Discrete Time Signal Processing

Class Notes for the Course ECSE-412

Benoît Champagne and Fabrice Labeau
Department of Electrical & Computer Engineering
McGill University

with updates by Peter Kabal

Winter 2004

Contents

Chapter 1

Introduction

This Chapter begins with a high-level definition of the concepts of signals and systems. This is followed by a general introduction to digital signal processing (DSP), including an overview of the basic components of a DSP system and their functionality. Some practical examples of real-life DSP applications are then discussed briefly. The Chapter ends with a presentation of the course outline.

1.1 The concepts of signal and system

Signal:

- A signal can be broadly defined as any quantity that varies as a function of time and/or space and has the ability to convey information.
- Signals are ubiquitous in science and engineering. Examples include:
  - Electrical signals: currents and voltages in AC circuits, radio communications signals, audio and video signals.
  - Mechanical signals: sound or pressure waves, vibrations in a structure, earthquakes.
  - Biomedical signals: electro-encephalogram, lung and heart monitoring, X-ray and other types of images.
  - Finance: time variations of a stock value or a market index.
- By extension, any series of measurements of a physical quantity can be considered a signal (temperature measurements for instance).

Signal characterization:

- The most convenient mathematical representation of a signal is via the concept of a function, say $x(t)$. In this notation:
  - x represents the dependent variable (e.g., voltage, pressure, etc.)
  - t the represents the independent variable (e.g., time, space, etc.).

- Depending on the nature of the independent and dependent variables, different types of signals can be identified:
  - Analog signal: $t \in \mathbb{R} \rightarrow x_a(t) \in \mathbb{R}$ or $\mathbb{C}$
    When t denotes the time, we also refer to such a signal as a continuous-time signal.
  - Discrete signal: $n \in \mathbb{Z} \rightarrow x[n] \in \mathbb{R}$ or $\mathbb{C}$
    When index n represents sequential values of time, we refer to such a signal as discrete-time.
  - Digital signal: $n \in \mathbb{Z} \rightarrow x[n] \in \mathbb{A}$
    where $\mathbb{A} = \{a_1, \ldots, a_L\}$ represents a finite set of L signal levels.
  - Multi-channel signal: $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t))$
  - Multi-dimensional signal: $x(t_1, \ldots, t_N)$;
- Distinctions can also be made at the model level, for example: whether $x[n]$ is considered to be deterministic or random in nature.

Example 1.1: Speech signal

► A speech signal consists of variations in air pressure as a function of time, so that it basically represents a continuous-time signal $x(t)$. It can be recorded via a microphone that translates the local pressure variations into a voltage signal. An example of such a signal is given in Figure 1.1(a), which represent the utterance of the vowel "a". If one wants to process this signal with a computer, it needs to be discretized in time in order to accommodate the discrete-time processing capabilities of the computer (Figure 1.1(b)), and also quantized, in order to accommodate the finite-precision representation in a computer (Figure 1.1(b)). These represent a continuous-time, discrete-time and digital signal respectively.

As we know from the sampling theorem, the continuous-time signal can be reconstructed from its samples taken with a sampling rate at least twice the highest frequency component in the signal. Speech signals exhibit energy up to say, 10 kHz. However, most of the intelligibility is conveyed in a bandwidth less than 4 kHz. In digital telephony, speech signals are filtered (with an anti-aliasing filter which removes energy above 4 kHz), sampled at 8 kHz and represented with 256 discrete (non-uniformly-spaced) levels. Wideband speech (often termed commentary quality) would entail sampling at a higher rate, often 16 kHz. ◄

Example 1.2: Digital Image

► An example of two-dimensional signal is a grayscale image, where t_1 and t_2 represent the horizontal and vertical coordinates, and $x(t_1, t_2)$ represents some measure of the intensity of the image at location (t_1, t_2). This example can also be considered in discrete-time (or rather in discrete space in this case): digital images are made up of a discrete number of points (or *pixels*), and the intensity of a pixel can be denoted by $x[n_1, n_2]$. Figure 1.2 shows an example of digital image. The rightmost part of the Figure shows a zoom on this image that clearly shows the pixels. This is an 8-bit grayscale image, i.e. each pixel (each signal sample) is represented by an 8-bit number ranging from 0 (black) to 255 (white). ◄

System:

- A physical entity that operates on a set of primary signals (the inputs) to produce a corresponding set of resultant signals (the outputs).

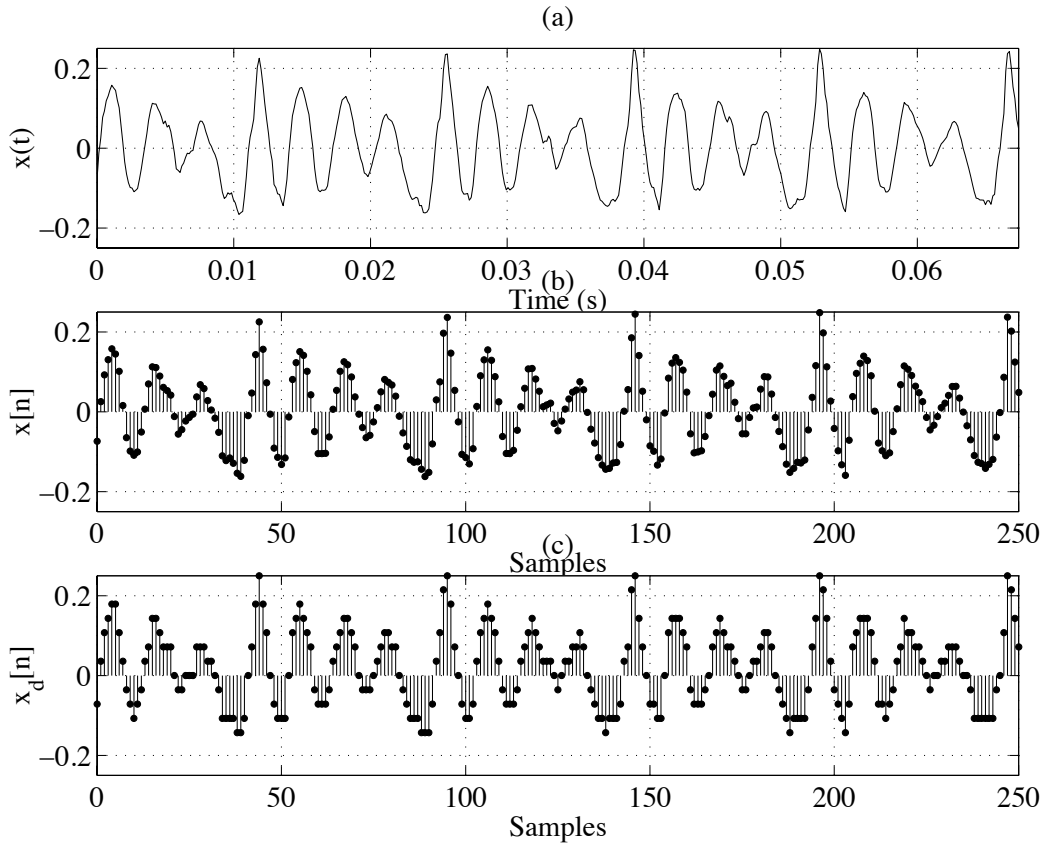


Fig. 1.1 An utterance of the vowel "a" in analog, discrete-time and digital format. Sampling at 4 kHz and quantization on 4 bits (16 levels).

Fig. 1.2 Digital image, and zoom on a region of the image to show the pixels

- The operations, or processing, may take several forms: modification, combination, decomposition, filtering, extraction of parameters, etc.

System characterization:

- A system can be represented mathematically as a transformation between two signal sets, as in $x[n] \in \mathbb{S}_1 \rightarrow y[n] = T\{x[n]\} \in \mathbb{S}_2$. This is illustrated in Figure 1.3

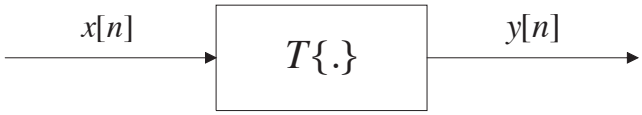


Fig. 1.3 A generic system

- Depending on the nature of the signals on which the system operates, different basic types of systems may be identified:
  - Analog or continuous-time system: the input and output signals are analog in nature.
  - Discrete-time system: the input and output signals are discrete.
  - Digital system: the input and outputs are digital.
  - Mixed system: a system in which different types of signals (i.e. analog, discrete and/or digital) coexist.

1.2 Digital signal processing

1.2.1 Digital processing of analog signals

Discussion:

- Early education in engineering focuses on the use of calculus to analyze various systems and processes at the analog level:
  - motivated by the prevalence of the analog signal model
  - e.g.: circuit analysis using differential equations
- Yet, due to extraordinary advances made in micro-electronics, the most common/powerful processing devices today are digital in nature.
- Thus, there is a strong, practical motivation to carry out the processing of analog real-world signals using such digital devices.
- This has lead to the development of an engineering discipline know as digital signal processing (DSP).

Digital signal processing (DSP):

- In its most general form, DSP refers to the processing of analog signals by means of discrete-time operations implemented on digital hardware.
- From a system viewpoint, DSP is concerned with mixed systems:
  - the input and output signals are analog
  - the processing is done on the equivalent digital signals.

1.2.2 Basic components of a DSP system

Generic structure:

- In its most general form, a DSP system will consist of three main components, as illustrated in Figure 1.4.
- The analog-to-digital (A/D) converter transforms the analog signal $x_a(t)$ at the system input into a digital signal $x_d[n]$. An A/D converter can be thought of as consisting of a sampler (creating a discrete-time signal), followed by a quantizer (creating discrete levels).
- The digital system performs the desired operations on the digital signal $x_d[n]$ and produces a corresponding output $y_d[n]$ also in digital form.
- The digital-to-analog (D/A) converter transforms the digital output $y_d[n]$ into an analog signal $y_a(t)$ suitable for interfacing with the outside world.
- In some applications, the A/D or D/A converters may not be required; we extend the meaning of DSP systems to include such cases.

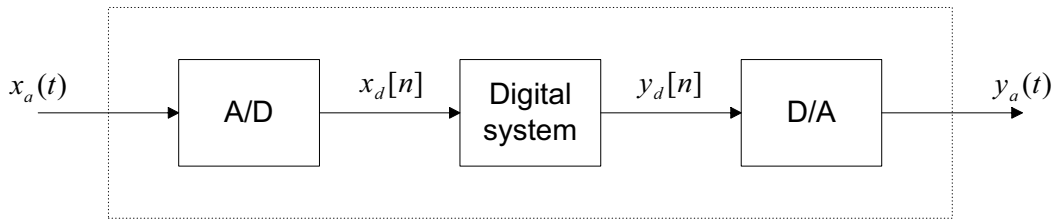


Fig. 1.4 A digital signal processing system

A/D converter:

- A/D conversion can be viewed as a two-step process:

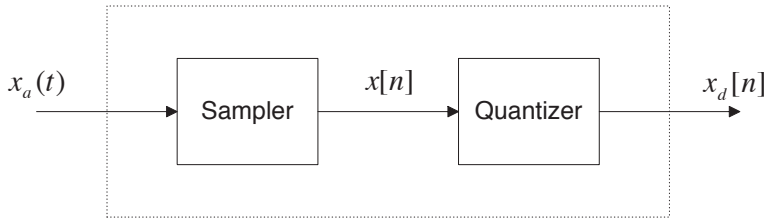


Fig. 1.5 A/D conversion

- Sampler: in which the analog input is transformed into a discrete-time signal, as in $x_a(t) \rightarrow x[n] = x_a(nT_s)$, where T_s is the sampling period.

- Quantizer: in which the discrete-time signal $x[n] \in \mathbb{R}$ is approximated by a digital signal $x_d[n] \in \mathbb{A}$, with only a finite set $\mathbb{A}$ of possible levels.

- The number of representation levels in the set $\mathbb{A}$ is hardware defined, typically 2^b where b is the number of bits in a word.

- In many systems, the set of discrete levels is uniformly spaced. This is the case for instance for WAVE files used to store audio signals. For WAVE files, the sampling rate is often 44.1 kHz (the sampling rate used for audio CD's) or 48 kHz (used in studio recording). The level resolution for WAVE files is most often 16 bits per sample, but some systems use up to 24 bits per samples.

- In some systems, the set of discrete levels is non-uniformly spaced. This is the case for digital telephony. Nearly all telephone calls are digitized to 8 bit resolution. However, the levels are not equally spaced — the spacing between levels increases with increasing amplitude.

Digital system:

- The digital system is functionally similar to a microprocessor: it has the ability to perform mathematical operations on a discrete-time basis and can store intermediate results of computation in internal memory.
- The operations performed by the digital system can usually be described by means of an algorithm, on which its implementation is based.
- The implementation of the digital system can take different forms:
  - Hardwired: in which dedicated digital hardware components are specially configured to accomplish the desired processing task.
  - Softwired: in which the desired operations are executed via a programmable digital signal processor (PDSP) or a general computer programmed to this end.
- The following distinctions are also important:
  - Real-time system: the computing associated to each sampling interval can be accomplished in a time $\leq$ the sampling interval.
  - Off-line system: A non real-time system which operates on stored digital signals. This requires the use of external data storage units.

D/A converter:

This operation can also be viewed as a two-step process, as illustrated in Figure 1.6.

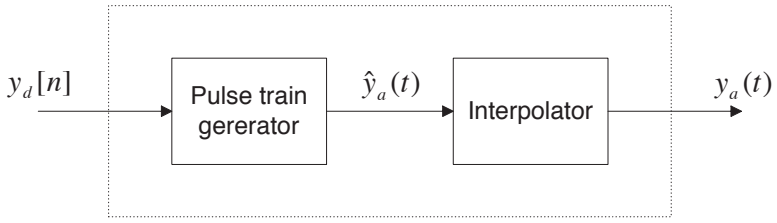


Fig. 1.6 D/A Conversion

- Pulse train generator: in which the digital signal $y_d[n]$ is transformed into a sequence of scaled, analog pulses.
- Interpolator: in which the high frequency components of $\hat{y}_a(t)$ are removed via low-pass filtering to produce a smooth analog output $y_a(t)$.

This two-step representation is a convenient mathematical model of the actual D/A conversion, though, in practice, one device takes care of both steps.

1.2.3 Pros and cons of DSP

Advantages:

- Robustness:
  - Signal levels can be regenerated. For binary signals, the zeros and ones can be easily distinguished even in the presence of noise as long as the noise is small enough. The process of regeneration make a hard decision between a zero and a one, effectively stripping off the noise.
  - Precision not affected by external factors. This means that one gets the results are reproducible.
- Storage capability:
  - DSP system can be interfaced to low-cost devices for storage. The retrieving stored digital signals (often in binary form) results in the regeneration of clean signals.
  - allows for off-line computations
- Flexibility:
  - Easy control of system accuracy via changes in sampling rate and number of representation bits.
  - Software programmable $\Rightarrow$ implementation and fast modification of complex processing functions (e.g. self-tunable digital filter)
- Structure:
  - Easy interconnection of DSP blocks (no loading problem)
  - Possibility of sharing a processor between several tasks

Disadvantages:

- Cost/complexity added by A/D and D/A conversion.
- Input signal bandwidth is technology limited.
- Quantization effects. Discretization of the levels adds quantization noise to the signal.
- Simple conversion of a continuous-time signal to a binary stream of data involves an increase in the bandwidth required for transmission of the data. This however can be mitigated by using compression techniques. For instance, coding an audio signal using MP3 techniques results in a signal which uses much less bandwidth for transmission than a WAVE file.

1.2.4 Discrete-time signal processing (DTSP)

Equivalence of analog and digital signal processing

- It is not at all clear that an arbitrary analog system can be realized as a DSP system.
- Fortunately, for the important class of linear time-invariant systems, this equivalence can be proved under the following conditions:

- the number of representation levels provided by the digital hardware is sufficiently large that quantization errors may be neglected.
- the sampling rate is larger than twice the largest frequency contained in the analog input signal (Sampling Theorem).

The DTSP paradigm

- Based on these considerations, it is convenient to break down the study of DSP into two distinct sets of issues:
  - Discrete-time signal processing (DTSP)
  - Study of quantization effects
- The main object of DTSP is the study of DSP systems under the assumption that finite-precision effects may be neglected $\Rightarrow$ DTSP paradigm.
- Quantization is concerned with practical issues resulting from the use of finite-precision digital hardware in the implementation of DTSP systems.

1.3 Applications of DSP

Typical applications:

- Signal enhancement via frequency selective filtering
- Echo cancellation in telephony:
  - Electric echoes resulting from impedance mismatch and imperfect hybrids.
  - Acoustic echoes due to coupling between loudspeaker and microphone.
- Compression and coding of speech, audio, image, and video signals:
  - Low bit-rate codecs (coder/decoder) for digital speech transmission.
  - Digital music: CD, DAT, DCC, MD,...and now MP3
  - Image and video compression algorithms such as JPEG and MPEG
- Digital simulation of physical processes:
  - Sound wave propagation in a room
  - Baseband simulation of radio signal transmission in mobile communications
- Image processing:
  - Edge and shape detection
  - Image enhancement

Example 1.3: Electric echo cancellation

► In classic telephony (see Figure 1.7), the signal transmitted through a line must pass through a hybrid before being sent to the receiving telephone. The hybrid is used to connect the local loops deserving the customers (2-wire connections) to the main network (4-wire connections). The role of the hybrid is to filter incoming signals so that they are oriented on the right line: signals coming from the network are passed on to the telephone set, signals from the telephone set are passed on to the transmitting line of the network. This separation is never perfect due to impedance mismatch and imperfect hybrids. For instance, the signal received from the network (on the left side of Figure 1.7) can partially leak into the transmitting path of the network, and be sent back to the transmitter (on the right side), with an attenuation and a propagation delay, causing an echo to be heard by the transmitter.
To combat such electric echoes, a signal processing device is inserted at each end of the channel. The incoming signal is monitored, processed through a system which imitates the effect of coupling, and then subtracted from the outgoing signal. ◄

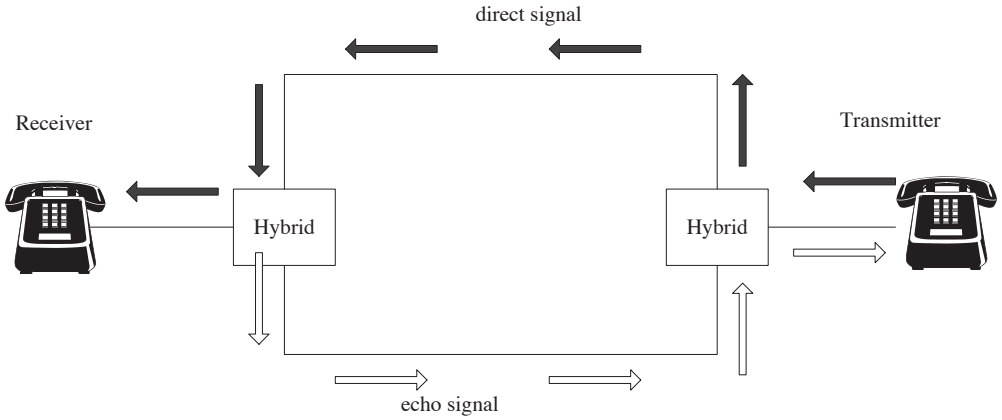


Fig. 1.7 Illustration of electrical echo in classic telephone lines.

Example 1.4: Edge detection

► An edge detection system can be easily devised for grayscale images. It is convenient to study the principle in one dimension before going to two dimensions. Figure 1.8(a) illustrates what we expect an ideal edge to look like: it is a transition between two flat regions, two regions with approximately the same grayscale levels. Of course, in practise, non ideal edges will exist, with smoother transitions and non flat regions. For the sake of explanation, let us concentrate on this ideal edge. Figure 1.8(b) shows the impulse response of a filter that will enable edge detection: the sum of all its samples is equal to one, so that the convolution of a flat region with this impulse response will yield 0, as the central peak will be compensated by the equal side values. On the other hand, when the convolution takes place on the edge, the values on the left of the peak and on the right of the peak will not be the same anymore, so that he output will be nonzero. So to detect edges in a signal in an automatic way, one only has to filter it with a filter like the one shown in Figure 1.8(b) and threshold the output. A result of this procedure is shown on Figure 1.9 for the image shown earlier. ◄

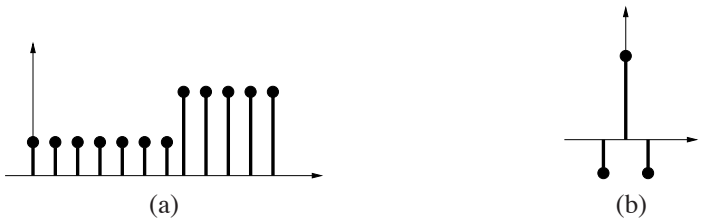


Fig. 1.8 (a) An ideal edge signal in one dimension and (b) an edge detecting filter

Fig. 1.9 Illustration of an edge detection system on image of Figure 1.2.

1.4 Course organization

Three main parts:

- Part I: Basic tools for the analysis of discrete-time signals and systems
  - Time-domain and transform-domain analysis
  - Discrete-time Fourier transform (DTFT) and Z-transform
  - Discrete Fourier transform (DFT)
- Part II: Issues related to the design and implementation of DSP systems:
  - Sampling and reconstruction of analog signals, A/D and D/A
  - Structures for realization of digital filters, signal flow-graphs
  - Digital filter design (FIR and IIR)
  - Study of finite precision effects (quantization noise,...)
  - Fast computation of the DFT (FFT)
  - Programmable digital signal processors (PDSP)
- Part III: Selected applications and advanced topics.
  - Applications of DFT to frequency analysis
  - Multi-rate digital signal processing
  - Introduction to adaptive filtering

Chapter 2

Discrete-time signals and systems

2.1 Discrete-time (DT) signals

Definition:

A DT signal is a sequence of real or complex numbers, that is, a mapping from the set of integers $\mathbb{Z}$ into either $\mathbb{R}$ or $\mathbb{C}$, as in:

$$n \in \mathbb{Z} \rightarrow x[n] \in \mathbb{R} \text{ or } \mathbb{C} \tag{2.1}$$

- n is called the discrete-time index.
- $x[n]$, the nth number in the sequence, is called a sample.
- To refer to the complete sequence, one of the following notations may be used: x, $\{x[n]\}$ or even $x[n]$ if there is no possible ambiguity.[1]
- Unless otherwise specified, it will be assumed that the DT signals of interest may take on complex values, i.e. $x[n] \in \mathbb{C}$.
- We shall denote by $\mathcal{S}$ the set of all complex valued DT signals.

Description:

There are several alternative ways of describing the sample values of a DT signal. Some of the most common are:

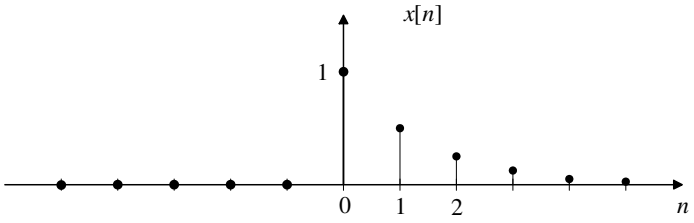
- Sequence notation:

$$x = \{\ldots, 0, 0, \bar{1}, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots\} \tag{2.2}$$

  where the bar on top of symbol 1 indicates origin of time (i.e. $n = 0$)

[1] The latter is a misuse of notation, since $x[n]$ formally refers to the nth signal sample. Following a common practice in the DSP literature, we shall often use the notation $x[n]$ to refer to the complete sequence, in which case the index n should be viewed as a dummy place holder.

- Graphical:



- Explicit mathematical expression:

$$x[n] = \begin{cases} 0 & n < 0, \\ 2^{-n} & n \geq 0. \end{cases} \tag{2.3}$$

- Recursive approach:

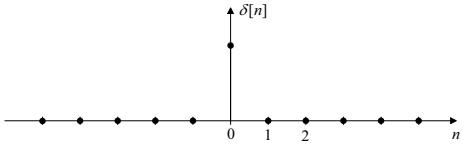
$$x[n] = \begin{cases} 0 & n < 0, \\ 1 & n = 0, \\ \frac{1}{2}x[n-1] & n > 0 \end{cases} \tag{2.4}$$

Depending on the specific sequence $x[n]$, some approaches may lead to more compact representation than others.

Some basic DT signals:

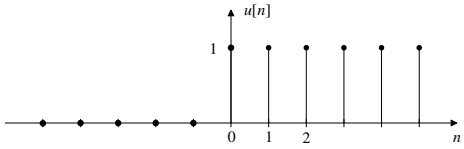
- Unit pulse:

$$\delta[n] = \begin{cases} 1 & n = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$



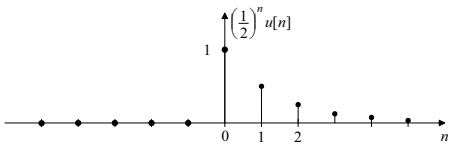
- Unit step:

$$u[n] = \begin{cases} 1 & n \geq 0, \\ 0 & n < 0. \end{cases} \tag{2.6}$$



- Exponential sequence:
  for some $\alpha \in \mathbb{C}$

$$x[n] = A\alpha^n, \quad n \in \mathbb{Z} \tag{2.7}$$



- Complex exponential sequence (CES):
  If $|\alpha| = 1$ in (2.6), i.e. $\alpha = e^{j\omega}$ for some $\omega \in \mathbb{R}$, we have

$$x[n] = Ae^{j\omega n}, \quad n \in \mathbb{Z} \tag{2.8}$$

  where ω is called the angular frequency (in radian).
  - One can show that a CES is periodic with period N, i.e. $x[n+N] = x[n]$, if and only if $\omega = 2\pi k/N$ for some integer k.
  - If $\omega_2 = \omega_1 + 2\pi$, then the two CES signals $x_2[n] = e^{j\omega_2 n}$ and $x_1[n] = e^{j\omega_1 n}$ are indistinguishable (see also Figure 2.1 below).
  - Thus, for DT signals, the concept of frequency response is really limited to an interval of size 2π, typically $[-\pi, \pi]$.

Example 2.1: Uniform sampling

▶ In DSP applications, a common way of generating DT signals is via uniform (or periodic) sampling of an analog signal $x_a(t), t \in \mathbb{R}$, as in:

$$x[n] = x_a(nT_s), \quad n \in \mathbb{Z} \tag{2.9}$$

where $T_s > 0$ is called the sampling period.
For example, consider an analog complex exponential signal given by $x_a(t) = e^{j2\pi Ft}$, where F denotes the analog frequency (in units of 1/time). Uniform sampling of $x_a(t)$ results in the discrete-time CES

$$x[n] = e^{j2\pi FnT_s} = e^{j\omega n} \tag{2.10}$$

where $\omega = 2\pi FT_s$ is the angular frequency (dimensionless).

Figure 2.1 illustrates the limitation of the concept of frequencies in the discrete-time domain. The continuous and dashed-dotted lines respectively show the real part of the analog complex exponential signals $e^{j\omega t}$ and $e^{j(\omega+2\pi)t}$. Upon uniform sampling at integer values of t (i.e. using $T_s = 1$ in (2.9)), the same sample values are obtained for both analog exponentials, as shown by the solid bullets in the figure. That is, the DT CES $e^{j\omega n}$ and $e^{j(\omega+2\pi)n}$ are indistinguishable, even tough the original analog signals are different. This is a simplified illustration of an important phenomenon known as frequency aliasing. ◀

As we saw in the example, for sampled data signals (discrete-time signals formed by sampling continuous-time signals), there are several frequency variables: F the frequency variable (units Hz) for the frequency response of the continuous-time signal, and ω the (normalized) radian frequency of the frequency response of the discrete-time signal. These are related by

$$\omega = 2\pi F/F_s, \tag{2.11}$$

where F_s ($F_s = 1/T_s$) is the sampling frequency. One could, of course, add a radian version of F ($\Omega = 2\pi F$) or a natural frequency version of ω ($f = \omega/(2\pi)$).

For a signal sampled at F_s, the frequency interval $-F_s/2 \le F \le F_s/2$ is mapped to the interval $-\pi \le \omega \le \pi$.

Basic operations on signal:

- Let $\mathcal{S}$ denote the set of all DT signals.

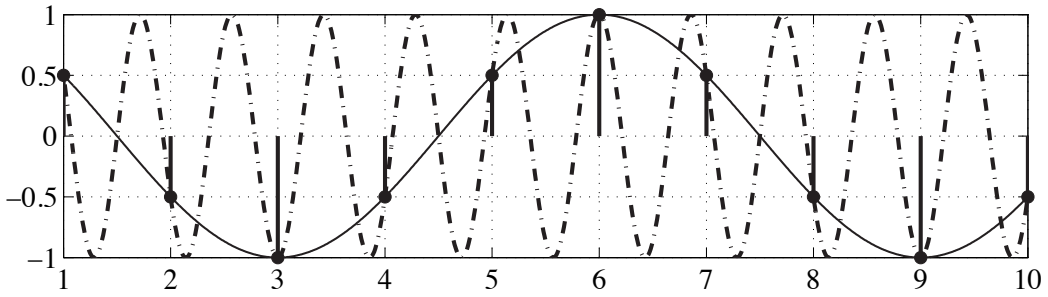


Fig. 2.1 Illustration of the limitation of frequencies in discrete-time.

- We define the following operations on $\mathcal{S}$:

$$\text{scaling:} \qquad (\alpha x)[n] = \alpha x[n], \quad \text{where } \alpha \in \mathbb{C} \tag{2.12}$$

$$\text{addition:} \qquad (x+y)[n] = x[n] + y[n] \tag{2.13}$$

$$\text{multiplication:} \quad (xy)[n] = x[n]y[n] \tag{2.14}$$

- Set $\mathcal{S}$ equipped with addition and scaling is a vector space.

Classes of signals:

The following subspaces of $\mathcal{S}$ play an important role:

- Energy signals: all $x \in \mathcal{S}$ with finite energy, i.e.

$$\mathcal{E}_x \triangleq \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty \tag{2.15}$$

- Power signals: all $x \in \mathcal{S}$ with finite power, i.e.

$$\mathcal{P}_x \triangleq \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2 < \infty \tag{2.16}$$

- Bounded signals: all $x \in \mathcal{S}$ that can be bounded, i.e. we can find $B_x > 0$ such that $|x[n]| \leq B_x$ for all $n \in \mathbb{Z}$
- Absolutely summable: all $x \in \mathcal{S}$ such that $\sum_{n=-\infty}^{\infty} |x[n]| < \infty$

Discrete convolution:

- The discrete convolution of two signals x and y in $\mathcal{S}$ is defined as

$$(x * y)[n] \triangleq \sum_{k=-\infty}^{\infty} x[k]y[n-k] \tag{2.17}$$

  The notation $x[n] * y[n]$ is often used instead of $(x * y)[n]$.

- The convolution of two arbitrary signals may not exist. That is, the sum in (2.17) may diverge. However, if both x and y are absolutely summable, $x*y$ is guaranteed to exist.
- The following properties of $*$ may be proved easily:

$$(a) \quad x*y = y*x \tag{2.18}$$
$$(b) \quad (x*y)*z = x*(y*z) \tag{2.19}$$
$$(c) \quad x*\delta = x \tag{2.20}$$

- For example, (a) is equivalent to

$$\sum_{k=-\infty}^{\infty} x[k]y[n-k] = \sum_{k=-\infty}^{\infty} y[k]x[n-k] \tag{2.21}$$

  which can be proved by changing the index of summation from k to $k' = n-k$ in the LHS summation (try it!)

2.2 DT systems

Definition:

A DT system is a mapping T from $\mathcal{S}$ into itself. That is

$$x \in \mathcal{S} \rightarrow y = Tx \in \mathcal{S} \tag{2.22}$$

An equivalent block diagram form is shown in Figure 2.2 We refer to x as the input signal or excitation, and to y as the output signal, or response.

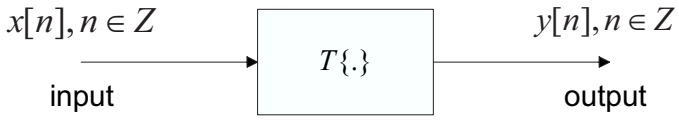


Fig. 2.2 A generic Discrete-Time System seen as a signal mapping

- Note that $y[n]$, the system output at discrete-time n, generally depends on $x[k]$ for all values of $k \in \mathbb{Z}$.
- Even tough the notation $y[n] = T\{x[n]\}$ is often used, the alternative notation $y[n] = (Tx)[n]$ is more precise.
- Some basic systems are described below.

Time reversal:

- Definition:

$$y[n] = (Rx)[n] \triangleq x[-n] \tag{2.23}$$

- Graphical interpretation: mirror image about origin (see Figure 2.3)

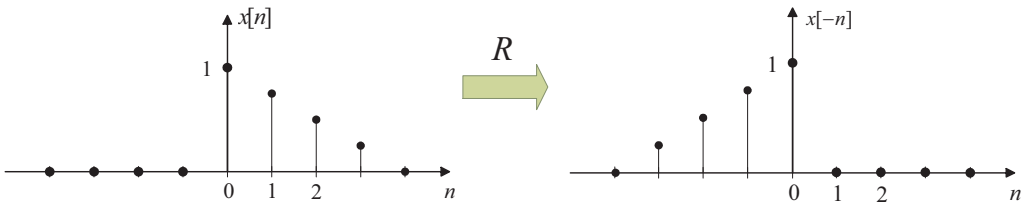


Fig. 2.3 Illustration of time reversal: left, original signal; right: result of time reversal.

Delay or shift by integer k:

- Definition:
$$y[n] = (D_k x)[n] \triangleq x[n-k] \tag{2.24}$$
- Interpretation:
  - $k \geq 0 \Rightarrow$ graph of $x[n]$ shifted by k units to the right
  - $k < 0 \Rightarrow$ graph of $x[n]$ shifted by $|k|$ units to the left
- Application: any signal $x \in \mathcal{S}$ can be expressed as a linear combination of shifted impulses:
$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k] \tag{2.25}$$

Other system examples:

- Moving average system:
$$y[n] = \frac{1}{M+N+1} \sum_{k=-M}^{N} x[n-k] \tag{2.26}$$
This system can be used to smooth out rapid variations in a signal, in order to easily view long-term trends.
- Accumulator:
$$y[n] = \sum_{k=-\infty}^{n} x[k] \tag{2.27}$$

Example 2.2: Application of Moving Average

► An example of Moving Average filtering is given in Figure 2.4: the top part of the figure shows daily NASDAQ index values over a period of almost two years. The bottom part of the figure shows the same signal after applying a moving average system to it, with $M+N+1 = 10$. Clearly, the output of the moving average system is smoother and enables a better view of longer-term trends. ◄

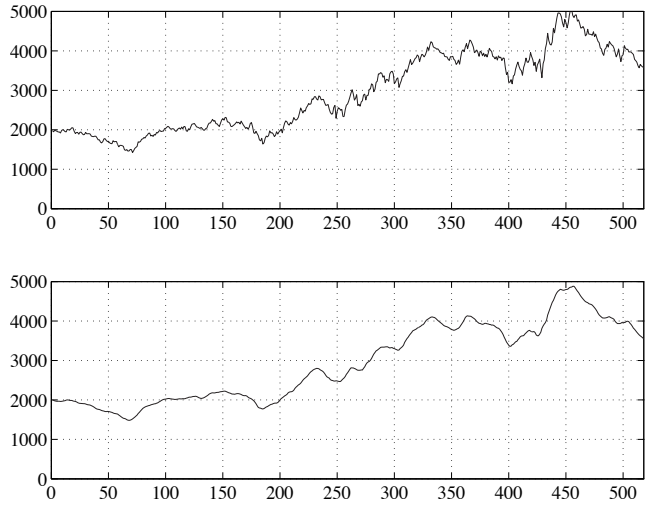


Fig. 2.4 Effect of a moving average filter. (Sample values are connected by straight lines to enable easier viewing)

Basic system properties:

- Memoryless: $y[n] = (Tx)[n]$ is a function of $x[n]$ only.

  Example 2.3:

  ► $y[n] = (x[n])^2$ is memoryless, but $y[n] = \frac{1}{2}(x[n-1] + x[n])$ is not. ◄

  Memoryless systems are also called *static* systems. Systems with memory are termed *dynamic* systems. Further, systems with memory can have finite (length) memory or infinite (length) memory.

- Causal: $y[n]$ only depends on values $x[k]$ for $k \leq n$.

- Anti-causal: $y[n]$ only depends on values $x[k]$ for $k \geq n$[2].

  Example 2.4:

  ► $y[n] = (Tx)[n] = \sum_{k=-\infty}^{n} x[k]$ is causal, $y[n] = \sum_{k=n}^{\infty} x[k]$ is anti-causal. ◄

- Linear: for any $\alpha_1, \alpha_2 \in \mathbb{C}$ and $x_1, x_2 \in \mathcal{S}$,

$$T(\alpha_1 x_1 + \alpha_2 x_2) = \alpha_1 T(x_1) + \alpha_2 T(x_2) \tag{2.28}$$

[2]This definition is not consistent in the literature. Some authors prefer to define an anti-causal system as one where $y[n]$ only depends on values $x[k]$ for $k > n$, i.e. the present output does only depend on future inputs, not even the present input sample.

Example 2.5:

► The moving average system is a good example of linear system, but e.g. $y[n] = (x[n])^2$ is obviously not linear. ◄

- Time-invariant: for any $k \in \mathbb{Z}$,

$$T(D_k x) = D_k(Tx) \tag{2.29}$$

or equivalently, $T\{x[n]\} = y[n] \Rightarrow T\{x[n-k]\} = y[n-k]$.

Example 2.6:

► The moving average system can easily be shown to be time invariant by just replacing n by $n-k$ in the system definition (2.26). On the other hand, a system defined by $y[n] = (Tx)[n] = x[2n]$, i.e. a decimation system, is not time invariant. This is easily seen by considering a delay of $k = 1$: without delay, $y[n] = x[2n]$; with a delay of 1 sample in the input, the first sample of the output is $x[-1]$, which is different from $y[-1] = x[-2]$. ◄

- Stable: x bounded $\Rightarrow y = Tx$ bounded. That is, if $|x[n]| \leq B_x$ for all n, then we can find B_y such that $|y[n]| \leq B_y$ for all n

Remarks:

In on-line processing, causality of the system is very important. A causal system only needs the past and present values of the input to compute the current output sample $y[n]$. A non-causal system would require at least some future values of the input to compute its output. If only a few future samples are needed, this translates into a processing delay, which may not be acceptable for real-time processing; on the other hand, some non-causal systems require the knowledge of all future samples of the input to compute the current output sample: these are of course not suited for on-line processing.

Stability is also a very desirable property of a system. When a system is implemented, it is supposed to meet certain design specifications, and play a certain role in processing its input. An unstable system could be driven to an unbounded output by a bounded input, which is of course an undesirable behaviour.

2.3 Linear time-invariant (LTI) systems

Motivation:

Discrete-time systems that are both linear and time-invariant (LTI) play a central role in digital signal processing:

- Many physical systems are either LTI or approximately so.
- Many efficient tools are available for the analysis and design of LTI systems (e.g. Fourier analysis).

Fundamental property:

Suppose T is LTI and let $y = Tx$ (x arbitrary). Then

$$\boxed{y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]} \tag{2.30}$$

where $h \triangleq T\delta$ is known as the unit pulse (or impulse) response of T.

Proof: Recall that for any DT signal x, we can write

$$x = \sum_{k=-\infty}^{\infty} x[k]\, D_k\delta \tag{2.31}$$

Invoking linearity and then time-invariance of T, we have

$$y = Tx = \sum_{k=-\infty}^{\infty} x[k]\, T(D_k\delta) \tag{2.32}$$

$$= \sum_{k=-\infty}^{\infty} x[k]\, D_k(T\delta) = \sum_{k=-\infty}^{\infty} x[k]\, D_k h \tag{2.33}$$

which is equivalent to (2.30) □.

Discussion:

- The input-output relation for the LTI system may be expressed as a convolution sum:

$$y = x * h \tag{2.34}$$

- For LTI system, knowledge of impulse response $h = T\delta$ completely characterizes the system. For any other input x, we have $Tx = x * h$.
- Graphical interpretation: to compute the sample values of y[n] according to (2.34), or equivalently (2.30), one may proceed as follows:
  - Time reverse sequence $h[k] \Rightarrow h[-k]$
  - Shift $h[-k]$ by n samples $\Rightarrow h[-(k-n)] = h[n-k]$
  - Multiply sequences $x[k]$ and $h[n-k]$ and sum over $k \Rightarrow y[n]$

Example 2.7: Impulse response of the accumulator

▶ Consider the accumulator given by (2.27). There are different ways of deriving the impulse response here. By setting $x[n] = \delta[n]$ in (2.27), we obtain:

$$h[n] = \sum_{k=-\infty}^{n} \delta[k] = \begin{cases} 1 & n \geq 0, \\ 0 & n < 0. \end{cases} \tag{2.35}$$

$$= u[n]$$

An alternative approach is via modification of (2.27), so as to directly reveal the convolution format in (2.30). For instance, by simply changing the index of summation from k to $l = n-k$, equation (2.27) then becomes:

$$\begin{aligned} y[n] &= \sum_{l=0}^{+\infty} x[n-l] \\ &= \sum_{l=-\infty}^{+\infty} u[l]x[n-l], \end{aligned}$$

so that by identification, $h[n] = u[n]$. ◄

Causality:

An LTI system is causal iff $h[n] = 0$ for $n < 0$.

Proof: The input-output relation for an LTI system can be expressed as:

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \tag{2.36}$$

$$= \cdots + h[-1]x[n+1] + h[0]x[n] + h[1]x[n-1] + \cdots \tag{2.37}$$

Clearly, $y[n]$ only depends on values $x[m]$ for $m \leq n$ iff $h[k] = 0$ for $k < 0$ □.

Stability:

An LTI system is stable iff the sequence $h[n]$ is absolutely summable, that is $\sum_{n=-\infty}^{\infty} |h[n]| < \infty$.

Proof: Suppose that h is absolutely summable, that is $\sum_n |h[n]| = M_h < \infty$. Then, for any bounded input sequence x, i.e. such that $|x[n]| \leq B_x < \infty$, we have for the corresponding output sequence y:

$$\begin{aligned} |y[n]| &= |\sum_k x[n-k]h[k]| \leq \sum_k |x[n-k]||h[k]| \\ &\leq \sum_k B_x|h[k]| = B_x M_h < \infty \end{aligned} \tag{2.38}$$

which shows that y is bounded. Now, suppose that $h[n]$ is not absolutely summable, i.e. $\sum_{n=-\infty}^{\infty} |h[n]| = \infty$. Consider the input sequence defined by

$$x[-n] = \begin{cases} h[n]^*/|h[n]| & \text{if } h[n] \neq 0 \\ 0 & \text{if } h[n] = 0. \end{cases}$$

Note that $|x[n]| \leq 1$ so that x is bounded. We leave it as an exercise to the student to verify that in this case, $y[0] = +\infty$, so that the output sequence y is unbounded and so, the corresponding LTI system is not bounded. □

Example 2.8:

▶ Consider LTI system with impulse response $h[n] = \alpha^n u[n]$

- Causality: $h[n] = 0$ for $n < 0 \Rightarrow$ causal
- Stability:

$$\sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=0}^{\infty} |\alpha|^n \tag{2.39}$$

Clearly, the sum diverges if $|\alpha| \geq 1$, while if $|\alpha| < 1$, it converges:

$$\sum_{n=0}^{\infty} |\alpha|^n = \frac{1}{1-|\alpha|} < \infty \tag{2.40}$$

Thus the system is stable provided $|\alpha| < 1$.

◀

FIR versus IIR

- An LTI system has finite impulse response (FIR) if we can find integers $N_1 \leq N_2$ such that

$$h[n] = 0 \quad \text{when} \quad n < N_1 \text{ or } n > N_2 \tag{2.41}$$

- Otherwise the LTI system has an infinite impulse response (IIR).
- For example, the LTI system with

$$h[n] = u[n] - u[n-N] = \begin{cases} 1 & 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases} \tag{2.42}$$

is FIR with $N_1 = 0$ and $N_2 = N-1$. The LTI system with

$$h[n] = \alpha^n u[n] = \begin{cases} \alpha^n & n \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.43}$$

is IIR (cannot find an N_2...)

- FIR systems are necessarily stable:

$$\sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=N_1}^{N_2} |h[n]| < \infty \tag{2.44}$$

Interconnection of LTI systems:

- Cascade interconnection (Figure 2.5):

$$y = (x * h_1) * h_2 = x * (h_1 * h_2) \tag{2.45}$$
$$= x * (h_2 * h_1) = (x * h_2) * h_1 \tag{2.46}$$

LTI systems commute; not true for arbitrary systems.

