n]\}_{n=0}^{\infty} = \{1, 0, -1, 2, \dots\}$. In both cases, we segment the input into nonoverlapping blocks of length 2.

For the overlap-and-add method, we start by zero padding each length-2 input block with $N_h - 1 = 2$ zeros, as shown at the top of Fig. D9.16a. Similarly, we use a single zero to pad the impulse response $h[n]$ to the same total length of 4. Next, we use the DFT to convolve the zero-padded input blocks with the zero-padded impulse response. Figure D9.16a shows the resulting output blocks, computed using MATLAB.

```

01 Nx = 2; Nh = 3; N = Nx+Nh-1; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1);
02 DN = exp(-1j*Omega0*k'*n); x0pad = [1;0;zeros(Nh-1,1)]; hpad = [3;2;3;zeros(Nx-1,1)];
03 X0pad = DN*x0pad; Hpad = DN*hpad; y0 = conj(DN)/N*(X0pad.*Hpad)
y0 = 3 2 3 0
04 x1pad = [-1;2;zeros(Nh-1,1)]; X1pad = DN*x1pad; y1 = conj(DN)/N*(X1pad.*Hpad)
y1 = -3 4 1 6

```

By adding the overlapping output blocks together, we obtain the final output $\{y[n]\}_{n=0}^{\infty} = \{3, 2, 0, 4, \dots\}$, also shown in Fig. D9.16a.

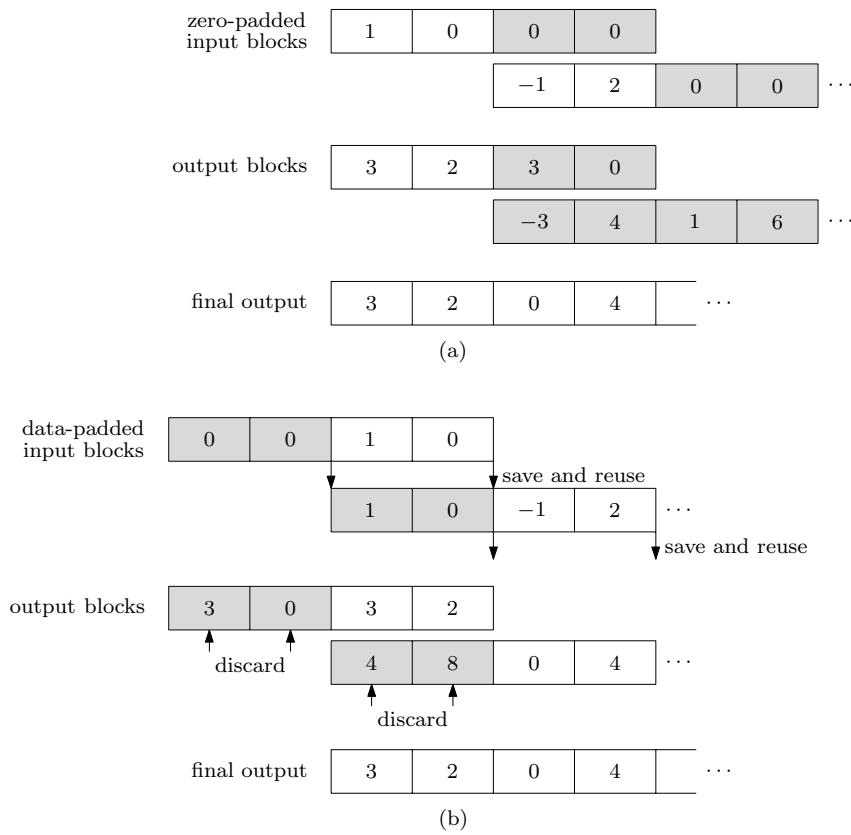


Figure D9.16

For the overlap-and-save method, we again segment the input into nonoverlapping length-2 blocks. Except for the first block, which uses zeros, these blocks are data-padded at their beginnings with $N_h - 1 = 2$ previous input values, as shown at the top of Fig. D9.16b. As in the overlap-and-add method, we use a single zero to pad the impulse response $h[n]$ to the same total length of 4. Next, we use the DFT to convolve the data-padded input blocks with the zero-padded impulse response. Figure D9.16b shows the resulting output blocks, computed using MATLAB.

```
05 x0pad = [zeros(Nh-1,1);1;0]; X0pad = DN*x0pad; y0 = conj(DN)/N*(X0pad.*Hpad)
```

```

y0 = 3 0 3 2
06 x1pad = [x0pad(end-1:end);-1;2]; X1pad = DN*x1pad; y1 = conj(DN)/N*(X1pad.*Hpad)
y1 = 4 8 0 4

```

By discarding the first $N_h - 1 = 2$ values of each output block, we obtain the final output $\{y[n]\}_{n=0}^{\infty} = \{3, 2, 0, 4, \dots\}$, as shown in Fig. D9.16b.

Drill 9.17 (Determining DIT FFT Input Order)

The input position of a 128-point DIT FFT requires a 7-bit number that ranges from 0 to 127. Counting from zero, the fourth input position is identified by the decimal value 3, which written as a 7-bit binary number is 0000011. Bit reversing this number yields binary 110000, which is 96 in decimal. Thus, the fourth input to a 128-point DIT FFT is $x[96]$.

Similarly, the 100th input position is identified by the decimal value 99, which is written in binary as 1100011. Bit reversing this number produces the same binary number, 1100011. Thus, the 100th input to a 128-point DIT FFT is $x[99]$.

Drill 9.18 (DTFS Using an Alternate Frequency Interval)

In this case, the fundamental frequency is $\Omega_0 = 0.1\pi$. Referring to Fig. 9.35, over $2\pi < \Omega \leq 2\pi$, there are two nonzero spectral coefficients, $\mathcal{D}[21] = -\frac{j}{2}$ and $\mathcal{D}[39] = \frac{j}{2}$. Using Eq. (9.56), the DTFS is thus computed as

$$\tilde{x}[n] = \sum_{k=21}^{40} \mathcal{D}[k] e^{j0.1\pi kn} = -\frac{j}{2} e^{j2.1\pi n} + \frac{j}{2} e^{j3.9\pi n}.$$

Due to the 2π periodicity of the complex exponential, we know that

$$e^{j2.1\pi n} = e^{j(2.1-2)\pi n} = e^{j0.1\pi n} \quad \text{and} \quad e^{j3.9\pi n} = e^{j(3.9-4)\pi n} = e^{-j0.1\pi n}.$$

Making these substitutions into our DTFS, we obtain

$$\tilde{x}[n] = -\frac{j}{2} e^{j0.1\pi n} + \frac{j}{2} e^{-j0.1\pi n} = \sin(0.1\pi n).$$

Clearly, the DTFS synthesis using the interval $2\pi < \Omega \leq 4\pi$ is equivalent to that of Eq. (9.58), which uses the interval $-\pi \leq \Omega < \pi$.

Drill 9.19 (Discrete-Time Fourier Series of a Sum of Two Sinusoids)

To begin, we write our signal as

$$\tilde{x}[n] = 4 \cos(2\pi \frac{1}{10} n) + 6 \sin(2\pi \frac{1}{4} n).$$

The first sinusoid has period 10, and the second sinusoid has period 4. To determine the period N of the combined signal $\tilde{x}[n]$, we simply compute the least common multiple of the two periods,

$$N = \text{LCM}(4, 10) = 20.$$

Thus, $\Omega_0 = \frac{2\pi}{N} = \frac{\pi}{10}$ and

$$\tilde{x}[n] = 4 \cos(2\Omega_0 n) + 6 \sin(5\Omega_0 n) = 2e^{j2\Omega_0 n} + 2e^{-j2\Omega_0 n} - 3je^{j5\Omega_0 n} + 3je^{-j5\Omega_0 n}.$$

This result is the DTFS of $\tilde{x}[n]$ constructed over $-\pi \leq \Omega < \pi$ ($-10 \leq k < 10$). To construct $\tilde{x}[n]$ over $0 \leq \Omega < 2\pi$, we need to exchange the spectral components over $-\pi \leq \Omega < 0$ ($-10 \leq k < 0$) with the equivalent components from the range $\pi \leq \Omega < 2\pi$ ($10 \leq k < 20$). This requires that we replace the $k = -2$ component with $k = -2 + N = 18$ and the $k = -5$ component with $k = -5 + N = 15$. Thus, constructed over $0 \leq \Omega < 2\pi$, the DTFS is

$$\tilde{x}[n] = 2e^{j2\Omega_0 n} - 3je^{j5\Omega_0 n} + 3je^{j15\Omega_0 n} + 2e^{j18\Omega_0 n}, \quad \text{where } \Omega_0 = \frac{\pi}{10}.$$

Drill 9.20 (Computing the DTFS Spectrum Using the DFT)

The DTFS spectrum is just the DFT scaled by $\frac{1}{N}$. Thus, we compute the DTFS spectrum using the scaled DFT as $\mathcal{D}_k = \frac{1}{N} \mathbf{D}_N \mathbf{x}$, where \mathbf{D}_N is the DFT matrix and vector \mathbf{x} is the signal $\tilde{x}[n]$ over $0 \leq n \leq N - 1$.

To demonstrate the procedure, let us use the DFT and MATLAB to compute the DTFS of the Ex. 9.17 periodic sampled gate function $\tilde{x}[n]$, which is shown in Fig. 9.36.

```

01 x = [ones(5,1);zeros(23,1);ones(4,1)];
02 N = 32; Omega0 = 2*pi/N; k = (0:N-1); n = (0:N-1);
03 DN = exp(-1j*Omega0*k'*n); Dk = (DN/N)*x;
04 stem(k*Omega0,Dk);

```

The results, shown in Fig. D9.20, exactly match the Fig. 9.37 results of Ex. 9.17.

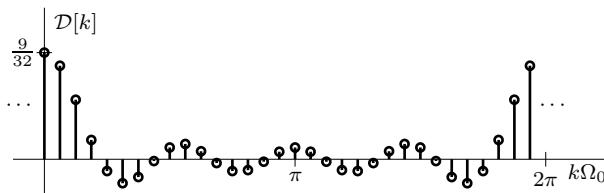


Figure D9.20

Index

- accumulator systems, 239, 240, 302, 447
as digital integrators, 243
- adders, 238, 447
- additivity property, 26, 250
distinction from homogeneity, 27
- adjustable windows, 521
- advance form, 240, 245, 254, 275
- advance-operator form, 275
- aliasing, 170–173, 373
disguised in s -to- z mapping, 224
downsampling and, 217, 257, 379, 384
fundamental-band representation and, 341
impulse invariance method and, 487–491
in circular convolution, 588–590, 597
in exponentials, 226–230
in sinusoids, 173–176, 226–230
multiple folding and, 171
prevented by filtering, 170
relation to apparent frequency, 174
sub-Nyquist sampling and, 180, 378
time-domain, 566
treachery of, 169
- allpass systems, 95, 542
- amplitude modulation, 58
- amplitude response, 513, 540
- analog filter transformations
digital filters designed by, 485–507
lowpass-to-bandpass, 117–118
lowpass-to-bandstop, 118–120
lowpass-to-highpass, 116–117
lowpass-to-lowpass, 115–116
- analog filters, 85–150, 247
digital realization of, 370
families of, 120–149
ideal, 100
practical specification of, 112
- analog resampling, 393
- analog signals, 3, 185
digital processing of, 370–379
- analog systems, *see* continuous-time systems
- analog-to-digital conversion, 185–199, 247
- analog-to-digital converters
bipolar, 189
- counting, 197
- dynamic errors, 194
- effective number of bits, 195
- flash, 198
- gain errors, 194
- missing codes, 195
- nonlinearity errors, 195
- offset errors, 194
- quantization errors, 189
- saturation errors, 189
- static errors, 194
- successive approximation, 197
- timing jitter, 195
- transfer characteristics of, 189
- unipolar, 189
- analysis equation
discrete Fourier transform, 560, 584, 600
discrete-time Fourier transform, 331, 356, 584
- Fourier series, 34, 67
- Fourier transform, 45, 67, 356
- Laplace transform, 69, 72
- z -transform, 397, 410, 434
- angles, *see* phase spectrum
- anti-aliasing filters, 170
importance of type, 173
in digital processing of analog signals, 371
used in decimation, 217, 257, 380, 384
- anti-causal signals, 15, 231
- anticipative systems, *see* systems, noncausal
- anti-mirror image polynomials, 514
- antisymmetry, *see* conjugate antisymmetry; odd symmetry
- aperiodic signals, 21, 233
- DT Fourier transform and, 331
- Fourier transform and, 45
- aperture effect, 200–201
- apparent frequency, 173–176, 226–230
multiple folding to determine, 175
- associative property, 292
- asymmetric quantization, *see* quantization, asymmetric
- asymptotic stability, *see* internal stability

- attenuation, 112, 150
 - maximum passband, 112
 - minimum stopband, 112, 529
- audio signals, 96, 187
 - nonlinear quantization of, 193
 - resampling, 380
- autocorrelation function
 - continuous-time, 61
 - discrete-time, 354
- auxiliary conditions, 278, 318, 438
- backward difference systems, 243, 255, 447
- backward difference transform, 491–492
- bandlimited interpolation, 367, 397
 - understanding resampling using, 380
- bandlimited signals, 156, 169, 171, 370
- bandpass filters, 465, 501, 504
 - group delay and, 97–99
 - ideal, 100, 362
 - impulse invariance method and, 489
 - poles and zeros of, 466
 - practical specification of, 112
 - restrictions of linear phase FIR, 514
 - transformation of lowpass to, *see filter transformations*
 - window method and, 533
- bandstop filters, 463, 465, 501, 506
 - ideal, 100, 362
 - impulse invariance method and, 374, 489
 - poles and zeros of, 467
 - practical specification of, 112
 - restrictions of linear phase FIR, 514
 - transformation of lowpass to, *see filter transformations*
 - window method and, 533
- bandwidth, 169, 368, 610
 - audio signal, 187
 - bandpass signal, 97, 176
 - discrete-time Fourier transform and, 347, 365
 - downsampling effect on, 381
 - essential, 63
 - Fourier transform and, 50, 53, 365
 - relationship between time constant and, 315
 - sampling theorem and, 156, 158, 161, 200
 - window functions and, 104, 109, 518
- Bartlett window, *see triangular window*
- basis signals, 69, 545
- basis vectors, 545
- baud rate, 186
- Bessel polynomials, reverse, 148
 - table of, 148
- Bessel-Thomson filters, 147–149
- BIBO stability, *see bounded-input bounded-output stability*
- bilateral Laplace transform, *see Laplace transform*
- bilateral z -transform, *see z-transform*
- bilinear transform, 491–497
 - discrete-time filter specifications and the, 500
 - simplified procedure for the, 499
 - with prewarping, 497–501
- binary signals, 3, 185, 608
- binomial coefficient, 249
- bit-reversed addressing, 608
- bits, 185
- Blackman window, 110, 520, 530
- block convolution, 593–599
 - overlap-and-add method, 594–597
 - overlap-and-save method, 594, 597–599
- block diagrams, 1, 445–457
 - analog-to-digital converters, 196
 - continuous-time systems, 25, 93
 - digital signal processing systems, 2
 - digital-to-analog converters, 199
 - discrete-time systems, 238, 356
 - sampling and reconstruction, 158, 162
- bounded-input bounded-output stability, 29, 256, 306, 444
 - asymptotic stability not guaranteed by, 309
 - frequency response and, 86, 355, 419
 - relation to internal stability, 307–309
 - steady-state response and, 458
 - transfer function and, 32, 444
- branch nodes, 238
- buffer zones, 125
- butterfly, 604
- Butterworth filters, 120–129
 - comparing Chebyshev and, 130
 - determination of cutoff frequency, 124
 - determination of order, 124
 - pole locations, 121
 - transfer function, 121
 - transformation to digital, 485–507
- Butterworth polynomials, normalized, 122
 - factored-form table of, 123
 - table of, 122
- canonical direct form, *see direct form II realization*
- canonical realizations, 450, 451, 508
- cascade
 - compressor and expander order of, 258–261