- Parallel interconnection (Figure 2.6):

$$y = x * h_1 + x * h_2 = x * (h_1 + h_2) \tag{2.47}$$

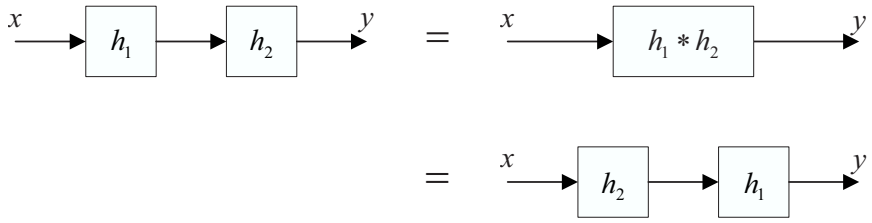


Fig. 2.5 Cascade interconnection of LTI systems

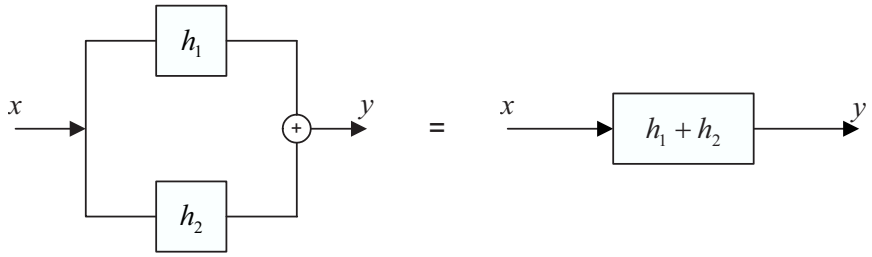


Fig. 2.6 Parallel interconnection of LTI systems

2.4 LTI systems described by linear constant coefficient difference equations (LCCDE)

Definition:

A discrete-time system can be described by an LCCDE of order N if, for any arbitrary input x and corresponding output y,

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{2.48}$$

where $a_0 \neq 0$ and $a_N \neq 0$.

Example 2.9: Difference Equation for Accumulator

► Consider the accumulator system:

$$x[n] \longrightarrow y[n] = \sum_{k=-\infty}^{n} x[k] \tag{2.49}$$

which we know to be LTI. Observe that

$$\begin{aligned} y[n] &= \sum_{k=-\infty}^{n-1} x[k] + x[n] \\ &= y[n-1] + x[n] \end{aligned} \tag{2.50}$$

This is an LCCDE of order $N = 1$ ($M = 0, a_0 = 1, a_1 = -1, b_0 = 1$) ◄

Remarks:

LCCDEs lead to efficient recursive implementation:

- Recursive because the computation of $y[n]$ makes use past output signal values (e.g. $y[n-1]$ in (2.50)).
- These past output signal values contain all the necessary information about earlier states of the system.
- While (2.49) requires an infinite number of adders and memory units, (2.50) requires only one adder and one memory unit.

Solution of LCCDE:

The problem of solving the LCCDE (2.48) for an output sequence $y[n]$, given a particular input sequence $x[n]$, is of particular interest in DSP. Here, we only look at general properties of the solutions $y[n]$. The presentation of a systematic solution technique is deferred to Chapter 4.

Structure of the general solution:

The most general solution of the LCCDE (2.48) can be expressed in the form

$$y[n] = y_p[n] + y_k[n] \tag{2.51}$$

- $y_p[n]$ is any particular solution of the LCCDE
- $y_h[n]$ is the general solution of the homogeneous equation

$$\sum_{k=0}^{N} a_k y_h[n-k] = 0 \tag{2.52}$$

Example 2.10:

▶ Consider the 1st order LCCDE

$$y[n] = ay[n-1] + x[n] \tag{2.53}$$

with input $x[n] = A\delta[n]$.
It is easy to verify that the following is a solution to (2.53):

$$y_p[n] = Aa^n u[n] \tag{2.54}$$

The homogeneous equation is

$$y_h[n] + ay_h[n-1] = 0 \tag{2.55}$$

Its general solution is given by

$$y_h[n] = Ba^n \tag{2.56}$$

where B is an arbitrary constant. So, the general solution of the LCCDE is given by:

$$y[n] = Aa^n u[n] + Ba^n \tag{2.57}$$

◀

Remarks on example:

- The solution of (2.53) is not unique: B is arbitrary
- (2.53) does not necessarily define a linear system: in the case $A = 0$ and $B = 1$, we obtain $y[n] \neq 0$ while $x[n] = 0$.
- The solution of (2.53) is not causal in general: the choice $B = -A$ gives

$$y[n] = \begin{cases} 0 & n \geq 0, \\ -Aa^n & n < 0. \end{cases} \tag{2.58}$$

  which is an anti-causal solution.

General remarks:

- The solution of an LCCDE is not unique; for an Nth order LCCDE, uniqueness requires the specification of N initial conditions.
- An LCCDE does not necessarily correspond to a causal LTI system.
- However, it can be shown that the LCCDE will correspond to a unique causal LTI system if we further assume that this system is *initially at rest*. That is:

$$x[n] = 0 \text{ for } n < n_0 \quad \Longrightarrow \quad y[n] = 0 \text{ for } n < n_0 \tag{2.59}$$

- This is equivalent to assuming zero initial conditions when solving LCCDE, that is: $y[n_0 - l] = 0$ for $l = 1, ..., N$.
- Back to example: Here $x[n] = A\delta[n] = 0$ for $n < 0$. Assuming zero-initial conditions, we must have $y[-1] = 0$. Using this condition in (2.57), we have

$$y[-1] = Ba^{-1} = 0 \Rightarrow B = 0 \tag{2.60}$$

  so that finally, we obtain a unique (causal) solution $y[n] = Aa^n u[n]$.
- A systematic approach for solving LCCDE will be presented in Chap. 4.

2.5 Problems

Problem 2.1: Basic Signal Transformations
Let the sequenc $x[n]$ be as illustrated in figure 2.7. Determine and draw the following sequences:

1. $x[-n]$
2. $x[n-2]$
3. $x[3-n]$

4. $x[n] * \delta[n-3]$
5. $x[n] * (u[n] - u[n-2])$
6. $x[2n]$.

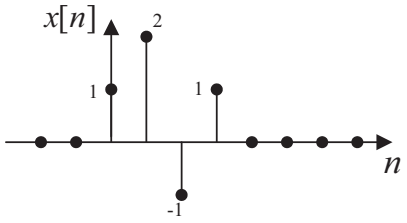


Fig. 2.7 Sequence to be used in problem 2.1.

Problem 2.2: Impulse response from LCCDE
Let the input-output relationship of an LTI system be described by the following difference equation:

$$y[n] + \frac{1}{3}y[n-1] = x[n] - 4x[n-1],$$

together with initial rest conditions.

1. Determine the impulse response $h[n]$ of the corresponding system.
2. Is the system causal, stable, FIR, IIR ?
3. Determine the output of this system if the input $x[n]$ is given by $x[n] = \left\{\ldots, 0, 0, \underline{\frac{1}{2}}, 0, -1, 0, 0, 0, \ldots\right\}$.

Problem 2.3:
Let T_i, $i = 1,2,3$, be stable LTI systems, with corresponding impulse responses $h_i[n]$.

1. Is the system illustrated in figure 2.8 LTI ? If so, prove it, otherwise give a counter-example.
2. If the system is LTI, what is its impulse response ?

Problem 2.4:
Let T_i, $i = 1,2$, be LTI systems, with corresponding impulse responses

$$\begin{aligned} h_1[n] &= \delta[n] - 3\delta[n-2] + \delta[n-3] \\ h_2[n] &= \frac{1}{2}\delta[n-1] + \frac{1}{4}\delta[n-2]. \end{aligned}$$

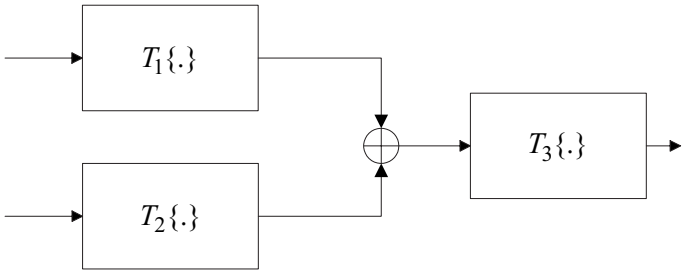


Fig. 2.8 System connections to be used in problem 2.3.

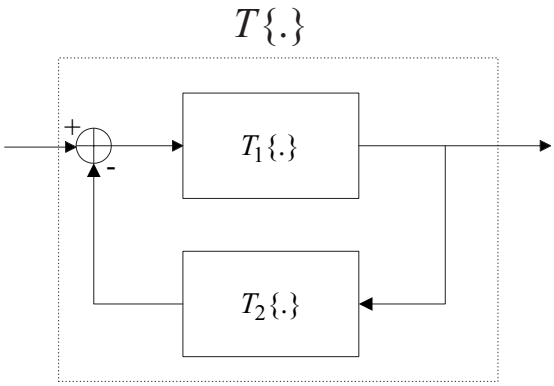


Fig. 2.9 Feedback connection for problem 2.4.

1. Write a linear constant coefficient difference equation (LCCDE) corresponding to the input-output relationships of T_1 and T_2.

2. Consider the overall system T in figure 2.9, with impulse response $h[n]$. Write a LCCDE corresponding to its input-output relationship. Explain how you could compute $h[n]$ from this LCCDE. Compute $h[0]$, $h[1]$ and $h[2]$.

Chapter 3

Discrete-time Fourier transform (DTFT)

3.1 The DTFT and its inverse

Definition:

The DTFT is a transformation that maps DT signal $x[n]$ into a complex-valued function of the real variable, namely

$$\boxed{X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}, \quad \omega \in \mathbb{R}} \tag{3.1}$$

Remarks:

- In general, $X(\omega) \in \mathbb{C}$
- $X(\omega + 2\pi) = X(\omega) \Rightarrow \omega \in [-\pi, \pi]$ is sufficient
- $X(\omega)$ is called the spectrum of $x[n]$:

$$X(\omega) = |X(\omega)|e^{j\angle X(\omega)} \Rightarrow \begin{cases} |X(\omega)| = & \text{magnitude spectrum,} \\ \angle X(\omega) = & \text{phase spectrum.} \end{cases} \tag{3.2}$$

  The magnitude spectrum is often expressed in decibel (dB):

$$|X(\omega)|_{\text{dB}} = 20\log_{10}|X(\omega)| \tag{3.3}$$

Inverse DTFT:

Let $X(\omega)$ be the DTFT of DT signal $x[n]$. Then

$$\boxed{x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}d\omega, \quad n \in \mathbb{Z}} \tag{3.4}$$

Proof: First note that $\int_{-\pi}^{\pi} e^{j\omega n} d\omega = 2\pi\delta[n]$. Then, we have

$$\begin{aligned}\int_{-\pi}^{\pi} X(\omega) e^{j\omega n} d\omega &= \int_{-\pi}^{\pi} \{ \sum_{k=-\infty}^{\infty} x[k] e^{-j\omega k} \} e^{j\omega n} d\omega \\ &= \sum_{k=-\infty}^{\infty} x[k] \int_{-\pi}^{\pi} e^{j\omega(n-k)} d\omega \\ &= 2\pi \sum_{k=-\infty}^{\infty} x[k]\, \delta[n-k] = 2\pi x[n] \quad \square \end{aligned} \tag{3.5}$$

Remarks:

- In (3.4), $x[n]$ is expressed as a weighted sum of complex exponential signals $e^{j\omega n}$, $\omega \in [-\pi, \pi]$, with weight $X(\omega)$
- Accordingly, the DTFT $X(\omega)$ describes the frequency content of $x[n]$
- Since the DT signal $x[n]$ can be recovered uniquely from its DTFT $X(\omega)$, we say that $x[n]$ together with $X(\omega)$ form a DTFT pair, and write:

$$x[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\omega) \tag{3.6}$$

- DTFT equation (3.1) is called analysis relation; while inverse DTFT equation (3.4) is called synthesis relation.

3.2 Convergence of the DTFT:

Introduction:

- For the DTFT to exist, the series $\sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$ must converge
- That is, the partial sum

$$X_M(\omega) = \sum_{n=-M}^{M} x[n] e^{-j\omega n} \tag{3.7}$$

  must converge to a limit $X(\omega)$ as $M \to \infty$.
- Below, we discuss the convergence of $X_M(\omega)$ for three different signal classes of practical interest, namely:
  - absolutely summable signals
  - energy signals
  - power signals
- In each case, we state without proof the main theoretical results and illustrate the theory with corresponding examples of DTFTs.

Absolutely summable signals:

- Recall: $x[n]$ is said to be absolutely summable (sometimes denoted as $\mathcal{L}_1$) iff

$$\sum_{n=-\infty}^{\infty} |x[n]| < \infty \tag{3.8}$$

- In this case, $X(\omega)$ always exists because:

$$|\sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}| \leq \sum_{n=-\infty}^{\infty} |x[n]e^{-j\omega n}| = \sum_{n=-\infty}^{\infty} |x[n]| < \infty \tag{3.9}$$

- Possible to show that $X_M(\omega)$ converges *uniformly* to $X(\omega)$, that is:

  For all $\varepsilon > 0$, can find M_ε such that $|X(\omega) - X_M(\omega)| < \varepsilon$ for all $M > M_\varepsilon$ and for all $\omega \in \mathbb{R}$.

- $X(\omega)$ is continuous and $d^pX(\omega)/d\omega^p$ exists and continuous for all $p \geq 1$.

Example 3.1:

▶ Consider the unit pulse sequence $x[n] = \delta[n]$. The corresponding the DTFT is simply $X(\omega) = 1$.

More generally, consider an arbitrary finite duration signal, say ($N_1 \leq N_2$)

$$x[n] = \sum_{k=N_1}^{N_2} c_k \delta[n-k]$$

Clearly, $x[n]$ is absolutely summable; its DTFT is given by the finite sum

$$X(\omega) = \sum_{n=N_1}^{N_2} c_n e^{-j\omega n}$$

◀

Example 3.2:

▶ The exponential sequence $x[n] = a^n u[n]$ with $|a| < 1$ is absolutely summable. Its DTFT is easily computed to be

$$X(\omega) = \sum_{n=0}^{\infty} (ae^{j\omega})^n = \frac{1}{1-ae^{-j\omega}}.$$

Figure 3.1 illustrates the uniform convergence of $X_M(\omega)$ as defined in (3.7) to $X(\omega)$ in the special case $a = 0.8$. As M increases, the whole $X_M(\omega)$ curve tends to $X(\omega)$ at every frequency. ◀

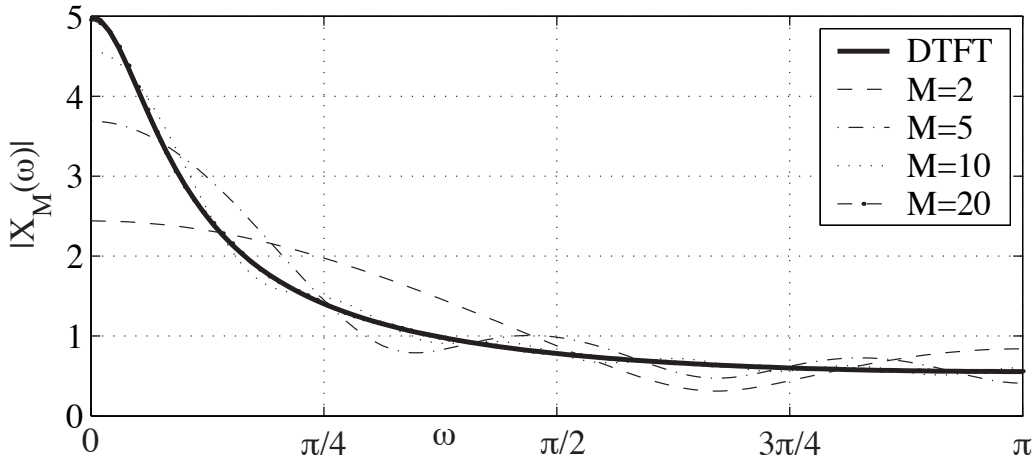


Fig. 3.1 Illustration of uniform convergence for an exponential sequence.

Energy signals:

- Recall: $x[n]$ is an energy signal (square summable or $\mathcal{L}_2$) iff

$$\mathcal{E}_x \triangleq \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty \tag{3.10}$$

- In this case, it can be proved that $X_M(\omega)$ converges in the *mean square* sense to $X(\omega)$, that is:

$$\lim_{M\to\infty} \int_{-\pi}^{\pi} |X(\omega) - X_M(\omega)|^2 d\omega = 0 \tag{3.11}$$

- Mean-square convergence is weaker than uniform convergence:
  - $\mathcal{L}_1$ convergence implies $\mathcal{L}_2$ convergence.
  - $\mathcal{L}_2$ convergence does not guarantee that $\lim_{M\to\infty} X_M(\omega)$ exists for all ω.
  - e.g., $X(\omega)$ may be discontinuous (jump) at certain points.

Example 3.3: Ideal low-pass filter

▶ Consider the DTFT defined by

$$X(\omega) = \begin{cases} 1 & |\omega| < \omega_c \\ 0 & \omega_c < \omega < \pi \end{cases}$$

whose graph is illustrated in Figure 3.2(a). The corresponding DT signal is obtained via (3.4) as follows:

$$x[n] = \frac{1}{2\pi} \int_{-\omega_c}^{\omega_c} e^{j\omega n} d\omega = \frac{\omega_c}{\pi} \operatorname{sinc}\left(\frac{\omega_c n}{\pi}\right) \tag{3.12}$$

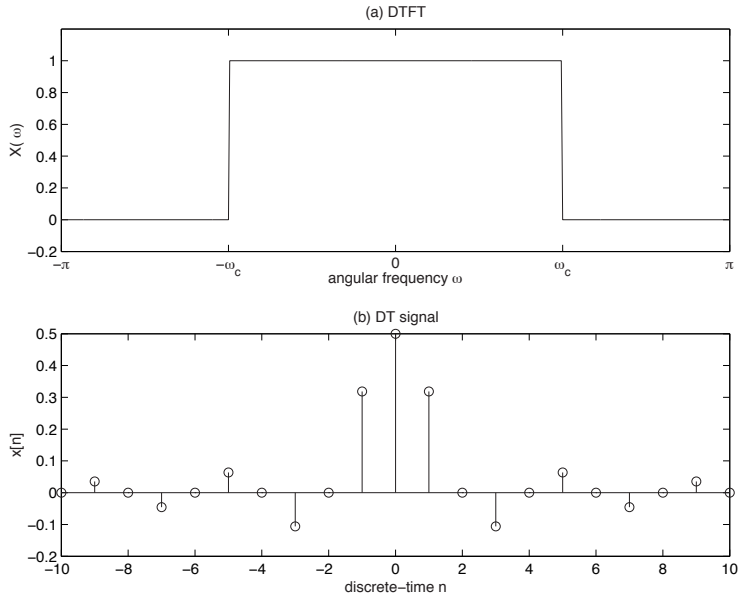


Fig. 3.2 Impulse response of the ideal low-pass filter.

where the sinc function is defined as

$$\mathrm{sinc}(x) = \frac{\sin \pi x}{\pi x}.$$

The signal $x[n]$ is illustrated in Figure 3.2(b) for the case $\omega_c = \pi/2$

It can be shown that:

- $\sum_n |x[n]| = \infty \Rightarrow x[n]$ not absolutely summable
- $\sum_n |x[n]|^2 < \infty \Rightarrow x[n]$ square summable

Here, $X_M(\omega)$ converges to $X(\omega)$ in the mean-square only; the convergence is not uniform. We have the so-called Gibb's phenomenon (Figure 3.3):

- There is an overshoot of size $\Delta X \approx 0.09$ near $\pm\omega_c$;
- The overshoot cannot be eliminated by increasing M;
- This is a manifestation of the non-uniform convergence;
- It has important implications in filter design.

◀

Power signals

- Recall: $x[n]$ is a power signal iff

$$\mathcal{P}_x \triangleq \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2 < \infty \tag{3.13}$$

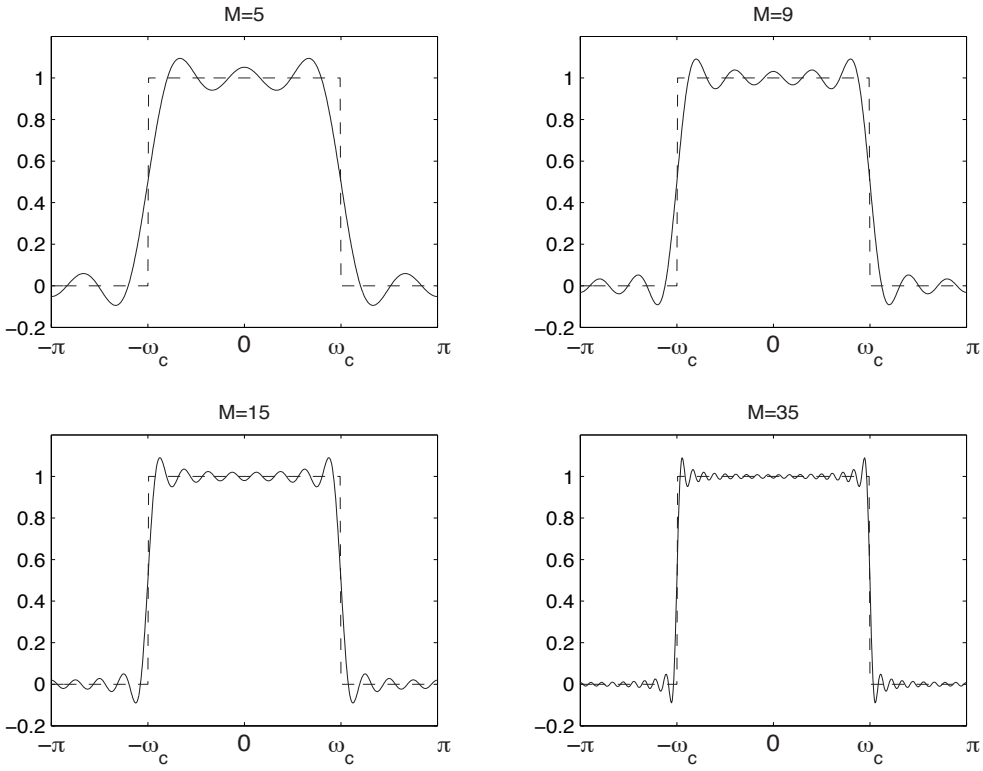


Fig. 3.3 Illustration of the Gibb's phenomenon.

- If $x[n]$ has infinite energy but finite power, $X_M(\omega)$ may still converge to a generalized function $X(\omega)$:
- The expression of $X(\omega)$ typically contains continuous delta functions in the variable ω.
- Most power signals do not have a DTFT even in this sense. The exceptions include the following useful DT signals:
  - Periodic signals
  - Unit step
- While the formal mathematical treatment of generalized functions is beyond the scope of this course, we shall frequently make use of certain basic results, as developed in the examples below.

Example 3.4:

▶ Consider the following DTFT

$$X(\omega) = 2\pi \sum_{k=-\infty}^{\infty} \delta_a(\omega - 2\pi k)$$

where $\delta_a(\omega)$ denotes an analog delta function centred at $\omega = 0$. By using the synthesis relation (3.4), one gets

$$\begin{aligned} x[n] &= \frac{1}{2\pi} \int_{-\pi}^{\pi} 2\pi \sum_{k=-\infty}^{\infty} e^{j\omega n} \delta_a(\omega - 2\pi k) d\omega \\ &= \sum_{k=-\infty}^{\infty} \int_{-\pi}^{\pi} e^{j\omega n} \delta_a(\omega - 2\pi k) d\omega \qquad (3.14) \\ &= \int_{-\pi}^{\pi} e^{j\omega n} \delta_a(\omega) d\omega \\ &= e^{j0n} = 1 \qquad (3.15) \end{aligned}$$

since in (3.14) the only value of k for which the delta function will be non-zero in the integration interval is $k = 0$. ◀

Example 3.5:

▶ Consider the unit step sequence $u[n]$ and let $U(\omega)$ denotes its DTFT. The following result is given without proof:

$$U(\omega) = \frac{1}{1 - e^{-j\omega}} + \pi \sum_{k=-\infty}^{\infty} \delta_a(\omega - 2\pi k)$$

◀

3.3 Properties of the DTFT

Notations:

- DTFT pairs:

$$\begin{aligned} x[n] &\overset{\mathcal{F}}{\longleftrightarrow} && X(\omega) \\ y[n] &\overset{\mathcal{F}}{\longleftrightarrow} && Y(\omega) \end{aligned}$$

- Real and imaginary parts:

$$\begin{aligned} x[n] &= x_R[n] + jx_I[n] \\ X(\omega) &= X_R(\omega) + jX_I(\omega) \end{aligned}$$

- Even and odd components:

$$x[n] = x_e[n] + x_o[n] \tag{3.16}$$

$$x_e[n] \triangleq \frac{1}{2}(x[n] + x^*[-n]) = x_e^*[-n] \quad \text{(even)} \tag{3.17}$$

$$x_o[n] \triangleq \frac{1}{2}(x[n] - x^*[-n]) = -x_o^*[-n] \quad \text{(odd)} \tag{3.18}$$

- Similarly:

$$X(\omega) = X_e(\omega) + X_o(\omega) \tag{3.19}$$

$$X_e(\omega) \triangleq \frac{1}{2}(X(\omega) + X^*(-\omega)) = X_e^*(-\omega) \quad \text{(even)} \tag{3.20}$$

$$X_o(\omega) \triangleq \frac{1}{2}(X(\omega) - X^*(-\omega)) = -X_o^*(-\omega) \quad \text{(odd)} \tag{3.21}$$

Basic symmetries:

$$x[-n] \overset{\mathcal{F}}{\longleftrightarrow} X(-\omega) \tag{3.22}$$

$$x^*[n] \overset{\mathcal{F}}{\longleftrightarrow} X^*(-\omega) \tag{3.23}$$

$$x_R[n] \overset{\mathcal{F}}{\longleftrightarrow} X_e(\omega) \tag{3.24}$$

$$jx_I[n] \overset{\mathcal{F}}{\longleftrightarrow} X_o(\omega) \tag{3.25}$$

$$x_e[n] \overset{\mathcal{F}}{\longleftrightarrow} X_R(\omega) \tag{3.26}$$

$$x_o[n] \overset{\mathcal{F}}{\longleftrightarrow} jX_I(\omega) \tag{3.27}$$

Real/imaginary signals:

- If $x[n]$ is real, then $X(\omega) = X^*(-\omega)$
- If $x[n]$ is purely imaginary, then $X(\omega) = -X^*(-\omega)$

Proof: $x[n] \in \mathbb{R} \Rightarrow x_I[n] = 0 \Rightarrow X_o(\omega) = 0$. In turns, this implies $X(\omega) = X_e(\omega) = X^*(-\omega)$. Similar argument for purely imaginary case. □

Remarks:

- For $x[n]$ real, symmetry $X(\omega) = X^*(-\omega)$ implies

$$|X(\omega)| = |X(-\omega)|, \quad \angle X(\omega) = -\angle X(-\omega) \tag{3.28}$$
$$X_R(\omega) = X_R(-\omega), \quad X_I(\omega) = -X_I(-\omega) \tag{3.29}$$

  This means that, for real signals, one only needs to specify $X(\omega)$ for $\omega \in [0, \pi]$, because of symmetry.

Linearity:

$$ax[n] + by[n] \overset{\mathcal{F}}{\longleftrightarrow} aX(\omega) + bY(\omega) \tag{3.30}$$

Time shift (very important):

$$x[n-d] \overset{\mathcal{F}}{\longleftrightarrow} e^{-j\omega d} X(\omega) \tag{3.31}$$

Frequency modulation:

$$e^{j\omega_0 n} x[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\omega - \omega_0) \tag{3.32}$$

Differentiation:

$$nx[n] \overset{\mathcal{F}}{\longleftrightarrow} j\frac{dX(\omega)}{d\omega} \tag{3.33}$$

Example 3.6:

▶ $y[n] = b_0 x[n] + b_1 x[n-1]$. By using the linearity and time shift properties, one gets that

$$Y(\omega) = (b_0 + b_1 e^{-j\omega}) X(\omega)$$

◀

Example 3.7:

► $y[n] = \cos(\omega_o n)x[n]$. Recall first that $e^{j\omega_0 n} = \cos(\omega_o n) + j\sin(\omega_o n)$, so that $\cos(\omega_o n) = \frac{1}{2}(e^{j\omega_0 n} + e^{-j\omega_0 n})$, and, by linearity and time shift:

$$Y(\omega) = \frac{1}{2}\left(X(\omega-\omega_0)+X(\omega+\omega_0)\right)$$

◄

Convolution:

$$x[n] * y[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\omega)Y(\omega) \tag{3.34}$$

Multiplication:

$$x[n]y[n] \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\phi)Y(\omega-\phi)d\phi \tag{3.35}$$

We refer to the RHS of (3.35) as the circular convolution of periodic functions $X(\omega)$ and $Y(\omega)$.

Thus:

- DT convolution $\overset{\mathcal{F}}{\longleftrightarrow}$ multiplication in ω while
- DT multiplication $\overset{\mathcal{F}}{\longleftrightarrow}$ circular convolution in ω

We say that these two properties are dual of each other

Parseval's relation:

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} |X(\omega)|^2 d\omega \tag{3.36}$$

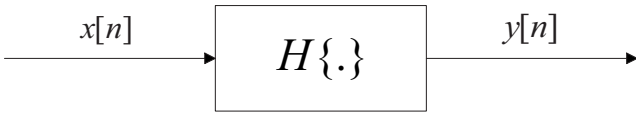
Plancherel's relation:

$$\sum_{n=-\infty}^{\infty} x[n]y[n]^* = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)Y(\omega)^* d\omega \tag{3.37}$$

We have met two kinds of convolution so far and one more is familiar from continuous-time systems. For functions of a continuous variable we have ordinary or linear convolution involving an integral from $-\infty$ to ∞. Here we have seen periodic or circular convolution of periodic functions of a continuous variable (in this case the periodic frequency responses of discrete-time systems) involving an integral over one period. For discrete-time systems, we have seen the ordinary or linear convolution which involves a sum from $-\infty$ to Later in the study of DFTs, we will meet the periodic or circular convolution of sequences involving a sum over one period.

3.4 Frequency analysis of LTI systems

LTI system (recap):



$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \tag{3.38}$$

$$h[n] = \mathcal{H}\{\delta[n]\} \tag{3.39}$$

Response to complex exponential:

- Let $x[n] = e^{j\omega n}$, $n \in \mathbb{Z}$.
- Corresponding output signal:

$$\begin{aligned} y[n] &= \sum_k h[k]x[n-k] \\ &= \sum_k h[k]e^{j\omega(n-k)} \\ &= \{\sum_k h[k]e^{-j\omega k}\}e^{j\omega n} = H(\omega)x[n] \end{aligned} \tag{3.40}$$

  where we recognize $H(\omega)$ as the DTFT of $h[n]$.

- Eigenvector interpretation:
  - $x[n] = e^{j\omega n}$ behaves as an eigenvector of LTI system $\mathcal{H}$: $\mathcal{H}x = \lambda x$
  - Corresponding eigenvalue $\lambda = H(\omega)$ provides the system gain

Definition:

Consider an LTI system $\mathcal{H}$ with impulse response $h[n]$. The frequency response of $\mathcal{H}$, denoted $H(\omega)$, is defined as the DTFT of $h[n]$:

$$H(\omega) = \sum_{n=-\infty}^{\infty} h[n]e^{-j\omega n}, \quad \omega \in \mathbb{R} \tag{3.41}$$

Remarks:

- If DT system $\mathcal{H}$ is stable, then the sequence $h[n]$ is absolutely summable and the DTFT converges uniformly:
  - $H(\omega)$ exists and is continuous.

- $H(\omega)$ known $\Rightarrow$ we can recover $h[n]$ from the inverse DTFT relation:

$$h[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(\omega)e^{j\omega n}d\omega \tag{3.42}$$

- We refer to $|H(\omega)|$ as the magnitude response of the system.
  We refer to $\angle H(\omega)$ as the phase spectrum.

Properties:

Let $\mathcal{H}$ be an LTI system with frequency response $H(\omega)$. Let $y[n]$ denote the response of $\mathcal{H}$ to an arbitrary input $x[n]$. We have

$$\boxed{Y(\omega) = H(\omega)X(\omega)} \tag{3.43}$$

Proof:

$$\mathcal{H} \text{ LTI } \Rightarrow\ y = h * x\ \Rightarrow\ Y(\omega) = H(\omega)X(\omega)$$

Interpretation:

- Recall that input signal $x[n]$ may be expressed as a weighted sum of complex exponential sequences via the inverse DTFT:

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}d\omega \tag{3.44}$$

- According to (3.43),

$$y[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(\omega)X(\omega)e^{j\omega n}d\omega \tag{3.45}$$

- Note filtering role of $H(\omega)$: each frequency component in $x[n]$, i.e. $X(\omega)e^{j\omega n}$, is affected by $H(\omega)$ in the final system output $y[n]$:
  - gain or attenuation of $|H(\omega)|$
  - phase rotation of $\angle H(\omega)$.

Example 3.8:

▶ Consider a causal moving average system:

$$y[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k] \tag{3.46}$$

- Impulse response:

$$\begin{aligned} h[n] &= \frac{1}{M}\sum_{k=0}^{M-1}\delta[n-k] \\ &= \begin{cases} \frac{1}{M} & 0 \le n < M \\ 0 & \text{otherwise} \end{cases} \\ &= \frac{1}{M}(u[n] - u[n-M]) \end{aligned}$$

- Frequency response:

$$\begin{aligned} H(\omega) &= \frac{1}{M}\sum_{n=0}^{M-1} e^{-j\omega n} \\ &= \frac{1}{M}\,\frac{e^{-j\omega M}-1}{e^{-j\omega}-1} \\ &= \frac{1}{M}\,\frac{e^{-j\omega M/2}}{e^{-j\omega/2}}\,\frac{e^{-j\omega M/2}-e^{j\omega M/2}}{e^{-j\omega/2}-e^{j\omega/2}} \\ &= \frac{1}{M}\,e^{-j\omega(M-1)/2}\,\frac{\sin(\omega M/2)}{\sin(\omega/2)} \end{aligned} \tag{3.47}$$

- The magnitude response is illustrated in Figure 3.5.
  - zeros at $\omega = 2\pi k/M$ ($k \neq 0$), where $\frac{\sin(\omega M/2)}{\sin(\omega/2)} = 0$
  - level of first sidelobe ≈ -13dB
- The phase response is illustrated in Figure 3.4.
  - negative slope of $-(M-1)/2$
  - jumps of π at $\omega = 2\pi k/M$ ($k \neq 0$), where $\frac{\sin(\omega M/2)}{\sin(\omega/2)}$ changes its sign.

◀

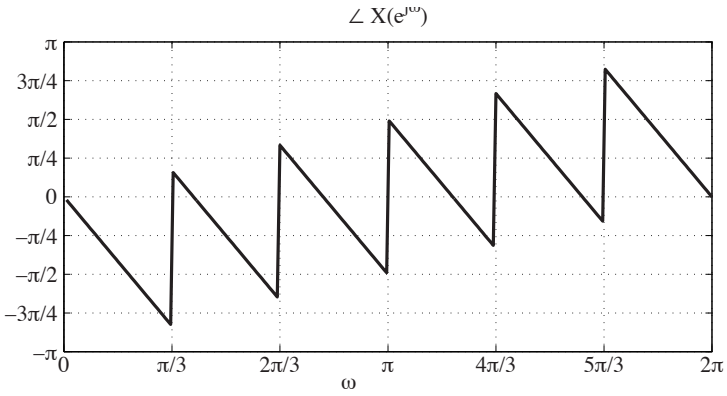


Fig. 3.4 Phase response of the causal moving average system ($M = 6$)

Further remarks:

- We have seen that for LTI system, $Y(\omega) = H(\omega)X(\omega)$:
  - $X(\omega) = 0$ at a given frequency $\omega \Rightarrow Y(\omega) = 0$
    if not: system is non-linear or time-varying
  - $H(\omega) = 0 \Rightarrow Y(\omega) = 0$, regardless of input
- If system is LTI, its frequency response may be computed as

$$H(\omega) = \frac{Y(\omega)}{X(\omega)} \tag{3.48}$$

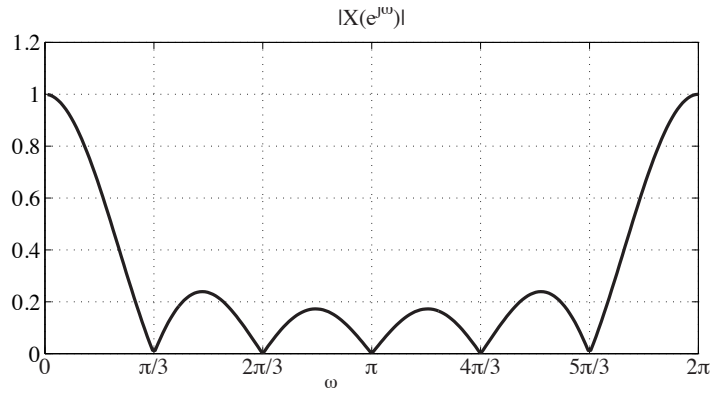


Fig. 3.5 Magnitude Response of the causal moving average system ($M = 6$)

3.5 LTI systems characterized by LCCDE

LCCDE (recap):

- DT system obeys LCCDE of order N if

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{3.49}$$

  where $a_0 \neq 0$ and $a_N \neq 0$.

- If we further assume initial rest conditions, i.e.:

$$x[n] = 0 \text{ for } n < n_0 \quad \Longrightarrow \quad y[n] = 0 \text{ for } n < n_0 \tag{3.50}$$

  LCCDE corresponds to unique causal LTI system.

Frequency response:

Taking DTFT on both sides of (3.49):

$$\begin{aligned} & \sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \\ \Rightarrow \quad & \sum_{k=0}^{N} a_k e^{-j\omega k} Y(\omega) = \sum_{k=0}^{M} b_k e^{-j\omega k} X(\omega) \\ \Rightarrow \quad & H(\omega) = \frac{Y(\omega)}{X(\omega)} = \frac{\sum_{k=0}^{M} b_k e^{-j\omega k}}{\sum_{k=0}^{N} a_k e^{-j\omega k}} \end{aligned} \tag{3.51}$$

3.6 Ideal frequency selective filters:

Ideal low-pass:

$$H_{LP}(\omega) = \begin{cases} 1 & \text{if } |\omega| < \omega_c \\ 0 & \text{if } \omega_c < |\omega| < \pi \end{cases}$$

$$h_{LP}[n] = \frac{\omega_c}{\pi}\,\text{sinc}\left(\frac{\omega_c n}{\pi}\right) \qquad (3.52)$$

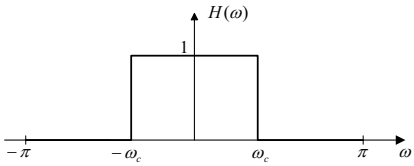


Figure 3.6: Low-pass filter

Ideal high-pass:

$$H_{HP}(\omega) = \begin{cases} 1 & \text{if } \omega_c < |\omega| < \pi \\ 0 & \text{if } |\omega| < \omega_c \end{cases}$$

$$h_{HP}[n] = \delta[n] - h_{LP}[n] \qquad (3.53)$$

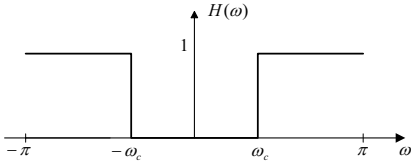


Figure 3.7: High-pass filter

Ideal band-pass:

$$H_{BP}(\omega) = \begin{cases} 1 & \text{if } |\omega - \omega_o| < B/2 \\ 0 & \text{else in } [-\pi, \pi] \end{cases}$$

$$h_{BP}[n] = 2\cos(\omega_o n) h_{LP}[n]|_{\omega_c \equiv B/2} \qquad (3.54)$$

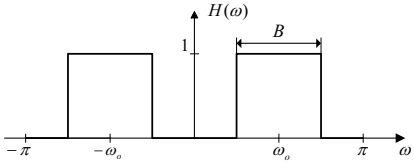


Figure 3.8: Band-pass filter

Ideal band-stop:

$$H_{BS}(\omega) = \begin{cases} 0 & \text{if } |\omega - \omega_o| < B/2 \\ 1 & \text{else in } [-\pi, \pi] \end{cases}$$

$$h_{BS}[n] = \delta[n] - h_{BP}[n] \qquad (3.55)$$

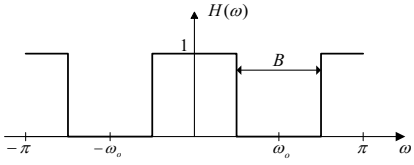


Figure 3.9: Stop-pass filter

Remarks:

- These filters are not realizable:
  - they are non-causal ($h[n] \neq 0$ for $n < 0$)
  - they are unstable ($\sum_n |h[n]| = \infty$)

- The main object of filter design is to derive practical approximations to such filters (more on this later...)

3.7 Phase delay and group delay

Introduction:

Consider an integer delay system, for which the input-output relationship is

$$y[n] = x[n-k], \qquad k \in \mathbb{Z}. \tag{3.56}$$

The corresponding frequency response is computed as

$$Y(\omega) = e^{-j\omega k}X(\omega) \Rightarrow H(\omega) = \frac{Y(\omega)}{X(\omega)} = e^{-j\omega k}. \tag{3.57}$$

This clearly shows that the phase of the system $\angle H(\omega) = -\omega k$ provides information about the delay incurred by the input.

Phase delay:

For arbitrary $H(\omega)$, we define the phase delay as

$$\tau_{ph} \triangleq -\frac{\angle H(\omega)}{\omega} \tag{3.58}$$

For the integer delay system, we get:

$$-\frac{\angle H(\omega)}{\omega} = \frac{\omega k}{\omega} = k \tag{3.59}$$

The concept of phase delay is useful mostly for wideband signals (i.e. occupying the whole band from $-\pi$ to π).

Group delay:

Definition:

$$\tau_{gr} \triangleq -\frac{d\angle H(\omega)}{d\omega} \tag{3.60}$$

This concept is useful when the system input $x[n]$ is a narrowband signal centred around ω_0, i.e.:

$$x[n] = s[n]e^{j\omega_0 n} \tag{3.61}$$

where $s[n]$ is a slowly-varying envelope. An example of such a narrowband signal is given in Figure 3.7.

The corresponding system output is then

$$y[n] \approx |H(\omega_0)|s[n-\tau_{gr}(\omega_0)]e^{j\omega_0[n-\tau_{ph}(\omega_0)]}. \tag{3.62}$$

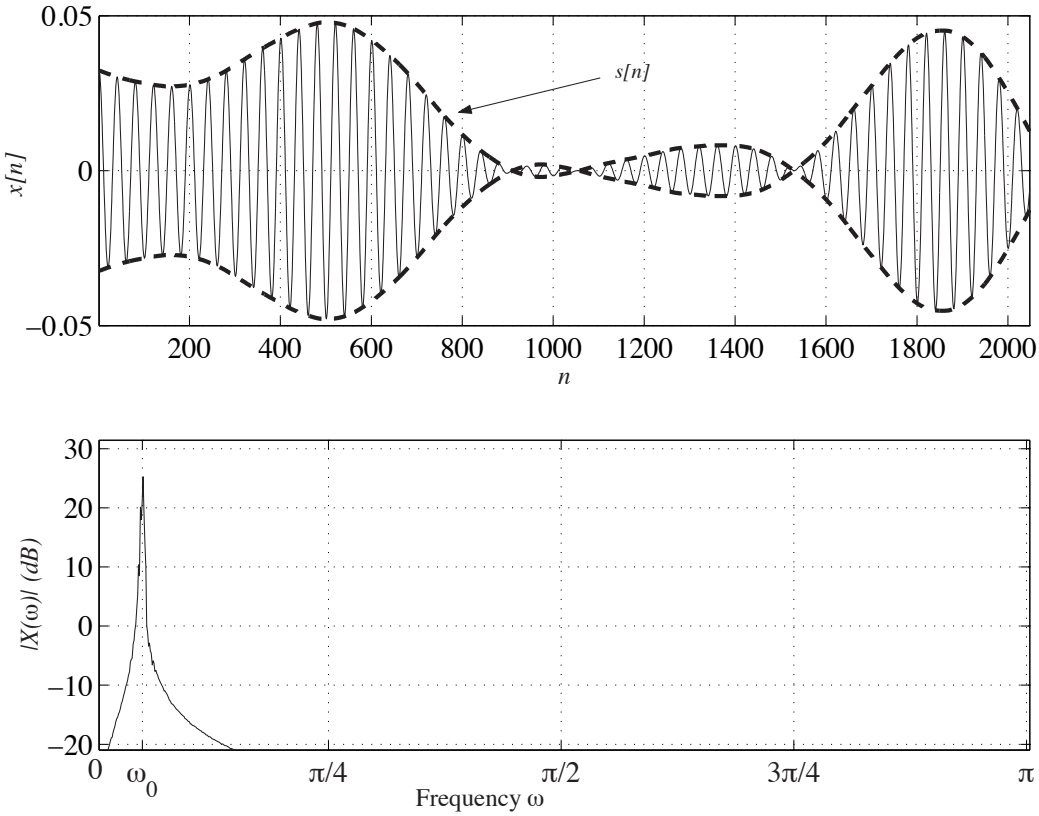


Fig. 3.10 An example of a narrowband signal as defined by $x[n] = s[n]\cos(\omega_0 n)$. Top: the signal $x[n]$ is shown, together with the dashed envelope $s[n]$ (Samples are connected to ease viewing). Bottom: the magnitude of the DTFT of $x[n]$.

The above equation shows that the phase delay $\tau_{ph}(\omega_0)$ contributes a phase change to the *carrier* $e^{j\omega_0 n}$, whereas the group delay $\tau_{gr}(\omega_0)$ contributes a delay to the envelope $s[n]$. Notice that this equation is strictly valid only for integer values of $\tau_{gr}(\omega_0)$, though a non-integer value of $\tau_{gr}(\omega_0)$ can still be interpreted as a non-integer delay[1].

Pure delay system:

- Generalization of the integer delay system.
- Defined directly in terms of its frequency response:

$$H(\omega) = e^{-j\omega\tau} \tag{3.63}$$

  where $\tau \in \mathbb{R}$ is not limited to take integer value.

- Clearly:

$$|H(\omega)| = 1 \Rightarrow \text{no magnitude distortion} \tag{3.64}$$

$$\tau_{ph}(\omega) = \tau_{gr}(\omega) = \tau \tag{3.65}$$

Linear phase:

Consider an LTI system with

$$H(\omega) = |H(\omega)|e^{j\angle H(\omega)}. \tag{3.66}$$

If $\angle H(\omega) = -\omega\tau$, we say that the system has linear phase. A more general definition of linear phase will be given later, but it is readily seen that a linear phase system does not introduce phase distortion, since its effect on the phase of the input amounts to a delay by τ samples.

One can define a distortionless system as one having a gain and a linear phase, i.e. $H(\omega) = Aexp(j\omega\tau)$. The gain is easily compensated for and the linear phase merely delays the signal by τ samples.

3.8 Problems

Problem 3.1: Ideal High-Pass Filter
Let $h[n]$ be the impulse reponse of an ideal high-pass filter, i.e. $H(\omega)$ is defined as:

$$H(\omega) = \begin{cases} 0 & |\omega| < \omega_c \\ 1 & \text{elsewhere} \end{cases},$$

for the main period $-\pi \leq \omega \leq \pi$, and for somr given ω_c, called the cut-off frequency..

1. plot $H(\omega)$.
2. compute $h[n]$ by using the definition of the inverse DTFT.

[1] For instance, a delay of $\frac{1}{2}$ sample amount to replacing $x[n]$ by a value interpolated between $x[n]$ and $x[n-1]$.

3. Given that $H(\omega)$ can be expressed as $H(\omega) = 1 - H_{LP}(\omega)$, where $H_LP(\omega)$ is the freqquency response of an ideal low-pass filter with cut-off frequency ω_c, compute $h[n]$ as a function of $h_{LP}[n]$.

4. What would be the frequency response $G(\omega)$ if $g[n]$ was $h[n]$ delayed by 3 samples ?

5. Is the flter $h[n]$ stable, causal, FIR, IIR ?

Problem 3.2: Windowing
Let $x[n] = \cos(\omega_0 n)$ for some discrete frequency ω_0, such that $0 \leq \omega_0 \leq \pi$. Give an expression of $X(\omega)$.

Let $w[n] = u[n] - u[n-M]$. Plot the sequence $w[n]$. Compute $W(\omega)$ by using the definition of the DTFT.

Let now $y[n] = x[n]w[n]$. Compute the frequency response $Y(\omega)$.

Problem 3.3:
Given a DTFT pair

$$x[n] \stackrel{\mathcal{F}}{\longleftrightarrow} X(\omega),$$

1. Compute the DTFT of $(-1)^n x[n]$;

2. Compute the DTFT of $(-1)^n x^*[M-n]$ for some $M \in \mathbb{Z}$;

3. What is the equivalent in the time -domain of the following frequency domain condition

$$|X(\omega)|^2 + |X(\pi - \omega)|^2 = 1?$$

[Hint: Recall that $|a|^2 = a^* a$.]

Chapter 4

The z-transform (ZT)

Motivation:

- While very useful, the DTFT has a limited range of applicability.
- For example, the DTFT of a simple signal like $x[n] = 2^n u[n]$ does not exist.
- One may view the ZT as a generalization of the DTFT that is applicable to a larger class of signals.
- The ZT is the discrete-time equivalent of the Laplace transform for continuous-time signals.

4.1 The ZT

Definition:

The ZT is a transformation that maps DT signal $x[n]$ into a function of the complex variable z, defined as

$$\boxed{X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}} \tag{4.1}$$

The domain of $X(z)$ is the set of all $z \in \mathbb{C}$ such that the series converges absolutely, that is:

$$\text{Dom}(X) = \{z \in \mathbb{C} : \sum_n |x[n]z|^{-n} < \infty\} \tag{4.2}$$

Remarks:

- The domain of $X(z)$ is called the region of convergence (ROC).
- The ROC only depends on $|z|$: if $z \in$ ROC, so is $ze^{j\phi}$ for any angle ϕ.
- Within the ROC, $X(z)$ is an analytic function of complex variable z. (That is, $X(z)$ is smooth, derivative exists, etc.)
- Both $X(z)$ and the ROC are needed when specifying a ZT.

Example 4.1:

▶ Consider $x[n] = 2^n u[n]$. We have

$$X(z) = \sum_{n=0}^{\infty} 2^n z^{-n} = \frac{1}{1-2z^{-1}}$$

where the series converges provided $|2z^{-1}| < 1$. Accordingly, ROC : $|z| > 2$ ◀

Connection with DTFT:

- The ZT is more general than the DTFT. Let $z = re^{j\omega}$, so that

$$\text{ZT}\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n] r^{-n} e^{-j\omega n} = \text{DTFT}\{x[n]r^{-n}\} \tag{4.3}$$

  With ZT, possibility of adjusting r so that series converges.

- Consider previous example:
  - $x[n] = 2^n u[n]$ does not have a DTFT (Note: $\sum_n |x[n]| = \infty$)
  - $x[n]$ has a ZT for $|z| = r > 2$

- If $z = e^{j\omega} \in$ ROC,

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n} = \text{DTFT}\{x[n]\} \tag{4.4}$$

- In the sequel, the DTFT is either denoted by $X(e^{j\omega})$, or simply by $X(\omega)$ when there is no ambiguity (as we did in Chapter 3).

Inverse ZT:

Let $X(z)$, with associated ROC denoted $\mathcal{R}_x$, be the ZT of DT signal $x[n]$. Then

$$\boxed{x[n] = \frac{1}{2\pi j} \oint_C X(z) z^{n-1} dz, \quad n \in \mathbb{Z}} \tag{4.5}$$

where C is any closed, simple contour around $z = 0$ within $\mathcal{R}_x$.

Remarks:

- In analogy with the inverse DTFT, signals z^n, with relative weight $X(z)$.
- In practice, we do not use (4.5) explicitly to compute the inverse ZT (more on this later); the use of (4.5) is limited mostly to theoretical considerations (e.g. next item).
- Since DT signal $x[n]$ can be recovered uniquely from its ZT $X(z)$ (and associated ROC $\mathcal{R}_x$), we say that $x[n]$ together with $X(z)$ and $\mathcal{R}_x$ form a ZT pair, and write:

$$x[n] \overset{Z}{\longleftrightarrow} X(z), z \in \mathcal{R}_x \tag{4.6}$$

4.2 Study of the ROC and ZT examples

Signal with finite duration:

- Suppose there exist integers $N_1 \leq N_2$ such that $x[n] = 0$ for $n < N_1$ and for $n > N_2$. Then, we have

$$\begin{aligned} X(z) &= \sum_{n=N_1}^{N_2} x[n] z^{-n} \\ &= x[N_1] z^{-N_1} + x[N_1+1] z^{-N_1-1} + \cdots + x[N_2] z^{-N_2} \end{aligned} \tag{4.7}$$

- ZT exists for all $z \in \mathbb{C}$, except possibly at $z = 0$ and $z = \infty$:
  - $N_2 > 0 \Rightarrow z = 0 \notin$ ROC
  - $N_1 < 0 \Rightarrow z = \infty \notin$ ROC

Example 4.2:

▶ Consider the unit pulse sequence $x[n] = \delta[n]$.

$$\begin{aligned} X(z) &= 1 \times z^0 = 1 \\ \text{ROC} &= \mathbb{C} \end{aligned} \tag{4.8}$$

◀

Example 4.3:

▶ Consider the finite length sequence $x[n] = \{1, \underline{1}, 2, 1\}$.

$$\begin{aligned} X(z) &= z + 1 + 2z^{-1} + z^{-2} \\ \text{ROC} &: 0 < |z| < \infty \end{aligned} \tag{4.9}$$

◀

Theorem:

To any power series $\sum_{n=0}^{\infty} c_n w^n$, we can associate a *radius of convergence*

$$R = \lim_{n \to \infty} \left| \frac{c_n}{c_{n+1}} \right| \tag{4.10}$$

such that

$$\begin{aligned} \text{if } |w| < R &\Rightarrow \text{the series converges absolutely} \\ \text{if } |w| > R &\Rightarrow \text{the series diverges} \end{aligned}$$

Causal signal:

Suppose $x[n] = 0$ for $n < 0$. We have

$$X(z) = \sum_{n=0}^{\infty} x[n]z^{-n} = \sum_{n=0}^{\infty} c_n w^n$$
$$c_n \equiv x[n], \quad w \equiv z^{-1} \tag{4.11}$$

Therefore, the ROC is given by

$$|w| < R_w = \lim_{n\to\infty} \left| \frac{x[n]}{x[n+1]} \right|$$
$$|z| > \frac{1}{R_w} \equiv r \tag{4.12}$$

The ROC is the exterior of a circle of radius r, as shown in Figure 4.1.

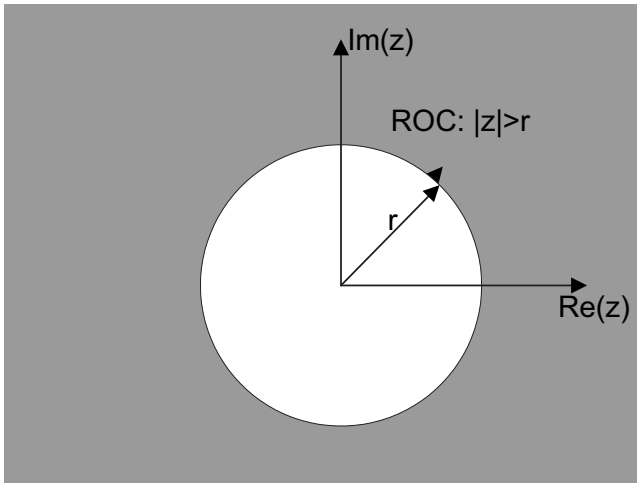


Fig. 4.1 Illustration of the ROC for a causal signal

Example 4.4:

- Consider the causal sequence

$$x[n] = a^n u[n] \tag{4.13}$$

where a is an arbitrary complex number. We have

$$X(z) = \sum_{n=0}^{\infty} a^n z^{-n}$$
$$= \sum_{n=0}^{\infty} (az^{-1})^n = \frac{1}{1-az^{-1}} \tag{4.14}$$

provided $|az^{-1}| < 1$, or equivalently, $|z| > |a|$. Thus,

$$\text{ROC} : |z| > |a| \tag{4.15}$$

Consistent with Figure 4.1, this ROC corresponds to the exterior of a circle of radius $r = |a|$ in the z-plane. ◀

Anti-causal signal:

Suppose $x[n] = 0$ for $n > 0$. We have

$$X(z) = \sum_{n=-\infty}^{0} x[n]z^{-n} = \sum_{n=0}^{\infty} x[-n]z^{n} \tag{4.16}$$

Therefore, the ROC is given by

$$|z| < R = \lim_{n\to\infty} \left| \frac{x[-n]}{x[-n-1]} \right| \tag{4.17}$$

The ROC is the interior of a circle of radius R in the z-plane, as shown in Figure 4.2.

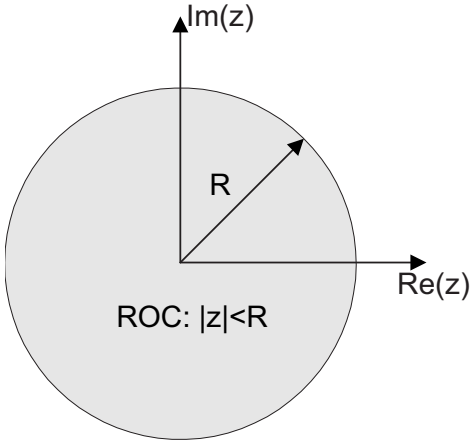


Fig. 4.2 Illustration of the ROC for an anti-causal signal.

Example 4.5:

▶ Consider the anti-causal sequence

$$x[n] = -a^{n}u[-n-1]. \tag{4.18}$$

We have

$$\begin{aligned} X(z) &= -\sum_{n=-\infty}^{-1} a^n z^{-n} \\ &= -\sum_{n=1}^{\infty} (a^{-1}z)^n \\ &= -\frac{a^{-1}z}{1-a^{-1}z} \quad \text{if } |a^{-1}z| < 1 \\ &= \frac{1}{1-az^{-1}} \end{aligned} \tag{4.19}$$

provided $|a^{-1}z| < 1$, or equivalently, $|z| < |a|$. Thus,

$$\text{ROC} : |z| < |a| \tag{4.20}$$

Consistent with Figure 4.2, the ROC is the interior of a circle of radius $R = |a|$. Note that in this and the previous example, we obtain the same mathematical expression for $X(z)$, but the ROCs are different. ◄

Arbitrary signal:

We can always decompose the series $X(z)$ as

$$X(z) = \underbrace{\sum_{n=-\infty}^{-1} x[n]z^{-n}}_{\text{needs } |z|>r} + \underbrace{\sum_{n=0}^{\infty} x[n]z^{-n}}_{\text{needs } |z|<R} \tag{4.21}$$

We distinguish two cases:

- If $r < R$, the ZT exists and ROC : $r < |z| < R$ (see Figure 4.3).

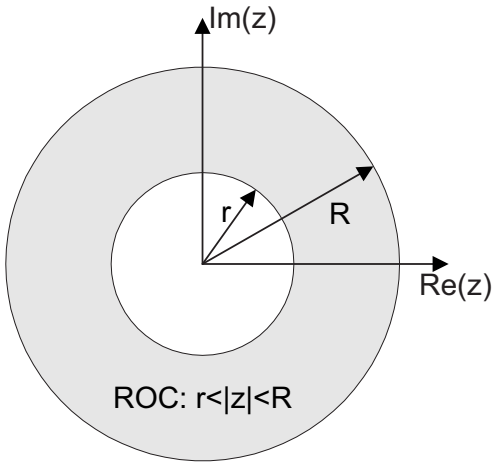


Fig. 4.3 General annular ROC.

- If $r > R$, the ZT does not exist.

Example 4.6:

▶ Consider

$$x[n] = (1/2)^n u[n] - 2^n u[-n-1]. \tag{4.22}$$

Here, we have

$$\begin{aligned} X(z) &= \underbrace{\sum_{n=0}^{\infty} (1/2)^n z^{-n}}_{\Downarrow |z|>1/2} - \underbrace{\sum_{n=-\infty}^{-1} 2^n z^{-n}}_{\Downarrow |z|<2} \\ &= \frac{1}{1-\frac{1}{2}z^{-1}} + \frac{1}{1-2z^{-1}} \\ &= \frac{2-\frac{5}{2}z^{-1}}{1-\frac{5}{2}z^{-1}+z^{-2}} \end{aligned} \tag{4.23}$$

The two series will converge simultaneously iff

$$\text{ROC} : 1/2 < |z| < 2. \tag{4.24}$$

This correspond to the situation shown in Figure 4.3 with $r = 1/2$ and $R = 2$. ◀

Example 4.7:

▶ Consider

$$x[n] = 2^n u[n] - (1/2)^n u[-n-1]. \tag{4.25}$$

Here, $X(z) = X_+(z) + X_-(z)$ where

$$\begin{aligned} X_+(z) &= \sum_{n=0}^{\infty} 2^n z^{-n} \text{ converges for } |z| > 2 \\ X_-(z) &= \sum_{n=-\infty}^{-1} (1/2)^n z^{-n} \text{ converges for } |z| < 1/2 \end{aligned}$$

Since these two regions do not intersect, the ROC is empty and the ZT does not exist. ◀

4.3 Properties of the ZT

Introductory remarks:

- Notations for ZT pairs:

$$\begin{aligned} x[n] &\overset{Z}{\longleftrightarrow} X(z), \quad z \in \mathcal{R}_x \\ y[n] &\overset{Z}{\longleftrightarrow} Y(z), \quad z \in \mathcal{R}_y \end{aligned}$$

  $\mathcal{R}_x$ and $\mathcal{R}_y$ respectively denote the ROC of $X(z)$ and $Y(z)$

- When stating a property, we must also specify the corresponding ROC.
- In some cases, the true ROC may be larger than the one indicated.

Basic symmetries:

$$x[-n] \overset{Z}{\longleftrightarrow} X(z^{-1}), \quad z^{-1} \in \mathcal{R}_x \tag{4.26}$$

$$x^*[n] \overset{Z}{\longleftrightarrow} X^*(z^*), \quad z \in \mathcal{R}_x \tag{4.27}$$

Linearity:

$$ax[n] + by[n] \overset{Z}{\longleftrightarrow} aX(Z) + bY(z), \quad z \in \mathcal{R}_x \cap \mathcal{R}_y \tag{4.28}$$

Time shift (very important):

$$x[n-d] \overset{Z}{\longleftrightarrow} z^{-d}X(z), \quad z \in \mathcal{R}_x \tag{4.29}$$

Exponential modulation:

$$a^n x[n] \overset{Z}{\longleftrightarrow} X(z/a), \quad z/a \in \mathcal{R}_x \tag{4.30}$$

Differentiation:

$$nx[n] \overset{Z}{\longleftrightarrow} -z\frac{dX(z)}{dz}, \quad z \in \mathcal{R}_x \tag{4.31}$$

Convolution:

$$x[n] * y[n] \overset{Z}{\longleftrightarrow} X(z)Y(z), \quad z \in \mathcal{R}_x \cap \mathcal{R}_y \tag{4.32}$$

Initial value:

For $x[n]$ causal (i.e. $x[n] = 0$ for $n < 0$), we have

$$\lim_{z\to\infty} X(z) = x[0] \tag{4.33}$$

Example 4.8:

▶ Consider

$$\begin{aligned} x[n] &= \cos(\omega_o n)u[n] \\ &= \frac{1}{2}e^{j\omega_o n}u[n] + \frac{1}{2}e^{-j\omega_o n}u[n] \end{aligned}$$

We have

$$\begin{aligned} X(z) &= \frac{1}{2}ZT\{e^{j\omega_o n}u[n]\} + \frac{1}{2}ZT\{e^{-j\omega_o n}u[n]\} \\ &= \frac{1}{2}\underbrace{\frac{1}{1-e^{j\omega_o}z^{-1}}}_{|z|>|e^{j\omega_o}|=1} + \frac{1}{2}\underbrace{\frac{1}{1-e^{-j\omega_o}z^{-1}}}_{|z|>|e^{-j\omega_o}|=1} \\ &= \frac{1-z^{-1}\cos\omega_o}{1-2z^{-1}\cos\omega_o+z^{-2}}, \quad \text{ROC}: |z|>1 \end{aligned}$$

◀

Example 4.9:

▶ Consider

$$x[n] = na^n u[n]$$

We have

$$\begin{aligned} X(z) &= -z\frac{d}{dz}\left\{\frac{1}{1-az^{-1}}\right\}, \quad |z|>|a| \\ &= \frac{az^{-1}}{(1-az^{-1})^2}, \quad \text{ROC}: |z|>|a| \end{aligned}$$

◀

Example 4.10:

▶ Consider the signals $x_1[n] = \{\underline{1}, -2a, a^2\}$ and $x_2[n] = \{\underline{1}, a, a^2, a^3, a^4\}$ where $a \in \mathbb{C}$. Let us compute the convolution $y = x_1 * x_2$ using the z-transform:

$$X_1(z) = 1 - 2az^{-1} + a^2z^{-2} = (1-az^{-1})^2$$

$$X_2(z) = 1 + az^{-1} + a^2z^{-2} + a^3z^{-3} + a^4z^{-4} = \frac{1-a^5z^{-5}}{1-az^{-1}}$$

$$\begin{aligned} Y(z) &= X_1(z)X_2(z) = (1-az^{-1})(1-a^5z^{-5}) \\ &= 1 - az^{-1} - a^5z^{-5} + a^6z^{-6} \end{aligned}$$

Therefore, $y[n] = \{\underline{1}, -a, 0, 0, 0, -a^5, a^6\}$. ◀

4.4 Rational ZTs

Rational function:

$X(z)$ is a rational function in z (or z^{-1}) if

$$X(z) = \frac{N(z)}{D(z)} \tag{4.34}$$

where $N(z)$ and $D(z)$ are polynomials in z (resp. z^{-1})

Remarks:

- Rational ZT plays a central role in DSP
- Essential for the realization of practical IIR filters.
- In this and the next Sections, we investigate two important issues related to rational ZT:
  - Pole-zero (PZ) characterization
  - Inversion via partial fraction expansion

Poles and zeros:

- $X(z)$ has a pole of order L at $z = p_o$ if

$$X(z) = \frac{\psi(z)}{(z-p_o)^L}, \quad 0 < |\psi(p_o)| < \infty \tag{4.35}$$

- $X(z)$ has a zero of order L at $z = z_o$ if

$$X(z) = (z-z_o)^L \psi(z), \quad 0 < |\psi(z_o)| < \infty \tag{4.36}$$

- We sometimes refer to the order L as the multiplicity of the pole/zero.

Poles and zeros at ∞:

- $X(z)$ has a pole of order L at $z = \infty$ if

$$X(z) = z^L \psi(z), \quad 0 < |\psi(\infty)| < \infty \tag{4.37}$$

- $X(z)$ has a zero of order L at $z = \infty$ if

$$X(z) = \frac{\psi(z)}{z^L}, \quad 0 < |\psi(\infty)| < \infty \tag{4.38}$$

Poles & zeros of a rational $X(z)$:

Consider rational function $X(z) = N(z)/D(z)$:

- Roots of $N(z) \Rightarrow$ zeros of $X(z)$
  Roots of $D(z) \Rightarrow$ poles of $X(z)$
- Must take into account pole-zero cancellation:
  common roots of $N(z)$ and $D(Z)$ do not count as zeros and poles.
- Repeated roots in $N(z)$ (or $D(z)$) lead to multiple zeros (respectively poles).
- If we include poles and zeros at 0 and ∞:

$$\boxed{\text{number of poles} = \text{number of zeros}} \tag{4.39}$$

Example 4.11:

► (1) Consider the rational function

$$X(z) = \frac{z^{-1}}{1-2z^{-1}+z^{-2}} = \frac{z}{z^2-2z+1} = \frac{z}{(z-1)^2}$$

The poles and zeros of $X(z)$ along with their order are as follows:

$$\begin{array}{rcl} \text{poles} & : & p_1 = 1, \quad L = 2 \\ \text{zeros} & : & z_1 = 0, \quad L = 1 \\ & & z_2 = \infty, \quad L = 1 \end{array}$$

(2) Let

$$X(z) = \frac{1-z^{-4}}{1+3z^{-1}} = \frac{z^4-1}{z^3(z+3)}$$

The corresponding poles and zeros are

$$\begin{array}{rcl} \text{zeros} & : & z_k = e^{j\pi k/2} \text{ for } k = 0,1,2,3, \quad L = 1 \\ \text{poles} & : & p_1 = 0, \quad L = 3 \text{ (triple pole)} \\ & & p_2 = -3, \quad L = 1 \end{array}$$

(3) As a final example, consider

$$X(z) = z - 1$$

The poles and zeros are

$$\begin{array}{rcl} \text{zeros} & : & z_1 = 1, \quad L = 1 \\ \text{poles} & : & p_1 = \infty, \quad L = 1 \end{array}$$

◄

Pole-zero (PZ) diagram:

For rational functions $X(z) = N(z)/D(Z)$, knowledge of the poles and zeros (along with their order) completely specify $X(z)$, up to a scaling factor, say $G \in \mathbb{C}$.

Example 4.12:

► Let us consider the following pole and zero values:

$$\left.\begin{array}{ll} z_1 = 1 & L = 1 \\ p_1 = 2 & L = 1 \end{array}\right\} \Longrightarrow X(z) = G\frac{z-1}{z-2} = G\frac{1-z^{-1}}{1-2z^{-1}} \tag{4.40}$$

◄

Remarks:

- Thus, we may represent $X(z)$ by a so-called PZ diagram, as illustrated in Figure 4.4.

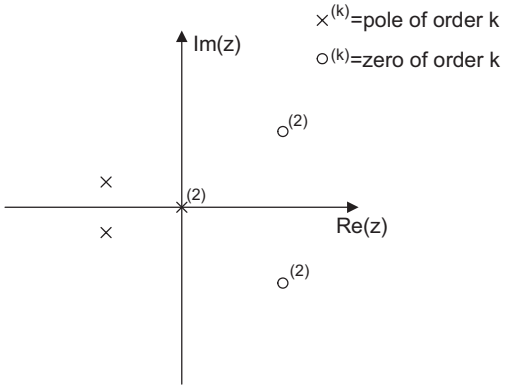


Fig. 4.4 Example of a PZ diagram.

- For completeness, the presence of poles or zeros at ∞ should be mentioned on the diagram.
- Note that it is also useful to indicate ROC on the pz-diagram.

Example 4.13:

► Consider $x[n] = a^n u[n]$, where $a > 0$. The corresponding ZT is

$$X(z) = \frac{1}{1-az^{-1}} = \frac{z}{z-a}, \quad \text{ROC} : |z| > a \tag{4.41}$$

$$\begin{array}{lll} z_1 = 0 & , & L = 1 \\ p_1 = a & , & L = 1 \end{array}$$

The PZ diagram is shown in Figure 4.5. ◄

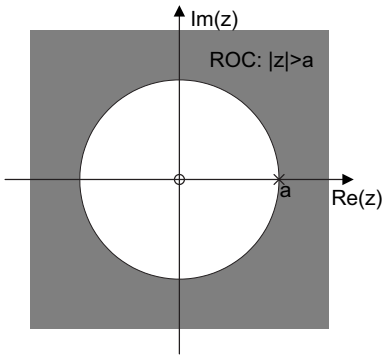


Fig. 4.5 PZ diagram for the signal $x[n] = a^n u[n]$.

ROC for rational ZT:

Let's summarize a few facts about ROC for rational ZTs:

- ROC does not contain poles (because $X(z)$ does not converge at a pole).
- ROC can always be extended to nearest pole
- ROC delimited by poles $\Rightarrow$ annular region between poles
- If we are given only $X(z)$, then several possible ROC:
  - any annular region between two poles of increasing magnitude.
  - accordingly, several possible DT signals $x[n]$

4.5 Inverse ZT

Introduction:

- Several methods do exist for the evaluation of $x[n]$ given its ZT $X(z)$ and corresponding ROC:
  - Contour integration via residue theorem
  - Partial fraction expansion (PFE)
  - Long division
- Partial fraction is by far the most useful technique in the context of rational ZTs
- In this section:
  - we present the PFE method in detail;
  - we discuss division only as a useful tool to put $X(z)$ in proper form for the PFE.

4.5.1 Inversion via PFE

PFE:

- Suppose that $X(z) = \frac{N(z)}{D(z)}$ where
  - $N(z)$ and $D(z)$ are polynomials in z^{-1}
  - degree of $D(z) >$ degree of $N(z)$
- Under these conditions, $X(z)$ may be expressed as

$$X(z) = \sum_{k=1}^{K} \sum_{l=1}^{L_k} \frac{A_{kl}}{(1-p_k z^{-1})^l} \tag{4.42}$$

  where
  - $p_1, \ldots, p_K$ are the distinct poles of $X(z)$
  - $L_1, \ldots, L_K$ are the corresponding orders
- The constants A_{kl} can be computed as follows:
  - simple poles ($L_k = 1$):

$$A_{kl} \equiv A_{k1} = (1-p_k z^{-1}) X(z)|_{z=p_k} \tag{4.43}$$

  - multiple poles ($L_k > 1$):

$$A_{kl} = \frac{1}{(L_k-l)!(-p_k)^{L_k-l}} \Big\{ \frac{d^{L_k-l}[(1-p_k z^{-1})^{L_k} X(z)]}{(dz^{-1})^{L_k-l}} \Big\}\Big|_{z=p_k} \tag{4.44}$$

Inversion method:

Given $X(z)$ as above with ROC : $r < |z| < R$, the corresponding DT signal $x[n]$ may be obtained as follows:

- Determine the PFE of $X(z)$:

$$X(z) = \sum_{k=1}^{K} \sum_{l=1}^{L_k} \frac{A_{kl}}{(1-p_k z^{-1})^l} \tag{4.45}$$

- Invoking linearity of the ZT, express $x[n]$ as

$$x[n] = \sum_{k=1}^{K} \sum_{l=1}^{L_k} A_{kl} \mathcal{Z}^{-1} \Big\{ \frac{1}{(1-p_k z^{-1})^l} \Big\} \tag{4.46}$$

  where $\mathcal{Z}^{-1}$ denotes the inverse z-transform.
- Evaluate the elementary inverse ZTs in (4.46):
  - simple poles ($L_k = 1$):

$$\frac{1}{1-p_k z^{-1}} \xrightarrow{\mathcal{Z}^{-1}} \begin{cases} p_k^n u[n] & \text{if } |p_k| \le r \\ -p_k^n u[-n-1] & \text{if } |p_k| \ge R \end{cases} \tag{4.47}$$

- higher order poles ($L_k > 1$):

$$\frac{1}{(1-p_k z^{-1})^l} \xrightarrow{\mathcal{Z}^{-1}} \begin{cases} \binom{n+l-1}{l-1} p_k^n u[n] & \text{if } |p_k| \le r \\ -\binom{n+l-1}{l-1} p_k^n u[-n-1] & \text{if } |p_k| \ge R \end{cases} \tag{4.48}$$

where $\binom{n}{r} = \frac{n!}{r!(n-r)!}$ (read n choose r)

Example 4.14:

▶ Consider

$$X(z) = \frac{1}{(1-az^{-1})(1-bz^{-1})}, \quad |a| < |z| < |b|$$

The PFE of $X(z)$ can be rewritten as:

$$X(z) = \frac{A_1}{1-az^{-1}} + \frac{A_2}{1-bz^{-1}},$$

with

$$\begin{aligned} A_1 &= (1-az^{-1})X(z)\big|_{z=a} = \left.\frac{1}{1-bz^{-1}}\right|_{z=a} = \frac{a}{a-b} \\ A_2 &= (1-bz^{-1})X(z)\big|_{z=b} = \left.\frac{1}{1-az^{-1}}\right|_{z=b} = \frac{b}{b-a} \end{aligned}$$

so that

$$\begin{aligned} x[n] &= \underbrace{A_1 \mathcal{Z}^{-1}\left\{\frac{1}{1-az^{-1}}\right\}}_{|z|>|a| \Rightarrow \text{causal}} + \underbrace{A_2 \mathcal{Z}^{-1}\left\{\frac{1}{1-bz^{-1}}\right\}}_{|z|<|b| \Rightarrow \text{anti-causal}} \\ &= A_1 a^n u[n] - A_2 b^n u[-n-1] \\ &= \frac{a^{n+1}}{a-b} u[n] - \frac{b^{n+1}}{b-a} u[-n-1] \end{aligned}$$

◀

4.5.2 Putting $X(z)$ in a suitable form

Introduction:

- When applying the above PFE method to $X(z) = N(z)/D(z)$, it is essential that
  - $N(z)$ and $D(z)$ be polynomials in z^{-1}
  - degree of $D(z) >$ degree of $N(z)$
- If either one of the above conditions are not satisfied, further algebraic manipulations must be applied to $X(z)$
- There are two common types of manipulations:
  - polynomial division
  - use of shift property

Polynomial division:

- Find $Q(z)$ and $R(z)$, such that
$$\frac{N(z)}{D(z)} = Q(z) + \frac{R(z)}{D(z)} \tag{4.49}$$
where $Q(z) = \sum_{n=N_1}^{N_2} x[n]z^{-n}$ and $R(z)$ is a polynomial in z^{-1} with degree less than that of $D(z)$
- Determination of $Q(z)$ and $R(z)$ via a division table:
$$\begin{array}{l} D(z) \quad \overline{\left| \begin{array}{r} Q(z) \\ N(z) \\ -Q(z)D(z) \\ \hline R(z) \end{array} \right.} \end{array} \tag{4.50}$$
- If we want the largest power of z in $N(z)$ to decrease, we simply express $D(z)$ and $N(z)$ in decreasing powers of z (e.g. $D(z) = 1 + 2z^{-1} + z^{-2}$)
- If we want the smallest power of z in $N(z)$ to increase we write down $D(z)$ and $N(z)$ in reverse order (e.g. $D(z) = z^{-2} + 2z^{-1} + 1$)

Example 4.15:

▶ Let us consider the z-transform
$$X(z) = \frac{-5 + 3z^{-1} + z^{-2}}{3 + 4z^{-1} + z^{-2}},$$
with the constraint that $x[n]$ is causal. The first step towards finding $x[n]$ is to use long division to make the degree of the numerator smaller than the degree of the denominator. [1]

$$\begin{array}{l} z^{-2} + 4z^{-1} + 3 \quad \overline{\left| \begin{array}{r} 1 \\ z^{-2} + 3z^{-1} - 5 \\ -(z^{-2} + 4z^{-1} + 3) \\ \hline -\ z^{-1} - 8 \end{array} \right.} \end{array} \tag{4.51}$$

so that $X(z)$ rewrites:
$$X(z) = 1 - \frac{z^{-1} + 8}{z^{-2} + 4z^{-1} + 3}.$$
The denominator of the second term has two roots, the poles at $z = -\frac{1}{3}$ and $z = -1$, hence the factorization:
$$X(z) = 1 - \frac{1}{3}\frac{z^{-1} + 8}{(1 + \frac{1}{3}z^{-1})(1 + z^{-1})}.$$
The PFE of the rational term in the above equation is given by two terms:
$$X(z) = 1 - \frac{1}{3}\left(\frac{A_1}{1 + \frac{1}{3}z^{-1}} + \frac{A_2}{1 + z^{-1}}\right),$$

[1] That is, we want the smallest power of z in $N(z)$ to increase from z^{-2} to z^{-1}. Accordingly, we write down $N(z)$ and $D(z)$ in reverse order when performing the division.

with

$$A_1 = \left.\frac{z^{-1}+8}{1+z^{-1}}\right|_{z=-\frac{1}{3}} = -\frac{5}{2} \tag{4.52}$$

$$A_2 = \left.\frac{z^{-1}+8}{1+\frac{1}{3}z^{-1}}\right|_{z=-1} = \frac{21}{2}, \tag{4.53}$$

so that

$$X(z) = 1 + \left(\frac{5}{6(1+\frac{1}{3}z^{-1})} - \frac{7}{2(1+z^{-1})}\right).$$

Now the constraint of causality of $x[n]$ determines the region of convergence of $X(z)$, which is supposed to be delimited by circles with radius $1/3$ and/or 1. Since the sequence is causal, its ROC must extend outwards from the outermost pole, so that the ROC is $|z| > 1$. The sequence $x[n]$ is then given by:

$$x[n] = \delta[n] + \frac{5}{6}\left(-\frac{1}{3}\right)^n u[n] - \frac{7}{2}(-1)^n u[n].$$

◀

Use of shift property:

- In some cases, a simple multiplication by z^k is sufficient to put $X(z)$ into a suitable format, that is:

$$Y(z) = z^k X(z) = \frac{N(z)}{D(z)} \tag{4.54}$$

  where $N(z)$ and $D(z)$ satisfy previous conditions
- The PFE method is then applied to $Y(z)$, yielding a DT signal $y[n]$
- Finally, the shift property is applied to recover $x[n]$:

$$x[n] = y[n-k] \tag{4.55}$$

Example 4.16:

▶ Consider

$$X(z) = \frac{1-z^{-128}}{1-z^{-2}} \quad |z| > 1$$

We could use division to work out this example but this would not be very efficient. A faster approach is to use a combination of linearity and the shift property. First note that

$$X(z) = Y(z) - z^{-128}Y(z)$$

where

$$Y(z) = \frac{1}{1-z^{-2}} = \frac{1}{(1-z^{-1})(1+z^{-1})}$$

The inverse z-transform of $Y(z)$ is easily obtained as (please try it)

$$y[n] = \frac{1}{2}(1+(-1)^n)u[n]$$

Therefore

$$x[n] = y[n] - y[n-128] = \frac{1}{2}(1+(-1)^n)(u[n]-u[n-128])$$

◄

4.6 The one-sided ZT

Definition

$$X^+(z) = \mathcal{Z}^+\{x[n]\} \triangleq \sum_{n=0}^{\infty} x[n]z^{-n} \tag{4.56}$$

- ROC: $|z| > r$, for some $r \geq 0$
- Information about $x[n]$ for $n < 0$ is lost
- Used to solve LCCDE with arbitrary initial conditions

Useful properties:

- Time shift to the right ($k > 0$):

$$x[n-k] \xrightarrow{\mathcal{Z}^+} z^{-k}X^+(z) + x[-k] + x[-k+1]z^{-1} + \cdots + x[-1]z^{-(k-1)} \tag{4.57}$$

- Time shift to the left ($k > 0$):

$$x[n+k] \xrightarrow{\mathcal{Z}^+} z^k X^+(z) - x[0]z^k - x[1]z^{k-1} - \cdots - x[k-1]z \tag{4.58}$$

Example 4.17:

► Consider the LCCDE

$$y[n] = \alpha y[n-1] + x[n], \quad n \geq 0 \tag{4.59}$$

where $x[n] = \beta u[n]$, α and β are real constants with $\alpha \neq 1$ and $y[-1]$ is an arbitrary initial condition. Let us find the solution of this equation, i.e. $y[n]$ for $n \geq 0$, using the unilateral z-transform $\mathcal{Z}^+$.

- Compute $\mathcal{Z}^+$ of $x[n]$:

$$x[n] \text{ causal } \Rightarrow X^+(z) = X(z) = \frac{\beta}{1-z^{-1}}, \quad |z| > 1$$

- Apply $\mathcal{Z}^+$ to both sides of (4.59) and solve for $Y^+(z)$:

$$\begin{aligned} Y^+(z) &= \alpha(z^{-1}Y^+(z)+y[-1])+X^+(z) \\ &= \alpha z^{-1}Y^+(z)+\alpha y[-1]+\frac{\beta}{1-z^{-1}} \\ \Rightarrow (1-\alpha z^{-1})Y^+(z) &= \alpha y[-1]+\frac{\beta}{1-z^{-1}} \\ \Rightarrow Y^+(z) &= \frac{\alpha y[-1]}{1-\alpha z^{-1}}+\frac{\beta}{(1-\alpha z^{-1})(1-z^{-1})} \\ &= \frac{\alpha y[-1]}{1-\alpha z^{-1}}+\frac{A}{1-\alpha z^{-1}}+\frac{B}{1-z^{-1}} \end{aligned}$$

  where

$$A=-\frac{\alpha\beta}{1-\alpha} \qquad B=\frac{\beta}{1-\alpha}$$

- To obtain $y[n]$ for $n\geq 0$, compute the inverse unilateral ZT. This is equivalent to computing the standard inverse ZT under the assumption of a causal solution, i.e. ROC : $|z|>\max(1,|\alpha|)$. Thus, for $n\geq 0$

$$\begin{aligned} y[n] &= \alpha y[-1]\alpha^n u[n]+A\alpha^n u[n]+B(1)^n u[n] \\ &= y[-1]\alpha^{n+1}+\beta\left(\frac{1-\alpha^{n+1}}{1-\alpha}\right), \quad n\geq 0 \end{aligned}$$

◀

4.7 Problems

Problem 4.1: Inverse ZT
Knowing that $h[n]$ is causal, determine $h[n]$ from its z-transform $H(z)$:

$$H(z)=\frac{2+2z^{-1}}{(1+\frac{1}{4}z^{-1})(1-\frac{1}{2}z^{-1})}.$$

Problem 4.2:
Let a stable system $H(z)$ be described by the pole-zero plot in figure 4.6, with the added specification that $H(z)$ is equal to 1 at $z=1$.

1. Using Matlab if necessary, find the impulse response $h[n]$.
2. From the pole-zero plot, is this a low-pass, high-pass, band-pass or band-stop filter ? Check your answer by plotting the magnitude response $|H(\omega)|$.

Problem 4.3:
Let a causal system have the following set of poles and zeros:

- zeros: 0, $\frac{3}{2}e^{\pm j\pi/4}$,$0.8$;

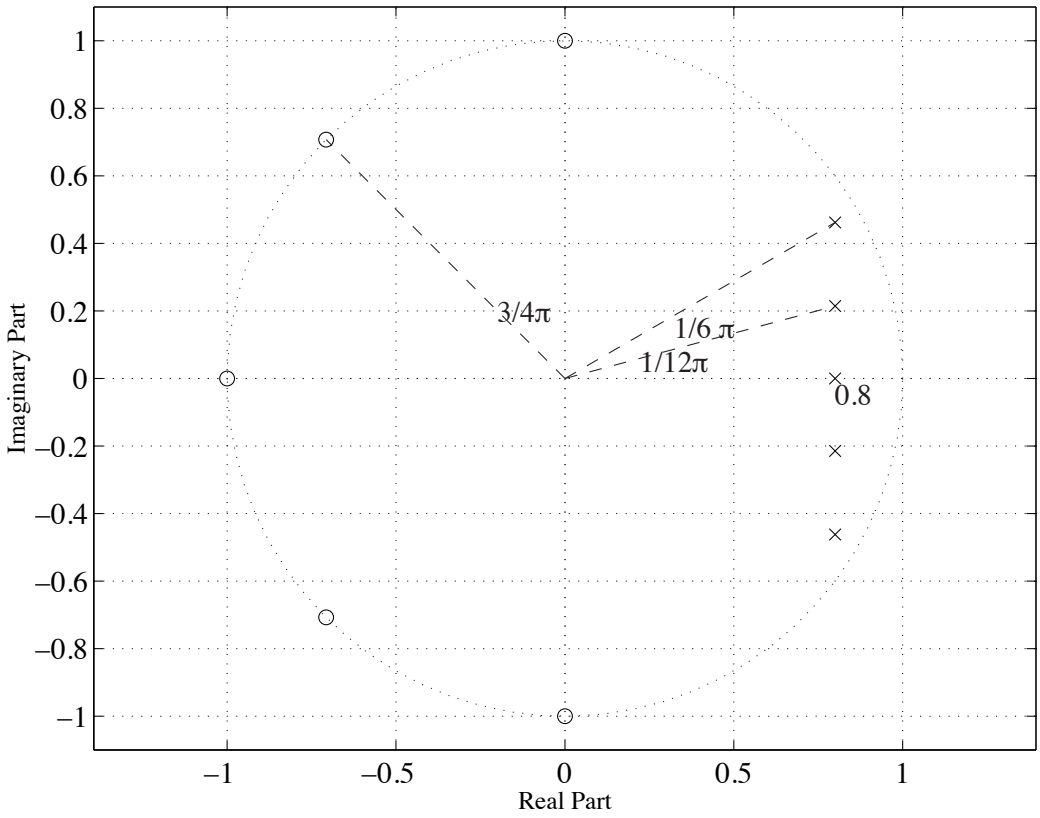


Fig. 4.6 Pole-Zero Plot for problem 4.2.

- poles: $\pm\frac{1}{2}, \frac{1}{2}e^{\pm j\pi/6}$.

It is also known that $H(z)|_{z=1} = 1$.

1. Compute $H(z)$ and determine the Region of Convergence;
2. Compute $h[n]$.

Problem 4.4: Difference Equation
Let a system be sepcified by the following LCCDE:

$$y[n] = x[n] - x[n-1] + \frac{1}{3}y[n-1],$$

with initial rest conditions.

1. What is the corresponding transfer function $H(z)$? Also determine its ROC.
2. Compute the impulse response $h[n]$
   (a) by inverting the$z$-transform $H(z)$;
   (b) from the LCCDE.
3. What is the output of this system if
   (a) $x[n] = \left(\frac{1}{2}\right)^n u[n]$;
   (b) $x[n] = \delta[n-3]$.

Chapter 5

Z-domain analysis of LTI systems

5.1 The system function

LTI system (recap):

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \tag{5.1}$$

$$h[n] = \mathcal{H}\{\delta[n]\} \tag{5.2}$$

Response to arbitrary exponential:

- Let $x[n] = z^n$, $n \in \mathbb{Z}$. The corresponding output signal:

$$\begin{aligned} \mathcal{H}\{z^n\} &= \sum_k h[k]z^{n-k} \\ &= \{\sum_k h[k]z^{-k}\}z^n = H(z)z^n \end{aligned} \tag{5.3}$$

  where we recognize $H(z)$ as the ZT of $h[n]$.