- identity systems created by, 257, 301, 446
interpolator and decimator, 394
loading effects, 445
second-order sections, 508
cascade realizations, 300, 445, 448, 453
 performance of, 457
cascade systems, 445
causal signals, 15, 231, 290
causal systems, 29, 254, 290, 447, *see also* causality
 causality, 29, 254
 realizable systems requirement of, 101, 362, 486
 unilateral Laplace transform and, 69
 unilateral z -transform and, 416
characteristic equations, 278
 of nonrecursive systems, 282
characteristic modes, 278, 282
 impulse response and, 285
 relationship between system behavior and, 311–317
characteristic polynomials, 278
characteristic roots, 32, 278
 of nonrecursive systems, 282
 relationship between system behavior and, 311–317
 stability determined from, 32, 307, 444
characteristic values, *see* characteristic roots
Chebyshev filters, 129–139
 determination of order, 131
 inverse, *see* inverse Chebyshev filters
 pole locations, 131
 ripple parameter, 130
 table of denominator coefficients, 133
 transfer function, 131
 transformation to digital, 485–507
Chebyshev polynomials, 129
circular convolution, 68, 351
 aliasing in, 588–590
 graphical interpretation of, 583
 linear convolution by, 585
 zero padding and, 586
circular convolution property, 581–582
circular correlation property, 582
circular shift, 575–578, *see also* modulo- N shift
circular shifting property, 580
clock jitter, 195
commutative property, 291
compact disc, 187
complex-conjugation property
 discrete Fourier transform, 580, 584
 discrete-time Fourier transform, 343, 356, 584
Fourier series, 67
Fourier transform, 52, 67, 356
Laplace transform, 72
 z -transform, 427, 434
complex exponential functions, *see* exponential functions
complex filters
 discrete Fourier transform computed using, 601
complex frequency, 12, 222, 396
 Laplace transform and, 69
 relating s and z , 222
 z -transform and, 410
complex frequency plane, 13, 222, 306, 396
 poles and zeros located in the, 90, 462
complex inputs, 294
complex poles, 422, *see* poles, complex
complex roots, 280
 realization of conjugate pairs, 454
complex signal processing, 17
complex zeros, *see* zeros, complex
compression
 continuous-time, 5, 53
 digital resampling and, 257
 discrete-time, 216, 258, 394, *see also* downsampling
 μ -law, 193
compressors, 257, *see also* compression
 block representation of, 258, 379
 order of expanders and, 258–261
computational complexity
 discrete Fourier transform, 600
 fast Fourier transform, 603, 608
 Goertzel's algorithm, 601–603
conformable, 630
conformal mapping, 492
conjugate antisymmetry, 20, 54, 233, 343
conjugate symmetry, 20, 54, 233, 343
conjugate-symmetry property
 discrete Fourier transform, 580
constant-parameter systems, *see* time-invariant systems
continuous-time Fourier transform, *see* Fourier transform
 discrete Fourier transform to estimate the, 610
continuous-time signals
 defined, 3
 discrete-time systems and, 213, 370
 exponential functions, 12–13

- Fourier analysis of, 33–68
 Laplace transform of, 68–73
 models of, 7–15
 sinc function, 13–15
 sinusoids, 12
 size of, 21
 unit gate function, 8
 unit impulse function, *see* Dirac delta function
 unit step function, 7
 unit triangle function, 8
 continuous-time systems, 25, 26, *see also* analog filters
 analog systems compared with, 370
 differential equation description of, 28, 89
 discrete-time systems compared with, 213
 discrete-time systems to approximate, 372
 frequency response of, 85–92
 Laplace transform analysis of, 31
 periodic inputs and, 34
 properties of, 25–30
 signal transmission through, 92–99
 stability of, 29, 32
 total response of, 27
 zero-input response of, 27
 zero-state response of, 27, 28
 continuous-to-discrete conversion, 213, 247, 371,
 see also analog-to-digital conversion
 controllability, 32, 309
 convergence
 in the mean, 41, 335
 of Fourier series, 41–42
 point-wise, 41
 region of, *see* region of convergence
 uniform, 41, 335
 convolution, 29, 253, 290
 block, *see* block convolution
 circular, *see* circular convolution
 fast, 590, 617
 overlap-and-add method, *see* block convolution
 overlap-and-save method, *see* block convolution
 polynomial multiplication by, 299
 with an impulse, 292
 zero-state response and, 28, 252, 288
 convolution integral, 29
 convolution property, 433
 discrete Fourier transform circular, 581, 584
 discrete-time Fourier transform, 351–354,
 356, 584
 discrete-time Fourier transform circular, 351–354, 356, 584
 Fourier series, 67
 Fourier transform, 59, 60, 67, 356
 Laplace transform, 72
 z -transform, 433, 434
 convolution sum, 253, 290
 from a table, 293
 graphical procedure, 294–296
 matrix form of, 327
 properties of the, 291
 sliding-tape method, 296–298
 Cooley, J. W., 603
 coprime, 234, 260, 394
 corner frequency, *see* cutoff frequency
 correlation
 continuous-time, 61
 convolution used to compute, 61
 discrete-time, 354
 importance of argument order in, 61
 correlation property
 discrete Fourier transform circular, 584
 discrete-time Fourier transform, 354, 356,
 584
 Fourier series, 67
 Fourier transform, 61, 67, 356
 counting converters, 197
 cross-correlation function
 continuous-time, 61
 discrete-time, 354
 cutoff frequency, 100, 315
 cyclic convolution, *see* circular convolution
 data truncation, 104–112
 dc component, 34
 decibel scale, 86
 decimation, 217, 380, 383–387
 decimation filters, 217, 257
 frequency response of ideal, 384
 impulse response of ideal, 391
 interpolation filter combined with, 261
 decimation-in-frequency algorithm, 609
 decimation-in-time algorithm, 604–608
 decimators, 257, 384
 block representation of, 258, 379
 interpolators in cascade with, 394–395
 output of ideal, 391
 decomposition property, 27, 251
 deconvolution, 327
 delay, 238, 447, *see also* shifting
 bandlimited, 374
 beneficial effect of filter, 102–103

- ideal, 31, 87, 94, 147
- delay form, 240, 254, 358, 440
- delay-operator form, 275
- delta modulation, 380
- deterministic signals, 25
- difference equations, 239, 245, 485
 - advance form, 240, 245, 254, 275
 - block realizations of, 447–457
 - classical solution of, 317
 - delay form, 240, 254, 255, 358, 440
 - differential equation kinship with, 246
 - ensuring causality of, 254
 - frequency response from, 358
 - initial conditions, 270
 - iterative solutions to, 270
 - linearity of, 251
 - nonrecursive form, 239, 273
 - order of, 245
 - preferred form, 255
 - recursive form, 239, 273
 - recursive solutions to, 270
 - z -transform solution of, 436–445
- differential equations, 28, 89
 - difference equation kinship with, 246
- differentiation property
 - discrete-time Fourier transform, 350, 356
 - Fourier series, 67
 - Fourier transform, 59, 67, 356
 - Fourier transform time, 59, 67
 - Laplace transform, 72
 - z -transform z -domain, 433, 434
- differentiators
 - backward difference transform and digital, 492
 - bandlimited digital, 377
 - bilinear transform unsuited to design digital, 496
 - digital, 241
 - ideal, 31, 88, 496, 527
 - stability of, 256
 - window method design of digital, 526
- digital filters, 247, 248, 465–468, 485–553
 - discrete-time systems referred as, 247
 - ideal, 362
- digital processing of analog signals, 370–379
- digital resampling, 257, 379–395
- digital signal processing, 370–379
 - advantages of, 248
 - block diagram of, 2
- digital signals, 3, 185, 248, *see also* analog-to-digital conversion
- binary, 185–194
- L -ary, 3, 185
- digital systems, *see* discrete-time systems
- digital-to-analog conversion, 199–202
- Dirac delta function, 9
 - connection to the unit step function, 10
 - impulse sampling using the, 155–161
 - properties of the, 10
- Dirac, P. A. M., 9
- direct form I realizations, 255, 448
- direct form II realizations, 449–450, 508
- direct form realizations, 447–450
 - transposed, 451–453
- Dirichlet conditions
 - Fourier series, 42
 - Fourier transform, 47
- Dirichlet kernel, 104
- discrete Fourier transform, 538, 559–569
 - analysis equation, 560, 584, 600
 - bins, 561
 - continuous-time Fourier transform estimated with the, 610
 - direct, 560, 561
 - discrete-time Fourier series equivalence to the, 612
 - discrete-time Fourier transform obtained by interpolation of the, 567–569
 - fast Fourier transform, 603
 - frequency resolution of the, 563
 - Goertzel's algorithm, 600–603
 - inverse, 561
 - matrix representation of the, 565–566
 - Parseval's theorem, 583
 - periodicity of the, 570
 - picket fence effect and the, 563
 - properties of the, 579–583
 - synthesis equation, 561, 584, 600
 - table of properties, 584
 - uniqueness, 569–572
 - zero padding and the, 563–565
- discrete-time Fourier series, 612–617
 - analysis equation, 612
 - discrete Fourier transform equivalence to the, 612
 - synthesis equation, 612
- discrete-time Fourier transform, 331–342
 - analysis equation, 331, 356, 584
 - continuous-time Fourier transform connection to the, 338, 364–370
 - difficulties of the, 559
 - direct, 332
 - existence of the, 335
 - generalization to z -transform, 395–397

- generalized limits of integration, 338
- inverse, 332
- LTID system analysis and the, 355
- nuisance of periodicity, 340
- obtained by discrete Fourier transform interpolation, 567–569
- Parseval's theorem, 355
- periodic and continuous nature of the, 337–338
- properties of the, 343–355
- synthesis equation, 332, 338, 356, 584
- table of pairs, 340
- table of pairs using the fundamental band, 341
- table of properties, 356, 584
- discrete-time sequences, *see* discrete-time signals
- discrete-time signals
 - apparent frequency of, 226
 - circular representation of, 573
 - defined, 3, 212
 - exponential functions, 222–230
 - Fourier analysis of, 331, 560
 - inherently bandlimited, 228
 - models of, 219–230
 - sinusoids, 222, 225–226, 233
 - size of, 236
 - unit impulse function, *see* Kronecker delta function
 - unit step function, 219
 - z -transform of, 410
- discrete-time systems, 1, 212, *see also* digital filters
 - alternate names for, 247
 - block diagrams of, 445–457
 - continuous-time systems approximated by, 372
 - continuous-time systems compared with, 213
 - difference equations of, 245, 254, 485
 - discrete-time Fourier transform analysis of, 355–358
 - examples of, 238–245
 - frequency response of, 304, 358, 457–468
 - intuitive insights into, 311–317
 - properties of, 248–257
 - signal transmission through, 359–362
 - stability of, 256, 305–311, 444
 - total response of, 250
 - z -transform analysis of, 410–485
 - zero-input response of, 250
 - zero-state response of, 250, 252
- discrete-to-continuous conversion, 213, 247, 371, *see also* digital-to-analog conversion
- distortion, 96
- distortionless transmission, 94, 359
 - in bandpass systems, 97, 360–362
 - measure of delay variation, 95, 360
- division
 - polynomial long, 425
- downsampling, 216, 344, 379
 - block representation of, 257, 379
 - foolishness of, 384
 - fractional sampling rate conversion and, 394
 - frequency-domain perspective of, 379, 383–387
 - time-domain perspective of, 391
- drill exercises with MATLAB, *see* MATLAB drill exercises
- duality
 - discrete-time Fourier transform and, 343
 - time-frequency, 50, 66, 68, 394
- duality property
 - discrete Fourier transform, 566, 579, 584
 - Fourier transform, 51, 67, 356
- duobinary pulse, 208
- dynamic errors, 194
- dynamic systems, 256
- effective number of bits, 195
- eigenfunctions, 31
- eigenvalues, 31, 278, *see also* characteristic roots
- elliptic filters, 144–147
 - transfer function, 145
- elliptic rational function, 145
- energy
 - computed as an inner product, 545
 - essential bandwidth and, 63
 - frequency-domain representation of, 61, 354, 582
 - of continuous-time signals, 21
 - of discrete-time signals, 236
- energy signals
 - all practical signals must be, 25
 - continuous-time, 22
 - discrete-time, 236
 - periodic replication of, 181
- energy spectral density, 62, 355
- envelope delay, *see* group delay
- equiripple
 - functions, 129
 - passband, 129, 144
 - stopband, 139, 144
- error

- minimum norm-squared, 545–546
essential bandwidth, 63
Euler's formula, 12
 variations of, 17
even symmetry, 18, 54, 232, 344
 in Fourier analysis, 66
 linear phase FIR filters with, 512
everlasting exponentials
 discrete Fourier transform and, 560
 discrete-time Fourier transform and, 332,
 419
 Fourier series and, 34
 Fourier transform and, 45, 70
 Laplace transform and, 70
 LTIC system response to, 30, 85
 LTID system response to, 303, 357
 z-transform and, 411, 419
everlasting signals, 16, 231
examples with MATLAB, *see* MATLAB examples
expanders, 257, 389, *see also* expansion
 block representation of, 258, 379
 order of compressors and, 258–261
expansion
 continuous-time, 5, 53
 digital resampling and, 257
 discrete-time, 218, 258, 394, *see also* upsampling
exponential Fourier series, *see* Fourier series
exponential functions
 apparent laziness of discrete-time, 229–230
 continuous-time, 12–13
 discrete-time, 222–230
 everlasting, *see* everlasting exponentials
 relationship between sinusoids and, 12
exponential inputs
 LTIC system response to, 30, 85
 LTID system response to, 303, 321, 357
external description of a system, 32, 444
external inputs
 system response to, *see* zero-state response
external stability, *see* bounded-input bounded-output stability

fast convolution, 590, 617
fast Fourier transform, 603–612
 butterfly, 604
 decimation-in-frequency algorithm, 609
 decimation-in-time algorithm, 604
 FIR filter design by the, 538
 mixed-radix, 610
 radix-2, 610

twiddle factor, 604
feedback coefficients, 452, 485
 FIR filter, 515
feedback systems, 446
feedforward coefficients, 452, 485
filter transformations
 lowpass-to-bandpass, 117, 504
 lowpass-to-bandstop, 118, 506
 lowpass-to-highpass, 116, 501
 lowpass-to-lowpass, 115
filters
 analog, 85–150
 anti-aliasing, 170
 bandpass, *see* bandpass filters
 bandstop, *see* bandstop filters
 causal, 101
 complex, *see* complex filters
 cutoff frequency of, 100
 defined, 100
 digital, 485–553
 discrete Fourier transform and, 590–599
 effects of poles and zeros on, 89, 463–464
 families of analog, 120–149
 finite impulse response, 511–552, *see also* finite impulse response filters
 frequency response of, 85–92, 304, 356, 457–468
 highpass, *see* highpass filters
 ideal, 100, 112, 362–363
 infinite impulse response, 485–507, *see also* infinite impulse response filters
 linear phase, 483, 511–515, 539
 lowpass, *see* lowpass filters
 passband of, 100
 practical, *see* practical filters
 practical specification of, 112–113, 500
 practical specification of FIR, 529
 realizable, 101, 363–364
 realization of, 447–457
 realization of FIR, 455, 515–517
 realization of IIR, 508–510
 stopband of, 100
 time constant of lowpass, 314
 window design of, 106–109
final value theorem, 435
finite-duration signals, 69, 104, 169, *see also* finite impulse response filters
discrete-time Fourier transform of, 342
finite impulse response filters, 288, 485, 511–552
 equiripple, 550
 Fourier series method design of, 521