- Eigenvector interpretation:
  - $x[n] = z^n$ behaves as an eigenvector of LTI system $\mathcal{H}$: $\mathcal{H}x = \lambda x$
  - Corresponding eigenvalue $\lambda = H(z)$ provides the system gain

Definition:

Consider LTI system $\mathcal{H}$ with impulse response $h[n]$. The system function of $\mathcal{H}$, denoted $H(z)$, is the ZT of $h[n]$:

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n}, \quad z \in \mathcal{R}_H \tag{5.4}$$

where $\mathcal{R}_H$ denotes the corresponding ROC.

Remarks:

- Specifying the ROC is essential: two different LTI systems may have the same $H(z)$ but different $\mathcal{R}_H$.
- If $H(z)$ and $\mathcal{R}_H$ are known, $h[n]$ can be recovered via inverse ZT.
- If $z = e^{j\omega} \in \mathcal{R}_H$, i.e. the ROC contains the unit circle, then

$$H(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h[n]e^{-j\omega n} \equiv H(\omega) \tag{5.5}$$

  That is, the system function evaluated at $z = e^{j\omega}$ corresponds to the frequency response at angular frequency ω.

Properties:

Let $\mathcal{H}$ be LTI system with system function $H(z)$ and ROC $\mathcal{R}_H$.

- If $y[n]$ denotes the response of $\mathcal{H}$ to arbitrary input $x[n]$, then

$$\boxed{Y(z) = H(z)X(z)} \tag{5.6}$$

- LTI system $\mathcal{H}$ is causal iff $\mathcal{R}_H$ is the exterior of a circle (including ∞).
- LTI system $\mathcal{H}$ is stable iff $\mathcal{R}_H$ contains the unit circle.

Proof:

- Input-output relation:

$$\mathcal{H} \text{ LTI} \Rightarrow y = h * x \Rightarrow Y(z) = H(z)X(z)$$

- Causality:

$$\mathcal{H} \text{ causal } \Leftrightarrow h[n] = 0 \text{ for } n < 0 \Leftrightarrow \mathcal{R}_H : r < |z| \leq \infty$$

- Stability:

$$\mathcal{H} \text{ stable } \Leftrightarrow \sum_n |h[n]| = \sum_n |h[n]e^{-j\omega n}| < \infty \Leftrightarrow e^{j\omega} \in \mathcal{R}_H \quad \square$$

5.2 LTI systems described by LCCDE

LCCDE (recap):

- DT system obeys LCCDE of order N if

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{5.7}$$

  where $a_0 \neq 0$ and $a_N \neq 0$.

- If we further assume initial rest conditions, i.e.:

$$x[n] = 0 \text{ for } n < n_0 \quad \Longrightarrow \quad y[n] = 0 \text{ for } n < n_0 \tag{5.8}$$

  LCCDE corresponds to unique causal LTI system.

System function:

Taking ZT on both sides of (5.7):

$$\begin{aligned} & \sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \\ \Rightarrow \quad & \sum_{k=0}^{N} a_k z^{-k} Y(z) = \sum_{k=0}^{M} b_k z^{-k} X(z) \\ \Rightarrow \quad & H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} \end{aligned} \tag{5.9}$$

Rational system:

More generally we say that LTI system $\mathcal{H}$ is rational iff

$$H(z) = z^{-L} \frac{B(z)}{A(z)} \tag{5.10}$$

where L is an arbitrary integer and

$$A(z) = \sum_{k=0}^{N} a_k z^{-k}, \qquad B(z) = \sum_{k=0}^{M} b_k z^{-k} \tag{5.11}$$

Factored form:

If the roots of $B(z)$ and $A(z)$ are known, one can express $H(z)$ as

$$H(z) = G z^{-L} \frac{\prod_{k=1}^{M} (1 - z_k z^{-1})}{\prod_{k=1}^{N} (1 - p_k z^{-1})} \tag{5.12}$$

where $G =$ system gain ($\in \mathbb{R}$ or $\mathbb{C}$), z_k's are non-trivial zeros (i.e $\neq 0$ or ∞), p_k's are non-trivial poles. Factor z^{-L} takes care of zeros/poles at 0 or ∞.

Properties:

- Rational system (5.12) is causal iff ROC = exterior of a circle:
  - $\Rightarrow$ no poles at $z = \infty \Rightarrow L \geq 0$
  - $\Rightarrow$ ROC: $|z| > \max_k |p_k|$
- Rational system is causal and stable if in addition to above:
  - $\Rightarrow$ unit circle contained in ROC,
  - $\Rightarrow$ that is: $|p_k| < 1$ for all k

Rational system with real coefficients:

- Consider rational system:
$$H(z) = z^{-L}\frac{B(z)}{A(z)} = z^{-L}\frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} \tag{5.13}$$
- In many applications, coefficients a_k's and b_k's $\in \mathbb{R}$. This implies
$$H^*(z) = H(z^*) \tag{5.14}$$
- Thus, if z_k is a zero of $H(z)$, then
$$H(z_k^*) = (H(z_k))^* = 0^* = 0 \tag{5.15}$$
which shows that z_k^* is also a zero of $H(z)$
- More generally, it can be shown that
  - If p_k is a pole of order l of $H(z)$, so is p_k^*
  - If z_k is a zero of order l of $H(z)$, so is z_k^*

  We say that complex poles (or zeros) occur in complex conjugate pairs.
- In the PZ diagram of $H(z)$, the above complex conjugate symmetry translates into a mirror image symmetry of the poles and zeros with respect to the real axis. An example of this is illustrate in Figure 5.1.

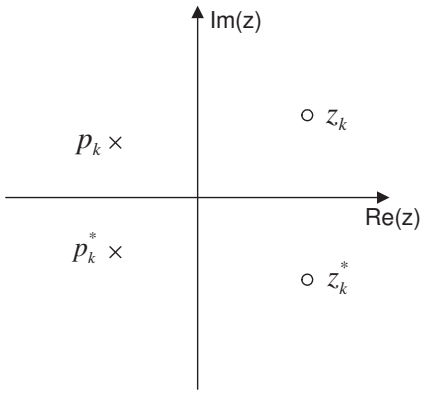


Fig. 5.1 PZ diagram with complex conjugate symmetry.

5.3 Frequency response of rational systems

The formulae:

- Alternative form of $H(z)$ (note $K = L+M-N$):
$$(5.12) \Rightarrow H(z) = Gz^{-K}\frac{\prod_{k=1}^{M}(z-z_k)}{\prod_{k=1}^{N}(z-p_k)} \tag{5.16}$$

- Frequency response:

$$H(\omega) \equiv H(z)|_{z=e^{j\omega}} = Ge^{-j\omega K}\frac{\prod_{k=1}^{M}(e^{j\omega}-z_k)}{\prod_{k=1}^{N}(e^{j\omega}-p_k)} \tag{5.17}$$

- Define:

$$\begin{aligned} V_k(\omega) &= |e^{j\omega}-z_k| \qquad & U_k(\omega) &= |e^{j\omega}-p_k| \\ \theta_k(\omega) &= \angle(e^{j\omega}-z_k) & \phi_k(\omega) &= \angle(e^{j\omega}-p_k) \end{aligned} \tag{5.18}$$

- Magnitude response:

$$|H(\omega)| = |G|\frac{V_1(\omega)\ldots V_M(\omega)}{U_1(\omega)\ldots U_N(\omega)} \tag{5.19}$$

$$|H(\omega)|_{\text{dB}} = |G|_{\text{dB}} + \sum_{k=1}^{M} V_k(\omega)|_{\text{dB}} - \sum_{k=1}^{N} U_k(\omega)|_{\text{dB}} \tag{5.20}$$

- Phase response:

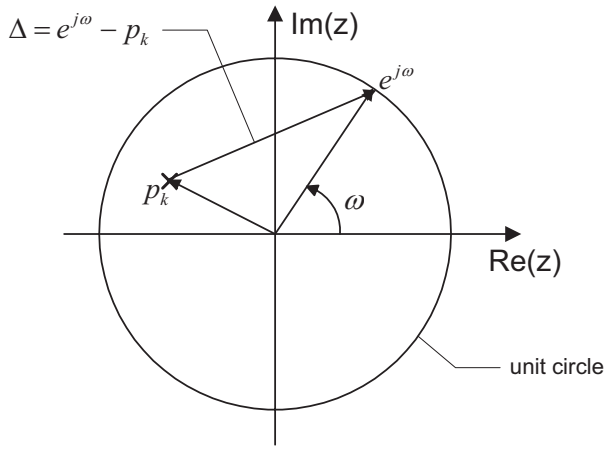
$$\angle H(\omega) = \angle G - \omega K + \sum_{k=1}^{M}\theta_k(\omega) - \sum_{k=1}^{N}\phi_k(\omega) \tag{5.21}$$

- Group delay:

$$\tau_{gr}(\omega) = -\frac{d}{d\omega}\angle H(\omega) = K + \sum_{k=1}^{M}\theta_k'(\omega) - \sum_{k=1}^{N}\phi_k'(\omega) \tag{5.22}$$

Geometrical interpretation:

- Consider pole p_k:



  - $\Delta = e^{j\omega} - p_k$: vector joining p_k to point $e^{j\omega}$ on unit circle
  - $U_k(\omega) = |\Delta|$: length of vector Δ
  - $\phi_k(\omega) = \angle\Delta$: angle between Δ and real axis

- A similar interpretation holds for the terms $V_k(\omega)$ and $\theta_k(\omega)$ associated to the zeros z_k...

Example 5.1:

► Consider the system with transfer function:

$$H(z) = \frac{1}{1-.8z^{-1}} = \frac{z}{z-.8}. \tag{5.23}$$

In this case, the only zero is at $z = 0$, while the only pole is at $z = .8$. So we have that

$$\begin{aligned} V_1(\omega) &= |e^{j\omega}| = 1 \qquad & \theta_1(\omega) &= \omega \\ U_1(\omega) &= |e^{j\omega} - .8| \qquad & \phi_1(\omega) &= \angle(e^{j\omega} - .8) \end{aligned}$$

The magnitude of the frequency response at frequency ω is thus given by the inverse of the distance between .8 and $e^{j\omega}$ in the z-plane, and the phase response is given by $\omega - \phi_1(\omega)$. Figure 5.2 illustrates this construction process. ◄

Some basic principles:

- For stable and causal systems, the poles are located inside the unit circles; the zeros can be anywhere
- Poles near the unit circle at $p = re^{j\omega_o}$ ($r < 1$) give rise to:
  - peak in $|H(\omega)|$ near ω_o
  - rapid phase variation near ω_o
- Zeros near the unit circle at $z = re^{j\omega_o}$ give rise to:
  - deep notch in $|H(\omega)|$ near ω_o
  - rapid phase variation near ω_o

5.4 Analysis of certain basic systems

Introduction

In this section, we study the PZ configuration and the frequency response of basic rational systems. In all cases, we assume that the system coefficients a_ks and b_k are real valued.

5.4.1 First order LTI systems

Description:

The system function is given by:

$$H(z) = G\frac{1-bz^{-1}}{1-az^{-1}} \tag{5.24}$$

The poles and zeros are:

- pole: $z = a$ (simple)
- zero: $z = b$ (simple)

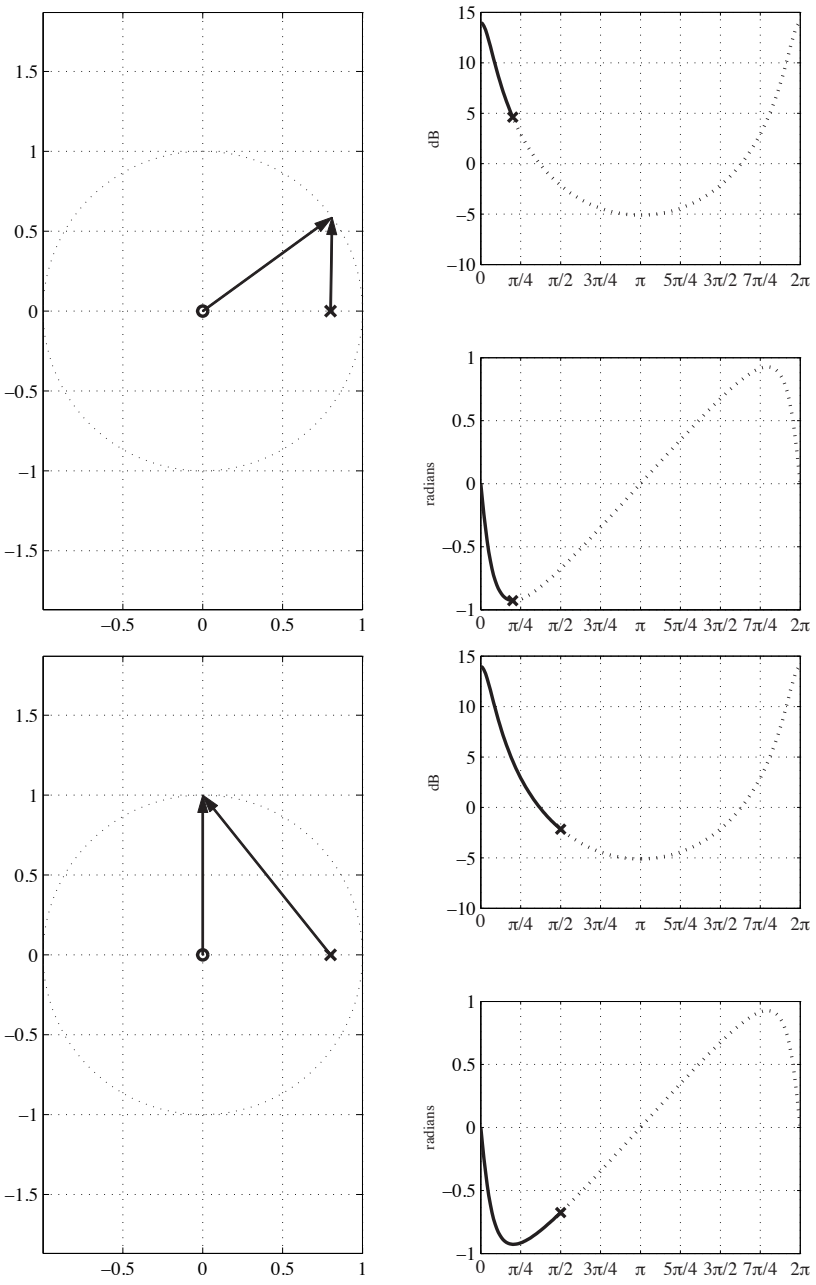


Fig. 5.2 Illustration of the geometric construction of the frequency response of the signal in (5.23), for $\omega = \pi/5$ (top) and $\omega = \pi/2$ (bottom)

Practical requirements:

- causality: ROC : $|z| > |a|$
- stability: $|a| < 1$

Impulse response (ROC: $|z| > |a|$):

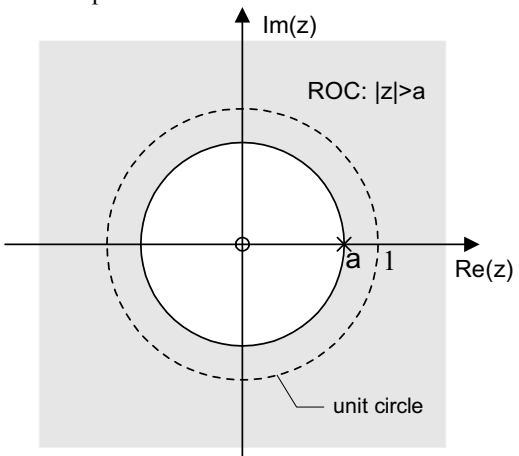
$$h[n] = G(1 - \frac{b}{a})a^n u[n] + G\frac{b}{a}\delta[n] \tag{5.25}$$

Low-pass case:

To get a low-pass behavior, one needs $a = 1 - \varepsilon$, where $0 < \varepsilon \ll 1$ (typically). Additional attenuation of high-frequency is possible by proper placement of the zero $z = b$.

Example 5.2: Low-Pass first order system

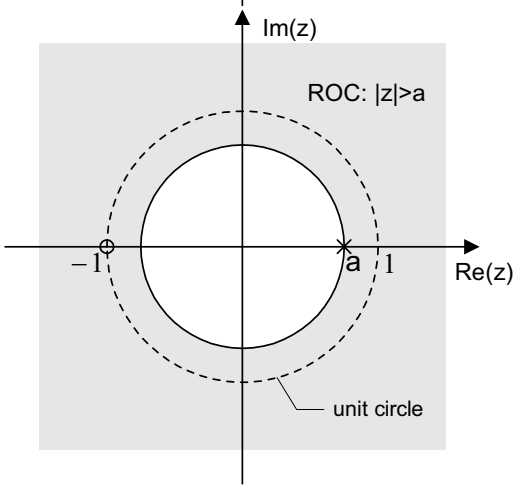
► Two examples of choices for b are shown below.



zero: $b = 0$

$$H_1(z) = G_1 \frac{1}{1 - az^{-1}}$$

$$G_1 = 1 - a \Rightarrow H_1(\omega = 0) = 1$$



zero: $b = -1$

$$H_2(z) = G_2 \frac{1 + z^{-1}}{1 - az^{-1}}$$

$$G_2 = \frac{1 - a}{2} \Rightarrow H_2(\omega = 0) = 1$$

The frequency responses of the corresponding first order low-pass systems are shown in Figure 5.3. ◄

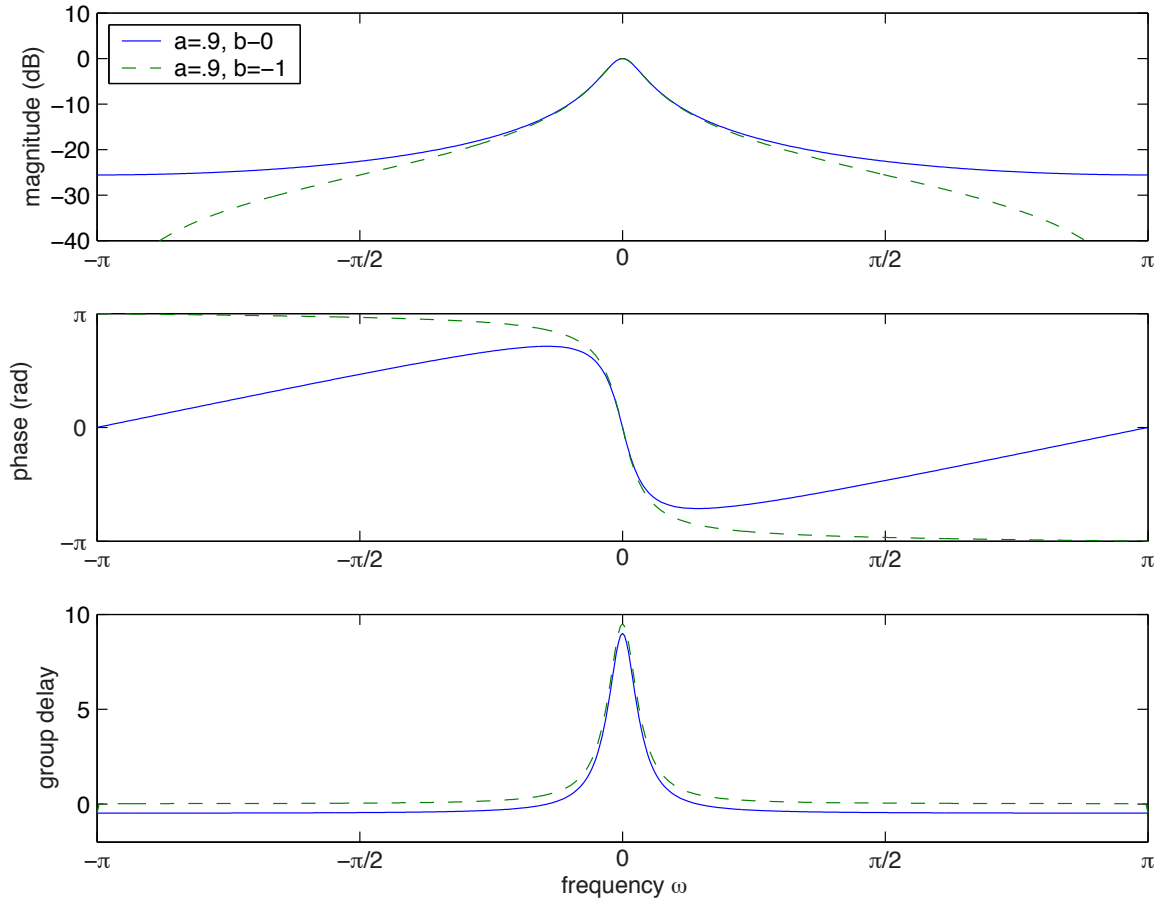


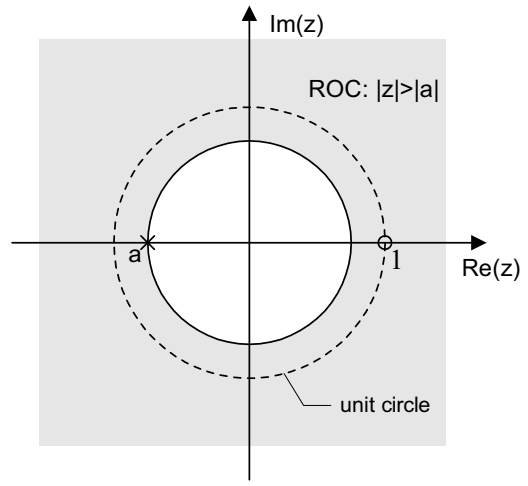
Fig. 5.3 Frequency response of first order system

High-pass case:

To get a high-pass behavior, one has to locate the pole at $a = -1 + \varepsilon$, where $0 < \varepsilon \ll 1$ To get a high attenuation of the DC component, one has to locate the zero at or near $b = 1$.

Example 5.3: High-Pass first order system

► The PZ diagram and transfer function of a high-pass first order system with $a = -.9$ and $b = 1$ are shown below.



zero: $b = 1$

$$H(z) = G\frac{1-z^{-1}}{1-az^{-1}}$$

$$G = \frac{1+a}{2} \Rightarrow H(-\pi) = 1$$

The corresponding frequency response is shown in Figure 5.4 ◄

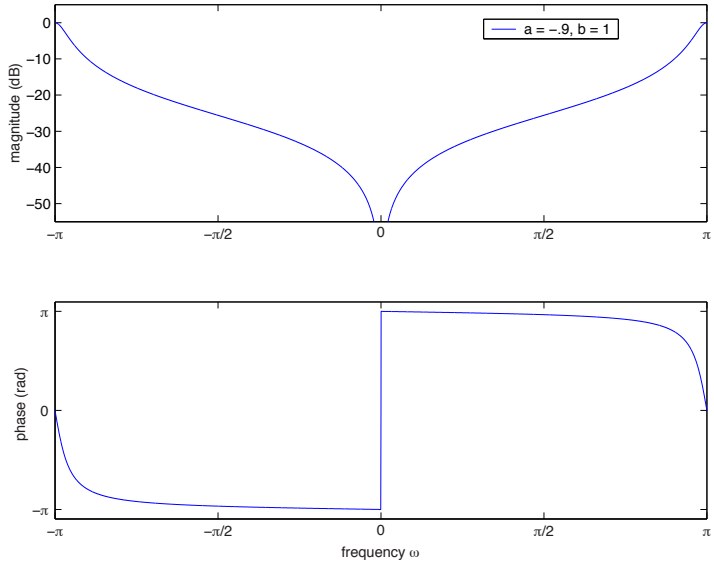


Fig. 5.4 Frequency response of a first-order high-pass filter with a pole at $z = -.9$ and a zero at $z = 1$

5.4.2 Second order systems

Description:

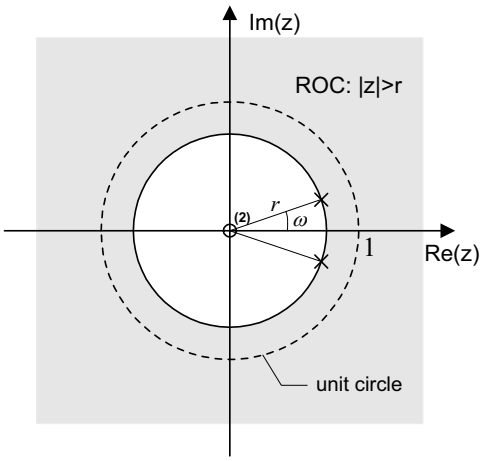
- System function:

$$H(z) = G\frac{1+b_1z^{-1}+b_2z^{-2}}{1+a_1z^{-1}+a_2z^{-2}} \tag{5.26}$$

- Poles:
  - if $a_1^2 > 4a_2$: 2 distinct poles (real) at $p_{1,2} = -\frac{a_1}{2} \pm \frac{1}{2}\sqrt{a_1^2 - 4a_2}$
  - if $a_1^2 = 4a_2$: double pole (real) at $p_1 = -\frac{a_1}{2}$
  - if $a_1^2 < 4a_2$: 2 distinct poles (complex) at $p_{1,2} = -\frac{a_1}{2} \pm j\frac{1}{2}\sqrt{4a_2 - a_1^2}$
- Practical requirements:
  - causality: ROC : $|z| > \max\{|p_1|, |p_2|\}$
  - stability: can show that $|p_k| < 1$ for all k iff

$$|a_2| < 1, \quad a_2 > |a_1| - 1 \tag{5.27}$$

Resonator:



$$\begin{aligned} p_1 &= re^{j\omega_o} \\ p_2 &= re^{-j\omega_o} = p_1^* \\ H(z) &= G\frac{1}{(1-re^{j\omega_o z^{-1}})(1-re^{-j\omega_o z^{-1}})} \\ &= G\frac{1}{1-2r\cos(\omega_o)z^{-1}+r^2z^{-2}} \end{aligned}$$

- The frequency response (Figure 5.5) clearly shows peaks around $\pm\omega_o$.
- For r close to 1 (but < 1), $|H(\omega)|$ attains a maximum at $\omega = \pm\omega_o$
- 3dB-bandwidth: for r close to 1, can show:

$$|H(\omega_o \pm \frac{\Delta\omega}{2})| = \frac{1}{\sqrt{2}}, \qquad \text{where } \Delta\omega = 2(1-r) \tag{5.28}$$

Notch filter:

A notch filter is an LTI system containing one or more notches in its frequency response, as the results of zeroes located on (or close to) the unit circle.

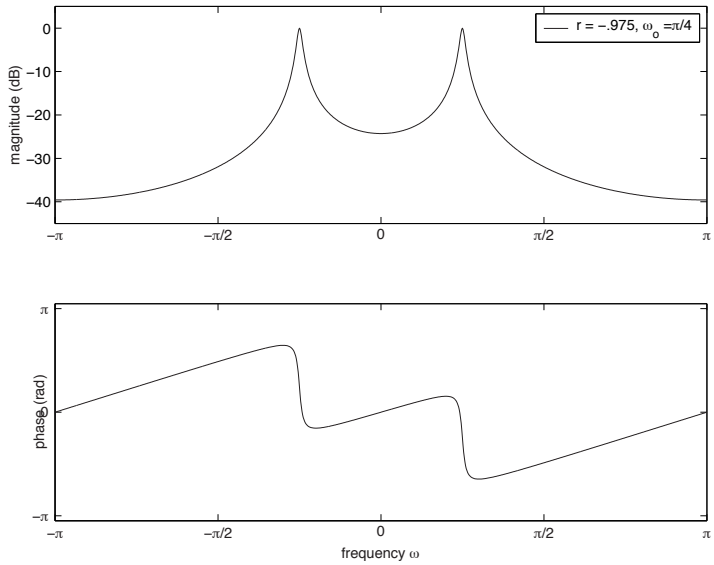
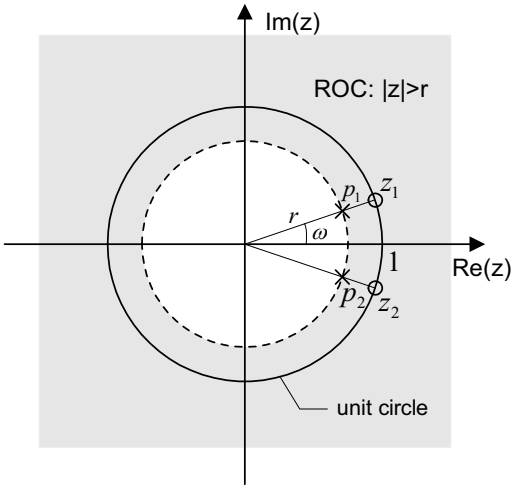


Fig. 5.5 Frequency response of a resonator system



$$
\begin{aligned}
z_1 &= e^{j\omega_o}, \quad z_2 = e^{-j\omega_o} = z_1^* \\
p_1 &= re^{j\omega_o}, \quad p_2 = re^{-j\omega_o} = p_1^*
\end{aligned}
$$

$$
\begin{aligned}
H(z) &= G\frac{(1-e^{j\omega_o}z^{-1})(1-e^{-j\omega_o}z^{-1})}{(1-re^{j\omega_o}z^{-1})(1-re^{-j\omega_o}z^{-1})} \\
&= G\frac{1-2\cos(\omega_o)z^{-1}+z^{-2}}{1-2r\cos(\omega_o)z^{-1}+r^2z^{-2}}
\end{aligned}
$$

The notches in the frequency response are clearly shown in Figure 5.6.

5.4.3 FIR filters

Description:

- System function:

$$
\begin{aligned}
H(z) &= B(z) = b_0 + b_1z^{-1} + \cdots + b_Mz^{-M} \\
&= b_0(1-z_1z^{-1})\cdots(1-z_Mz^{-1})
\end{aligned} \tag{5.29}
$$

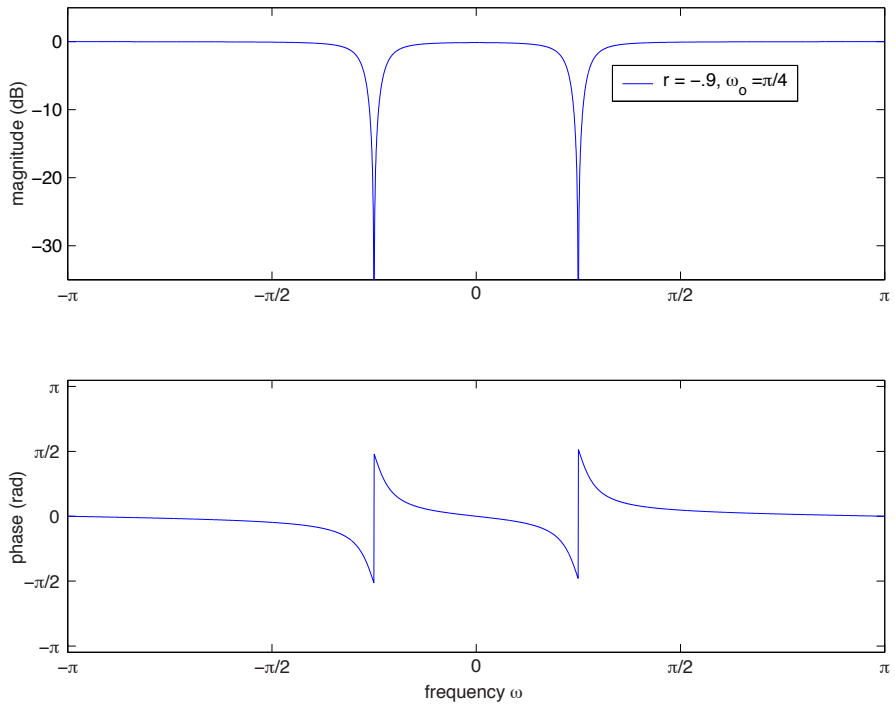


Fig. 5.6 Frequency response of a notch filter.

- This is a zeroth order rational system (i.e. $A(z) = 1$)
  The M zeros z_k can be anywhere in the complex plane
  There is a multiple pole of order M at $z = 0$.

- Practical requirement: none
  Above system is always causal and stable

- Impulse response:

$$h[n] = \begin{cases} b_n & 0 \le n \le M \\ 0 & \text{otherwise} \end{cases} \tag{5.30}$$

Moving average system:

- Difference equation:

$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k] \tag{5.31}$$

- System function:

$$H(z) = \frac{1}{M} \sum_{k=0}^{M-1} z^{-k} = \frac{1}{M} \frac{1 - z^{-M}}{1 - z^{-1}} \tag{5.32}$$

- PZ analysis: Roots of the numerator:

$$z^M = 1 \Rightarrow z = e^{j2\pi k/M}, \quad k = 0,1,...,M-1 \tag{5.33}$$

Note: there is no pole at $z = 1$ because of PZ cancellation. Thus:

$$H(z) = \frac{1}{M}\prod_{k=1}^{M-1}(1 - e^{j2\pi k/M}z^{-1}) \tag{5.34}$$

- The PZ diagram and frequency response for $M = 8$ are shown in Figures 5.7 and 5.8 respectively.

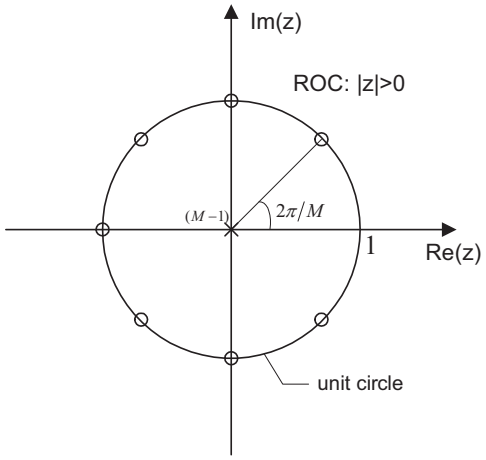


Fig. 5.7 Zero/Pole diagram for a Moving Average System ($M = 8$).

5.5 More on magnitude response

Preliminaries:

- Consider stable LTI system with impulse response $h[n]$
- Stable $\Rightarrow e^{j\omega} \in$ ROC $\Rightarrow$

$$\begin{aligned} H(\omega) &= \sum_n h[n]e^{-j\omega n} \\ &= \sum_n h[n]z^{-n}|_{z=e^{j\omega}} = H(z)|_{z=e^{j\omega}} \end{aligned} \tag{5.35}$$

- Recall that: $h[n] \in \mathbb{R} \Longleftrightarrow H^*(\omega) = H(-\omega) \Longleftrightarrow H^*(z) = H(z^*)$

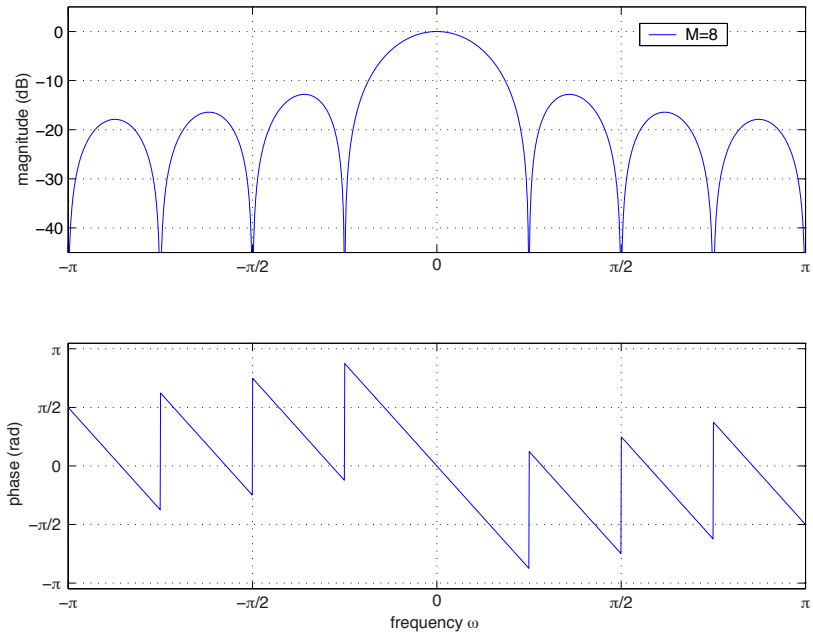


Fig. 5.8 Frequency response of a Moving Average system ($M = 8$).

Properties:

The squared magnitude response of a stable LTI system can be expressed in terms of the system function as follows:

$$\begin{aligned} |H(\omega)|^2 &= H(z)H^*(1/z^*)|_{z=e^{j\omega}} \\ &= H(z)H(1/z)|_{z=e^{j\omega}} \text{ if } h[n] \in \mathbb{R} \end{aligned} \tag{5.36}$$

Proof: Using (5.35), the square magnitude response can be expressed as

$$|H(\omega)|^2 = H(\omega)H^*(\omega) = H(z)H^*(z)|_{z=e^{j\omega}} \tag{5.37}$$

Note that when $z = e^{j\omega}$, we can write $z = 1/z^*$. Hence:

$$H^*(z) = H^*(1/z^*)|_{z=e^{j\omega}} \tag{5.38}$$

When $h[n] \in \mathbb{R}$, we have

$$H^*(z) = H(z^*) = H(1/z)|_{z=e^{j\omega}} \tag{5.39}$$

To complete the proof, substitute (5.38) or (5.39) in (5.37) □.

Magnitude square function:

- $C(z) \triangleq H(z)H^*(1/z^*)$

- Important PZ property:
  - z_k = zero of $H(z) \Rightarrow z_k$ and $1/z_k^*$ are zeros of $C(z)$
  - p_k = pole of $H(z) \Rightarrow p_k$ and $1/p_k^*$ are poles of $C(z)$
- This is called conjugate reciprocal symmetry (see Figure 5.9).

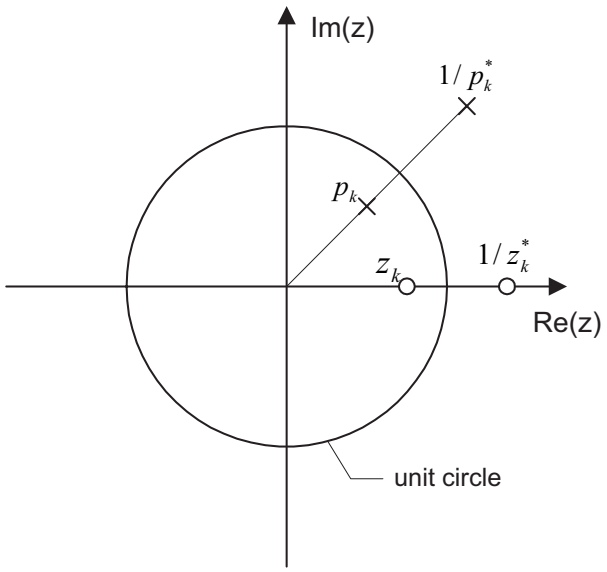


Fig. 5.9 Typcial Zero/Pole plot for a magnitude square function $C(z)$.

- Suppose magnitude response $|H(\omega)|^2$ is known as a function of ω, as well as the number of poles and zeros of $H(z)$:
  - we can always find the PZ diagram of $C(z)$
  - from there, only a finite number of possibilities for $H(z)$

5.6 All-pass systems

Definition:

We say that LTI system $\mathcal{H}$ is all-pass (AP) iff its frequency response, $H(\omega)$, satisfies

$$|H(\omega)| = K, \quad \text{for all } \omega \in [-\pi, \pi] \tag{5.40}$$

where K is a positive constant. In the sequel, this constant is set to 1.

Properties:

Let $H(z)$ be an AP system.

- From definition of AP system,

$$|H(\omega)|^2 = H(z)H^*(1/z^*)|_{z=e^{j\omega}} = 1, \quad \text{for all } \omega \in [-\pi,\pi] \tag{5.41}$$

This implies

$$\boxed{H(z)H^*(1/z^*) = 1, \quad \text{for all } z \in \mathbb{C}} \tag{5.42}$$

- From above relation, it follows that

$$H(1/z^*) = 1/H^*(z) \tag{5.43}$$

Therefore

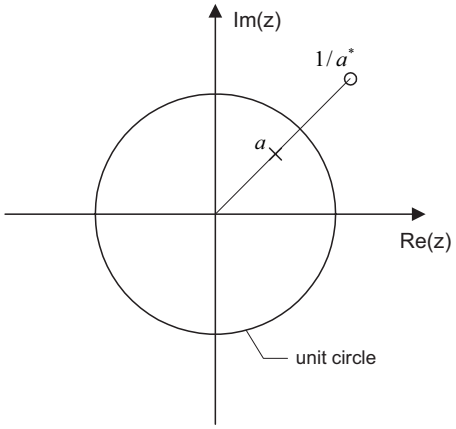
$$\boxed{\begin{array}{l} z_o = \text{zero of order } l \text{ of } H(z) \Longleftrightarrow 1/z_o^* = \text{pole of order } l \text{ of } H(z) \\ p_o = \text{pole of order } l \text{ of } H(z) \Longleftrightarrow 1/p_o^* = \text{zero of order } l \text{ of } H(z) \end{array}}$$

Trivial AP system:

- Consider integer delay system system: $H(\omega) = e^{-j\omega k}$
- This is an all-pass system: $|H(\omega)| = 1$

First order AP system:

From the above PZ considerations, we obtain the following PZ diagram:



System function:

$$\begin{aligned} H_{AP}(z) &= G\,\frac{z-\frac{1}{a^*}}{z-a} \\ &= \cdots \\ &= \frac{z^{-1}-a^*}{1-az^{-1}}. \end{aligned} \tag{5.44}$$

where the system gain G has been chosen such that $H(z) = 1$ at $z = 1$.

The frequency response is shown in Figure 5.10.

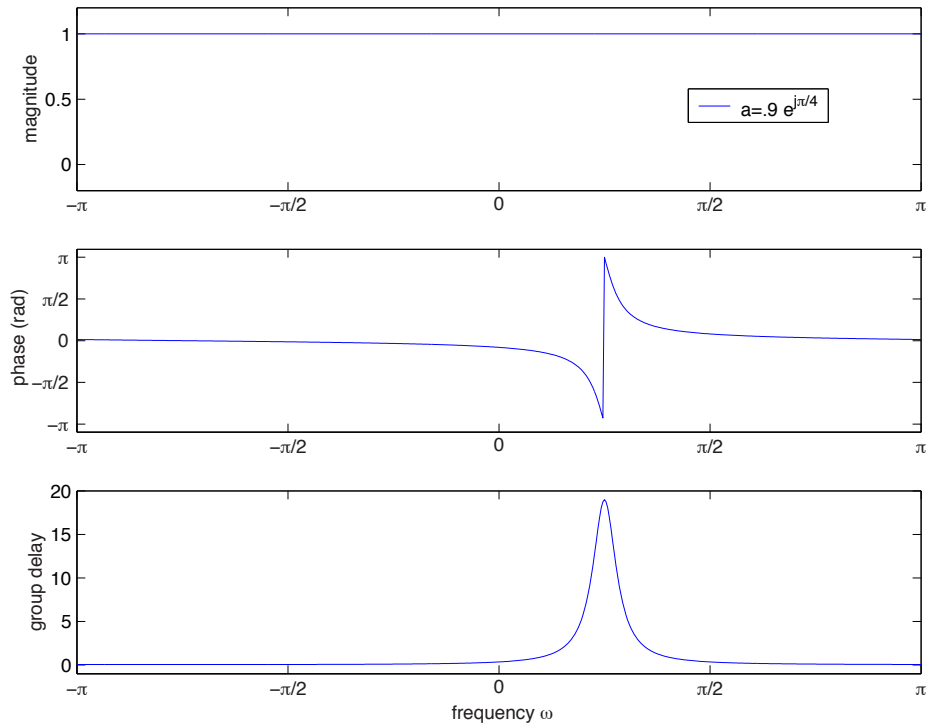


Fig. 5.10 Frequency response of a first order all-pass system.

General form of rational AP system:

- Consider rational system:

$$H(z) = z^{-L}\frac{B(z)}{A(z)} = z^{-L}\frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} \tag{5.45}$$

- In order for this system to be AP, we need:

$$\begin{aligned} B(z) &= z^{-N}A^*(1/z^*) \\ &= \sum_{k=0}^{N} a_k^* z^{k-N} \end{aligned} \tag{5.46}$$

- AP system will be causal and stable if, in addition to above:
  - $L \geq 0$
  - all poles (i.e. zeros of $A(z)$) inside U.C.

Remark:

- Consider two LTI systems with freq. resp. $H_1(\omega)$ and $H_2(\omega)$.