- frequency-domain methods to design, 537–552
- frequency-weighted least squares, 544–550
- linear phase, 511–515, 539, *see also* linear phase FIR filters
- realization of, 455, 515–517
- window method design of, 521–522, *see also* window method FIR filters
- windows, 517–521
- finite-length sequences, 414
- finite word-length effects, 469–474
- first-order factors
- method of, 422
- first-order hold, 168, 206
- flash converters, 198
- flat-top sampling, 206
- folding frequency, 170
- forced response, 317–321
- from a table, 319
- forward difference systems, 243, 255
- Fourier integral, *see* Fourier transform; discrete-time Fourier transform
- Fourier series, 33–45
- analysis equation, 34, 67
 - convergence at jump discontinuities, 36
 - Dirichlet conditions and convergence of the, 42
 - discrete-time, *see* discrete-time Fourier series
 - finality property, 37
 - frequency spectrum, 34
 - Gibbs phenomenon and truncation of the, 43
 - minimizing mean square error property, 36
 - orthogonality principle to derive the, 547
 - properties of the, 66–68
 - symmetry properties, 66
 - synthesis equation, 34, 67
 - table of properties, 67
 - trigonometric forms of the, 37
- Fourier series method FIR filters, 521
- Fourier transform, 45–68
- analysis equation, 45, 67, 356
 - computed from signal samples, 370
 - direct, 45, 50
 - discrete, *see* discrete Fourier transform
 - discrete-time, *see* discrete-time Fourier transform
 - discrete-time Fourier transform connection to the, 338, 364–370
 - existence of the, 47
 - fast, *see* fast Fourier transform
 - inverse, 45, 50
- Laplace transform connection to the, 70
- Parseval's theorem, 62
- properties of the, 50–66
- symmetry properties, 66
- synthesis equation, 45, 67, 356
- table of pairs, 48
- table of properties, 67, 356
- fractional sampling rate conversion, 258–261, 394
- frequency
- apparent, *see* apparent frequency
 - complex, *see* complex frequency
 - cutoff, 100, 315
 - folding, 170
 - fundamental, 34, 233
 - negative, 36–37, 172
- frequency bins, *see* discrete Fourier transform, bins
- frequency-convolution property, *see* convolution property
- frequency-differentiation property, *see* differentiation property
- frequency-domain analysis, 30, 92, 351
- frequency resolution, 563
- frequency response, 86, 304, 356
- continuous nature of discrete-time system, 461
 - effects of poles and zeros on, 90, 463–464
 - finite word-length effects on, 472
 - from difference equations, 358
 - from pole-zero locations, 462–468
 - LTIC system, 85–92
 - LTID system, 457–468
 - periodic nature of discrete-time system, 461
- frequency-reversal property, *see* reversal property
- frequency sampling filters, 542–544
- frequency sampling method FIR filters, 537
- with windowing, 544
- frequency scaling, 115
- frequency-scaling property, *see* scaling property
- frequency shifting, 57, 346–350
- circular, 581
- frequency-shifting property, *see* shifting property
- frequency transformations, 114
- lowpass-to-bandpass, 117–118
 - lowpass-to-bandstop, 118–120
 - lowpass-to-highpass, 116–117
 - lowpass-to-lowpass, 115–116
- frequency warping, 494, 496
- frequency-weighted least-squares FIR filters, 544–550
- fun, 1–731

- fundamental band, 171, 224, 227, 335
signals outside the, 228
- fundamental frequency, 34, 233
- fundamental period, 21, 233
- gain errors, 194
- gate function, *see* unit gate function
- generalized functions, 10, 335
- generalized linear phase, 97, 360
- Gibbs phenomenon, 43, 520
- Goertzel's algorithm, 600–603
efficient second-order realization of, 602
- Gray code, 189
- group delay, 95, 97–99, 360
- half-wave symmetry, 80
- Hamming window, 110, 520, 530
- Hann window, 110, 520, 530
- Heaviside cover-up method, 421
- hieroglyphics, 187
- highpass filters, 465, 501
ideal, 100, 362
impulse invariance method and, 374, 489
practical specification of, 112
restrictions of linear phase FIR, 514
transformation of lowpass to, *see* filter transformations
- window method and, 533
- Hilbert transformer, 95
- homogeneity property, 26, 250
- ideal delay, *see* delay, ideal
- ideal differentiators, *see* differentiators, ideal
- ideal filters, *see* filters, ideal
- ideal integrators, *see* integrators, ideal
- ideal interpolation, *see* interpolation, ideal
- ideal linear phase systems, 95, 359, 360
- identity systems, 257, 301, 446
- images, 388
- imaginary part, 16
- imaginary signals, 16, 232
- impulse function, *see* unit impulse function
- impulse invariance method, 373, 374
filter design by the, 486–491
impact of sampling interval on the, 487–489
limitations of the, 489
table of pairs for the, 487
- impulse response, 29, 86, 253, 254, 284
closed-form solution, 285–288
of interconnected systems, 300
of nonrecursive systems, 287
system behavior revealed by the, 312
transfer function connection to the, 85, 417
- zero-state response computed using the, 29, 92, 253, 355
- impulse sampling, 155–158
equivalence to point sampling, 366
- infinite impulse response filters, 288, 485–507
bilinear transform design of, 491–507
digital bandpass, 504
digital bandstop, 506
digital highpass, 501
families of analog, 120–149
impulse invariance method design of, 486–491
realization of, 508
second-order section cascade of, 508
- information theory, 187
- initial conditions, 27, 250, 270, 438
internal stability and, 30, 256
unilateral z -transform and, 436
unilateral Laplace transform and, 73
- initial value theorem, 435
- inner product, 545
- inputs, 25, 238
complex, 294
exponential, 30, 85, 303, 321, 357
multiple, 213, 293
sinusoidal, 30, 86, 314, 321, 458
- instability, *see* unstable systems
- instantaneous systems, 256
- integration property
Fourier transform, 356
Fourier transform time, 59, 67
Laplace transform, 72
- integrators
backward difference transform and digital, 492
digital, 243
ideal, 31, 89
trapezoidal approximation of, 244
- interconnected systems, 300, 445
- internal conditions
system response to, *see* zero-input response
- internal stability, 30, 256
poles to determine, 32, 306, 444
relation to bounded-input bounded-output stability, 307–309
- interpolation, 164, 219, 387–391
bandlimited, *see* bandlimited interpolation
discrete-time Fourier transform obtained by discrete Fourier transform, 567
- first-order hold, 168
- ideal, 164, 213, 380, 390
- spectral, 183–184

- zero-order hold, 167
- interpolation filters, 219, 257, *see also* reconstruction filters
 - decimation filter combined with, 261
 - frequency response of ideal, 164, 389
 - impulse response of ideal, 164, 391
- interpolation formula, 165, 213, 371
 - spectral, 184
- interpolation function, *see* sinc function
- interpolators, 257
 - block representation of, 258, 379
 - decimators in cascade with, 394–395
 - output of ideal, 391
- intuitive insights into system behavior, 311–317
 - frequency response provides, 86–87, 93
 - poles and zeros provide, 32
- inverse Chebyshev filters, 139–144
 - determination of order, 140
 - pole locations, 140
 - ripple parameter, 140
 - transfer function, 140
 - transformation to digital, 485–507
 - zero locations, 140
- inverse discrete Fourier transform, *see* discrete Fourier transform, inverse
- inverse discrete-time Fourier transform, *see* discrete-time Fourier transform, inverse
- inverse Fourier transform, *see* Fourier transform, inverse
- inverse Laplace transform, *see* Laplace transform, inverse
- inverse systems, 301, 446
- inverse z -transform, *see* z -transform, inverse
- invertibility, 257
- iterative solutions
 - to difference equations, 270
 - to impulse responses, 284
- Kaiser window, 110–112, 520, 530
 - filter design using, 532
- Kronecker delta function, 220
 - connection to the unit step function, 221
 - expansion represented using the, 222
 - properties of the, 220
 - sampling property, 220
- L -ary digital signals, *see* digital signals, L -ary
- Laplace transform, 31, 68–73
 - analysis equation, 69, 72
 - bilateral, 69
 - Fourier transform connection to the, 70
 - inverse, 69
- properties of the, 72–73
- region of convergence, 69
- synthesis equation, 69, 72
- table of pairs, 71
- table of properties, 72
- unilateral, 69–70, 72–73
- z -transform connection to the, 410, 474–476
- leakage, *see* spectral leakage
- left half-plane, 12, 224
 - system stability and the, 32
- left shift, 4, 214, 431
- left-sided signals, 16, 69, 231, 414
- Leibnitz, Gottfried Wilhelm, 188
- linear convolution, *see* convolution
- linear interpolation, 168, 206, 392
- linear phase, 54, 95, 359
 - distortionless transmission and, 94, 359
 - generalized, *see* generalized linear phase
 - physical explanation of, 57, 345
- linear phase FIR filters, 483, 511–515, 539, *see also* finite impulse response filters
 - realization of, 515
 - table of amplitude and phase responses, 513
 - table of restrictions, 514
 - table of types, 512
- linear phase systems
 - ideal, 95, 359, 360
- linear quantization, 193, *see also* quantization
- linear systems, 26, 250
- linear time-invariant systems, 25, 248, 290, *see also* continuous-time systems; discrete-time systems
- linear time-variant systems, 28, 252, 290
- linear vector spaces, 545
- linearity, 26, 250
 - visualizing, 26, 251
- linearity property
 - discrete Fourier transform, 579, 584
 - discrete-time Fourier transform, 343, 356, 584
 - Fourier series, 67
 - Fourier transform, 52, 67, 356
 - Laplace transform, 72
 - z -transform, 427, 434
- lowpass filters, 465
 - ideal, 100, 362
 - impulse invariance method and, 489
 - practical specification of, 112
 - practical specification of FIR, 529
 - restrictions of linear phase FIR, 514
- lowpass-to-bandpass filter transformations, 117
- lowpass-to-bandstop filter transformations, 118

- lowpass-to-highpass filter transformations, 116
lowpass-to-lowpass filter transformations, 115
- Maclaurin series, 426
magnitude response, 86, 94, 357, 458
 conditions to be realizable, 101
 effects of poles and zeros on, 90, 463–464
 expressed in decibel scale, 86
- magnitude spectrum
 discrete Fourier transform, 562
 discrete-time Fourier transform, 334
 Fourier series, 34
 Fourier transform, 46
- main lobe, 104–112, 517–521, 525
 widths of common window functions, 520
- mapping, 224
- marginal stability, 30, 256
 poles to determine, 32, 307, 444, 445
- MATLAB drill exercises
 aliasing, 726
 Bessel-Thomson filters, 663
 bilinear transform, 708–710
 block convolution, 727
 Butterworth filters, 660, 661
 Chebyshev filters, 661, 662, 708, 710
 circular convolution, 725, 726
 convolution, 686, 727
 discrete Fourier transform, 719, 720, 725–727, 730
 discrete-time Fourier series, 730
 downsampling, 678
 fractional sampling rate, 678
 frequency response, 655, 656, 691, 693, 706, 707
 frequency sampling method, 717
 frequency-weighted least squares, 718
 impulse response, 681, 691, 696
 interpolation, 665, 671, 693, 694
 inverse Chebyshev filters, 709
 iterative solution of difference equations, 679
 partial fraction expansions, 695–697, 700–704
 plotting signals, 670, 674
 pole/zero plots, 656, 661
 stability, 704
 upsampling, 671, 672, 678
 window functions, 659
 window method, 691, 712–715
 zero-input response, 679, 700–702
 zero-state response, 700–703
- MATLAB examples
 aliasing, 217, 224
- apparent frequency, 229
bilinear transform, 494, 497, 502, 504, 506
block convolution, 595, 598
Butterworth filters, 123, 125, 126, 489, 494, 497, 504
Chebyshev filters, 133, 135, 137, 502, 506
circular convolution, 587
circular shifting, 581
convolution, 298, 299, 587, 591, 592
decimation, 387
discrete Fourier transform, 566, 581, 591, 592, 595, 598, 610
discrete-time Fourier transform, 342
downsampling, 217
elliptic filters, 145
energy, 22
equiripple filters, 551
essential bandwidth, 64
even/odd decompositions, 19
Fourier series, 35, 38, 44
Fourier transform, 46, 54, 55, 610
frequency response, 90, 117, 119, 359, 364, 459, 466, 467, 516
frequency sampling method, 538, 539, 542
frequency-weighted least squares, 549
impulse invariance, 489
impulse response, 284, 285, 287, 364
interpolation, 166, 392
inverse Chebyshev filters, 141, 143
iterative solution of difference equations, 240, 270, 272, 284
partial fraction expansions, 357, 420, 423, 436, 441, 442, 454
plotting signals, 14, 234, 305
pole/zero plots, 90
quantization, 192
unit step function, 14
window functions, 111
window method, 364, 523, 526, 530, 532, 534
zero-input response, 278–280, 436
zero-state response, 253, 357, 436, 441, 442
- matrix
 discrete Fourier transform, 565
 permutation, 623
 Toeplitz, 549
- maximally flat responses, 121, 139, 147
maximum passband attenuation, 112
maximum stopband gain, 112
memory, 245, 256, 448
memoryless systems, 256
method of residues, 420
minimum passband gain, 112

- minimum phase systems, 32
- minimum stopband attenuation, 112
- mirror image polynomials, 514
- missing codes, 195
- mixed-radix fast Fourier transform, 610
- modified partial fractions, 420
- modulation
 - amplitude, 58
 - delta, 380
 - pulse-code, 186, 380
- modulation property
 - discrete-time Fourier transform, 347–350
 - Fourier transform, 58
- modulo- N operation, 560, 572–573
- modulo- N reflection, 574
- modulo- N shift, 572, 574–578
- modulo operation, 174
- monotonic exponentials, 12, 222
- moving-average systems, 269, 325
- μ -law compression, 193
- multiple-input, multiple-output systems, 213
- multiple-input, single-output systems, 213
- multiple inputs, 293
- multiplication
 - discrete-time convolution and polynomial, 299
 - of a function by an impulse, 10, 220
 - scalar, 238, 447, 545
- multiplication property, *see* convolution property
- multirate systems, 216, 257
- natural binary code, 186
- natural modes, *see* characteristic modes
- natural response, 317–321
- natural sampling, 206
- negative frequency, 36–37, 172
- neper frequency, 12
- noise, 25, 88, 192
 - aliasing and, 170, 172
 - quantization, *see* quantization error
- non-anticipative systems, *see* causal systems
- non-bandlimited signals, 105, 169, 171, 486
- noncausal signals, 15, 231
- noncausal systems, 29, 254
- non-integer shift, 368–370
- nonlinear quantization, 193
- nonlinear systems, 27, 193, 250, 251
- nonlinearity errors, 195
- nonrecursive form, 239, 255, 273, 300
- nonrecursive systems
 - impulse response of, 287
 - zero-input response of, 282
- nonuniqueness, 226, 570, 571
- norm, 545
- notch filters, *see* bandstop filters
- Nyquist interval, 156
- Nyquist rate, 155–161, 168, 370
 - downsampling and the, 216
 - reasons to exceed, 201
- Nyquist samples, 156, 171, 397
- objective functions, 64
- observability, 32, 309
- odd symmetry, 18, 54, 232, 344
 - in Fourier analysis, 66
 - linear phase FIR filters with, 512
- offset binary, 189
- offset errors, 194
- operator notation, 275–276
 - danger of treating algebraically, 276
- order, 245
 - of difference equations, 245
- orthogonality, 545
- orthogonality principle, 545–547
 - Fourier series derived using the, 547
- orthonormality, 547
- outputs, 25, 238
 - complex, 294
 - exponential, 30, 85, 303, 321, 357
 - multiple, 213
 - sinusoidal, 30, 86, 314, 321, 458
- overlap-and-add method, *see* block convolution
- overlap-and-save method, *see* block convolution
- Paley-Wiener criterion, 101, 112, 168
- parallel converters, *see* flash converters
- parallel realizations, 453
 - of frequency sampling filters, 542
 - performance of, 457
- parallel systems, 300, 445
- Parks-McClellan algorithm, 551
- Parseval's theorem
 - discrete Fourier transform, 583, 584
 - discrete-time Fourier transform, 355, 356, 584
 - Fourier series, 67
 - Fourier transform, 62, 67, 356
- partial fraction expansions, *see* MATLAB examples, partial fraction expansions
- partial fractions
 - modified, 420
- passbands, 100, 101, 112, 362
 - specification of, 112, 529
- peak-to-peak ripple, 130