- $|H_1(\omega)| = |H_2(\omega)|$ for all $\omega \in [-\pi, \pi]$ iff $H_2(z) = H_1(z)H_{ap}(z)$ for some all-pass system $H_{ap}(z)$

5.7 Inverse system

Definition:

A system $\mathcal{H}$ is invertible iff for any arbitrary input signals x_1 and x_2, we have:

$$x_1 \neq x_2 \Longrightarrow \mathcal{H}x_1 \neq \mathcal{H}x_2 \tag{5.47}$$

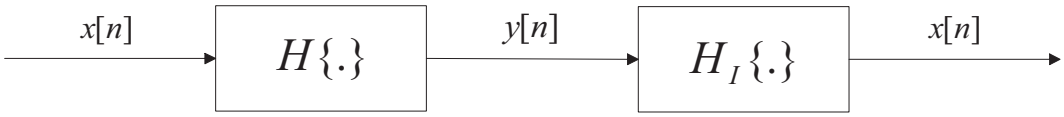
Definition:

Let $\mathcal{H}$ be an invertible system. Its inverse, denoted $\mathcal{H}_I$, is such that for any input signal x in the domain of $\mathcal{H}$, we have:

$$\mathcal{H}_I(\mathcal{H}x) = x \tag{5.48}$$

Remarks:

- Invertible means that there is a one-to-one correspondence between the set of possible input signals (domain of $\mathcal{H}$) and the set of corresponding output signals (range).
- When applied to output signal $y = \mathcal{H}x$, the inverse system produces the original input x



Fundamental property:

Let $\mathcal{H}$ be an invertible LTI system with system function $H(z)$. The inverse system $\mathcal{H}_I$ is also LTI with system function

$$\boxed{H_I(z) = 1/H(z)} \tag{5.49}$$

Remarks:

- Several ROC are usually possible for $H_I(z)$ (corresponding to causal, mixed or anti-causal impulse response $h_I[n]$)
- Constraint: $\text{ROC}(\mathcal{H}) \cap \text{ROC}(\mathcal{H}_I)$ cannot be empty
- From (5.49), it should be clear that the zeros of $H(z)$ become the poles of $H_I(z)$ and vice versa (with the order being preserved).

- Thus, causal and stable inverse will exist iff all the zero of $H_I(z)$ are inside the unit circle.

Proof (the first two parts are optional):

- *Linearity:* Let $y_1 = \mathcal{H}x_1$ and $y_2 = \mathcal{H}x_2$. Since $\mathcal{H}$ is linear by assumption, we have $\mathcal{H}(a_1x_1 + a_2x_2) = a_2y_1 + a_2y_2$. By definition of an inverse system,

  $$\mathcal{H}_I(a_2y_1 + a_2y_2) = a_1x_1 + a_2x_2 = a_1\mathcal{H}_I(y_1) + a_2\mathcal{H}_I(y_2)$$

  which shows that $\mathcal{H}_I$ is linear.
- *Time invariance:* Let $y = \mathcal{H}x$ and let $\mathcal{D}_k$ denote the shift by k operation. Since $\mathcal{H}$ is assumed to be time-invariant, $\mathcal{H}(\mathcal{D}_k x) = \mathcal{D}_k y$. Therefore, $\mathcal{H}_I(\mathcal{D}_k y) = \mathcal{D}_k x = \mathcal{D}_k(\mathcal{H}_I y)$ which shows that $\mathcal{H}_I$ is also TI.
- *Eq. (5.49):* For any x in the domain of $\mathcal{H}$, we must have $\mathcal{H}_I(\mathcal{H}x) = x$. Since both $\mathcal{H}$ and $\mathcal{H}_I$ are LTI systems, this implies that

  $$H_I(z)H(z)X(z) = X(z)$$

  from which (5.49) follows immediately □.

Example 5.4:

► Consider FIR system with impulse response ($0 < a < 1$):

$$h[n] = \delta[n] - a\delta[n-1]$$

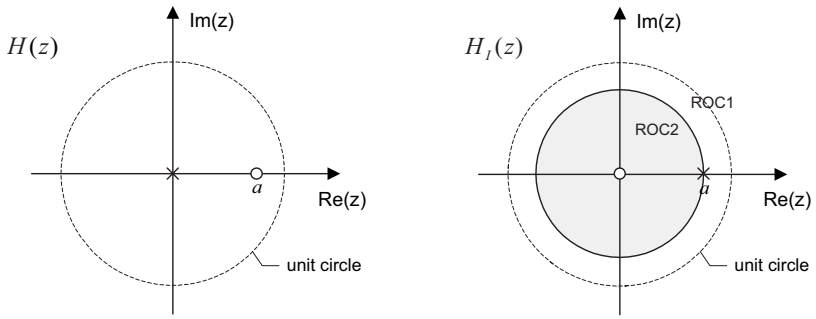
The corresponding system function

$$H(z) = 1 - az^{-1}, \quad |z| > 0$$

Invoking (5.49), we obtain

$$H_I(z) = \frac{1}{1 - az^{-1}}$$

The corresponding PZ diagrams are shown below



Note that there are two possible ROC for the inverse system $H_I(z)$:

- $\text{ROC}_1 : |z| > a \Rightarrow$

  $$h_I[n] = a^n u[n] \quad \text{causal \& stable}$$

- $\text{ROC}_2 : |z| < a \Rightarrow$

  $$h_I[n] = -a^n u[-n-1] \quad \text{anti-causal \& unstable}$$

◄

5.8 Minimum-phase system

5.8.1 Introduction

It is of practical importance to know when a causal & stable inverse $\mathcal{H}_I$ exists.

Example 5.5: Inverse system and ROC

► Consider the situation described by the pole/zero plot in Figure 5.11 (note: $H_I(z) = 1/H(z)$). There are 3 possible ROC for the inverse system $\mathcal{H}_I$:
- $\text{ROC}_1 : |z| < 1/2 \Rightarrow$ anti-causal & unstable
- $\text{ROC}_2 : 1/2 < |z| < 3/2 \Rightarrow$ mixed & stable
- $\text{ROC}_1 : |z| > 3/2 \Rightarrow$ causal & unstable

A causal and stable inverse does not exist! ◄

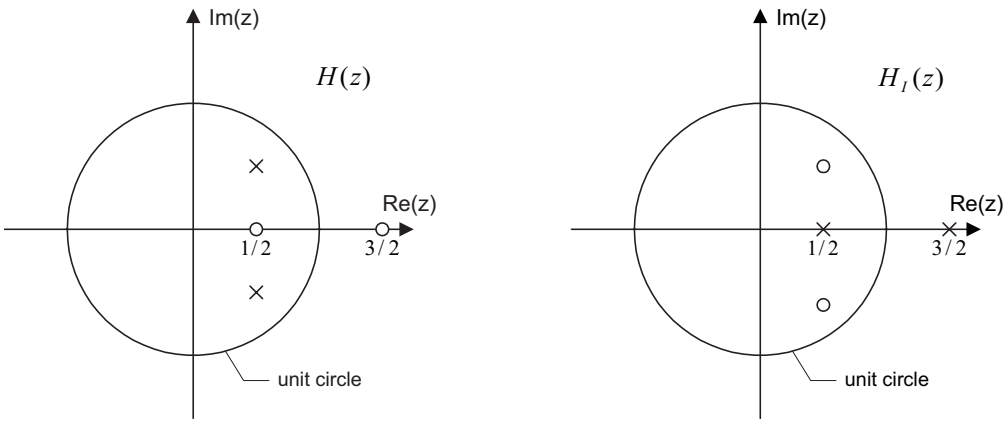


Fig. 5.11 Example of pole/zero plot of an LTI system and its inverse.

From the above example, it is clear that:

$$\mathcal{H}_I \text{ causal and stable} \iff \text{poles of } H_I(z) \text{ inside u.c.} \iff \text{zeros of } H(z) \text{ inside u.c.}$$

These considerations lead to the definition of *minimum phase* systems, which fulfill the above conditions.

Definition:

A causal LTI system $\mathcal{H}$ is said to be minimum phase (MP) iff all its poles and zeros are inside the unit circle.

Remarks

- Poles inside u.c. $\Rightarrow h[n]$ can be chsoen to be causal and stable
  Zeros inside u.c. $\Rightarrow h_I[n]$ can be chsoen to be causal and stable exists

- Minimum phase systems play a very important role in practical applications of digital filters.

5.8.2 MP-AP decomposition

Any rational system function can be decomposed as the product of a minimum-phase and an all-pass component, as illustrated in Figure 5.12:

$$H(z) = \underbrace{H_{min}(z)}_{\text{MP}}\underbrace{H_{ap}(z)}_{\text{AP}} \tag{5.50}$$

Fig. 5.12 Minimum phase/All pass decomposition of a rational system.

Example 5.6:

▶ Consider for instance the transfer function

$$H(z) = \frac{(1-\frac{1}{2}z^{-1})(1-\frac{3}{2}z^{-1})}{1-z^{-1}+\frac{1}{2}z^{-2}},$$

whose pole/zero plot is shown in Figure 5.11. The zero at $z = 3/2$ is not inside the unit circle, so that $H(z)$ is not minimum-phase. The annoying factor is thus $1-\frac{3}{2}z^{-1}$ on the numerator of $H(z)$, and it will have to be included in the all-pass component. This means that the all-pass component $H_{ap}(z)$ will have a zero at $z = 3/2$, and thus a pole at $z = 2/3$. The all-pass component is then

$$H_{ap}(z) = G\frac{1-\frac{3}{2}z^{-1}}{1-\frac{2}{3}z^{-1}}.$$

while $H_{min}(z)$ is chosen so as to have $H(z) = H_{min}(z)H_{ap}(z)$:

$$H_{min}(z) = \frac{H(z)}{H_{ap}(z)} = \frac{1}{G}\frac{(1-\frac{1}{2}z^{-1})(1-\frac{2}{3}z^{-1})}{1-z^{-1}+\frac{1}{2}z^{-2}}$$

The corresponding pole-zero plots are shown in Figure 5.13. If we further require the all-pass component to have unit gain, i.e. $H_{ap}(z=1) = 1$, then we must set $G = -3/2$. ◀

5.8.3 Frequency response compensation

In many applications, we need to remove distortion introduced by a channel on its input via a compensating filter (e.g. channel equalization), like illustrated in Figure 5.14.

Suppose $H(\omega)$ is not minimum phase; in this case we know that there exists no stable and causal inverse. Now, consider the MP-AP decomposition of the transfer function: $H(\omega) = H_{min}(\omega)H_{ap}(\omega)$ A possible

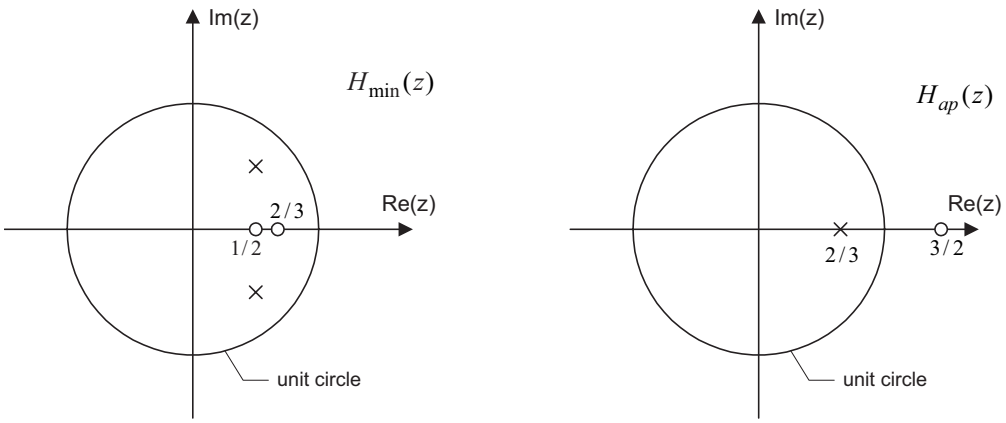


Fig. 5.13 Pole/Zero plot for a decomposition in Minimum-phase and all-pass parts.

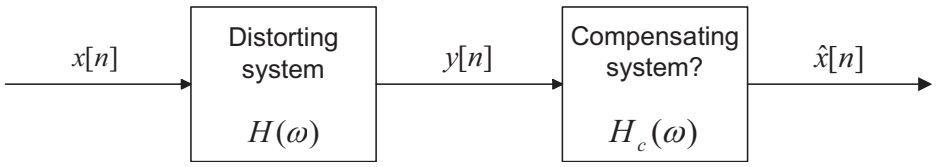


Fig. 5.14 Frequecny response Compensation

choice for the compensating filter (which is then stable and causal) is the inverse of the minimum phase component of $H(\omega)$):

$$H_c(\omega) = 1/H_{min}(\omega). \tag{5.51}$$

In this case, the compounded frequency response of the cascaded systems in Figure 5.14 is given by

$$\frac{\hat{X}(\omega)}{X(\omega)} = H_c(\omega)H(\omega) = H_{ap}(\omega), \tag{5.52}$$

Accordingly, the magnitude spectrum of the compensator output signal $\hat{x}[n]$ is identical to that of the input signal $x[n]$, whereas whereas their phases differ by the phase response of the all pass component $H_{ap}(z)$.

$$\begin{aligned} |\hat{X}(\omega)| &= |X(\omega)| \Rightarrow \text{ exact magnitude compensation} \\ \angle\hat{X}(\omega) &= \angle X(\omega) + \angle H_{ap}(\omega) \Rightarrow \text{ phase distortion} \end{aligned} \tag{5.53}$$

Thanks to this decomposition, at least the effect of $H(\omega)$ on the input signal's magnitude spectrum can be compensated for.

5.8.4 Properties of MP systems

Consider a causal MP system with frequency response $H_{min}(\omega)$. For any causal system $H(\omega)$ with the same magnitude response (i.e. $|H(\omega)| = |H_{min}(\omega)|$):

(1) group delay of $H(\omega) \geq$ group delay $H_{min}(\omega)$

(2) $\sum_{n=0}^{M} |h[n]|^2 \leq \sum_{n=0}^{M} |h_{min}[n]|^2$, for all integer $M \geq 0$

According to (1), the minimum phase system has minimum processing delay among all systems with the same magnitude response. Property (2) states that for the MP system, the energy concentration of the impulse response $h_{min}[n]$ around $n = 0$ is maximum (i.e. minimum energy delay).

5.9 Problems

Problem 5.1: Pole-zero plots
For each of the pole-zero plots in figure 5.15, state whether it can correspond to

1. an allpass system;
2. a minmum phase system;
3. a system with real impulse response;
4. an FIR system;
5. a Generalized Linear Phase system;
6. a system with a stable inverse.

In each case, specify any additional constraint on the ROC in order for the property to hold.

Problem 5.2: Inversion Moving Average system
An M-point moving average system is defined by the impulse response

$$h[n] = \frac{1}{M}(u[n] - u[n-M]).$$

1. find the z-transform of $h[n]$ (use the formula for a sum of a geometric series to simplify your expression);
2. Draw a pole-zero plot of $H(z)$. What is the region of converhence ?
3. Compute the z-transform for all possible inverses $h_I[n]$ of $h[n]$. Is any of these inverses stable ? What is the geenral shape of the correspndiong impulse responses $h_I[n]$?
4. A modified Moving Average system with forgetting factor is

$$h[n] = \frac{1}{M}a^n(u[n] - u[n-M]),$$

where a is real, positive and $a < 1$. Compute $G(z)$ and draw a pole-zero plot. Find a stable and causal inverse $G_I(z)$ and compute its impulse response $g_I[n]$.

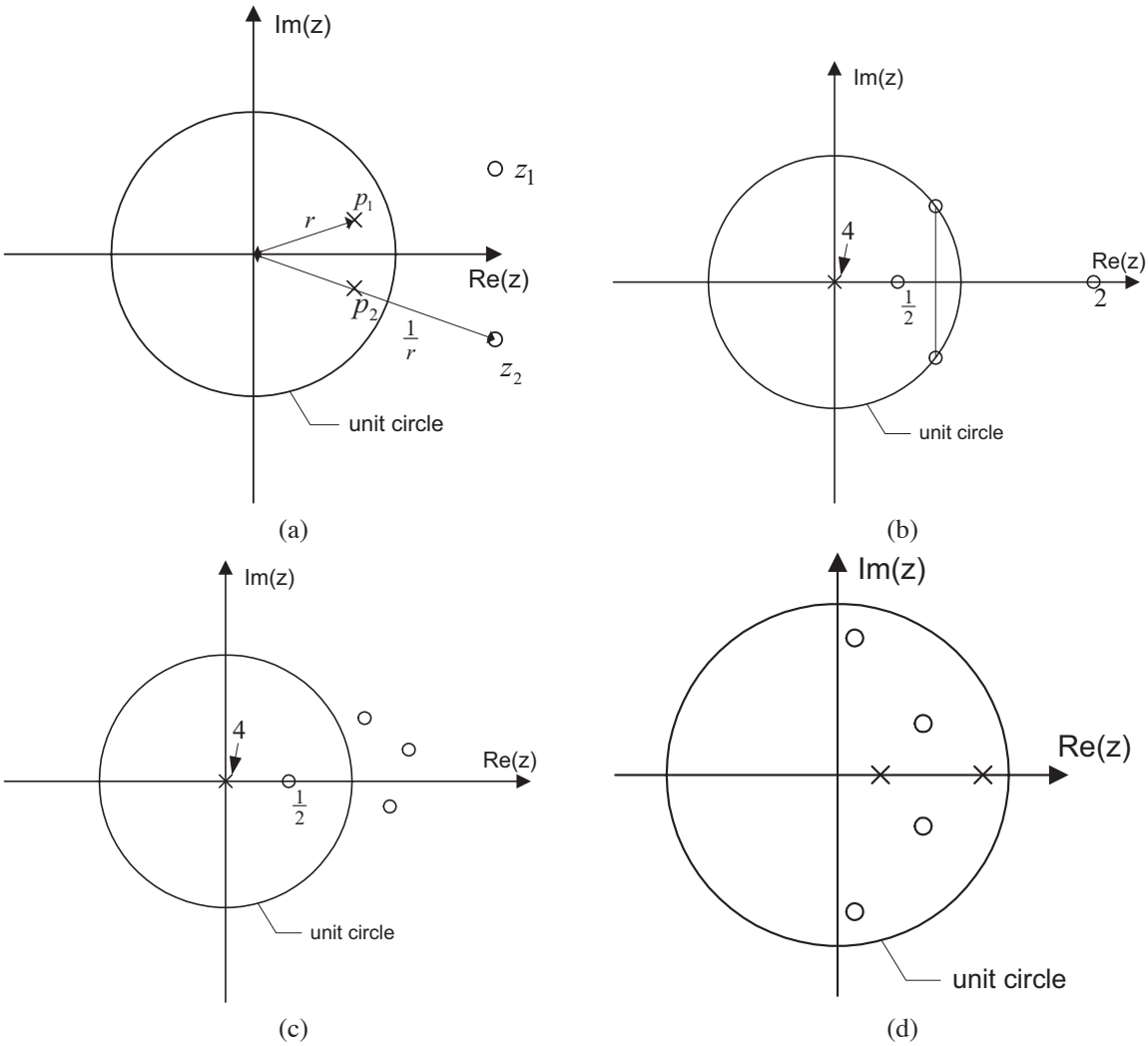


Fig. 5.15 Pole-Zero plots pertaining to problem 5.1.

Problem 5.3: Implementation of an approximate noncausal inverse
Let a system be defined by

$$H(z) = \frac{1 - bz^{-1}}{1 - az^{-1}}, \qquad |z| > |a|,$$

where $|b| > 1 > |a|$.

1. Is this system stable? Is it causal ?
2. Find the impulse response $h[n]$.
3. Find the z-transforms $H_I(z)$ of all possible inverses for this system. In each case, specify a region of convergence, and whether the inverse is stable and/or caussal.
4. For each of the above inverses, compute the impulse response and skecth it for $a = 0.5, b = 1.5$.
5. Consider the anticausal inverse $h_I[n]$ above. You want to implement a delayed and truncated version of this system. Find the number of samples of $h_I[n]$ to keep and the corresponding delay, so as to keep 99.99% of the energy of the impulse response $h_I[n]$.

Problem 5.4: Minimum-phase/All-Pass decomposition
Let a system be defined in the z domain by the following set of poles and zeros:

- zeros: $2, 0.7e^{\pm j\pi/8}$;
- poles: $\frac{3}{4}, 0.3e^{\pm j\pi/12}$.

Furthermore, $H(z)$ is equal to 1 at $z = 1$.

1. Draw a pole-zero plot for this system;
2. Assume that the system is causal. Is it stable ?
3. Compute the factorization of $H(z)$ into

$$H(z) = H_{min}(z)H_{ap}(z),$$

where $H_{min}(z)$ is minmimum-phase , and $H_{ap}(z)$ is allpass. Draw a pole-zero plot for $H_{min}(z)$ and $H_{ap}(z)$.

Problem 5.5: Two-sided sequence
Let a stable system have the transfer function

$$H(z) = \frac{1}{(1 - \frac{3}{2}z^{-1})(1 - \frac{1}{2}e^{j\pi/4}z^{-1})(1 - \frac{1}{2}e^{-j\pi/4}z^{-1})}.$$

1. Draw a pole-zero plot for $H(z)$ and specify its ROC;
2. compute the inverse z-transform $h[n]$.

Chapter 6

The discrete Fourier Transform (DFT)

Introduction:

The DTFT has proven to be a valuable tool for the theoretical analysis of signals and systems. However, if one looks at its definition, i.e.:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}, \quad \omega \in [-\pi, \pi], \tag{6.1}$$

from a computational viewpoint, it becomes clear that it suffers several drawbacks Indeed, its numerical evaluation poses the following difficulties:

- the summation over n is infinite;
- the variable ω is continuous.

In many situations of interest, it is either not possible, or not necessary, to implement the infinite summation $\sum_{n=-\infty}^{\infty}$ in (6.1):

- only the signal samples $x[n]$ from $n = 0$ to $n = N-1$ are available;
- the signal is known to be zero outside this range; or
- the signal is periodic with period N.

In all these cases, we would like to analyze the frequency content of signal $x[n]$ based only on the finite set of samples $x[0], x[1], \ldots, x[N-1]$. We would also like a frequency domain representation of these samples in which the frequency variable only takes on a finite set of values, say ω_k for $k = 0, 1, \ldots, N-1$, in order to better match the way a processor will be able to compute the frequency representation.

The discrete Fourier transform (DFT) fulfills these needs. It can be seen as an approximation to the DTFT.

6.1 The DFT and its inverse

Definition:

The N-point DFT is a transformation that maps DT signal samples $\{x[0],\ldots,x[N-1]\}$ into a periodic sequence $X[k]$, defined by

$$X[k] = \text{DFT}_N\{x[n]\} \triangleq \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N}, \quad k \in \mathbb{Z} \tag{6.2}$$

Remarks:

- Only the samples $x[0],...,x[N-1]$ are used in the computation.
- The N-point DFT is periodic, with period N:

$$X[k+N] = X[k]. \tag{6.3}$$

  Thus, it is sufficient to specify $X[k]$ for $k = 0,1,...,N-1$ only.

- The DFT $X[k]$ may be viewed as an approximation to the DTFT $X(\omega)$ at frequency $\omega_k = 2\pi k/N$.
- The "D" in DFT stands for discrete frequency (i.e. ω_k)
- Other common notations:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} \quad \text{where} \quad \omega_k \triangleq 2\pi k/N \tag{6.4}$$

$$= \sum_{n=0}^{N-1} x[n]W_N^{kn} \quad \text{where} \quad W_N \triangleq e^{-j2\pi/N} \tag{6.5}$$

Example 6.1:

▶ (a) Consider

$$x[n] = \begin{cases} 1 & n = 0, \\ 0 & n = 1,\ldots,N-1. \end{cases}$$

We have

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N} = 1, \quad \text{all } k \in \mathbb{Z}$$

(b) Let

$$x[n] = a^n, \quad n = 0,1,\ldots,N-1$$

We have

$$\begin{aligned} X[k] &= \sum_{n=0}^{N-1} a^n e^{-j2\pi kn/N} \\ &= \sum_{n=0}^{N-1} \rho_k^n \quad \text{where} \quad \rho_k \triangleq a\, e^{-j2\pi k/N} \\ &= \begin{cases} N & \text{if } \rho_k = 1, \\ \dfrac{1-\rho_k^N}{1-\rho} & \text{otherwise.} \end{cases} \end{aligned}$$

Note the following special cases of interest:

$$a = 1 \implies X[k] = \begin{cases} N & \text{if } k = 0, \\ 0 & \text{if } k = 1, \ldots, N-1. \end{cases}$$

$$a = e^{j2\pi l/N} \implies X[k] = \begin{cases} N & \text{if } k = l \text{ modulo } N, \\ 0 & \text{otherwise .} \end{cases}$$

◀

Inverse DFT (IDFT):

The N-point IDFT of the samples $X[0], \ldots, X[N-1]$ is defined as the periodic sequence

$$\tilde{x}[n] = \text{IDFT}_N\{X[k]\} \triangleq \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}, \quad n \in \mathbb{Z} \tag{6.6}$$

Remarks:

- In general, $\tilde{x}[n] \neq x[n]$ for all $n \in \mathbb{Z}$ (more on this later).
- Only the samples $X[0], \ldots, X[N-1]$ are used in the computation.
- The N-point IDFT is periodic, with period N:

$$\tilde{x}[n+N] = \tilde{x}[n] \tag{6.7}$$

- Other common notations:

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\omega_k n} \quad \text{where} \quad \omega_k \triangleq 2\pi k/N \tag{6.8}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn} \quad \text{where} \quad W_N \triangleq e^{-j2\pi/N} \tag{6.9}$$

IDFT Theorem:

If $X[k]$ is the N-point DFT of $\{x[0], \ldots, x[N-1]\}$, then

$$x[n] = \tilde{x}[n] = \text{IDFT}_N\{X[k]\}, \quad n = 0, 1, \ldots, N-1 \tag{6.10}$$

Remarks:

- The theorem states that $x[n] = \tilde{x}[n]$ for $n = 0, 1, ..., N-1$ only. Nothing is said about sample values of $x[n]$ outside this range.
- In general, it is not true that $x[n] = \tilde{x}[n]$ for all $n \in \mathbb{Z}$: the IDFT $\tilde{x}[n]$ is periodic with period N, whereas no such requirement is imposed on the original signal $x[n]$.
- Therefore, the values of $x[n]$ for $n < 0$ and for $n \geq N$ cannot in general be recovered from the DFT samples $X[k]$. This is understandable since these sample values are not used when computing $X[k]$.
- However, there are two important special cases when the complete signal $x[n]$ can be recovered from the DFT samples $X[k]$ ($k = 0, 1, ..., N-1$):
  - $x[n]$ is periodic with period N
  - $x[n]$ is known to be zero for $n < 0$ and for $n \geq N$

Proof of Theorem:

- First note that:

$$\frac{1}{N}\sum_{k=0}^{N-1} e^{-j2\pi kn/N} = \begin{cases} 1 & \text{if } n = 0, \pm N, \pm 2N, ... \\ 0 & \text{otherwise} \end{cases} \tag{6.11}$$

  Indeed, the above sum is a sum of terms of a geometric series. If $n = lN$, $l \in \mathbb{Z}$, then all the N terms are equal to one, so that the sum is N. Otherwise, the sum is equal to

$$\frac{1 - e^{-j2\pi kn}}{1 - e^{-j2\pi n/N}},$$

  whose numerator is equal to 0.

- Starting from the IDFT definition:

$$\begin{aligned} \tilde{x}[n] &= \frac{1}{N}\sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N} \\ &= \frac{1}{N}\sum_{k=0}^{N-1} \left\{ \sum_{l=0}^{N-1} x[l] e^{-j2\pi kl/N} \right\} e^{j2\pi kn/N} \\ &= \sum_{l=0}^{N-1} x[l] \left\{ \frac{1}{N}\sum_{k=0}^{N-1} e^{-j2\pi k(l-n)/N} \right\} \end{aligned} \tag{6.12}$$

  For the special case when $n \in \{0, 1, ..., N-1\}$, we have $-N < l - n < N$. Thus, according to (6.11), the bracketed term is zero unless $l = n$. Therefore:

$$\tilde{x}[n] = x[n], \quad n = 0, 1, \ldots, N-1 \quad \square \tag{6.13}$$

6.2 Relationship between the DFT and the DTFT

Introduction

The DFT may be viewed as a finite approximation to the DTFT:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} \approx X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

at frequency $\omega = \omega_k = 2\pi k/N$ As pointed out earlier, in general, an arbitrary signal $x[n]$, $n \in \mathbb{Z}$, cannot be recovered entirely from its N-point DFT $X[k], k \in \{0,...,N-1\}$. Thus, it should not be possible to recover the DTFT exactly from the DFT.

However, in the following two special cases:

- finite length signals
- N-periodic signals,

$x[n]$ can be completely recovered from its N-point DFT. In these two cases, the DFT is not merely an approximation to the DTFT: the DTFT can be evaluated exactly, at any given frequency $\omega \in [-\pi,\pi]$, if the N-point DFT $X[k]$ is known.

6.2.1 Finite length signals

Assumption: Suppose $x[n] = 0$ for $n < 0$ and for $n \geq N$.

Inverse DFT: In this case, $x[n]$ can be recovered entirely from its N-point DFT $X[k], k = 0,1,...,N-1$ Let $\tilde{x}[n]$ denote the IDFT of $X[k]$:

$$\tilde{x}[n] = \text{IDFT}\{X[k]\} = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j2\pi kn/N}, \quad n \in \mathbb{Z}. \tag{6.14}$$

For $n = 0,1,...,N-1$, the IDFT theorem yields: $x[n] = \tilde{x}[n]$. For $n < 0$ and for $n \geq N$, we have $x[n] = 0$ by assumption. Therefore:

$$x[n] = \begin{cases} \tilde{x}[n] & \text{if } 0 \leq n < N, \\ 0 & \text{otherwise.} \end{cases} \tag{6.15}$$

Relationship between DFT and DTFT: Since the DTFT $X(\omega)$ is a function of the samples $x[n]$, it should be clear that in this case, it is possible to completely reconstruct the DTFT $X(\omega)$ from the N-point DFT $X[k]$.

First consider the frequency $\omega_k = 2\pi k/N$:

$$\begin{aligned} X(\omega_k) &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega_k n} \\ &= \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} = X[k] \end{aligned} \tag{6.16}$$

The general case, i.e. ω arbitrary, is handled by the following theorem.

Theorem: Let $X(\omega)$ and $X[k]$ respectively denote the DTFT and N-point DFT of signal $x[n]$. If $x[n] = 0$ for $n < 0$ and for $n \geq 0$, then:

$$X(\omega) = \sum_{k=0}^{N-1} X[k] P(\omega - \omega_k) \tag{6.17}$$

where

$$P(\omega) \triangleq \frac{1}{N} \sum_{n=0}^{N-1} e^{-j\omega n} \tag{6.18}$$

Remark: This theorem provides a kind of interpolation formula for evaluating $X(\omega)$ in between adjacent values of $X(\omega_k) = X[k]$.

Proof:

$$\begin{aligned} X(\omega) &= \sum_{n=0}^{N-1} x[n] e^{-j\omega n} \\ &= \sum_{n=0}^{N-1} \left\{ \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\omega_k n} \right\} e^{-j\omega n} \\ &= \sum_{k=0}^{N-1} X[k] \left\{ \frac{1}{N} \sum_{n=0}^{N-1} e^{-j(\omega-\omega_k)n} \right\} \\ &= \sum_{k=0}^{N-1} X[k] P(\omega - \omega_k) \quad \square \end{aligned} \tag{6.19}$$

Properties of $P(\omega)$:

- Periodicity: $P(\omega + 2\pi) = P(\omega)$
- If $\omega = 2\pi l$ (l integer), then $e^{-j\omega n} = e^{-j2\pi l n} = 1$ so that $P(\omega) = 1$.
- If $\omega \neq 2\pi l$, then

$$\begin{aligned} P(\omega) &= \frac{1}{N} \frac{1 - e^{-j\omega N}}{1 - e^{j\omega}} \\ &= \frac{1}{N} e^{-j\omega(N-1)/2} \frac{\sin(\omega N/2)}{\sin(\omega/2)} \end{aligned} \tag{6.20}$$

- Note that at frequency $\omega_k = 2\pi k/N$:

$$P(\omega_k) = \begin{cases} 1 & k = 0, \\ 0 & k = 1, \ldots, N-1. \end{cases} \tag{6.21}$$

  so that (6.17) is consistent with (6.16).
- Figure 6.1 shows the magnitude of $P(\omega)$ (for $N = 8$).

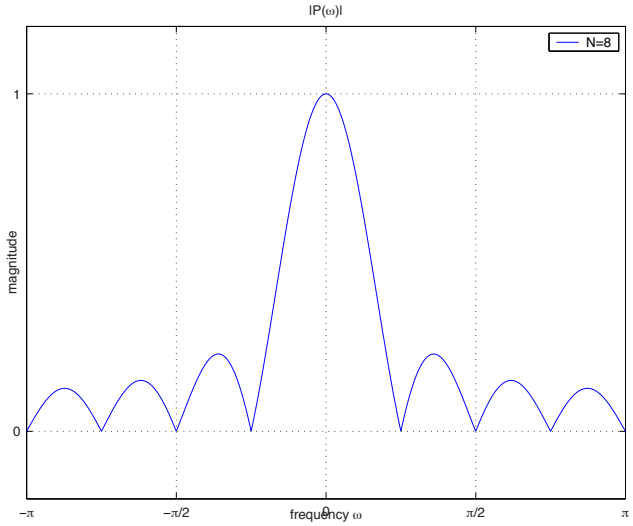


Fig. 6.1 Magnitude spectrum of the frequency interpolation function $P(\omega)$ for $N = 8$.

Remarks: More generally, suppose that $x[n] = 0$ for $n < 0$ and for $n \geq L$. As long as the DFT size N is larger than or equal to L, i.e. $N \geq L$, the results of this section apply. In particular:

- One can reconstruct $x[n]$ entirely from the DFT samples $X[k]$
- Also, from the theorem: $X[k] = X(\omega_k)$ at $\omega_k = 2\pi k/N$.

Increasing the value of N beyond the minimum required value, i.e. $N = L$, is called *zero-padding*:

- $\{x[0],...,x[L-1]\} \Rightarrow \{x[0],...,x[L-1],0,...,0\}$
- The DFT points obtained give a nicer graph of the underlying DTFT because $\Delta\omega_k = 2\pi/N$ is smaller.
- However, no new information about the original signal is introduced by increasing N in this way.

6.2.2 Periodic signals

Assumption: Suppose $x[n]$ is N-periodic, i.e. $x[n+N] = x[n]$.

Inverse DFT: In this case, $x[n]$ can be recovered entirely from its N-point DFT $X[k]$, $k = 0,1,...,N-1$. Let $\tilde{x}[n]$ denote the IDFT of $X[k]$, as defined in (6.14). For $n = 0,1,...,N-1$, the IDFT theorem yields: $x[n] = \tilde{x}[n]$. Since both $\tilde{x}[n]$ and $x[n]$ are known to be N-periodic, it follows that $x[n] = \tilde{x}[n]$ must also be true for $n < 0$ and for $n \geq N$. Therefore

$$x[n] = \tilde{x}[n], \quad \forall n \in \mathbb{Z} \tag{6.22}$$

Relationship between DFT and DTFT: Since the N-periodic signal $x[n]$ can be recovered completely from its N-point DFT $X[k]$, it should also be possible in theory to reconstruct the DTFT $X(\omega)$ from $X[k]$.

The situation is more complicated here because the DTFT of a periodic signal contains infinite impulses. The desired relationship is provided by the following theorem.

Theorem: Let $X(\omega)$ and $X[k]$ respectively denote the DTFT and N-point DFT of N-periodic signal $x[n]$. Then:

$$X(\omega) = \frac{2\pi}{N} \sum_{k=-\infty}^{\infty} X[k]\delta_a(\omega - \omega_k) \tag{6.23}$$

where $\delta_a(\omega)$ denotes an analog delta function centered at $\omega = 0$.

Remarks: According to the Theorem, one can recover the DTFT $X(\omega)$ from the N-point DFT $X[k]$. $X(\omega)$ is made up of a periodic train of infinite impulses in the ω domain (i.e. line spectrum). The amplitude of the impulse at frequency $\omega_k = 2\pi k/N$ is given by $2\pi X[k]/N$

Proof of Theorem (optional):

$$\begin{aligned}
X(\omega) &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \\
&= \sum_{n=-\infty}^{\infty} \left\{ \frac{1}{N} \sum_{k=0}^{N-1} X[k]e^{j\omega_k n} \right\} e^{-j\omega n} \\
&= \frac{1}{N} \sum_{k=0}^{N-1} X[k] \sum_{n=-\infty}^{\infty} e^{j\omega_k n} e^{-j\omega n} \\
&= \frac{1}{N} \sum_{k=0}^{N-1} X[k]\,\mathrm{DTFT}\{e^{j\omega_k n}\} \\
&= \frac{1}{N} \sum_{k=0}^{N-1} X[k]\, 2\pi \sum_{r=-\infty}^{\infty} \delta_a(\omega - \omega_k - 2\pi r) \\
&= \frac{2\pi}{N} \sum_{r=-\infty}^{\infty} \sum_{k=0}^{N-1} X[k]\,\delta_a(\omega - \frac{2\pi}{N}(k + rN))
\end{aligned} \tag{6.24}$$

Realizing that $X[k]$ is periodic with period N, and that $\sum_{r=-\infty}^{\infty} \sum_{k=0}^{N-1} f(k+rN)$ is equivalent to $\sum_{k=-\infty}^{\infty} f(k)$ we finally have:

$$\begin{aligned}
X(\omega) &= \frac{2\pi}{N} \sum_{r=-\infty}^{\infty} \sum_{k=0}^{N-1} X[k+rN]\,\delta_a(\omega - \frac{2\pi}{N}(k+rN)) \\
&= \frac{2\pi}{N} \sum_{k=-\infty}^{\infty} X[k]\,\delta_a(\omega - \frac{2\pi k}{N}) \quad \square
\end{aligned}$$

Remarks on the Discrete Fourier series: In the special case when $x[n]$ is N-periodic, i.e. $x[n+N] = x[n]$, the DFT admits a Fourier series interpretation. Indeed, the IDFT provides an expansion of $x[n]$ as a sum of

harmonically related complex exponential signals $e^{j\omega_k n}$:

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j\omega_k n}, \quad n \in \mathbb{Z} \tag{6.25}$$

In some textbooks (e.g. [10]), this expansion is called discrete Fourier series (DFS). The DFS coefficients $X[k]$ are identical to the DFT:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n}, \quad n \in \mathbb{Z} \tag{6.26}$$

In these notes, we treat the DFS as a special case of the DFT, corresponding to the situation when $x[n]$ is N-periodic.

6.3 Signal reconstruction via DTFT sampling

Introduction:

Let $X(\omega)$ be the DTFT of signal $x[n]$, $n \in \mathbb{Z}$, that is:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \quad \omega \in \mathbb{R}.$$

Consider the sampled values of $X(\omega)$ at uniformly spaced frequencies $\omega_k = 2\pi k/N$ for $k = 0,\ldots,N-1$. Suppose we compute the IDFT of the samples $X(\omega_k)$:

$$\hat{x}[n] = \text{IDFT}\{X(\omega_k)\} = \frac{1}{N}\sum_{k=0}^{N-1} X(\omega_k)e^{j\omega_k n} \tag{6.27}$$

What is the relationship between the original signal $x[n]$ and the reconstructed sequence $\hat{x}[n]$?

Note that $\hat{x}[n]$ is N-periodic, while $x[n]$ may not be. Even for $n = 0,\ldots,N-1$, there is no reason for $\hat{x}[n]$ to be equal to $x[n]$. Indeed, the IDFT theorem does not apply since $X(\omega_k) \neq \text{DFT}_N\{x[n]\}$.

The answer to the above question turns out to be very important when using DFT to compute linear convolution...

Theorem

$$\boxed{\hat{x}[n] = \text{IDFT}\{X(\omega_k)\} = \sum_{r=-\infty}^{\infty} x[n-rN]} \tag{6.28}$$

Proof:

$$\begin{aligned}\hat{x}[n] &= \frac{1}{N}\sum_{k=0}^{N-1} X(\omega_k)e^{j\omega_k n} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\left\{\sum_{l=-\infty}^{\infty} x[l]e^{-j\omega_k l}\right\}e^{j\omega_k n} \\ &= \sum_{l=-\infty}^{\infty} x[l]\left\{\frac{1}{N}\sum_{k=0}^{N-1} e^{j2\pi(n-l)k/N}\right\} \end{aligned} \tag{6.29}$$

From (6.11), we note that the bracketed quantity in (6.29) is equal to 1 if $n-l=rN$, i.e. $l=n-rN$ (r integer) and is equal to 0 otherwise. Therefore

$$\hat{x}[n] = \sum_{r=-\infty}^{\infty} x[n-rN] \quad \square \tag{6.30}$$

Interpretation

$\hat{x}[n]$ is an infinite sum of the sequences $x[n-rN]$, $r\in\mathbb{Z}$. Each of these sequences $x[n-rN]$ is a shifted version of $x[n]$ by an integer multiple of N. Depending on whether or not these shifted sequences overlap, we distinguish two important cases.

- *Time limited signal:* Suppose $x[n]=0$ for $n<0$ and for $n\geq N$. Then, there is no temporal overlap of the sequences $x[n-rN]$. We can recover $x[n]$ exactly from one period of $\hat{x}[n]$:

$$x[n] = \begin{cases}\hat{x}[n] & n=0,1,...,N-1, \\ 0 & \text{otherwise.}\end{cases} \tag{6.31}$$

  This is consistent with the results of Section 6.2.1. Figure 6.2 shows an illustration of this reconstruction by IDFT.

- *Non time-limited signal:* Suppose that $x[n]\neq 0$ for some $n<0$ or $n\geq N$. Then, the sequences $x[n-rN]$ for different values of r will overlap in the time-domain. In this case, it is not true that $\hat{x}[n]=x[n]$ for all $0\leq n\leq N-1$. We refer to this phenomenon as temporal aliasing, as illustrated in Figure 6.3.

6.4 Properties of the DFT

Notations:

In this section, we assume that the signals $x[n]$ and $y[n]$ are defined over $0\leq n\leq N-1$. Unless explicitly stated, we make no special assumption about the signal values outside this range. We denote the N-point DFT of $x[n]$ and $y[n]$ by $X[k]$ and $Y[k]$:

$$\begin{aligned} x[n] &\overset{\text{DFT}_N}{\longleftrightarrow} X[k] \\ y[n] &\overset{\text{DFT}_N}{\longleftrightarrow} Y[k] \end{aligned}$$

We view $X[k]$ and $Y[k]$ as N-periodic sequences, defined for all $k\in\mathbb{Z}$

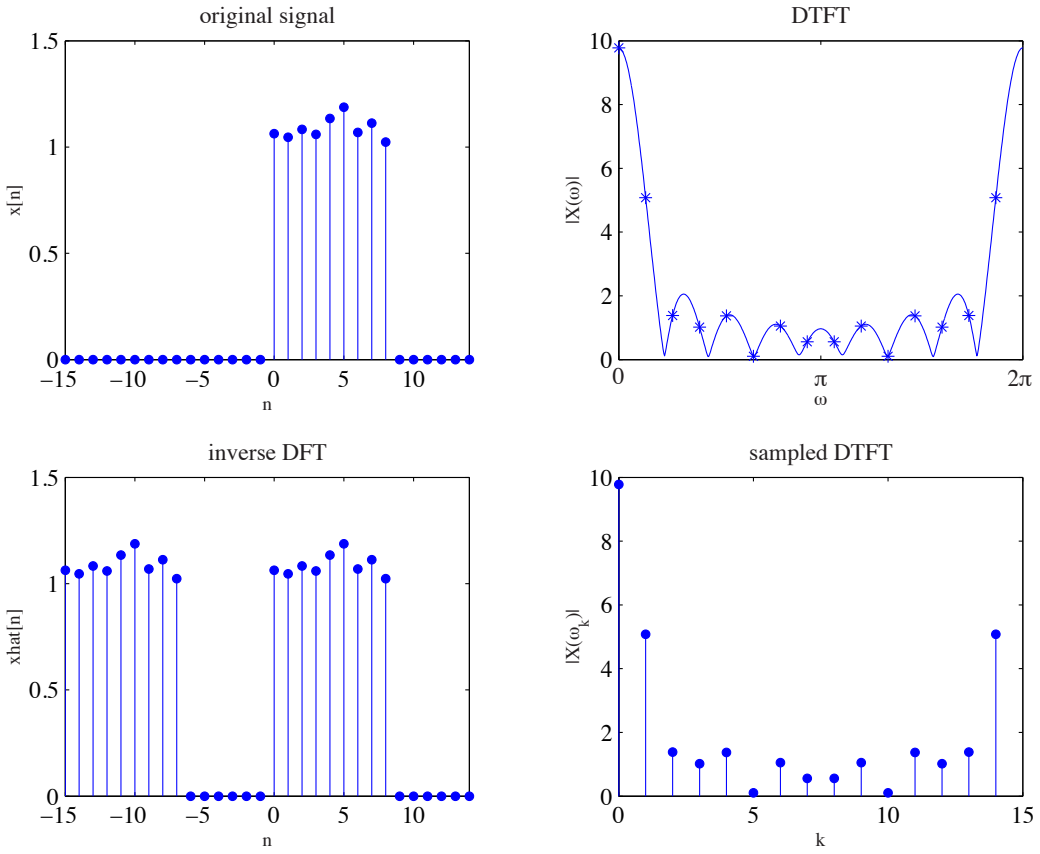


Fig. 6.2 Illustration of reconstruction of a signal from samples of its DTFT. Top left: the original signal $x[n]$ is time limited. Top right, the original DTFT $X(\omega)$, from which 15 samples are taken. Bottom right: the equivalent impulse spectrum corresponds by IDFT to a 15-periodic sequence $\hat{x}[n]$ shown on the bottom left. Since the original sequence is zero outside $0 \leq n \leq 14$, there is no overlap between replicas of the original signal in time.