- periodic convolution, *see* circular convolution
periodic replication, 21, 156, 161
 circular representation and, 574
 discrete Fourier transform and, 570
 discrete-time Fourier transform from Fourier transform, 366
 discrete-time Fourier transform fundamental band and, 340, 350
 spectral sampling produces time-domain, 181
 time-domain sampling results in frequency-domain, 374
periodic signals, 21, 233
 continuous-time systems and, 32–33
 discrete Fourier transform and, 570, 578
 discrete-time Fourier series and, 612
 Fourier series and, 33
 Fourier transform and, 49
permutation matrix, 623
phase delay, 361
phase response, 86, 94, 357, 458
 effects of poles and zeros on, 90, 463–464
 principal value of the, 362
phase spectrum
 discrete Fourier transform, 562
 discrete-time Fourier transform, 334
 Fourier series, 34
 Fourier transform, 46
 principal value of the, 55
physical systems, *see* causal systems
pick-off nodes, 238, 452
picket fence effect, 563
Pingala, 188
point sampling, 155
 equivalence to impulse sampling, 366
point-wise convergence, 41
Poisson sum formula, 210
pole-zero plots, 89–92, 462–468, 508–510
poles, 32, 448
 complex, 280, 508
 controlling gain by, 32, 149, 463–464
 effect on frequency response, 90, 463–464
 finite word-length effects on, 469
 realization of repeated, 454
 repeated, 279, 421, 487
 system stability and, 32, 307, 444
 system stability and repeated, 32, 307, 444
polynomial multiplication by convolution, 299
polynomials
 anti-mirror image, 514
 mirror image, 514
polyphase decomposition, 604
power
 computed as an inner product, 547
 frequency-domain representation of, 65
 of continuous-time signals, 23
 of discrete-time signals, 236
power series expansion, 425
power signals
 continuous-time, 24
 discrete-time, 236
power spectral density, 65
practical filters, 100–113, 362–364
 analog, 120, 149
 digital, 485–552
predictive first-order hold, 206
prewarping, 497–501
principal values
 phase using, 55, 362
prototype filters, 114, 501
pulse-code modulation, 186, 380
pulse dispersion, 315
pulse sampling, 161–164
quadratic factors
 method of, 422
quadratic formula, 127, 136
quadrature systems, 404
quantization, 185
 asymmetric, 189, 190
 linear, 193
 nonlinear, 193
 rounding, 189, 190
 symmetric, 189, 190
 truncating, 189, 190
quantization errors, 189, 201–202
quantization levels, 185
quantization noise, *see* quantization error
radix-2 fast Fourier transform, 610
ramp invariance method, 554
random signals, 25
rational functions, 420
real part, 16
real signals, 16, 232
real systems, 85
 complex roots of, 280
realizable filters, *see* practical filters
reconstruction
 practical difficulties in, 168–176
reconstruction filters, 164–176, 199–201
rectangle function, *see* unit gate function
rectangular window, 104–110, 517–521, 530
 optimality of the, 522

- recursion, 270
 recursive form, 239, 273
 reflection, *see* reversal property; time reversal
 region of convergence, 69, 411–415
 for finite-length signals, 414
 poles never found within the, 412
 unilateral transforms and, 70, 416
 region of existence, *see* region of convergence
 Remez exchange algorithm, 551
 repeated poles, *see* poles, repeated
 repeated roots, 279, *see also* poles, repeated
 resampling
 analog, 393
 digital, 257, 379–395
 resonance, 32, 283, 311, 315–317
 reversal, *see* reversal property; time reversal
 reversal property
 discrete Fourier transform, 580, 584
 discrete-time Fourier transform, 344, 356,
 584
 Fourier series, 67
 Fourier transform, 53, 67, 356
 Laplace transform, 72
 z -transform, 428, 434
 reverse Bessel polynomials, 148
 table of, 148
 right half-plane, 12, 224
 right shift, 4, 214, 430
 right-sided signals, 16, 69, 231, 414
 ripple parameters, 112, 529, 536
 rise time, 314
 rolloff, 107, 519
 roots
 characteristic, *see* characteristic roots
 complex, 280
 repeated, *see* repeated roots
 transfer function, 31, 89, 462
 rounding quantization, *see* quantization, rounding
 sample-and-hold, 196
 sampled continuous-time sinusoids, 458–461
 sampling, 155–164
 first-order, 178
 flat-top, 206
 impulse, 10, 155–158, 366
 natural, 206
 nonuniform, 177
 Nyquist rate, *see* Nyquist rate
 of bandpass signals, 176–181
 point, 155, 366
 practical, 161–164
 practical difficulties in, 168–176
 pulse, 161–164
 second-order, 177
 signal reconstruction and, 164–176
 spectral, 181–183, 537, 544
 uniform, 155
 sampling frequency, *see* sampling rate
 sampling interval, 155
 sampling property
 continuous-time, 10
 discrete-time, 220, 289
 sampling rate, 155, *see also* Nyquist rate
 conversion of, 216, 257, 394
 permissible for bandpass signals, 176
 spectral, 183
 sampling rates
 permissible for bandpass signals, 181
 sampling theorem, 155–161, 235
 spectral, 181–183
 saturation errors, 189
 savings account example, 240
 scalar multiplication, 238, 447, 545
 scaling
 continuous-time signals, 5
 discrete-time signals, 216
 shifting combined with, 6
 scaling property, 26, 250, *see also* time scaling
 discrete-time Fourier transform lack of the,
 344
 Fourier series lack of the, 68
 Fourier transform, 53, 67, 381
 Laplace transform, 72
 z -domain, 432, 434
 z -transform lack of time-domain, 428
 second-order sections, 447–457
 cascade of, 508
 sequences, *see* discrete-time signals
 series expansion, *see* power series expansion
 shifting
 continuous-time signals, 4
 discrete-time signals, 214
 reversal combined with, 215
 scaling combined with, 6
 shifting property, *see also* time shifting
 convolution, 292
 discrete Fourier transform circular, 580, 584
 discrete-time Fourier transform, 345–350,
 356, 584
 Fourier series, 67
 Fourier transform, 54, 57, 67, 356
 Laplace transform, 72
 z -transform time-domain, 428, 434

- side lobes, 104–112, 517–521, 525
sifting property, *see also* sampling property
 continuous-time, 10, 11
 discrete-time, 220, 289
signal distortion, 96
signal energy
 computed as an inner product, 545
 continuous-time, 21
 discrete-time, 236
 essential bandwidth and, 63
 frequency-domain representation of, 61, 354,
 582
signal power
 computed as an inner product, 547
 continuous-time, 23
 discrete-time, 236
 frequency-domain representation of, 65
signal reconstruction, 164–176, *see also* interpolation
signal to noise power ratio, 25
signal transmission, 92–99, 359–362
signals, 2–25, 212–238
 analog, *see* analog signals
 anti-causal, *see* anti-causal signals
 aperiodic, *see* aperiodic signals
 approximating with basis functions, 545–550
 audio, *see* audio signals
 bandlimited, *see* bandlimited signals
 categorization of, 3–4
 causal, *see* causal signals
 classifications of, 15–25, 231–238
 conjugate-antisymmetric, *see* conjugate antisymmetry
 conjugate-symmetric, *see* conjugate symmetry
 continuous-time, *see* continuous-time signals
 defined, 2
 deterministic, *see* deterministic signals
 digital, *see* digital signals
 discrete-time, *see* discrete-time signals
 energy, *see* energy signals
 even, *see* even symmetry
 everlasting, *see* everlasting signals; two-sided signals
 finite-duration, *see* finite-duration signals
 imaginary, *see* imaginary signals
 left-sided, *see* left-sided signals
 models of useful, 7–15, 219–230
 modulation, 58
 non-bandlimited, *see* non-bandlimited signals
 noncausal, *see* noncausal signals
odd, *see* odd symmetry
orthogonal, *see* orthogonality
orthonormal, *see* orthonormality
periodic, *see* periodic signals
power, *see* power signals
probabilistic, *see* random signals
random, *see* random signals
real, *see* real signals
size of, *see* energy; power
useful operations, 214–219
useful operations on the independent variable of, 4–7
video, *see* video signals
signals and systems
 elementary structures, 1
sinc function, 13–15
 interpolation using the, 165
 spectral interpolation using the, 184
single-input, multiple-output systems, 213
single-input, single-output systems, 212
sinusoidal inputs and outputs, 86, 314, 458
sinusoidal steady-state response, 85, 458
sinusoids, 12, 222, 225–226, *see also* exponential functions
 aliasing in, 173–176, 226–230
 apparent frequency of, 174
 frequency response and, 85–86
 periodicity of discrete-time, 233–236
 relationship between exponential functions and, 12
sampled continuous-time, 226, 458–461
sliding-tape method, 296–298
spectra, 331
 discrete Fourier transform, 560
 discrete-time Fourier transform, 334
 Fourier series, 34
 Fourier transform, 46
 width of periodic, 354
spectral density
 energy, 62, 355
 power, 65
spectral folding, 170, *see also* aliasing
spectral images, 388
spectral interpolation, 183–184
spectral leakage, 104–112, 517–521, 525, 539
 remedy for, 109
spectral resolution, 109
spectral sampling, 181–183, 537, 544
spectral sampling method FIR filters, 537
 with windowing, 544
spectral sampling rate, 183
spectral sampling theorem, 181–183

- spectral smearing, *see* spectral spreading
 spectral spreading, 104–112, 517–521, 525
 remedy for, 109
 spectral resolution and, 109
 stability, 29, 256, 305–311, 444–445
 bounded-input bounded-output, *see*
 bounded-input bounded-output stability
 external, *see* bounded-input bounded-output stability
 internal, *see* internal stability
 marginal, *see* marginal stability
 poles to determine, 32, 307, 444
 static errors, 194–196
 static systems, 256
 steady-state response, 85, 458
 step function, *see* unit step function
 step invariance method, 554
 stopband rolloff, 107
 stopbands, 100, 101, 112, 362
 specification of, 112, 529
 sub-Nyquist sampling, 171, 378
 bandpass signals and, 176–181
 successive approximation converters, 197
 sums
 table of, 290
 superposition property, 26, 250, 293
 zero-state response and the, 28, 252
 symbols, 185
 M-ary, 188
 symmetric quantization, *see* quantization, symmetric
 symmetry
 conjugate, *see* conjugate symmetry
 conjugate anti-, *see* conjugate antisymmetry
 relations of Fourier analysis, 66
 synthesis equation
 discrete Fourier transform, 561, 584, 600
 discrete-time Fourier series, 612
 discrete-time Fourier transform, 332, 338, 356, 584
 Fourier series, 34, 67
 Fourier transform, 45, 67, 356
 Laplace transform, 69, 72
 z -transform, 397, 410, 434
 system poles, *see* poles
 system realizations, 238, 445–457
 cascade, *see* cascade realizations
 direct form, *see* direct form realizations
 finite impulse response, 455
 of complex-conjugate poles, 454
 of repeated poles, 454
 parallel, *see* parallel realizations
 performance differences in, 457
 system zeros, *see* zeros
 systems, 25, 238–248
 accumulator, *see* accumulator
 backward difference, *see* backward difference systems
 cascade, *see* cascade; cascade realizations
 causal, *see* causal systems
 continuous-time, *see* continuous-time systems
 discrete-time, *see* discrete-time systems
 dynamic, *see* dynamic systems
 feedback, *see* feedback systems
 forward difference, *see* forward difference systems
 frequency response of, *see* frequency response
 identity, *see* identity systems
 instantaneous, *see* instantaneous systems
 interconnected, *see* interconnected systems
 inverse, *see* inverse systems
 linear, *see* linear systems
 magnitude response of, *see* magnitude response
 minimum phase, *see* minimum phase systems
 multiple-input multiple-output, *see*
 multiple-input multiple-output systems
 multiple-input single-output, *see* multiple-input single-output systems
 noncausal, *see* noncausal systems
 nonlinear, *see* nonlinear systems
 parallel, *see* parallel systems; parallel realizations
 phase response of, *see* phase response
 properties of, 25–30, 248–257
 single-input multiple-output, *see* single-input multiple-output systems
 single-input single-output, *see* single-input single-output systems
 stable, *see* stability
 time-invariant, *see* time-invariant systems
 time-variant, *see* time-variant systems
 tables
 collection of useful, 640–645
 Tacoma Narrows Bridge failure, 317
 tapered windows, 109–110, 517–521
 tapped delay lines, 455, 515
 Taylor series, 426

- time constant, 312–315
relationship between bandwidth and, 315
- time convolution, *see* convolution; convolution property; convolution sum
- time delay, *see* time shifting
variation with frequency, 95
- time differentiation, *see* differentiation property
- time-domain analysis
of continuous-time systems, 25–30
of discrete-time systems, 270–323
of interpolation, 164–168
- time-frequency duality, *see* duality
- time-integration, *see* integration property
- time invariance, 27, 248
visualizing, 28, 249
- time-invariant systems, 27, 248
linear, *see* linear time-invariant systems
- time reversal
continuous-time, 5
discrete-time, 215
shifting combined with, 215
- time-reversal property, *see* reversal property
- time scaling, *see also* scaling property
continuous-time, 5, 53
discrete-time, 216–219
- time shifting, *see also* shifting property
continuous-time, 4
discrete-time, 214
- time-variant systems, 28, 248–250, 252, 290
- timelimited signals, *see* finite-duration signals
- timing jitter, 195
- Toeplitz matrix, 549
- total response
of continuous-time systems, 27
of discrete-time systems, 250, 304, 317
- transfer functions
block diagram realizations of, 445–457
discrete-time system realization of
continuous-time system, 486
of continuous-time systems, 31
of discrete-time systems, 303, 439–444
roots of, 31, 89, 462
- transformations of filters, *see* filter transformations
- transition bands, 107, 112, 530
multiple, 112
- transposed realizations, 451–453
canonical direct form, 451
direct form I, 453
direct form II, 451
- transversal filters, 455, 515
- trapezoidal rule
- bilinear transform and the, 492
- triangle function, *see* unit triangle function
- triangular window, 104–110, 520
- trigonometric Fourier series, 37, *see also* Fourier series
- truncating quantization, *see* quantization, truncating
- Tukey, J. W., 603
- twiddle factor, 604
- two's complement, 189
- two-sided signals, 16, 231, 414
- uncontrollable systems, 309
- undersampling, *see* sub-Nyquist sampling
- undetermined coefficients
method of, 318
- uniform sampling, 155
- uniformly convergent series, 41
- unilateral Laplace transform, *see* Laplace transform, unilateral
- unilateral z -transform, *see* z -transform, unilateral
- uniqueness
discrete Fourier transform and, 569–572
- unit circle, 223, 463
- unit delay, 275
transfer function of the, 448
- unit gate function, 8
- unit impulse function
continuous-time, *see* Dirac delta function
discrete-time, *see* Kronecker delta function
- unit impulse response, *see* impulse response
- unit impulse train, 40, 157
- unit step function
connection to the Dirac delta function, 10
connection to the Kronecker delta function, 221
continuous-time, 7
difference between continuous-time and discrete-time, 231
discontinuity of the continuous-time, 7
discrete-time, 219
- unit triangle function, 8
- unobservable systems, 309
- unstable systems, 30, 256, 306, 307, *see also* stability
pole locations and, 32, 307, 445
- upsampling, 218–219, 344, 380
block representation of, 257, 379
fractional sampling rate conversion and, 394
frequency-domain perspective of, 379, 387–391

- time-domain perspective of, 391
- vector spaces, 545
- vectors
 - basis, *see* basis vectors
 - error, 546
 - orthogonality principle demonstrated using, 547
- video signals, 96
- warping, *see also* prewarping
 - frequency, 494, 496
- weighting function
 - frequency, 548
- width property
 - convolution integral, 104
 - convolution sum, 292
- window functions, 104–112
 - adjustable, 521
 - Blackman, *see* Blackman window
 - choosing, 110
 - Hamming, *see* Hamming window
 - Hann, *see* Hann window
 - impairments caused by, 104–105
 - Kaiser, *see* Kaiser window
 - rectangular, *see* rectangular window
 - table of, 110, 520
 - table of filtering characteristics, 530
 - triangular, *see* triangular window
- window method FIR filters, 521–522, 533–536
 - achieving given specifications with, 529–530
 - examples of, 523–529
 - Kaiser window and, 532
 - suboptimality of, 536
- window operation, 104
- z*-domain scaling property, 432
- z*-transform, 395–397, 410–477
 - analysis equation, 397, 410, 434
 - bilateral, 410
 - difference equations solved using the, 436–445
 - direct, 411
 - discrete-time Fourier transform connection to the, 418
 - existence of the, 413
 - final value theorem, 435
 - initial value theorem, 435
 - inverse, 411, 419–427
 - inverse by power series expansion, 425
 - Laplace transform connection to the, 474–476
 - linearity of the, 414
 - properties of the, 427–436
 - region of convergence, 411–415
 - synthesis equation, 397, 410, 434
 - table of pairs, 418
 - table of properties, 434
 - unilateral, 416–417, 420–423, 434, 436
 - zero-input response by the, 438
 - zero-state response by, 439–444
 - zero-state response by the, 438
- zero-input response
 - insights into the, 282–283
 - iterative solution, 271
 - of continuous-time systems, 27
 - of discrete-time systems, 250, 277
 - of nonrecursive systems, 282
 - z*-transform and the, 438
- zero-order hold, 167
 - distortion from, 200
- zero padding, 563–565
 - circular convolution and, 586
- zero state, 27, 250
- zero-state response
 - convolution and the, 288
 - iterative solution, 272
 - of continuous-time systems, 27, 28
 - of discrete-time systems, 250, 252
 - transfer function and the, 439–444
 - z*-transform and the, 438
- zeros, 32, 448
 - complex, 508
 - controlling gain by, 32, 149, 463–464
 - effect on frequency response, 90, 463–464
 - finite word-length effects on, 469