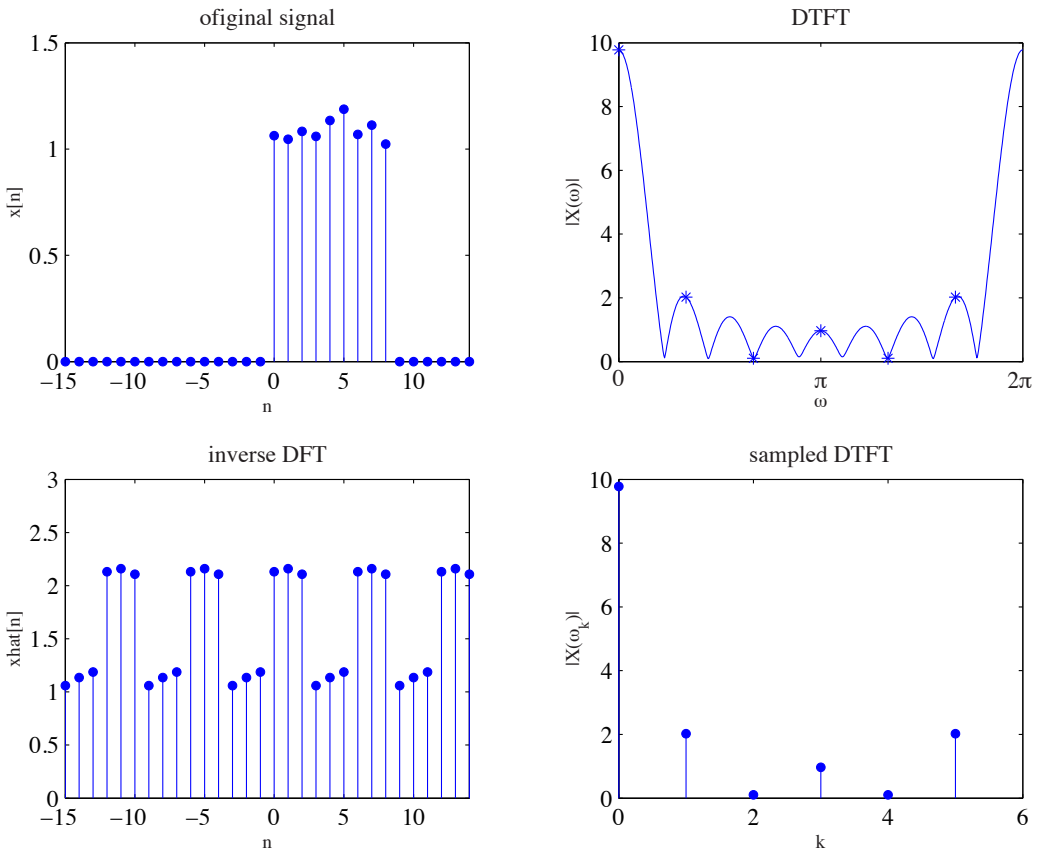


Fig. 6.3 Illustration of reconstruction of a signal from samples of its DTFT. Top left: the original signal $x[n]$ is time limited. Top right, the original DTFT $X(\omega)$, from which 6 samples are taken. Bottom right: the equivalent impulse spectrum corresponds by IDFT to a 6-periodic sequence $\hat{x}[n]$ shown on the bottom left. Since the original sequence is not zero outside $0 \leq n \leq 5$, there is overlap between replicas of the original signal in time, and thus aliasing.

Modulo N operation:

Any integer $n \in \mathbb{Z}$ can be expressed uniquely as $n = k + rN$ where $k \in \{0, 1, ..., N-1\}$ and $r \in \mathbb{Z}$. We define

$$(n)_N = n \text{ modulo } N \triangleq k. \tag{6.32}$$

For example: $(11)_8 = 3$, $(-1)_6 = 5$.

Note that the sequence defined by $x[(n)_N]$ is an N-periodic sequence made of repetitions of the values of $x[n]$ between 0 and $N-1$, i.e.

$$x[(n)_N] = \sum_{r=-\infty}^{\infty} \xi[n - rN] \tag{6.33}$$

where $\xi[n] = x[n](u[n] - u[n-N])$.

6.4.1 Time reversal and complex conjugation

Circular time reversal: Given a sequence $x[n]$, $0 \le n \le N-1$, we define its circular reversal (CR) as follows:

$$\text{CR}\{x[n]\} = x[(-n)_N], \quad 0 \le n \le N-1 \tag{6.34}$$

Example 6.2:

▶ Let $x[n] = 6 - n$ for $n = 0, 1, .., 5$:

n	0	1	2	3	4	5
$x[n]$	6	5	4	3	2	1

Then, for $N = 6$, we have $\text{CR}\{x[n]\} = x[(-n)_6]$:

n	0	1	2	3	4	5
$(-n)_6$	0	5	4	3	2	1
$x[(-n)_6]$	6	1	2	3	4	5

This is illustrated Figure 6.4. ◀

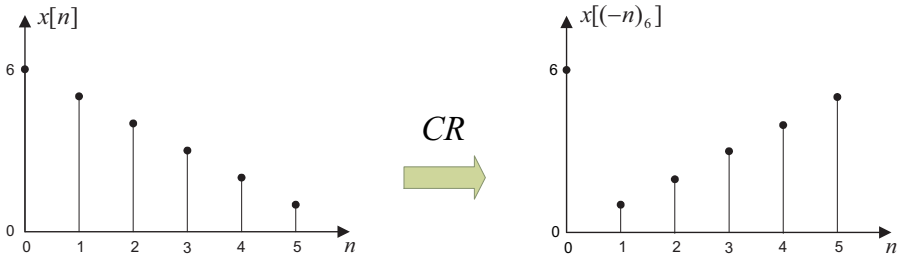


Fig. 6.4 Illustration of circular time reversal

Interpretation: Circular reversal can be seen as an operation on the set of signal samples $x[0],\ldots,x[N-1]$:

- $x[0]$ is left unchanged, whereas
- for $k=1$ to $N-1$, samples $x[k]$ and $x[N-k]$ are exchanged.

One can also see the circular reversal as an operation consisting in:

- periodizing the samples of $x[n], 0 \leq n \leq N-1$ with period N;
- time-reversing the periodized sequence;
- Keeping only the samples between 0 and $N-1$.

Figure 6.5 gives a graphical illustration of this process.

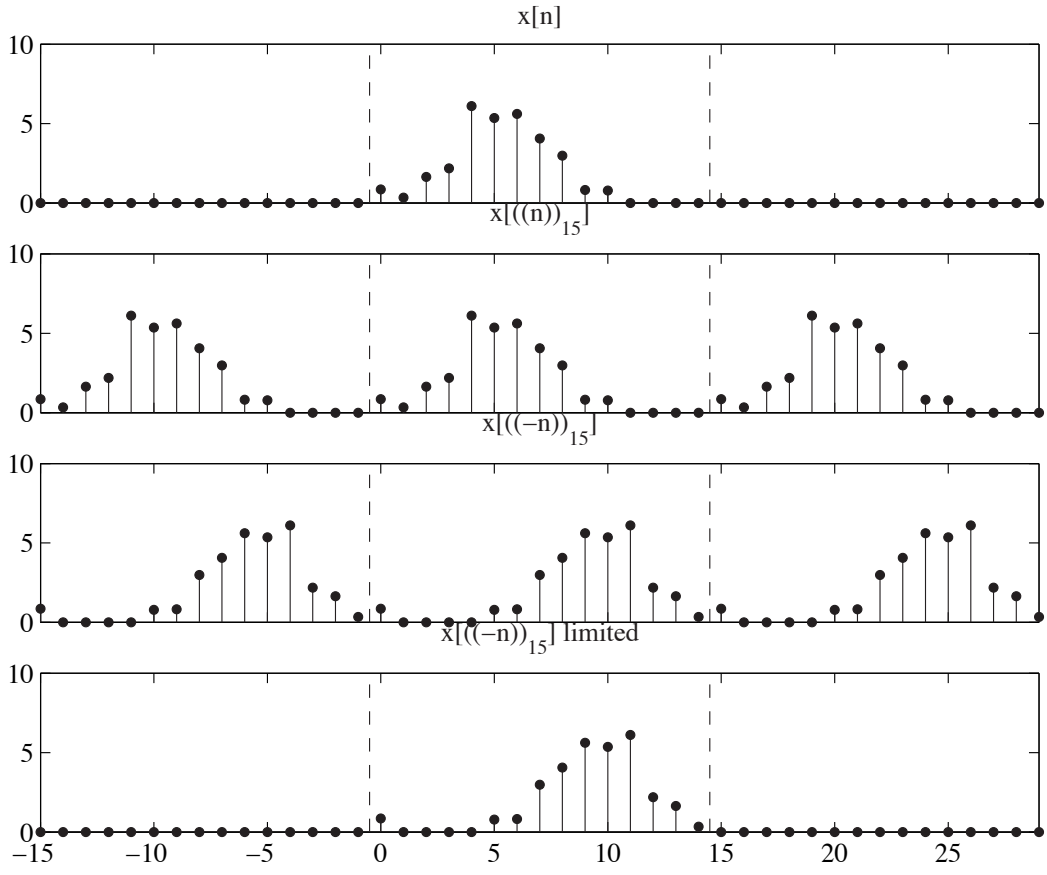


Fig. 6.5 Graphical illustration of circular reversal, with $N=15$.

Property:

$$x[(-n)_N] \overset{\text{DFT}_N}{\longleftrightarrow} X[-k] \tag{6.35}$$
$$x^*[n] \overset{\text{DFT}_N}{\longleftrightarrow} X^*[-k] \tag{6.36}$$
$$x^*[(-n)_N] \overset{\text{DFT}_N}{\longleftrightarrow} X^*[k] \tag{6.37}$$

Remarks:

- Since $X[k]$ is N-periodic, we note that $X[-k] = X[(-k)_N]$. For example, if $N = 6$, then $X[-1] = X[5]$ in (6.35).
- Property (6.36) leads to the following conclusion for real-valued signals:

$$x[n] = x^*[n] \Longleftrightarrow X[k] = X^*[-k] \tag{6.38}$$

  or equivalently, iff $|X[k]| = |X[-k]|$ and $\measuredangle X[k] = -\measuredangle X[-k]$.
- Thus, for real-valued signals:
  - $X[0]$ is real;
  - if N is even: $X[N/2]$ is real;
  - $X[N-k] = X^*[k]$ for $1 \leq k \leq N-1$.

  Figure 6.6 illustrates this with a graphical example.

6.4.2 Duality

Property:

$$X[n] \overset{\text{DFT}_N}{\longleftrightarrow} Nx[(-k)_N] \tag{6.39}$$

Proof: First note that since $(-k)_N = -k + rN$, for some integer r, we have

$$e^{-j2\pi kn/N} = e^{j2\pi n(-k)_N/N} \tag{6.40}$$

Using this identity and recalling the definition of the IDFT:

$$\begin{aligned} \sum_{n=0}^{N-1} X[n]e^{-j2\pi nk/N} &= \sum_{n=0}^{N-1} X[n]e^{j2\pi n(-k)_N/N} \\ &= N \times \text{IDFT}\{X[n]\} \text{ at index value } (-k)_N \\ &= Nx[(-k)_N] \quad \square \end{aligned}$$

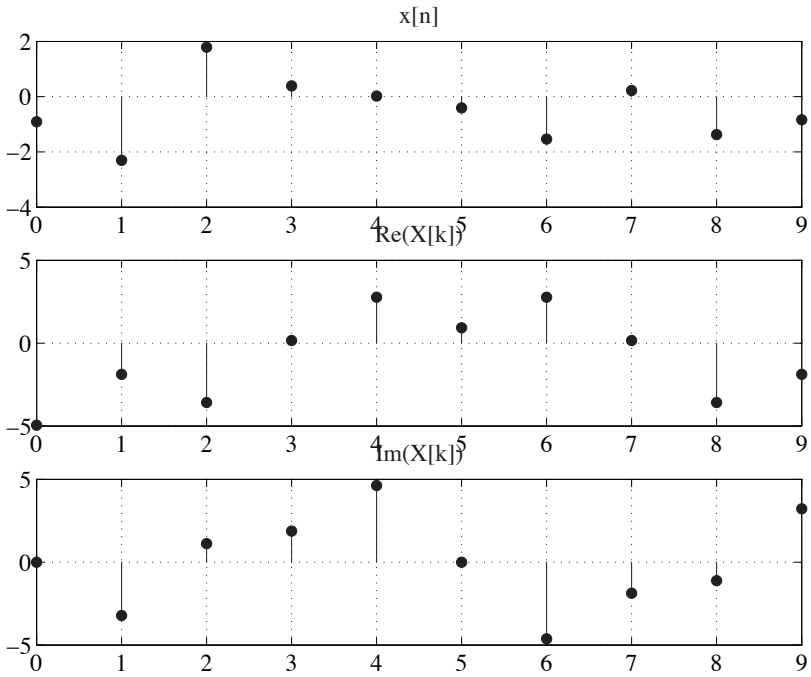


Fig. 6.6 Illustration of the symmetry properties of the DFT of a real-values sequence ($N = 10$).

6.4.3 Linearity

Property:

$$ax[n]+by[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad aX[k]+bY[k] \tag{6.41}$$

6.4.4 Even and odd decomposition

Conjugate symmetric components: The even and odd conjugate symmetric components of finite sequence $x[n], 0 \leq n \leq N-1$, are defined as:

$$x_{e,N}[n] \triangleq \frac{1}{2}(x[n]+x^*[(-n)_N]) \tag{6.42}$$

$$x_{o,N}[n] \triangleq \frac{1}{2}(x[n]-x^*[(-n)_N]) \tag{6.43}$$

In the case of a N-periodic sequence, defined for $n \in \mathbb{Z}$, the modulo N operation can be omitted. Accordingly, we define

$$X_e[k] \triangleq \frac{1}{2}(X[k]+X^*[-k]) \tag{6.44}$$

$$X_o[k] \triangleq \frac{1}{2}(X[k]-X^*[-k]) \tag{6.45}$$

Note that, e.g. $x_{e,N}[0]$ is real, whereas $x_{e,N}[N-n] = x_{e,N}[n], n = 1,\ldots,N-1$.

Property:

$$\text{Re}\{x[n]\} \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad X_e[k] \tag{6.46}$$

$$j\text{Im}\{x[n]\} \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad X_o[k] \tag{6.47}$$

$$x_{e,N}[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad \text{Re}\{X[k]\} \tag{6.48}$$

$$x_{o,N}[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad j\text{Im}\{X[k]\} \tag{6.49}$$

6.4.5 Circular shift

Definition: Given a sequence $x[n]$ defined over the interval $0 \leq n \leq N-1$, we define its circular shift by k samples as follows:

$$\text{CS}_k\{x[n]\} = x[(n-k)_N], \quad 0 \leq n \leq N-1 \tag{6.50}$$

Example 6.3:

▶ Let $x[n] = 6-n$ for $n = 0,1,..,5$.

n	0	1	2	3	4	5
$x[n]$	6	5	4	3	2	1

For $N = 6$, we have $\mathrm{CS}_k\{x[n]\} = x[(n-k)_6]$ In particular, for $k = 2$:

n	0	1	2	3	4	5
$(n-2)_6$	4	5	0	1	2	3
$x[(n-2)_6]$	2	1	6	5	4	3

This is illustrated Figure 6.7. ◀

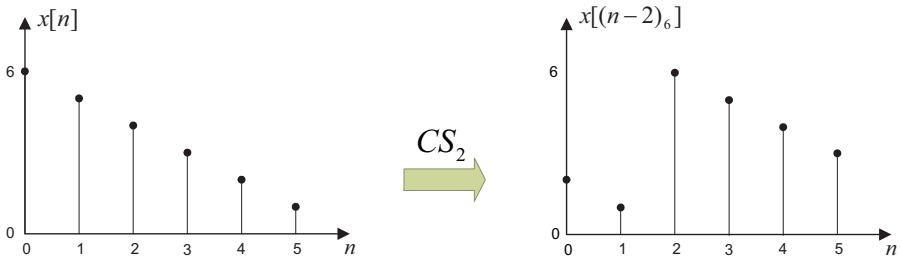


Fig. 6.7 Illustration of circular time reversal

Interpretation: Circular shift can be seen as an operation on the set of signal samples $x[n]$ ($n \in \{0,\ldots,N-1\}$) in which:

- signal samples $x[n]$ are shifted as in a conventional shift;
- any signal sample leaving the interval $0 \le n \le N-1$ from one end reenters by the other end.

Alternatively, circular shift may be interpreted as follows:

- periodizing the samples of $x[n], 0 \le n \le N-1$ with period N;
- delaying the periodized sequence by k samples;
- keeping only the samples between 0 and $N-1$.

Figure 6.8 gives a graphical illustration of this process. It is clearly seen from this figure that a circular shift by k samples amounts to taking the last k samples in the window of time from 0 to $N-1$, and placing these at the beginning of the window, shifting the rest of the samples to the right.

Circular Shift Property:

$$x[(n-m)_N] \quad \overset{\mathrm{DFT}_N}{\longleftrightarrow} \quad e^{-j2\pi mk/N}X[k]$$

Proof: To simplify the notations, introduce $\omega_k = 2\pi k/N$.

$$\begin{aligned}\mathrm{DFT}_N\{x[(n-m)_N]\} &= \sum_{n=0}^{N-1} x[(n-m)_N]e^{-j\omega_k n} \\ &= e^{-j\omega_k m}\sum_{n=-m}^{N-1-m} x[(n)_N]e^{-j\omega_k n}\end{aligned}$$

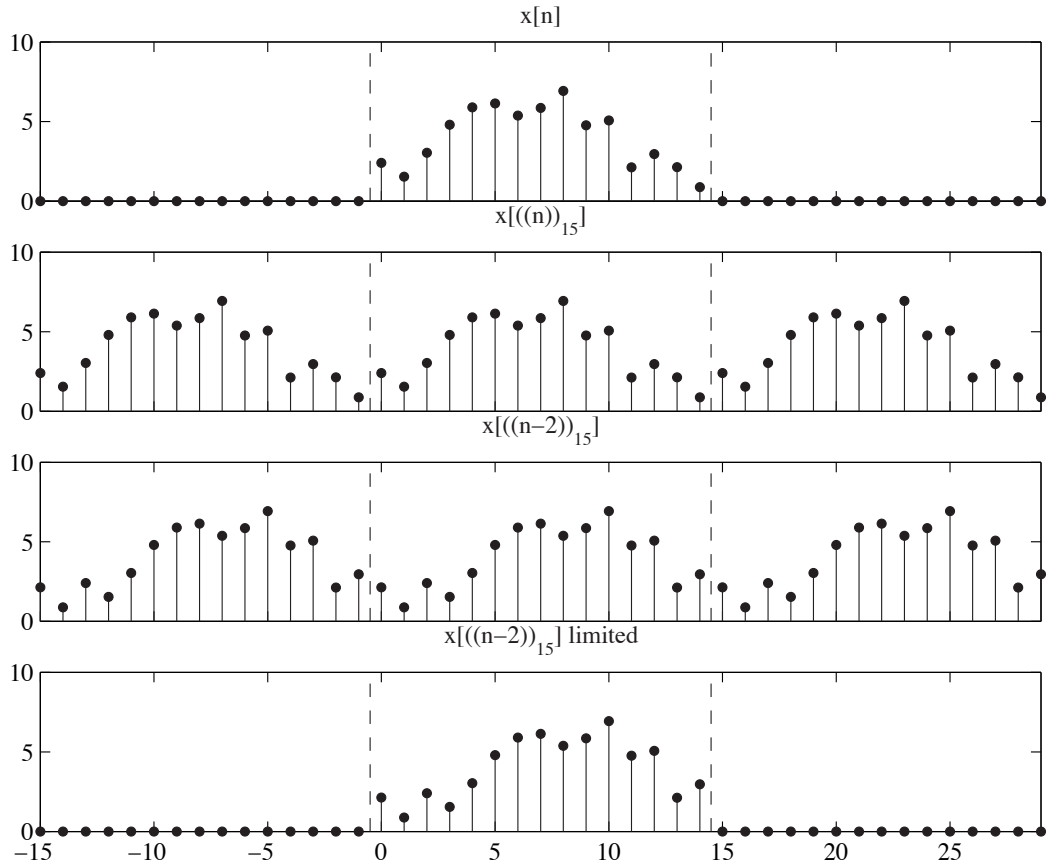


Fig. 6.8 Graphical illustration of circular shift, with $N = 15$ and $k = 2$.

Observe that the expression to the right of $\sum_{n=-m}^{N-1-m}$ is N-periodic in the variable n. Thus, we may replace that summation by $\sum_{n=0}^{N-1}$ without affecting the result (in both cases, we sum over one complete period)

$$\begin{aligned} \text{DFT}_N\{x[(n-m)_N]\} &= e^{-j\omega_k m}\sum_{n=0}^{N-1} x[(n)_N]e^{-j\omega_k n} \\ &= e^{-j\omega_k m}\sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} = e^{-j\omega_k m}X[k] \quad \square \end{aligned}$$

Frequency shift Property:

$$e^{j2\pi mn/N}x[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad X[k-m]$$

Remark: Since the DFT $X[k]$ is already periodic, the modulo N operation is not needed here, that is: $X[(k-m)_N] = X[k-m]$

6.4.6 Circular convolution

Definition: Let $x[n]$ and $y[n]$ be two sequences defined over the interval $0 \le n \le N-1$. The N-point circular convolution of $x[n]$ and $y[n]$ is given by

$$x[n] \circledast y[n] \triangleq \sum_{m=0}^{N-1} x[m]y[(n-m)_N], \quad 0 \le n \le N-1 \tag{6.51}$$

Remarks: This is conceptually similar to the linear convolution of Chapter 2 (i.e. $\sum_{n=\infty}^{\infty} = x[m]y[n-m]$) except for two important differences:

- the sum is finite, running from 0 to $N-1$
- circular shift is used instead of a linear shift.

The use of $(n-m)_N$ ensures that the argument of $y[\,]$ remains in the range $0, 1, ..., N-1$.

Circular Convolution Property:

$$x[n] \circledast y[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad X[k]Y[k] \tag{6.52}$$

Multiplication Property:

$$x[n]y[n] \quad \overset{\text{DFT}_N}{\longleftrightarrow} \quad \frac{1}{N}X[k] \circledast Y[k] \tag{6.53}$$

6.4.7 Other properties

Plancherel's relation:

$$\sum_{n=0}^{N-1} x[n]y^*[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]Y^*[k] \tag{6.54}$$

Parseval's relation:

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 \tag{6.55}$$

Remarks:

- Define signal vectors
$$\mathbf{x} = [x[0], \ldots, x[N-1]], \quad \mathbf{y} = [y[0], \ldots, y[N-1]]$$
Then, the LHS summation in (6.54) may be interpreted as the inner product between signal vectors $\mathbf{x}$ and $\mathbf{y}$ in vector space $\mathbb{C}^N$. Similarly, up to scaling factor $1/N$, the RHS of (6.54)may be interpreted as the inner product between the DFT vectors
$$\mathbf{X} = [X[0], \ldots, Y[N-1]], \quad \mathbf{Y} = [Y[0], \ldots, Y[N-1]]$$
Identity (6.54) is therefore equivalent to the statement that the DFT operation preserves the inner-product between time-domain vectors.
- Eq. (6.55) is a special case of (6.54) with $y[n] = x[n]$. It allows the computation of the energy of the signal samples $x[n]$ ($n = 0, \ldots, N-1$) directly from the DFT samples $X[k]$.

6.5 Relation between linear and circular convolutions

Introduction:

The circular convolution $\circledast$ admits a fast (i.e. very low complexity) realization via the so-called fast Fourier transform (FFT) algorithms. If the linear and circular convolution were equivalent, it would be possible to evaluate the linear convolution efficiently as well via the FFT.

The problem, then, is the following: under what conditions are the linear and circular convolutions equivalent?

In this section, we investigate the general relationship between the linear and circular convolutions. In particular, we show that both types of convolution are equivalent if

- the signals of interest have finite length,
- and the DFT size is sufficiently large.

Linear convolution:

Time domain expression:

$$y_l[n] = x_1[n] * x_2[n] = \sum_{k=-\infty}^{\infty} x_1[k] x_2[n-k], \quad n \in \mathbb{Z} \tag{6.56}$$

Frequency domain representation via DTFT:

$$Y_l(\omega) = X_1(\omega) X_2(\omega), \quad \omega \in [0, 2\pi] \tag{6.57}$$

Circular convolution:

Time domain expression:

$$y_c[n] = x_1[n] \circledast x_2[n] = \sum_{k=0}^{N-1} x_1[k] x_2[(n-k)_N], \quad 0 \le n \le N-1 \tag{6.58}$$

Frequency domain representation via N-point DFT:

$$Y_c[k] = X_1[k] X_2[k], \quad k \in \{0, 1, ..., N-1\} \tag{6.59}$$

Observation:

We say that the circular convolution (6.58) and the linear convolution (6.56) are equivalent if

$$y_l[n] = \begin{cases} y_c[n] & \text{if } 0 \le n < N \\ 0 & \text{otherwise} \end{cases} \tag{6.60}$$

Clearly, this can only be true in general if the signals $x_1[n]$ and $x_2[n]$ both have finite length.

Finite length assumption:

We say that $x[n]$ is time limited (TL) to $0 \le n < N$ iff $x[n] = 0$ for $n < 0$ and for $n \ge N$.

Suppose that $x_1[n]$ and $x_2[n]$ are TL to $0 \le n < N_1$ and $0 \le n < N_2$, respectively. Then, the linear convolution $y_l[n]$ in (6.56) is TL to $0 \le n < N_3$, where $N_3 = N_1 + N_2 - 1$.

Example 6.4:

► Consider the TL signals

$$x_1[n] = \{\underline{1}, 1, 1, 1\} \quad x_2[n] = \{\underline{1}, 1/2, 1/2\}$$

where $N_1 = 3$ and $N_2 = 4$. A simple calculation yields

$$y_l[n] = \{1, 1.5, 2, 2, 1, .5\}$$

with $N_3 = N_1 + N_2 - 1 = 6$. This is illustrated in figure 6.9. ◄

Condition for equivalence:

Suppose that $x_1[n]$ and $x_2[n]$ are TL to $0 \le n < N_1$ and $0 \le n < N_2$.

Based on our previous discussion, a necessary condition for equivalence of $y_l[n]$ and $y_c[n]$ is that $N \ge N_1 + N_2 - 1$. We show below that this is a sufficient condition.

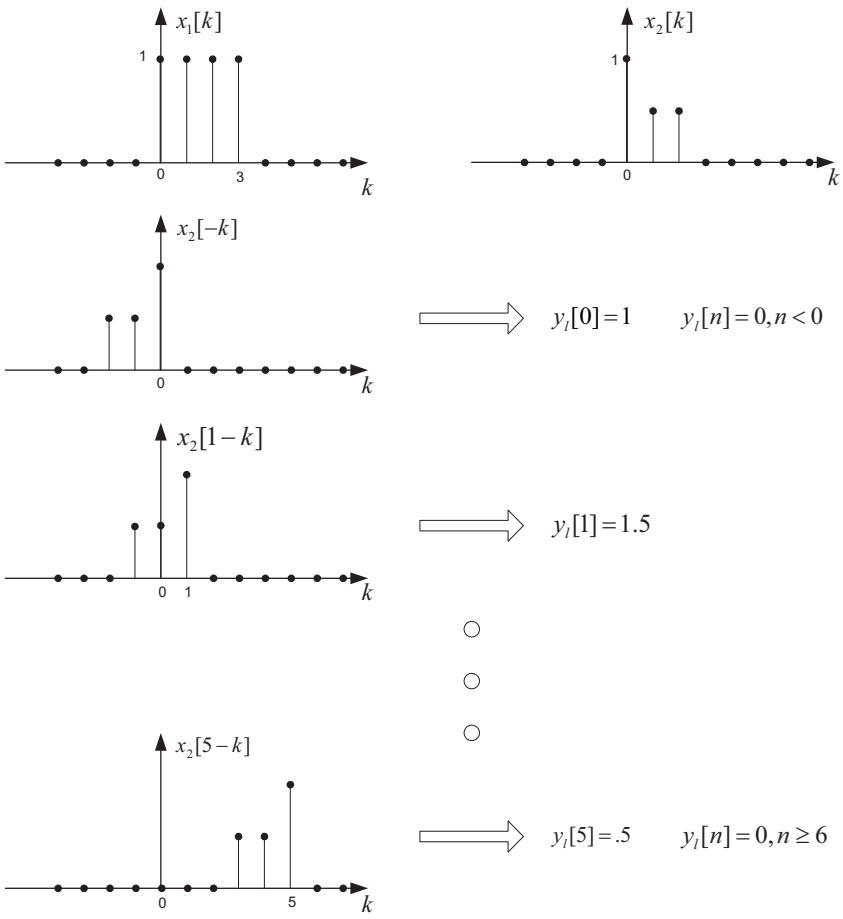


Fig. 6.9 Illustration of the sequences used in example 6.4. Top: $x_1[n]$ and $x_2[n]$; bottom: $y_l[n]$, their linear convolution.

Linear convolution:

$$\begin{aligned} y_l[n] &= \sum_{k=-\infty}^{\infty} x_1[k]x_2[n-k] \\ &= \sum_{k=0}^{n} x_1[k]x_2[n-k], \quad 0 \le n \le N-1 \end{aligned} \tag{6.61}$$

Circular convolution:

$$\begin{aligned} y_c[n] &= \sum_{k=0}^{N-1} x_1[k]x_2[(n-k)_N] \\ &= \sum_{k=0}^{n} x_1[k]x_2[n-k] + \sum_{k=n+1}^{N-1} x_1[k]x_2[N+n-k] \end{aligned} \tag{6.62}$$

If $N \ge N_2 + N_1 - 1$, it can be verified that the product $x_1[k]x_2[N+n-k]$ in the second summation on the RHS of (6.62) is always zero. Under this condition, it thus follows that $y_l[n] = y_c[n]$ for $0 \le n \le N-1$.

Conclusion: the linear and circular convolution are equivalent iff

$$\boxed{N \ge N_1 + N_2 - 1} \tag{6.63}$$

Example 6.5:

▶ Consider the TL signals

$$x_1[n] = \{\underline{1}, 1, 1, 1\} \quad x_2[n] = \{\underline{1}, 1/2, 1/2\}$$

where $N_1 = 3$ and $N_2 = 4$. First consider the circular convolution $y_c = x_1 \circledast x_2$ with $N = N_1 + N_2 - 1 = 6$, as illustrated in figure 6.10.

The result of this computation is

$$y_c[n] = \{\underline{1}, 1.5, 2, 2, 1, .5\}$$

which is identical to the result of the linear convolution in example 6.4.

Now consider the circular convolution $y_c = x_1 \circledast x_2$ with $N = 4$, as illustrated in figure 6.11.

The result of this computation is

$$y_c[n] = \{\underline{2}, 2, 2, 2\}$$

◀

Remarks:

When $N \ge N_1 + N_2 - 1$, the CTR operation has no effect on the convolution.

On the contrary, when $N = N_1 + N_2 - 1$, the CTR will affect the result of the convolution due to overlap between $x_1[k]$ and $x_2[N+n-k]$.

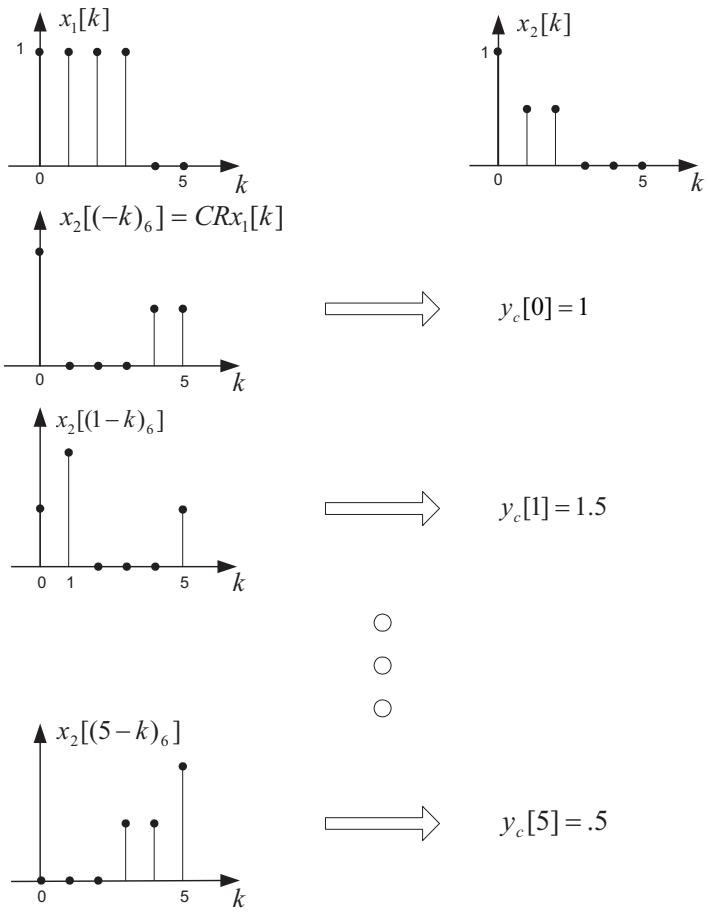


Fig. 6.10 Illustration of the sequences used in example 6.5. Top: $x_1[n]$ and $x_2[n]$; bottom: $y_c[n]$, their circular convolution with $N = 6$.

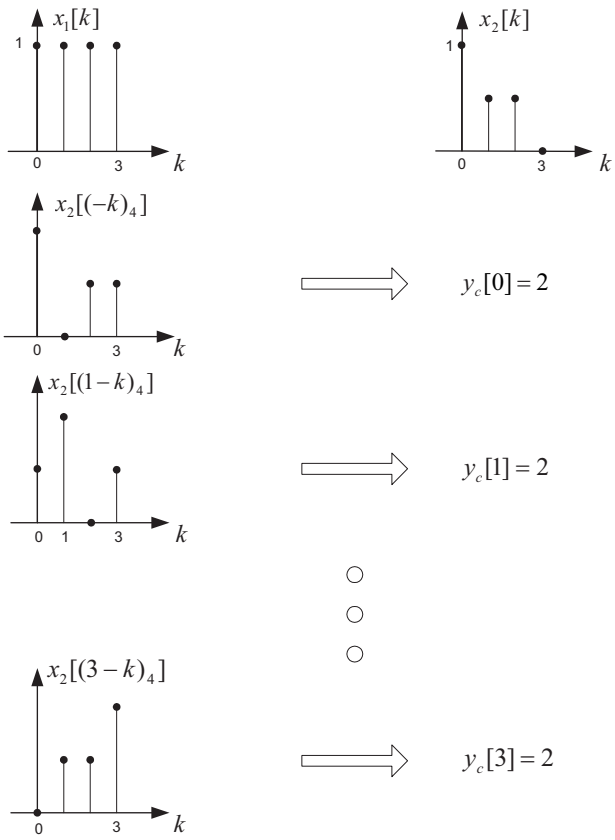


Fig. 6.11 Illustration of the sequences used in example 6.5. Top: $x_1[n]$ and $x_2[n]$; bottom: $y_c[n]$, their circular convolution with $N = 4$.

Relationship between $y_c[n]$ and $y_l[n]$:

Assume that $N \geq \max\{N_1, N_2\}$. Then the DFT of each of the two sequences $x_1[n]$ and $x_2[n]$ are samples of the corresponding DTFT:

$$\begin{aligned} N \geq N_1 &\implies X_1[k] = X_1(\omega_k), \quad \omega_k = 2\pi k/N \\ N \geq N_2 &\implies X_2[k] = X_2(\omega_k) \end{aligned}$$

The DFT of the circular convolution is just the product of DFT:

$$\begin{aligned} Y_c[k] &= X_1[k]X_2[k] \\ &= X_1(\omega_k)X_2(\omega_k) = Y_l(\omega_k) \end{aligned} \tag{6.64}$$

This shows that $Y_c[k]$ is also made of uniformly spaced samples of $Y_l(\omega)$, the DTFT of the linear convolution $y_l[n]$. Thus, the circular convolution $y_c[n]$ can be computed as the N-point IDFT of these frequency samples:

$$y_c[n] = \text{IDFT}_N\{Y_c[k]\} = \text{IDFT}_N\{Y_l(\omega_k)\}$$

Making use of (6.28), we have

$$y_c[n] = \sum_{r=-\infty}^{\infty} y_l[n - rN], \quad 0 \leq n \leq N-1, \tag{6.65}$$

so that the circular convolution is obtained by

- an N-periodic repetition of the linear convolution $y_l[n]$
- and a windowing to limit the nonzero samples to the instants 0 to $N-1$.

To get $y_c[n] = y_l[n]$ for $0 \leq n \leq N-1$, we must avoid temporal aliasing. We need: lenght of DFT $\geq$ length of $y_l[n]$, i.e.

$$N \geq N_3 = N_1 + N_2 - 1 \tag{6.66}$$

which is consistent with our earlier development. The following examples illustrate the above concepts:

Example 6.6:

▶ The application of (6.65) for $N < N_1 + N_2 - 1$ is illustrated in Figure 6.12. The result of a linear convolution $y_l[n]$ is illustrated (top) together with two shifted versions by $+N$ and $-N$ samples, respectively. These signals are added together to yield the circular convolution $y_c[n]$ (bottom) according to (6.65). In this case, $N = 11$ and $N_3 = N_1 + N_2 - 1 = 15$, so that there is time aliasing. Note that out of the $N = 11$ samples in the circular convolution $y_c[n]$, i.e. within the interval $0 \leq n \leq 10$, only the first $N_3 - N$ samples are affected by aliasing (as shown by the white dots), whereas the other samples are common to $y_l[n]$ and $y_c[n]$. ◀

Example 6.7:

▶ To convince ourselves of the validity of (6.65), consider again the sequences $x_1[n] = \{\underline{1}, 1, 1, 1\}$ and $x_2[n] = \{\underline{1}, .5, .5\}$ where $N_1 = 3$ and $N_2 = 4$. From Example 6.4, we know that

$$y_l[n] = \{\underline{1}, 1.5, 2, 2, 1, .5\}$$

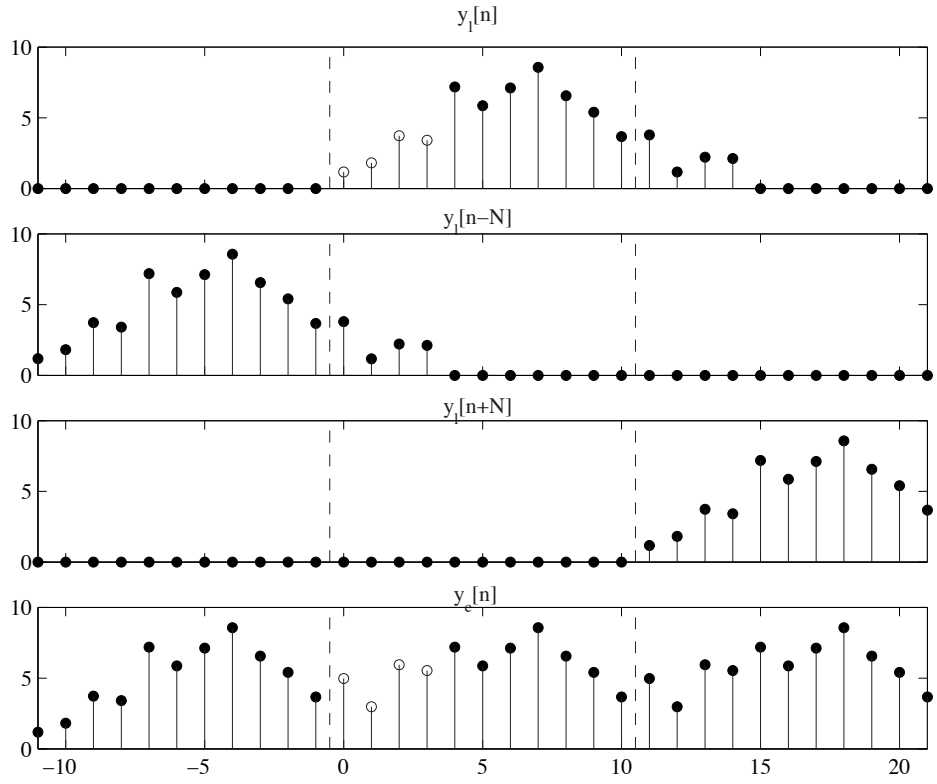


Fig. 6.12 Circular convolution seen as linear convolution plus time aliasing:

while from Example 6.5, the circular convolution for $N = 4$ yields

$$y_c[n] = \{\underline{2}, 2, 2, 2\}$$

The application of (6.65) is illustrated in Figure 6.13. It can be seen that the sample values of the circular

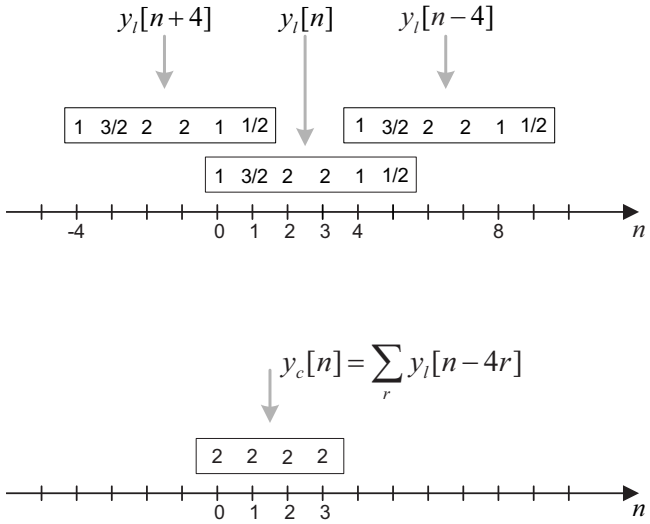


Fig. 6.13 Illustration of the circular convolution as a time-aliased verion of the linear convolution (see examepl 6.7).

convolution $y_c[n]$, as computed from (6.65), are identical to those obtained in Example 6.5. ◄

6.5.1 Linear convolution via DFT

We can summarize the way one can compute a linear convolution via DFT by the following steps:

- Suppose that $x_1[n]$ is TL to $0 \le n < N_1$ and $x_2[n]$ is TL to $0 \le n < N_2$
- Select DFT size $N \ge N_1 + N_2 - 1$ (usually, $N = 2^K$).
- For $i = 1, 2$, compute
$$X_i[k] = \text{DFT}_N\{x_i[n]\}, \quad 0 \le k \le N-1 \tag{6.67}$$
- Compute:
$$x_1[n] * x_2[n] = \begin{cases} \text{IDFT}_N\{X_1[k]X_2[k]\} & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{6.68}$$

Remarks: In practice, the DFT is computed via an FFT algorithm (more on this later). For large N, say $N \ge 30$, the computation of a linear convolution via FFT is more efficient than direct time-domain realization. The accompanying disadvantage is that it involves a processing delay, because the data must be buffered.

Example 6.8:

► Consider the sequences

$$x_1[n] = \{\underline{1}, 1, 1, 1\} \quad x_2[n] = \{\underline{1}, .5, .5\}$$

where $N_1 = 3$ and $N_2 = 4$. The DFT size must satisfy $N \geq N_1 + N_2 - 1 = 6$ for the circular and linear convolutions to be equivalent.

Let us first consider the case $N = 6$. The 6-point DFT of the *zero-padded* sequences $x_1[n]$ and $x_2[n]$ are:

$$\begin{aligned} X_1[k] &= \text{DFT}_6\{1,1,1,1,0,0\} \\ &= \{4, -1.7321j, 1, 01, +1.7321j\} \end{aligned}$$

$$\begin{aligned} X_1[k] &= \text{DFT}_6\{1,0.5,0.5,0,0,0\} \\ &= \{2, 1-0.866j, 1/2, 1, 1/2, 1+0.866j\} \end{aligned}$$

The product of the DFT yields

$$Y_c[k] = \{8, -1.5 - 1.7321j, 0.5, 0, 0.5, -1.5 - 1.7321j\}$$

Taking the 6-point IDFT, we obtain:

$$\begin{aligned} y_c[n] &= \text{IDFT}_6\{Y[k]\} \\ &= \{1, 1.5, 2, 2, 1, .5\} \end{aligned}$$

which is identical to $y_l[n]$ previously computed in Example 6.4. The student is invited to try this approach in the case $N = 4$.

◄

6.6 Problems

Problem 6.1: DFT of Periodic Sequence
Let $x[n]$ be aperiodic sequence with period N. Let $X[k]$ denote its N-point DFT. Let $Y[k]$ represent its $2N$-point DFT.

Find the relationship between $X[k]$ and $Y[k]$.

Problem 6.2: DFT of a complex exponential
Find the N-point DFT of $x[n] = e^{j2\pi pn/N}$.

Chapter 7

Digital processing of analog signals

As a motivation for Discrete-Time Signal Processing, we have stated in the introduction of this course that DTSP is a convenient and powerful approach for the processing of real-world analog signals. In this chapter, we will further study the theoretical and practical conditions that enable digital processing of analog signals.

Structural overview:

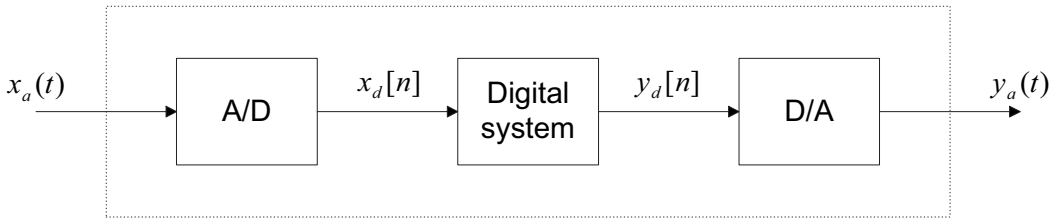


Fig. 7.1 The three main stages of digital processing of analog signals

Digital processing of analog signals consists of three main components, as illustrated in Figure 7.1:

- The A/D converter transforms the analog input signal $x_a(t)$ into a digital signal $x_d[n]$. This is done in two steps:
  - uniform sampling
  - quantization
- The digital system performs the desired operations on the digital signal $x_d[n]$ and produces a corresponding output $y_d[n]$ also in digital form.
- The D/A converter transforms the digital output $y_d[n]$ into an analog signal $y_a(t)$ suitable for interfacing with the outside world.

In the rest of this chapter, we will study the details of each of these building blocks.

7.1 Uniform sampling and the sampling theorem

Uniform (or periodic) sampling refers to the process in which an analog, or continuous-time (CT), signal $x_a(t), t \in \mathbb{R}$, is mapped into a discrete-time signal $x[n], n \in \mathbb{Z}$, according to the following rule:

$$x_a(t) \rightarrow x[n] \triangleq x_a(nT_s), \quad n \in \mathbb{Z} \tag{7.1}$$

where T_s is called the *sampling period*. Figure 7.2 gives a graphic illustration of the process.

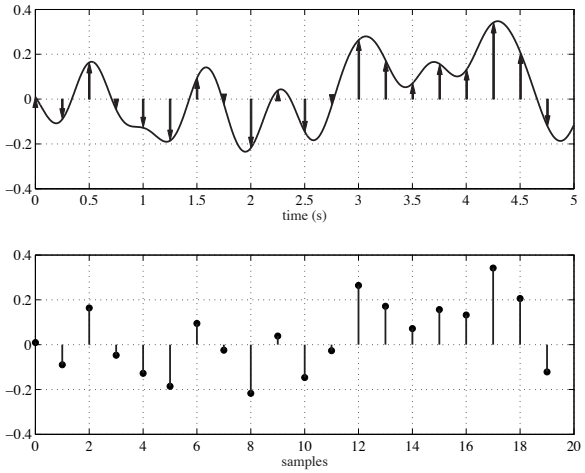


Fig. 7.2 Illustration of Uniform Sampling: top, the analog signal, where the sampling points $t = nT_s, n \in \mathbb{Z}$ are shown by arrows; bottom: the corresponding discrete-time sequence. Notice the difference in scale betwwen the x-axis of the two plots.

Based on the sampling period T_s, the *Sampling Frequency* can be defined as

$$F_s \triangleq 1/T_s, \text{ or} \tag{7.2}$$

$$\Omega_s \triangleq 2\pi F_s = 2\pi/T_s, \tag{7.3}$$

We shall also refer to uniform sampling as *ideal analog-to-discrete-time (A/DT) conversion*. We say ideal because the amplitude of $x[n]$ is not quantized, i.e. it may take any possible values within the set $\mathbb{R}$.

Ideal A/DT conversion is represented in block diagram form as follows:

In practice, uniform sampling is implemented with analog-to-digital (A/D) converters. These are non-ideal devices: only a finite set of possible values is allowed for the signal amplitudes $x[n]$ (see section 7.3).

It should be clear from the above presentation that, for an arbitrary continuous-time signal $x_a(t)$, information will be lost through the uniform sampling process $x_a(t) \rightarrow x[n] = x_a(nT_s)$.

This is illustrated by the following example.

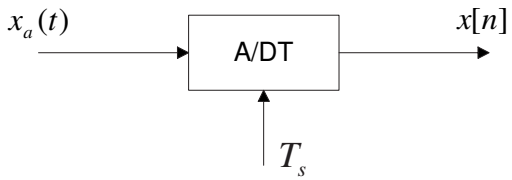


Fig. 7.3 Block diagram representation of ideal A/DT conversion.

Example 7.1: Information Loss in Sampling

► Consider the analog signals

$$\begin{aligned} x_{a1}(t) &= \cos(2\pi F_1 t), \quad F_1 = 100Hz \\ x_{a2}(t) &= \cos(2\pi F_2 t), \quad F_2 = 300Hz \end{aligned}$$

Uniform sampling at the rate $F_s = 200$Hz yields

$$\begin{aligned} x_1[n] = x_{c1}(nT_s) &= \cos(2\pi F_1 n/F_s) = \cos(\pi n) \\ x_2[n] = x_{c2}(nT_s) &= \cos(2\pi F_2 n/F_s) = \cos(3\pi n) \end{aligned}$$

Clearly,

$$x_1[n] = x_2[n]$$

This shows that uniform sampling is a non-invertible, many-to-one mapping. ◄

This example shows that it is not possible in general to recover the original analog signal $x_a(t)$ for all $t \in \mathbb{R}$, from the set of samples $x[n], n \in \mathbb{Z}$.

However, if we further assume that $x_a(t)$ is band-limited and select F_s to be sufficiently large, it is possible to recover $x_a(t)$ from its samples $x[n]$. This is the essence of the sampling theorem.

7.1.1 Frequency domain representation of uniform sampling

In this section, we first investigate the relationship between the Fourier transform (FT) of the analog signal $x_a(t)$ and the DTFT of the discrete-time signal $x[n] = x_a(nT_s)$. The sampling theorem will follow naturally from this relationship.

FT of analog signals: For an analog signal $x_a(t)$, the Fourier transform, denoted here as $X_a(\Omega), \Omega \in \mathbb{R}$, is defined by

$$X_a(\Omega) \triangleq \int_{-\infty}^{\infty} x_a(t) e^{-j\Omega t} dt \tag{7.4}$$

The corresponding inverse Fourier transform relationship is given by:

$$x_a(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X_a(\Omega) e^{j\Omega t} d\Omega \tag{7.5}$$

We assume that the student is familiar with the FT and its properties (if not, Signals and Systems should be reviewed).

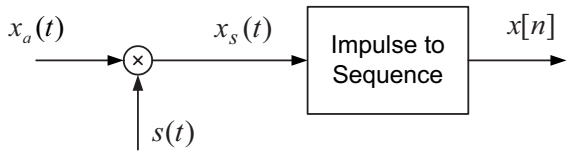


Fig. 7.4 Uniform sampling model.

A sampling model: It is convenient to represent uniform sampling as shown in Figure 7.4.
In this model:

- $s(t)$ is an impulse train, i.e.:
$$s(t) = \sum_{n=-\infty}^{\infty} \delta_a(t - nT_s) \tag{7.6}$$
- $x_s(t)$ is a pulse-amplitude modulated (PAM) waveform:
$$\begin{aligned} x_s(t) &= x_a(t)s(t) && (7.7)\\ &= \sum_{n=-\infty}^{\infty} x_a(nT_s)\delta_a(t - nT_s) && (7.8)\\ &= \sum_{n=-\infty}^{\infty} x[n]\delta_a(t - nT_s) && (7.9) \end{aligned}$$
- The sample value $x[n]$ is encoded in the amplitude of the pulse $\delta_a(t - nT_s)$. The information content of $x_s(t)$ and $x[n]$ are identical, i.e. one can be reconstructed from the other and vice versa.

FT/DTFT Relationship: Let $X(\omega)$ denote the DTFT of the samples $x[n] = x_a(nT_s)$, that is, $X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$. The following equalities hold:

$$\boxed{X(\omega)|_{\omega=\Omega T_s} = X_s(\Omega) = \frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_a(\Omega - k\Omega_s)} \tag{7.10}$$

Proof: The first equality is proved as follows:

$$\begin{aligned} X_s(\Omega) &= \int_{-\infty}^{\infty} x_s(t)e^{-j\Omega t}dt\\ &= \int_{-\infty}^{\infty} \{\sum_{n=-\infty}^{\infty} x[n]\delta_a(t - nT_s)\}e^{-j\Omega t}dt\\ &= \sum_{n=-\infty}^{\infty} x[n]\int_{-\infty}^{\infty} \delta_a(t - nT_s)e^{-j\Omega t}dt\\ &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega nT_s}\\ &= X(\omega) \text{ at } \omega = \Omega T_s && (7.11) \end{aligned}$$

Now consider the second equality: From ECSE 304, we recall that

$$\begin{aligned} S(\Omega) &= \text{FT}\{s(t)\} \\ &= \text{FT}\{\sum_{n=-\infty}^{\infty} \delta_a(t-nT_s)\} \\ &= \frac{2\pi}{T_s}\sum_{k=-\infty}^{\infty} \delta_a(\Omega - k\Omega_s) \end{aligned} \tag{7.12}$$

Finally, since $x_s(t) = x_a(t)s(t)$, it follows from the multiplicative property of the FT that

$$\begin{aligned} X_s(\Omega) &= \frac{1}{2\pi} X_a(\Omega) * S(\Omega) \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X_a(\Omega-\Omega')\,\frac{2\pi}{T_s}\sum_{k=-\infty}^{\infty}\delta_a(\Omega'-k\Omega_s)\,d\Omega' \\ &= \frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_a(\Omega - k\Omega_s)\ \square \end{aligned} \tag{7.13}$$

Remarks: The LHS of (7.10) is the DTFT of $x[n]$ evaluated at

$$\omega = \Omega T_s = \frac{\Omega}{F_s} = 2\pi\frac{F}{F_s} \tag{7.14}$$

We shall refer to $\frac{\Omega}{F_s}$ as the normalized frequency. The middle term of (7.10) is the FT of the amplitude modulated pulse train $x_s(t) = \sum_n x[n]\delta_a(t-nT_s)$. The RHS represents an infinite sum of scaled, shifted copies of $X_a(\Omega)$ by integer multiples of Ω_s.

Interpretation: Suppose $x_a(t)$ is a band-limited (BL) signal, i.e. $X_a(\Omega) = 0$ for $|\Omega| \geq \Omega_N$, where Ω_N represents a maximum frequency.

- Case 1: suppose $\Omega_s \geq 2\Omega_N$ (see Figure 7.5). Note that for different values of k, the spectral images $X_a(\Omega - k\Omega_s)$ do not overlap. Consequently, $X_a(\Omega)$ can be recovered from $X(\omega)$:

$$X_a(\Omega) = \begin{cases} T_s X(\Omega T_s) & |\Omega| \leq \Omega_s/2 \\ 0 & \text{otherwise} \end{cases} \tag{7.15}$$

- Case 2: suppose $\Omega_s < 2\Omega_N$ (see Figure 7.6). In this case, the different images $X_a(\Omega - k\Omega_s)$ do overlap. As a result, it is not possible to recover $X_a(\Omega)$ from $X(\omega)$.

The critical frequency $2\Omega_N$ (i.e. twice the maximum frequency) is called the *Nyquist rate* for uniform sampling. In case 2, i.e. $\Omega_s < 2\Omega_N$, the distortion resulting from spectral overlap is called *aliasing*. In case 1, i.e. $\Omega_s \geq 2\Omega_N$, the fact that $X_a(\Omega)$ can be recovered from $X(\omega)$ actually implies that $x_a(t)$ can be recovered from $x[n]$: this will be stated in the sampling theorem.

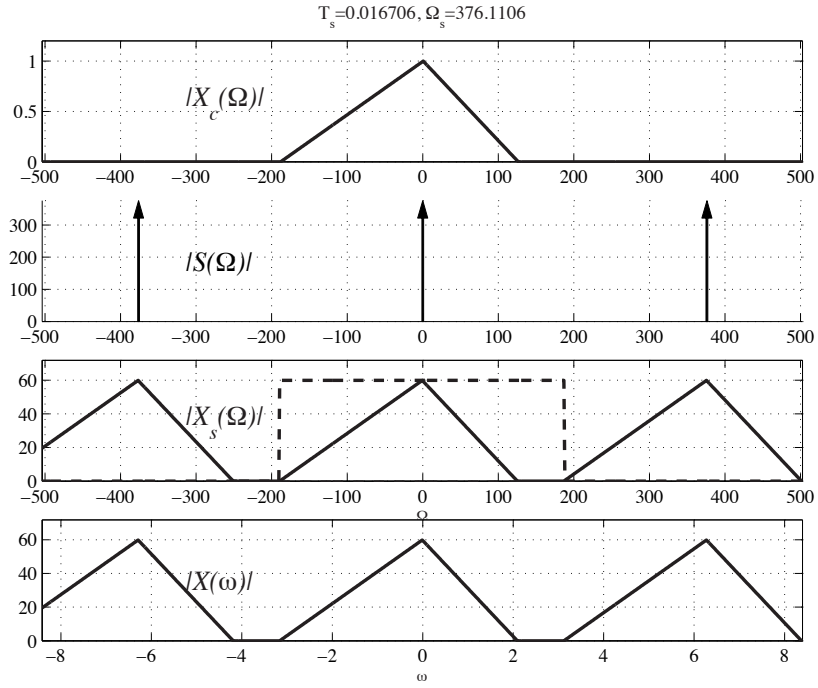


Fig. 7.5 Illustration of sampling in the frequency domain, when $\Omega_s > 2\Omega_N$

7.1.2 The sampling theorem

The following theorem is one of the cornerstones of discrete-time signal processing.

Sampling Theorem:

Suppose that $x_a(t)$ is Band-Limited to $|\Omega| < \Omega_N$. Provided the sampling frequency

$$\Omega_s = \frac{2\pi}{T_s} \geq 2\Omega_N \tag{7.16}$$

then $x_a(t)$ can be recovered from its samples $x[n] = x_a(nT_s)$ via

$$x_a(t) = \sum_{n=-\infty}^{\infty} x[n] h_r(t - nT_s), \qquad \text{any } t \in \mathbb{R} \tag{7.17}$$

where

$$h_r(t) \triangleq \frac{\sin(\pi t/T_s)}{\pi t/T_s} = \text{sinc}(t/T_s) \tag{7.18}$$

Proof: From our interpretation of the FT/DTFT relationship (7.10), we have seen under Case 1 that when $\Omega_s \geq 2\Omega_N$, it is possible to recover $X_a(\Omega)$ from the DTFT $X(\omega)$, or equivalently $X_s(\Omega)$, by means of (7.15). This operation simply amounts to keep the low-pass portion of $X_s(\Omega)$.

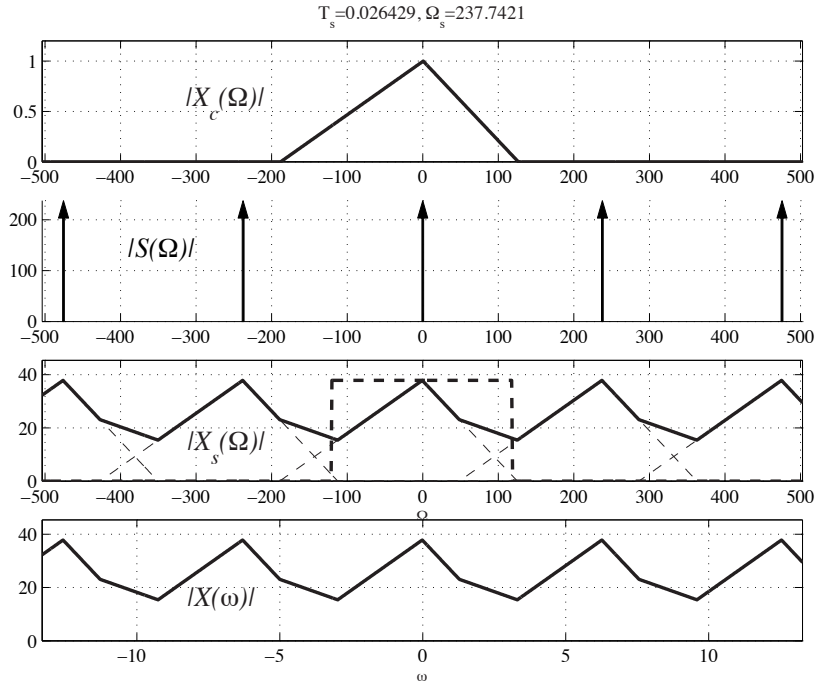


Fig. 7.6 Illustration of sampling in the frequency domain, when $\Omega_s < 2\Omega_N$

This can be done with an ideal continuous-time low-pass filter with gain T_s, defined as (see dashed line in Figure 7.5):

$$H_r(\Omega) = \begin{cases} T_s & |\Omega| \leq \Omega_s/2 \\ 0 & |\Omega| > \Omega_s/2 \end{cases} .$$

The corresponding impulse response is found by taking the inverse *continuous-time* Fourier Transform of $H_r(\Omega)$:

$$\begin{aligned} h_r(t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} H_r(\Omega)e^{j\Omega t}d\Omega = \frac{1}{2\pi}\int_{-\Omega_s/2}^{\Omega_s/2} T_s e^{j\Omega t}d\Omega \\ &= \frac{T_s}{2\pi}\frac{e^{j\Omega_s t/2} - e^{-j\Omega_s t/2}}{jt} \\ &= \frac{\sin(\pi t/T_s)}{\pi t/T_s}. \end{aligned}$$

Passing $x_s(t)$ through this impulse response $h_r(t)$ yields the desired signal $x_a(t)$, as the continuous-time

convolution:

$$\begin{aligned} x_a(t) &= x_s(t) * h_r(t) = \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta_a(\tau - nTs) x_a(nT_s) \frac{\sin(\pi(t-\tau)/T_s)}{\pi(t-\tau)/T_s} d\tau \\ &= \sum_{n=-\infty}^{\infty} x_a(nT_s) \int_{-\infty}^{\infty} \delta_a(\tau - nTs) \frac{\sin(\pi(t-\tau)/T_s)}{\pi(t-\tau)/T_s} d\tau \\ &= \sum_{n=-\infty}^{\infty} x_a(nT_s) \frac{\sin(\pi(t-nT_s)/T_s)}{\pi(t-nT_s)/T_s}, \end{aligned}$$

the last equality coming from the fact that $\int_{-\infty}^{\infty} \delta_a(t-t_0) f(t) dt = f(t_0)$. □

Interpretation:

The function $h_r(t)$ in (7.18) is called reconstruction filter. The general shape of $h_r(t)$ is shown in Figure 7.7. In particular, note that

$$h_r(mT_s) = \delta[m] = \begin{cases} 1 & m = 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.19}$$

Thus, in the special case $t = mT_s$, (7.17) yields

$$x_a(mT_s) = \sum_{n=-\infty}^{\infty} x[n] h_r((m-n)T_s) = x[m] \tag{7.20}$$

which is consistent with the definition of the samples $x[m]$. In the case $t \neq mT_s$, (7.17) provides an interpolation formula:

$$x_a(t) = \sum \text{ scaled and shifted copies of } \text{sinc}(t/T_s) \tag{7.21}$$

Equation (7.17) is often called the ideal *band-limited signal reconstruction* formula. Equivalently, it may be viewed as an ideal form of discrete-to-continuous (D/C) conversion. The reconstruction process is illustrated on the right of Figure 7.7.

7.2 Discrete-time processing of continuous-time signals

Now that we have seen under which conditions an analog signal can be converted to discrete-time without loss of information, we will develop in this section the conditions for the discrete-time implementation of a system to be equivalent to its continuous-time implementation.

Figure 7.8 shows the structure that we will consider for discrete-time processing of analog signals $x_s(t)$. This is an ideal form of processing in which the amplitudes of DT signals $x[n]$ and $y[n]$ are not constrained. In a practical digital signal processing system, these amplitudes would be quantized (see section 7.3).

The purpose of this section will be to answer the following question: Under what conditions is the DT structure of Figure 7.8 equivalent to an analog LTI system?

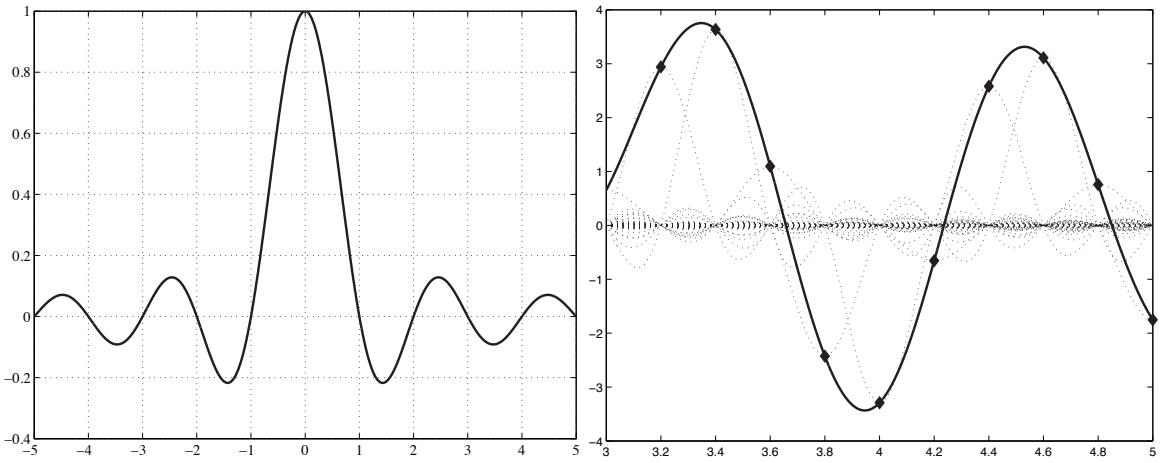


Fig. 7.7 Band-limited interpolation: Left, the general shape of the reconstruction filter $h_r(t)$ in time (T_s = 1 unit); Right: the actual interpolation by superposition of scaled and shifted versions of $h_r(t)$.

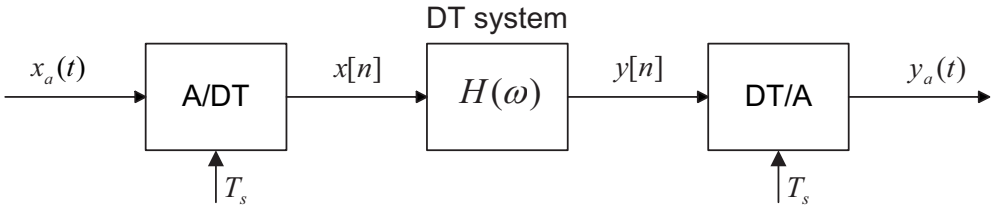


Fig. 7.8 Setting for discrete-time processing of continuous-time signals.

7.2.1 Study of input-output relationship

Mathematical characterization: The three elements of Figure 7.8 can be characterized by the following relationships:

- Ideal C/D converter:

$$X(\omega)|_{\omega=\Omega T_s} = \frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_a(\Omega - k\Omega_s) \tag{7.22}$$

  where $\Omega_s = 2\pi/T_s$.

- Discrete-time LTI filter:

$$Y(\omega) = H(\omega)X(\omega) \tag{7.23}$$

  where $H(\omega)$ is an arbitrary DT system function

- Ideal D/C converter (see proof of sampling theorem):

$$Y_a(\Omega) = H_r(\Omega)Y(\Omega T_s) \tag{7.24}$$

where $H_r(\Omega)$ is the frequency response of an ideal continuous-time low-pass filter, with cut-off frequency $\Omega_s/2$, i.e.

$$H_r(\Omega) = \begin{cases} T_s & |\Omega| < \Omega_s/2 \\ 0 & \text{otherwise} \end{cases} \tag{7.25}$$

Overall Input-output relationship: Combining (7.22)–(7.24) into a single equation, we obtain

$$Y_a(\Omega) = H_r(\Omega)H(\Omega T_s)\frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_a(\Omega - k\Omega_s). \tag{7.26}$$

Recall that for an analog LTI system, the Fourier Transforms of the input $x_a(t)$ and corresponding output $y_a(t)$ are related through

$$Y_a(\Omega) = G(\Omega)X_a(\Omega). \tag{7.27}$$

In general, (7.26) does not reduce to that form because of the spectral images $X_a(\Omega - k\Omega_s)$ for $k \neq 0$.

However, if we further assume that $x_a(t)$ is band-limited to $|\Omega| < \Omega_N$ and that $\Omega_s \geq 2\Omega_N$, then the products

$$H_r(\Omega)X_a(\Omega - k\Omega_s) = 0, \quad \text{for all } k \neq 0. \tag{7.28}$$

In this case, (7.26) reduces to

$$Y_a(\Omega) = H(\Omega T_s)X_a(\Omega) \tag{7.29}$$

which corresponds to an LTI analog system.

Theorem 1 (DT processing of analog signals) *Provided that analog signal $x_a(t)$ is band-limited to $|\Omega| < \Omega_N$ and that $\Omega_s \geq 2\Omega_N$, discrete-time processing of $x[n] = x_a(nT_s)$ with $H(\omega)$ is equivalent to analog processing of $x_a(t)$ with*

$$H_a(\Omega) = \begin{cases} H(\Omega T_s) & \text{if } |\Omega| < \Omega_s/2 \\ 0 & \text{otherwise} \end{cases} \tag{7.30}$$

Remarks: Note that $H_a(\Omega)$ is set to zero for $|\Omega| \geq \Omega_s/2$ since the FT of the input is zero over that range, i.e. $X_a(\Omega) = 0$ for $|\Omega| \geq \Omega_s/2$. The Theorem states that the two systems shown in Figure 7.9 are equivalent under the given conditions.

In practice, many factors limit the applicability of this result:

- input signal $x_a(t)$ not perfectly band-limited
- non-ideal C/D conversion
- non-ideal D/C conversion

These issues are given further considerations in the following sections.

Example 7.2:

▶ Consider a continuous-time speech signal limited to 8000 Hz. If it is desired to send this signal over a telephone line, it has to be band-limited between 300 Hz and 3400 Hz. The band-pass filter to be used is then specified as:

$$H_a(\Omega) = \begin{cases} 1 & 600\pi \leq |\Omega| \leq 6800\pi \\ 0 & \text{otherwise} \end{cases}$$

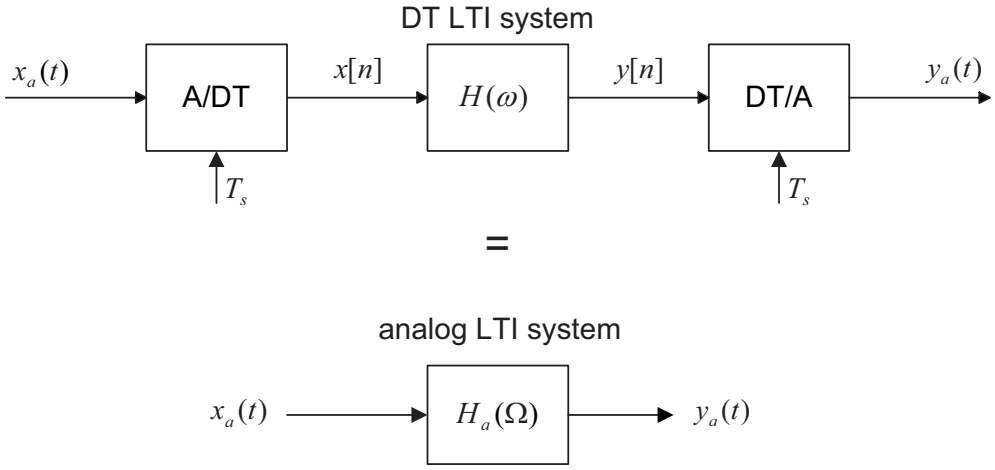


Fig. 7.9 Equivalence between Analog and Discrete-Time processing.

If this filter is to be implemented in discrete-time without loss of information, sampling must take place above the Nyquist rate, i.e. with $F_s \geq 16$ kHz, or $\Omega_s \geq 32000\pi$. If $\Omega_s = 32000\pi$ is chosen, the discrete-time filter which will implement the same band-pass filtering as $H_a(\Omega)$ is given by:

$$H(\omega) = \begin{cases} 1 & 600\pi/16000 \leq |\omega| \leq 6800\pi/16000 \\ 0 & \text{otherwise in } [-\pi,\pi], \end{cases}$$

the specifications outside $[-\pi,\pi]$ follow by periodicity of $H(\omega)$. ◄

7.2.2 Anti-aliasing filter (AAF)

Principle of AAF:

The AAF is an analog filter, used prior to C/D conversion, to remove high-frequency content of $x_a(t)$ when the latter is not BL to $\pm\Omega_s/2$. Its specifications are given by:

$$H_{aa}(\Omega) = \begin{cases} 1 & |\Omega| < \Omega_s/2 \\ 0 & \text{otherwise} \end{cases}, \tag{7.31}$$

and are illustrated in Figure 7.10.

The AAF avoids spectral aliasing during the sampling process, but is also useful in general even if the signal to be sampled is already band-limited: it removes high-frequency noise which would be aliased in the band of interest by the sampling process.

The AAF is necessary, but has some drawbacks: if the signal extends over the $\Omega_s/2$ limit, useful signal information may be lost; moreover, the direct analog implementation of $H_{aa}(\Omega)$ with sharp cut-offs is costly, and such filters usually introduce phase-distortion near the cut-off frequency.

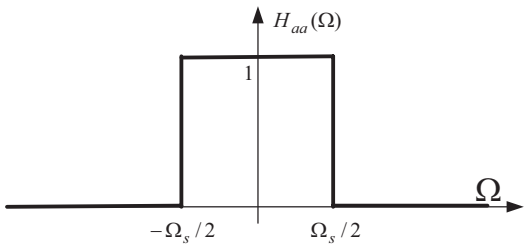


Fig. 7.10 Anti-aliasing filter template.

Implementation issues:

AAF are always present in DSP systems that interface with the analog world. Several approaches can be used to circumvent difficulties associated to the sharp cut-off requirements of the AAF filter $H_{aa}(\Omega)$ in (7.31). Two such approaches are described below and illustrated in Figure (7.11):

- Approach #1:
  - Fix $\Omega_N =$ maximum frequency of interest (e.g. $\Omega_N = 20$kHz)
  - Set $\Omega_s = (1+r)2\Omega_N$ where $0 < r < 1$ (e.g. $r = 0.1 \Rightarrow \Omega_s = 44$kHz)
  - Use of non-ideal $H_{aa}(\Omega)$ with transition band between Ω_N and $(1+r)\Omega_N$. In this case, a smooth transition band of width $r\Omega_N$ is available.
- Approach #2: M-times Oversampling:
  - Use cheap analog $H_{aa}(\Omega)$ with very large transition band
  - Significantly oversample $x_a(t)$ (e.g. $\Omega_s = 8 \times 2\Omega_N$)
  - Complete the anti-aliasing filtering in the DT domain by a sharp low-pass DT filter $H_{aa}(\omega)$, and then reduce the sampling rate.

The second approach is superior in terms of flexibility and cost. It is generally used in commercial audio CD players. We will investigate it in further detail when we study multi-rate systems in chapter **??**.

7.3 A/D conversion

Introduction

Up to now, we have considered an idealized form of uniform sampling, also called *ideal Analog-to-Discrete Time (A/DT) conversion* (see Figure 7.12), in which the sampling takes place instantaneously and the signal amplitude $x[n]$ can take any arbitrary real value ,i.e. $x[n] \in \mathbb{R}$, so that $x[n]$ is a discrete-time signal.

A/DT conversion cannot be implemented exactly due to hardware limitations; it is used mainly as a mathematical model in the study of DSP. Practical DSP systems use non-ideal *Analog-to-Digital (A/D)* converters instead. These are characterized by:

- Finite response time (the sampling is not quite instantaneous),

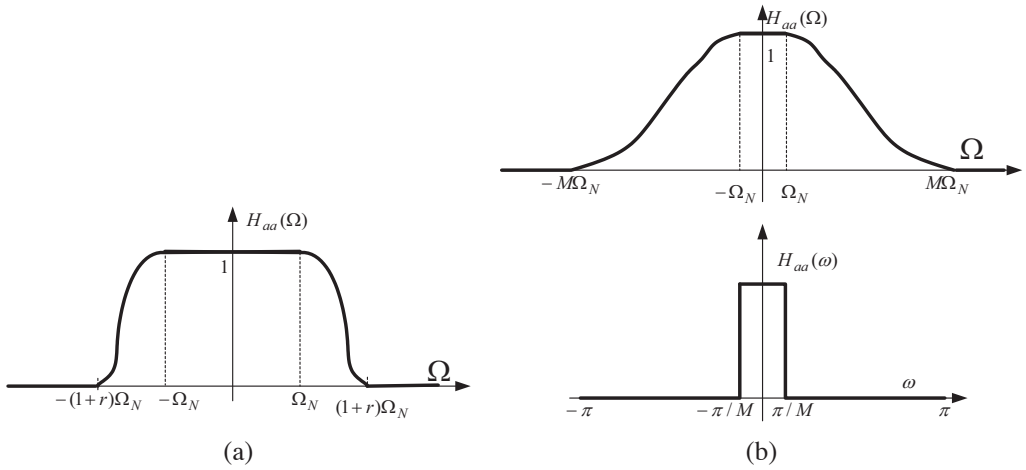


Fig. 7.11 Illustration of practical AAF implementations. (a) Excess sampling by factor $1+r$ (b) Oversampling by factor M.

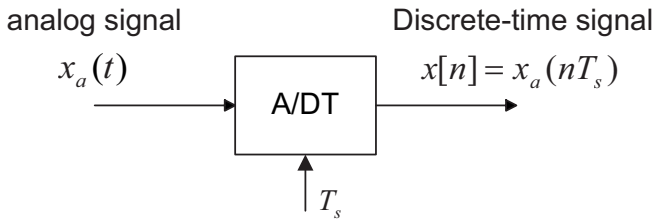


Fig. 7.12 Ideal A/DT Conversion.

- Finite number of permissible output levels for $x[n]$
  (i.e. $x[n] \in$ finite set $\Rightarrow$ digital signal),
- Other types of imperfections like timing jitter, A/D nonlinearities, ...

7.3.1 Basic steps in A/D conversion

Analog-to-Digital Conversion (A/D) may be viewed as a two-step process, as illustrated in Figure 7.13:

- The Sample-and-Hold device (S&H) performs uniform sampling of its input at times nT_s and maintains a constant output voltage of $x[n]$ until the next sampling instant.
- During that time, the quantizer approximates $x[n]$ with a value $\hat{x}[n]$ selected from a finite set of possible representation levels.

The operation of each process is further developed below.

Sample-and-hold (S&H): The S&H performs uniform sampling of its input at times nT_s; its output voltage, denoted $\tilde{x}(t)$, remains constant until the next sampling instant. Figure 7.14 illustrates the input and

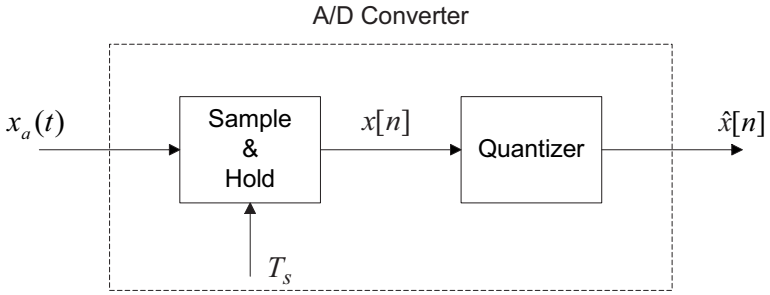


Fig. 7.13 Practical A/D Conversion

output signals of the sample-and-hold device.

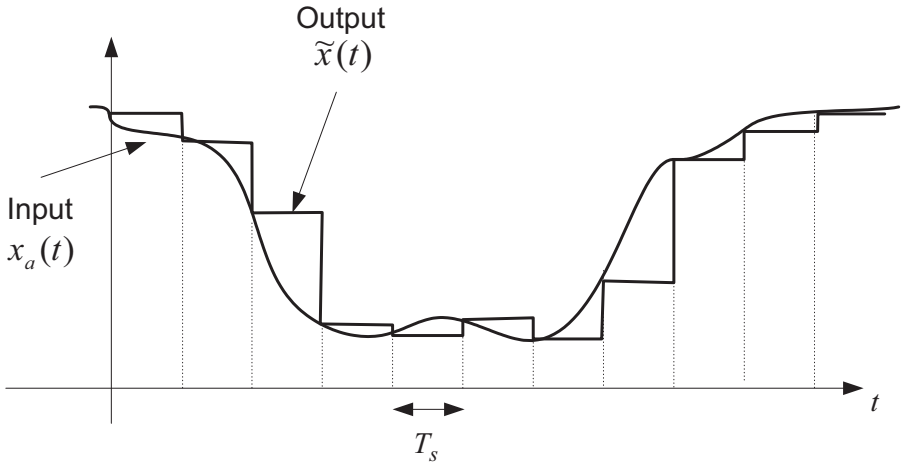


Fig. 7.14 Illustration of a Sample-and-Hold operation on a continuous-time signal.

Thus, the S&H output $\tilde{x}(t)$ is a PAM waveform ideally given by

$$\tilde{x}(t) = \sum_{n=-\infty}^{\infty} x[n]\, p(t - nT_s) \tag{7.32}$$

$$x[n] = x_a(nT_s) \tag{7.33}$$

$$p(t) = \begin{cases} 1 & \text{if } 0 \le t \le T_s \\ 0 & \text{otherwise} \end{cases} \tag{7.34}$$

Practical S&H suffer from several imperfections:

- The sampling is not instantaneous: $x[n] \approx x_a(nT_s)$;
- The output voltage does not remain constant but slowly decays within a sampling period of duration T_s.

Quantizer: The quantizer is a memoryless device, represented mathematically by a function $Q(.)$, also called quantizer characteristic:

$$\hat{x}(n) = Q(x[n]). \tag{7.35}$$

Assuming a constant input voltage $x[n]$, the function Q selects, among a finite set of K permissible levels, the one that best represents $x[n]$:

$$x[n] \in \mathbb{R} \xrightarrow{Q} \hat{x}[n] \in \mathcal{A} = \{\hat{x}_k\}_{k=1}^{K} \tag{7.36}$$

where $\hat{x}_k$ ($k = 1, \ldots, K$) represent the permissible levels and $\mathcal{A}$ denotes the set of all such levels. In practice, $B+1$ bits are used for the representation of signal levels (B bits for magnitude + 1 bit for sign), so that the the total number of levels is $K = 2^{B+1}$. Since the output levels of the quantizer are usually represented in binary form, the concept of binary coding is implicit within the function $Q(.)$ (more on this in a later chapter).

For an input signal $x[n]$ to the quantizer, the quantization error is defined as

$$e[n] \triangleq \hat{x}[n] - x[n] \tag{7.37}$$

Uniform Quantizer: The most commonly used form of quantizer is the uniform quantizer, in which the representation levels are uniformly spaced within an interval $[-X_{\text{fs}}, X_{\text{fs}}]$, where X_{fs} (often specified as a positive voltage) is known as the full-scale level of the quantizer. Specifically, the representation levels are constrained to be

$$\hat{x}_k = -X_{\text{fs}} + (k-1)\Delta, \quad k = 1, \ldots, K. \tag{7.38}$$

- The minimum value representable by the quantizer is $\hat{x}_{min} = \hat{x}_1 = -X_{\text{fs}}$
- Δ, called the quantization step size, is the constant distance between representation levels. In practice, we have

$$\Delta = \frac{2X_{\text{fs}}}{K} = 2^{-B} X_{\text{fs}} \tag{7.39}$$

- The maximum value representable by the quantizer is $\hat{x}_{max} = \hat{x}_K = X_{\text{fs}} - \Delta$
- The dynamic range of the uniform quantizer is defined as the interval

$$\text{DR} = [\hat{x}_1 - \frac{\Delta}{2}, \hat{x}_K + \frac{\Delta}{2}] \tag{7.40}$$

While several choices are possible for the non-linear quantizer characteristic $Q(.)$ in (7.35), a common approach is to round the input to the closest level:

$$Q(x[n]) = \begin{cases} \hat{x}_1 & \text{if } x[n] < \hat{x}_1 - \frac{\Delta}{2} \\ \hat{x}_k & \text{if } \hat{x}_k - \frac{\Delta}{2} \le x[n] < \hat{x}_k + \frac{\Delta}{2} \text{ for } k = 1, ..., K \\ \hat{x}_K & \text{if } x[n] \ge \hat{x}_K + \frac{\Delta}{2}. \end{cases} \tag{7.41}$$

Figure 7.15 illustrates the input-output relationship of a 3-bit uniform quantizer. In this case, $K = 8$ different levels exist.

For the uniform quantizer, the quantization error $e[n]$ satisfies:

$$\begin{aligned} x[n] \in \text{DR} \quad &\Rightarrow \quad |e[n]| \le \Delta/2 \\ x[n] \notin \text{DR} \quad &\Rightarrow \quad |e[n]| > \Delta/2 \text{ (clipping)} \end{aligned} \tag{7.42}$$

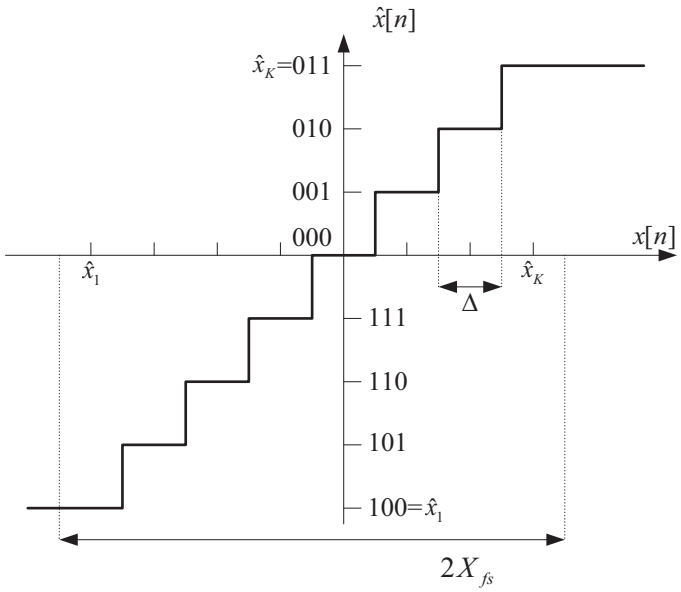


Fig. 7.15 3-bit Uniform Quantizer Input-output characteristic. The reproduction levels are labelled with their two's complement binary representation.

7.3.2 Statistical model of quantization errors

In the study of DSP systems, it is important to know how quantization errors will affect the system output. Since it is difficult to handle such errors deterministically, a statistical model is usually assumed. A basic statistical model is derived below for the uniform quantizer.

Model structure: From the definition of quantization error:

$$e[n] \triangleq \hat{x}[n] - x[n] \Rightarrow \hat{x}[n] = x[n] + e[n], \tag{7.43}$$

one can derive an equivalent additive model for the quantization noise, as illustrated in Figure 7.16. The

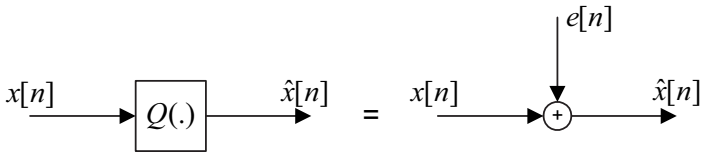


Fig. 7.16 Additive noise model of a quantizer

non-linear device Q is replaced by a linear operation, and the error $e[n]$ is modelled as a white noise sequence (see below).

White noise assumption: The following assumptions are made about the input sequence $x[n]$ and the corresponding quantization error sequence $e[n]$:

- $e[n]$ and $x[n]$ are uncorrelated, i.e.

$$E\{x[n]e[m]\} = 0 \text{ for any } n,m \in \mathbb{Z} \tag{7.44}$$

  where $E\{\,\}$ denotes expectation.

- $e[n]$ is a white noise sequence, i.e.:

$$E\{e[n]e[m]\} = 0 \text{ if } m \neq n \tag{7.45}$$

- $e[n]$ is uniformly distributed within $\pm\Delta/2$, which implies

$$\text{mean} = E\{e[n]\} = 0 \tag{7.46}$$

$$\text{variance} = E\{e[n]^2\} = \int_{-\Delta/2}^{\Delta/2} \frac{e^2}{\Delta} de = \frac{\Delta^2}{12} \triangleq \sigma_e^2 \tag{7.47}$$

These assumptions are valid provided the signal is sufficiently "busy", i.e.:

- $\sigma_x \triangleq$ RMS value of $x[n] \gg \Delta$;
- $x[n]$ changes rapidly.

SQNR computation: the signal to quantization noise ratio (SQNR) is a good characterization of the effect of the quantizer on the input signal. It is defined as the ratio between the signal power and the quantization noise power at the output of the quantizer, in dB:

$$\text{SQNR} = 10\log_{10}\frac{\sigma_x^2}{\sigma_e^2},$$

where σ_x and σ_e represent the RMS value of the signal and noise respectively. With the above assumption on the noise, we have that

$$\sigma_e^2 = \frac{\Delta^2}{12} = \frac{2^{-2B}X_{\text{fs}}^2}{12},$$

where X_{fs} is the full scale level of the quantizer, as illustrated in Figure 7.15, so that

$$\text{SQNR} = 20\log_{10}\frac{\sigma_x}{X_{\text{fs}}} + 10\log_{10}12 + 20B\log_{10}2$$

or finally

$$\boxed{\text{SQNR} = 6.02\,B + 10.8 - 20\log_{10}\frac{X_{\text{fs}}}{\sigma_x}} \quad (\text{dB}) \tag{7.48}$$

The above SQNR expression states that for each added bit, the SQNR increases by 6 dB. Moreover, it includes a penalty term for the cases where the quantizer range is not matched to the input dynamic range. As an example, if the dynamic range is chosen too big ($X_{\text{fs}} \gg \sigma_x$), then this term will add a big negative contribution to the SQNR, because the usage of the available quantizer reproduction levels will be very bad (only a few levels around 0 will be used). This formula does not take into account the complementary problem (quantizer range too small, resulting in clipping of useful data), because he underlying noise model is based on a no-clipping assumption.

7.4 D/A conversion

Figure 7.17 shows the ideal discrete time-to-analog conversion (DT/C) as considered up to now. It uses an ideal low-pass reconstruction filter $h_r(t)$, as defined in equation (7.18). The corresponding frequency and

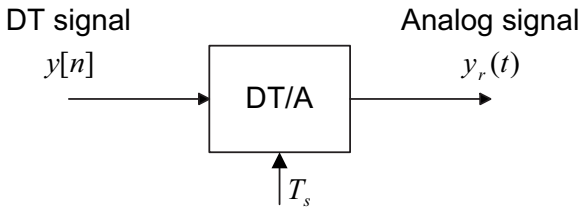


Fig. 7.17 Ideal Discrete Time-to-Analog Conversion

time input/output expressions are:

$$y_r(t) = \sum_{n=-\infty}^{\infty} y[n] h_r(t - nT_s) \tag{7.49}$$

and

$$Y_r(\Omega) = Y(\Omega T_s) H_r(\Omega) \tag{7.50}$$

where

$$H_r(\Omega) = \begin{cases} T_s & |\Omega| < \Omega_s/2 \\ 0 & \text{otherwise} \end{cases} \tag{7.51}$$

This is an idealized form of reconstruction, since all the signal samples $y[n]$ are needed to reconstruct $y_r(t)$. The interpolating function $h_r(t)$ corresponds to an ideal low-pass filtering operation (i.e. non-causal and unstable).

Ideal DT/A conversion cannot be implemented exactly due to physical limitations; similarly to A/DT, it is used as a mathematical model in DSP. Practical DSP systems use non-ideal D/A converters instead characterized by:

- a realizable, low-cost approximation to $h_r(t)$.
- the output signal $y_r(t)$ is not perfectly band-limited.

7.4.1 Basic steps in D/A conversion:

D/A may be viewed as a two-step process, illustrated in Figure 7.18:

- The hold circuitry transforms its digital input into a continuous-time PAM-like signal $\tilde{y}(t)$;
- The post-filter smoothes out the PAM signal (spectral reshaping).

Hold circuit: A Time-domain description of its operation is given by:

$$\tilde{y}(t) = \sum_{n=-\infty}^{\infty} y[n] h_0(t - nT_s) \tag{7.52}$$

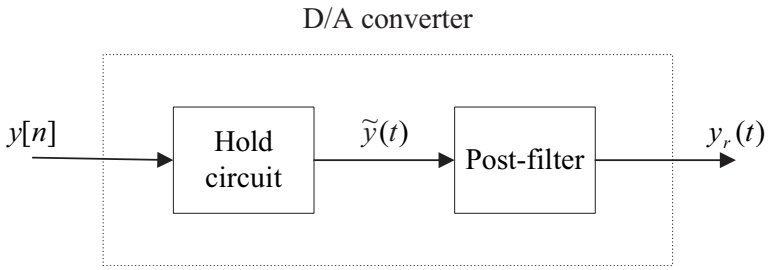


Fig. 7.18 Two-step D/A Conversion.

where $h_0(t)$ is the basic pulse shape of the circuit. In the frequency domain:

$$\begin{aligned}\tilde{Y}_r(\Omega) &= \int_{-\infty}^{\infty} \tilde{y}(t)e^{-j\Omega t}dt \\ &= \sum_{n=-\infty}^{\infty} y[n]\int_{-\infty}^{\infty} h_0(t-nTs)e^{-j\Omega t}dt \\ &= \sum_{n=-\infty}^{\infty} y[n]e^{-j\Omega T_s n}\int_{-\infty}^{\infty} h_0(t)e^{-j\Omega t}dt \\ &= Y(\Omega T_s)\,H_0(\Omega) \end{aligned} \tag{7.53}$$

In general, $H_0(\Omega) \neq H_r(\Omega)$, so that the output $\tilde{y}(t)$ of the hold circuit is different from that of the ideal DT/A conversion. The most common implementation is the *zero-order hold*, which uses a rectangular pulse shape:

$$h_0(t) = \begin{cases} 1 & \text{if } 0 \le t \le T_s \\ 0 & \text{otherwise} \end{cases} \tag{7.54}$$

The corresponding frequency response is

$$H_0(\Omega) = T_s\,\mathrm{sinc}(\Omega/\Omega_s)\,e^{-j\pi\Omega/\Omega_s} \tag{7.55}$$

Post-filter: The purpose of the post-filter $H_{pf}(\Omega)$ is to smooth out the output of the hold circuitry so that it resembles more that of an ideal DT/A function.

From the previous discussions, it is clear that the desired output of the D/A converter is

$$Y_r(\Omega) = Y(\Omega T_s)H_r(\Omega),$$

whereas the output of hold circuitry is:

$$\tilde{Y}(\Omega) = Y(\Omega T_s)H_0(\Omega).$$

Clearly, the output of the post filter will be identical to that of the ideal DT/A converter if

$$H_{pf}(\Omega) = H_r(\Omega)/H_0(\Omega) = \begin{cases} T_s/H_0(\Omega) & \text{if } |\Omega| < \Omega_s/2 \\ 0 & \text{otherwise} \end{cases} \tag{7.56}$$

In practice,the post filter $H_{pf}(\Omega)$ can only be approximated but the results are still satisfactory.

Example 7.3: Zero-order hold circuit

► The zero-order hold frequency response is shown on the top of Figure 7.19, as compared to the ideal brick-wall low-pass reconstruction filter $H_r(\Omega)$. The compensating filter in this case is shown in the bottom part of Figure 7.19. ◄

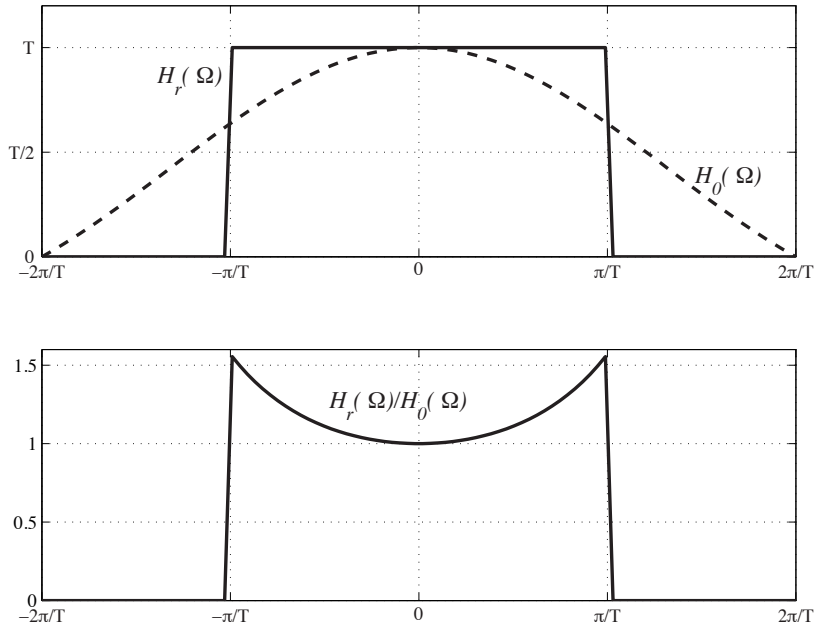


Fig. 7.19 Illustration of the zero-order hold frequency response (Top, compared with ideal low-pass interpolation filter), and the associated compensation post-filter (bottom).(T denotes the sampling period)

Chapter 8

Structures for the realization of DT systems

Introduction:

Consider a causal LTI system $\mathcal{H}$ with rational system function:

$$H(z) = \mathcal{Z}\{h[n]\} = \frac{B(z)}{A(z)}. \tag{8.1}$$

In theory, $H(z)$ provides a complete mathematical characterization of the system $\mathcal{H}$. However, even if we specify $H(z)$, the input-output transformation $x[n] \rightarrow y[n] = \mathcal{H}\{x[n]\}$, where $x[n]$ denotes and arbitrary input and $y[n]$ the corresponding output, can be computed in a multitude of different ways.

Each of the different ways of performing the computation $y[n] = \mathcal{H}\{x[n]\}$ is called a *realization* of the system. Specifically:

- A realization of a system is a specific (and complete) description of its internal computational structure.
- This description may involve difference equations, block diagrams, pseudo-code (Matlab like), etc.

For a given system $\mathcal{H}$, there is an infinity of such realizations. The choice of a particular realization is guided by practical considerations, such as:

- computational requirements;
- internal memory requirements;
- effects of finite precision representation;
- chip area and power consumption in VLSI implementation;
- etc.

In this chapter, we discuss some of the most well-known structures for the realization of FIR and IIR discrete-time systems. We will use Signal Flow-Graph representation to illustrate the different compuational structures. These are reviewed in the next section.

8.1 Signal flow-graph representation

A signal flow-graph (SFG) is a network of directed *branches* connecting at *nodes*. It is mainly used to graphically represent the relationship between signals. It in fact gives detailed information on the actual algorithm that is used to compute the samples of one of several signals based on the samples of one or several other signals.

Basic terminology and operation principles of SFG are detailed below:

Node: A node is represented by a small circle, as shown in Figure 8.1(a). To each node is associated a node value, say $w[n]$, that may or not appear next to the node. The node value represents either an external input, the result of an internal computation, or an output value (see below).

Branch: A branch is an oriented line segment. The signal flow along the branch is indicated by an arrow. The relationship between the branch input and output is provided by the branch transmittance, as shown in FIgure 8.1(b).

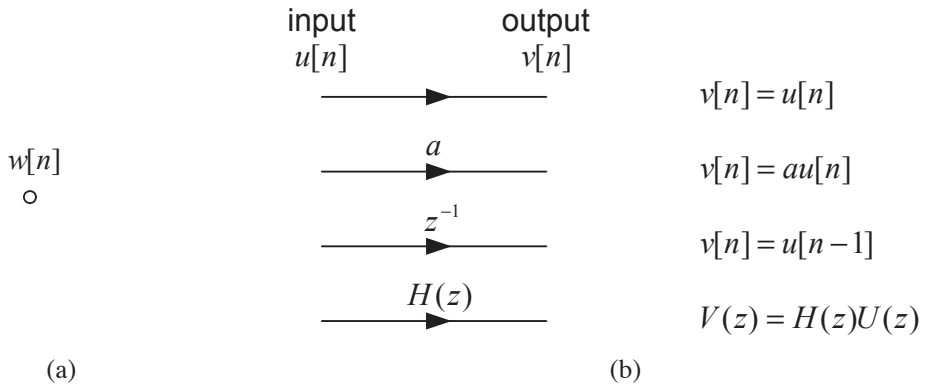


Fig. 8.1 Basic constituents of a signal flow-graph: (a) a single isolated node; (b) different branches along with their input-output characteristic.

Internal node: An internal node is a node with one or more input branches (i.e. entering the node) and one or more output branches (i.e. leaving the node), as shown in Figure 8.2(a).

The node value $w[n]$ (not shown in the figure) is the sum of the outputs of all the branches entering the node, i.e.

$$w[n] \triangleq \sum_{k=1}^{K} u_k[n]. \tag{8.2}$$

The inputs to the branches leaving the node are all identical and equal to the node value:

$$v_1[n] = \cdots = v_L[n] = w[n]. \tag{8.3}$$

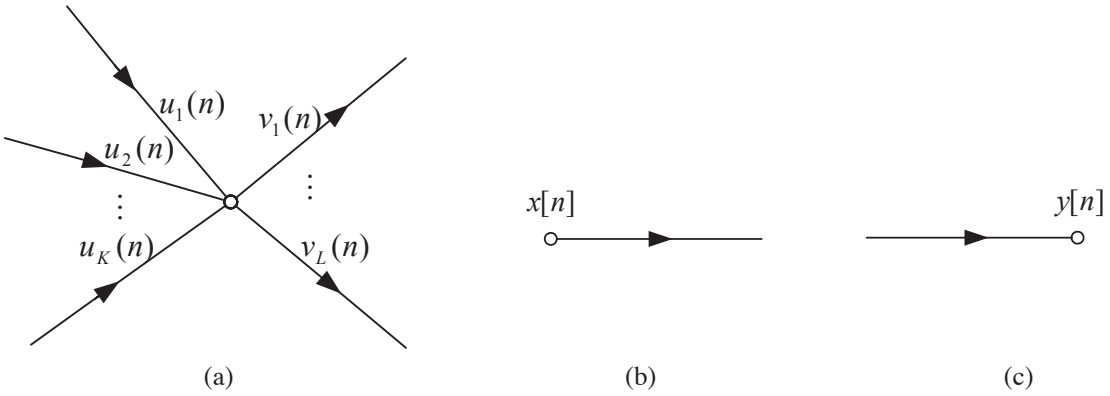


Fig. 8.2 Different types of nodes: (a) internal node; (b) source node; (c) sink node.

Source node: A source node is a node with no input branch, as shown in Figure 8.2(b). The node value $x[n]$ is provided by an external input to the network.

Sink node: A sink node is a node with no output branch, as shown in Figure 8.2(c). The node value $y[n]$, computed in the same way as an internal node, may be used as a network output.

Remarks: SFGs provides a convenient tool for describing various system realizations. In many cases, non-trivial realizations of a system can be obtained via simple modifications to its SFG.

In the examples below, we illustrate the use of SFG for the representation of rational LTI systems. In particular, we illustrate how a SFG can be derived from an LCCDE description of the system and vice versa.

Example 8.1:

- Consider the SFG in figure 8.3, where a and b are arbitrary constants:

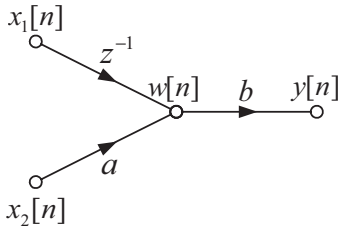


Fig. 8.3 SFG for example 8.1.

The node value can be expressed in terms of input signals $x_1[n]$ and $x_2[n]$ as

$$w[n] = x_1[n-1] + ax_2[n]$$

From there, an expression for the output signal $y[n]$ in terms of $x_1[n]$ and $x_2[n]$ is obtained as

$$\begin{aligned} y[n] &= bw[n] \\ &= bx_1[n-1] + abx_2[n] \end{aligned}$$

◄

Example 8.2:

► Consider the system function

$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 - az^{-1}}$$

To obtain a SFG representation of $H(z)$, we first obtain the corresponding LCCDE, namely

$$y[n] = ay[n-1] + b_0 x[n] + b_1 x[n-1]$$

The derivation of the SFG can be made easier by expressing the LCCDE as a set of two equations as follows:

$$\begin{aligned} w[n] &= b_0 x[n] + b_1 x[n-1] \\ y[n] &= ay[n-1] + w[n] \end{aligned}$$

From there, one immediately obtains the SFG in figure 8.4.

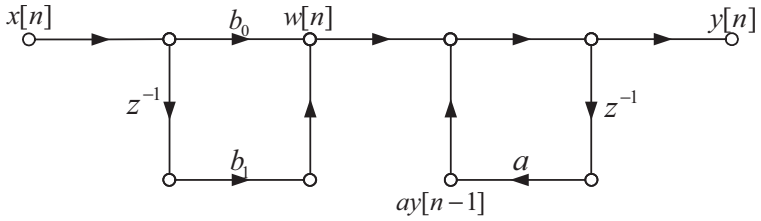


Fig. 8.4 SFG for example 8.2

Note the presence of the feedback loop in this diagram which is typical of recursive LCCDE ◄

Example 8.3:

► Let us find the system function $H(z)$ corresponding to the SFG in figure 8.5

In this type of problem, one has to be very systematic to avoid possible mistakes. We proceed in 4 steps as follows:

(1) Identify and label non-trivial internal nodes. Here, we identify two non-trivial nodes, labelled $w_1[n]$ and $w_2[n]$ in Figure 8.5.

(2) Find the z-domain input/output (I/O) relation for each of the non-trivial nodes and the output node. Here, there are 2 non-trivial nodes and 1 output node, so that we need a total of 3 linearly independent relations, namely:

$$Y(z) = b_0 W_1(z) + b_1 W_2(z) \tag{8.4}$$

$$W_2(z) = z^{-1} W_1(z) \tag{8.5}$$

$$W_1(z) = X(z) + aW_2(z) \tag{8.6}$$

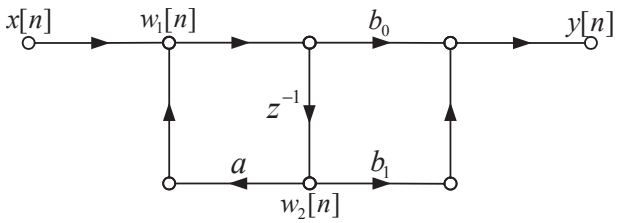


Fig. 8.5 SFG for example 8.3.

(3) By working out the algebra, solve the I/O relations for $Y(z)$ in terms of $X(z)$. Here:

$$\begin{aligned}(8.5)+(8.6) \quad &\Longrightarrow \quad W_1(z) = X(z) + az^{-1}W_1(z) \\ &\Longrightarrow \quad W_1(z) = \frac{X(z)}{1-az^{-1}} \end{aligned} \tag{8.7}$$

$$\begin{aligned}(8.4)+(8.5)+(8.7) \quad &\Longrightarrow \quad Y(z) = (b_0 + b_1 z^{-1})W_1(z) \\ &\Longrightarrow \quad Y(z) = \frac{b_0 + b_1 z^{-1}}{1-az^{-1}} X(z) \end{aligned} \tag{8.8}$$

(4) Finally, the system function is obtained as

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1}}{1-az^{-1}}$$

As a final remark, we note that the above system function is identical to that considered in Example 8.2, even tough the SFG in considered here is different from that in Example 8.3. ◀

8.2 Realizations of IIR systems

We consider IIR systems with rational system functions:

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{N} a_k z^{-k}} \triangleq \frac{B(z)}{A(z)}. \tag{8.9}$$

The minus signs in the denominator are introduced to simplify the Signal Flow-Graphs (be careful, they are often a source of confusion).

For this general IIR system function, we shall derive several SFGs that correspond to different system realizations.

8.2.1 Direct form I

The LCCDE corresponding to system function $H(z)$ in (8.9) is given by:

$$y[n] = \sum_{k=1}^{N} a_k y[n-k] + \sum_{k=0}^{M} b_k x[n-k] \tag{8.10}$$

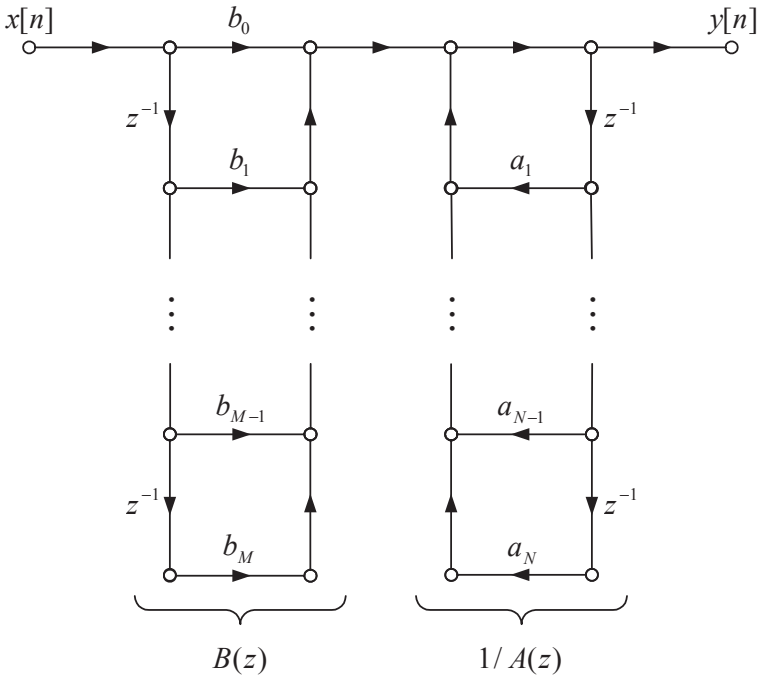


Fig. 8.6 Direct Form I implementation of an IIR system.

Proceeding as in Example 8.2, this leads directly to the SFG shown in Figure 8.6, known as direct form I (DFI).

In a practical implementation of this SFG, each unit delay (i.e. z^{-1}) requires one storage location (past values need to be stored).

Finally, note that in Figure 8.6, the section of the SFG labeled $B(z)$ corresponds to the zeros of the system, while the section labelled $1/A(z)$ corresponds to the poles.

8.2.2 Direct form II

Since LTI systems do commute, the two sections identified as $B(z)$ and $1/A(z)$ in Figure 8.6 may be interchanged. This leads to the intermediate SFG structure shown in Figure 8.7.

Observe that the two vertical delay lines in the middle of the structure compute the same quantities, namely $w[n-1], w[n-2], \ldots$, so that only one delay line is actually needed. Eliminating one of these delay lines and merging the two sections lead to the structure shown in Figure 8.8, known as the direct form II (DFII).

Note that $N = M$ is assumed for convenience, but if it is not the case, branches corresponding to $a_i = 0$ or $b_i = 0$ will simply disappear.

The main advantage of DFII over DFI is its reduced storage requirement. Indeed, while the DFI contains $N+M$ unit delays, the DFII only contains $\max\{N,M\} \leq N+M$ unit delays.

Referring to Figure 8.8, one can easily see that the following difference equations describe the DFII realiza-

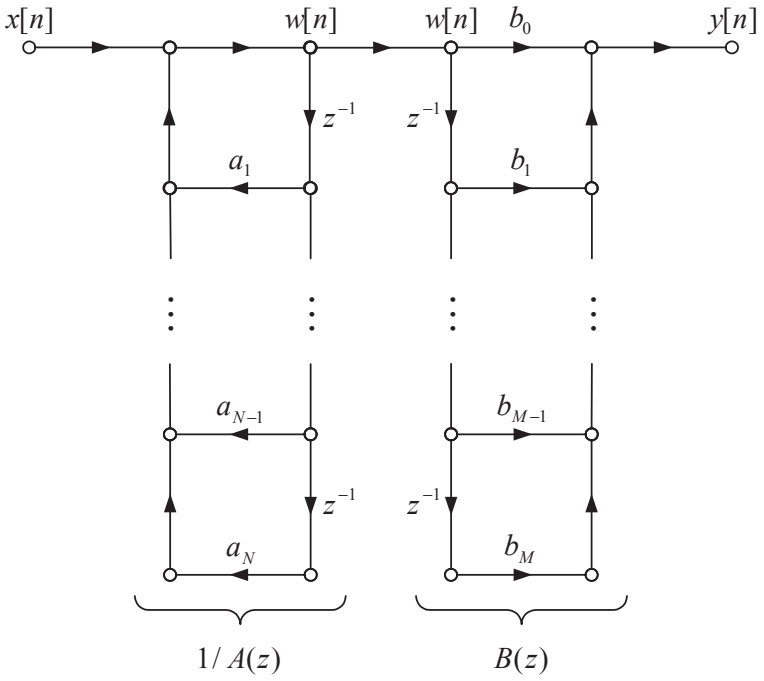


Fig. 8.7 Structure obtained from a DFI by commuting the implementation of the poles with the implementation of the zeros.

tion:

$$w[n] = a_1 w[n-1] + \cdots a_N w[n-N] + x[n] \tag{8.11}$$
$$y[n] = b_0 w[n] + b_1 w[n-1] + \cdots + b_M w[n-M] \tag{8.12}$$

8.2.3 Cascade form

A cascade form is obtained by first factoring $H(z)$ as a product:

$$H(z) = \frac{B(z)}{A(z)} = H_1(z)H_2(z)\ldots H_K(z) \tag{8.13}$$

where

$$H_k(z) = \frac{B_k(z)}{A_k(z)}, \quad k = 1,2,...,K \tag{8.14}$$

represent IIR sections of low-order (typically 1 or 2). The corresponding SFG is shown in Figure 8.9. Note that each box needs to be expanded with the proper sub-SFG (often in DFII or transposed DFII).

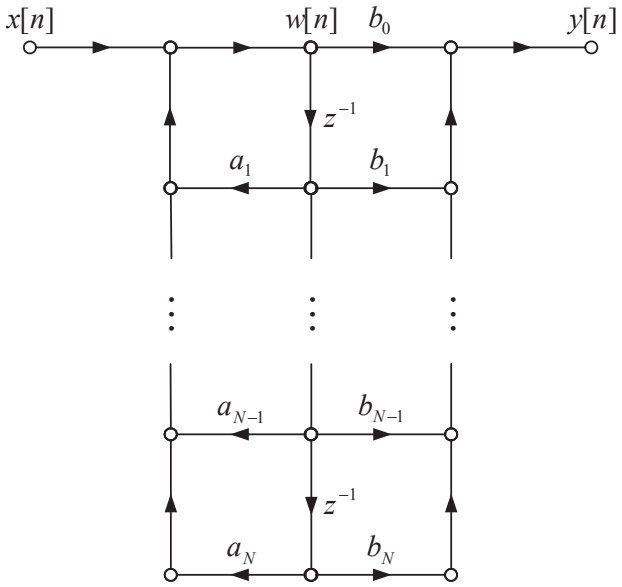


Fig. 8.8 Direct Form II realization of an IIR system.

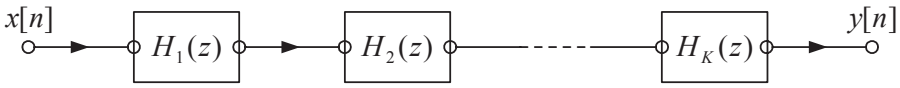


Fig. 8.9 Cascade Form for an IIR system. Note that each box needs to be expanded with the proper sub-SFG (often in DFII or transposed DFII).

For a typical real coefficient second-order section (SOS):

$$H_k(z) = G_k \frac{(1-z_{k1}z^{-1})(1-z_{k2}z^{-1})}{(1-p_{k1}z^{-1})(1-p_{k2}z^{-1})} \tag{8.15}$$

$$= \frac{b_{k0}+b_{k1}z^{-1}+b_{k2}z^{-2}}{1-a_{k1}z^{-1}-a_{k2}z^{-2}} \tag{8.16}$$

where $p_{k2} = p_{k1}^*$ and $z_{k2} = z_{k1}^*$ so that $a_{ki}, b_{ki} \in \mathbb{R}$. It should be clear that such a cascade decomposition of an arbitrary $H(z)$ is not unique:

- There are many different ways of grouping the zeros and poles of $H(z)$ into lower order sections;
- The gains of the individual sections may be changed (provided their product remains constant).

Example 8.4:

▶ Consider the system function

$$H(z) = G\frac{(1-e^{j\pi/4}z^{-1})(1-e^{-j\pi/4}z^{-1})(1-e^{j3\pi/4}z^{-1})(1-e^{-j3\pi/4}z^{-1})}{(1-.9z^{-1})(1+.9z^{-1})(1-.9jz^{-1})(1+.9jz^{-1})}$$

There are several ways of pairing the poles and the zeros of $H(z)$ to obtain 2nd order sections. Whenever possible, complex conjugate poles and zeros should be paired so that the resulting second order section has real coefficient. For example, one possible choice that fulfils this requirement is:

$$H_1(z) = \frac{(1-e^{j\pi/4}z^{-1})(1-e^{-j\pi/4}z^{-1})}{(1-.9z^{-1})(1+.9z^{-1})} = \frac{1-\sqrt{2}z^{-1}+z^{-2}}{1-.81z^{-2}}$$

$$H_2(z) = \frac{(1-e^{j3\pi/4}z^{-1})(1-e^{-j3\pi/4}z^{-1})}{(1-.9jz^{-1})(1+.9jz^{-1})} = \frac{1+\sqrt{2}z^{-1}+z^{-2}}{1+.81z^{-2}}$$

The corresponding SFG is illustrated in figure 8.10 where the above 2nd order sections are realized in DFII. Note that we have arbitrarily incorporated the gain factor G in between the two sections $H_1(z)$ and

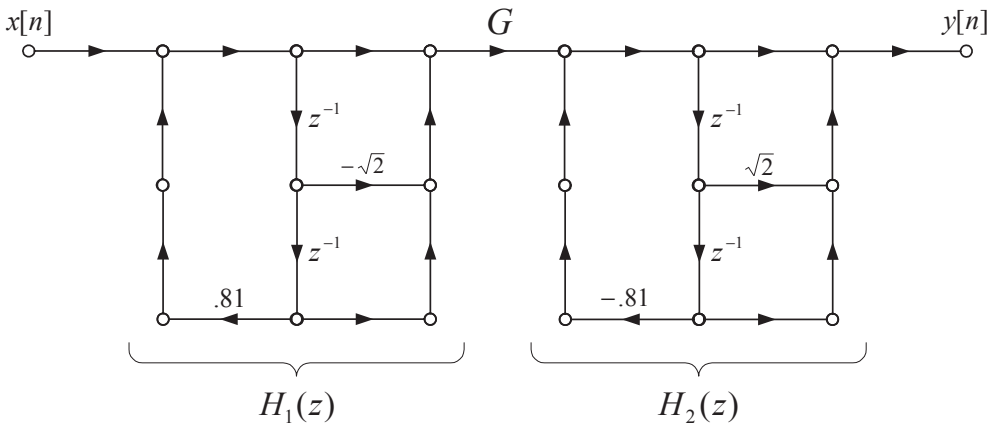


Fig. 8.10 Example of a cascade realization

$H_2(z)$. In practice, the gain G could be distributed over the SFG so as to optimize the use of the available dynamic range of the processor (more on this later). ◀

8.2.4 Parallel form

Based on the partial fraction expansion of $H(z)$, one can easily express the latter as a sum:

$$H(z) = C(z) + \sum_{k=1}^{K} H_K(z) \tag{8.17}$$

$$H_k(z): \text{ low-order IIR sections} \tag{8.18}$$

$$C(z): \text{ FIR section (if needed)} \tag{8.19}$$

The corresponding SFG is shown in Figure 8.11, where each box needs to be expanded with a proper sub-SFG. Typically, second-order IIR sections $H_k(z)$ would be obtained by combining terms in the PFE that correspond to complex conjugate poles:

$$H_k(z) = \frac{A_k}{1-p_k z^{-1}} + \frac{A_k^*}{1-p_k^* z^{-1}} \tag{8.20}$$

$$= \frac{b_{k0}+b_{k1}z^{-1}}{1-a_{k1}z^{-1}-a_{k2}z^{-2}} \tag{8.21}$$

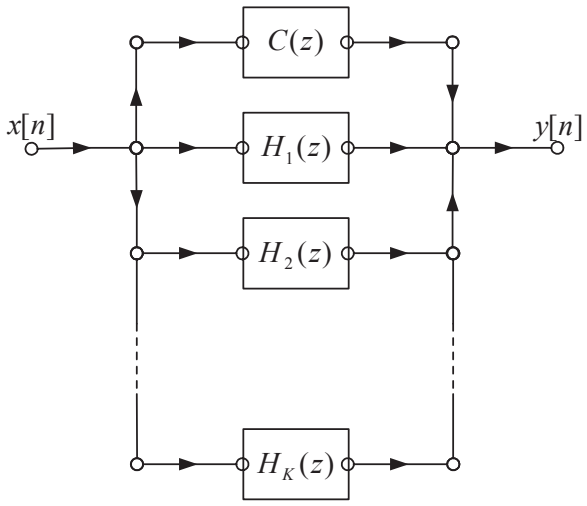


Fig. 8.11 Parallel Realization of an IIR system.

where $a_{ki}, b_{ki} \in \mathbb{R}$. A DFII or transposed DFII structure would be used for the realization of (8.21).

Example 8.5:

► Consider the system function in Example 8.4

$$H(z) = G\frac{(1-e^{j\pi/4}z^{-1})(1-e^{-j\pi/4}z^{-1})(1-e^{j3\pi/4}z^{-1})(1-e^{-j3\pi/4}z^{-1})}{(1-.9z^{-1})(1+.9z^{-1})(1-.9jz^{-1})(1+.9jz^{-1})}$$

It is not difficult to verify that it has the following PFE:

$$H(z) = A + \frac{B}{1-.9z^{-1}} + \frac{C}{1+.9z^{-1}} + \frac{D}{1-.9jz^{-1}} + \frac{D^*}{1+.9jz^{-1}}$$

where A, B and C are appropriate real valued coefficients and D is complex valued. Again, there are several ways of pairing the poles of $H(z)$ to obtain 2nd order sections. In practice, terms corresponding to complex conjugate poles should be combined so that the resulting second order section has real coefficients. Applying this approach, we obtain

$$H_1(z) = \frac{B}{1-.9z^{-1}} + \frac{C}{1+.9z^{-1}} = \frac{b_{10}+b_{11}z^{-1}}{1-.81z^{-2}}$$

$$H_2(z) = \frac{D}{1-.9jz^{-1}} + \frac{D^*}{1+.9jz^{-1}} = \frac{b_{20}+b_{21}z^{-1}}{1+.81z^{-2}}$$

The corresponding SFG is illustrated below with $H_1(z)$ and $H_2(z)$ realized in DFII: ◄

8.2.5 Transposed direct form II

Transposition theorem:

Let $\mathcal{G}_1$ denote a SFG with system function $H_1(z)$. Let $\mathcal{G}_2$ denote the SFG obtained from $\mathcal{G}_1$ by:

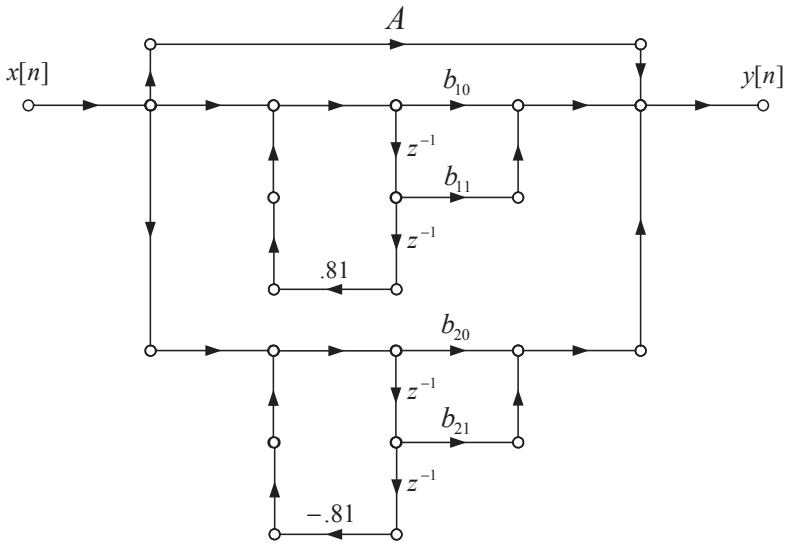


Fig. 8.12 Example of a parallel realization

(1) reversing the direction of all the branches, and

(2) interchanging the source $x[n]$ and the sink $y[n]$.

Then

$$H_2(z) = H_1(z) \tag{8.22}$$

Note that only the branch direction is changed, not the transmittance. Usually, we redraw $\mathcal{G}_2$ so that the source node $x[n]$ is on the left.

Example 8.6:

► An example flowgraph is represented on the left of Figure 8.13. Its transfer function is easily found to be

$$H_1(z) = \frac{\frac{bz^{-1}}{1-abz^{-1}}}{1-\frac{cbz^{-1}}{1-abz^{-1}}}.$$

The application of the transposition theorem yields the SFG on the right of Figure 8.13, which can easily be checked to have the same transfer function. ◄

Application to DFII:

Consider the general rational transfer function

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1-\sum_{k=1}^{N} a_k z^{-k}}. \tag{8.23}$$

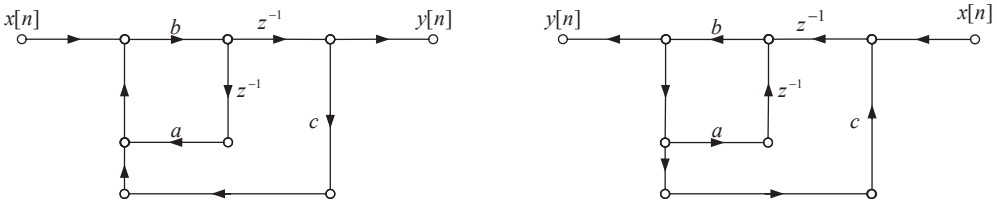


Fig. 8.13 Example of application of the transposition theorem: left, original SFG and right, transposed SFG.

with its DFII realization, as shown in Figure 8.8. Applying the transposition theorem as in the previous example, we obtain the SFG shown in figure 8.14, known as the transposed direct form II. Note that $N = M$ is assumed for convenience.

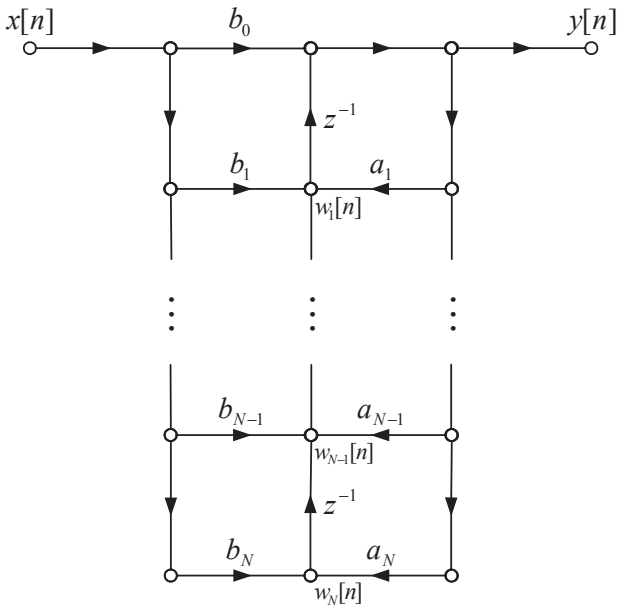


Fig. 8.14 Transposed DFII realization of a rational system.

The corresponding LCCDEs are :

$$\begin{aligned} w_N[n] &= b_N x[n] + a_N y[n] \\ w_{N-1}[n] &= b_{N-1} x[n] + a_{N-1} y[n] + w_N[n-1] \\ &\vdots \\ w_1[n] &= b_1 x[n] + a_1 y[n] + w_2[n-1] \\ y[n] &= w_1[n-1] + b_0 x[n] \end{aligned} \tag{8.24}$$

8.3 Realizations of FIR systems

8.3.1 Direct forms

In the case of an FIR system, the denominator in the rational system function (8.9) reduces to $A(z) = 1$. That is

$$H(z) = B(z) = \sum_{k=0}^{M} b_k z^{-k}, \tag{8.25}$$

corresponding to the following non-recursive LCCDE in the time-domain:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{8.26}$$

Accordingly, the DFI and DFII realizations become equivalent, as the structural component labelled $1/A(z)$ in Figure 8.6 disappears. The resulting SFG is shown in Figure 8.15; it is also called a tapped delay line or a transversal filter structure.

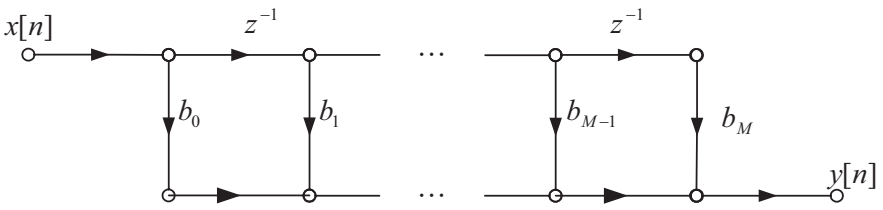


Fig. 8.15 DF for an FIR system (also called tapped delay line or transversal filter).

Note that the coefficients b_k in Figure 8.15 directly define the impulse response of the system. Applying a unit pulse as the input, i.e. $x[n] = \delta[n]$, produces an output $y[n] = h[n]$, with

$$h[n] = \begin{cases} b_n & \text{if } n \in \{0,1,\ldots,M\} \\ 0 & \text{otherwise.} \end{cases} \tag{8.27}$$

8.3.2 Cascade form

In the case of an FIR system, a cascade form is also possible but the problem is in fact simpler than for IIR since only the zeros need to be considered (because $A(z) = 1$).

To decompose $H(z)$ in (8.25) as a product of low-order FIR sections, as in

$$H(z) = H_1(z)H_2(z)\cdots H_K(z)$$

one needs to find the zeros of $H(z)$, say z_l ($l = 1,\ldots,M$), and form the desired low-order sections by properly grouping the corresponding factors $1 - z_l z^{-1}$ and multiplying by the appropriate gain term. For example, a typical 2nd order section would have the form

$$H_k(z) = G_k(1 - z_{k_1}z^{-1})(1 - z_{k_2}z^{-1}) = b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}.$$

where the coefficients b_{ki} can be made real by proper choice of complex conjugate zeros (i.e. $z_{k_1} = z_{k_2}^*$).

In practice, cascade realizations of FIR system use sections of order 1, 2 and/or 4. Sections of order 4 are useful in the efficient realization of linear-phase system...

Example 8.7:

▶ Consider the FIR system function

$$H(z) = (1 - e^{j\pi/4}z^{-1})(1 - e^{-j\pi/4}z^{-1})(1 - e^{j3\pi/4}z^{-1})(1 - e^{-j3\pi/4}z^{-1})$$

A cascade realization of H(z) as a product of 2nd order sections can be obtained as follows:

$$H_1(z) = (1 - e^{j\pi/4}z^{-1})(1 - e^{-j\pi/4}z^{-1}) = 1 - \sqrt{2}z^{-1} + z^{-2}$$

$$H_2(z) = (1 - e^{j3\pi/4}z^{-1})(1 - e^{-j3\pi/4}z^{-1}) = 1 + \sqrt{2}z^{-1} + z^{-2}$$

◀

8.3.3 Linear-phase FIR systems

Definition: A LTI system is said to have generalized linear-phase (GLP) if its frequency response is expressible as

$$H(\omega) = A(\omega)e^{-j(\alpha\omega - \beta)} \tag{8.28}$$

where the function $A(\omega)$ and the coefficients α and β are real valued.

Note that $A(\omega)$ can be negative, so that in general, $A(\omega) \neq |H(\omega)|$. In terms of $A(\omega)$, α and β, we have

$$\measuredangle H(\omega) = \begin{cases} -\alpha\omega + \beta & \text{if } A(\omega) > 0 \\ -\alpha\omega + \beta + \pi & \text{if } A(\omega) < 0 \end{cases} \tag{8.29}$$

Despite the phase discontinuities, it is common practice to refer to $H(\omega)$ as a linear phase system.

Theorem: Consider an LTI system $\mathcal{H}$ with FIR $h[n]$, such that

$$\begin{aligned} &h[n] = 0 \text{ for } n < 0 \text{ and for } n > M \\ &h[0] \neq 0 \text{ and } h[M] \neq 0 \end{aligned} \tag{8.30}$$

System $\mathcal{H}$ is GLP if and only if $h[n]$ is either

(i) symmetric, i.e.:

$$h[n] = h[M-n], \quad n \in \mathbb{Z} \tag{8.31}$$

(ii) or anti-symmetric, i.e.:

$$h[n] = -h[M-n], \quad n \in \mathbb{Z} \tag{8.32}$$

Note that the GLP conditions (8.31) and (8.32) can be combined into a single equation, namely

$$h[n] = \varepsilon\, h[M-n] \tag{8.33}$$

where ε is either equal to $+1$ (symmetric) or -1 (anti-symmetric).

Example 8.8:

► Consider the LTI system with impulse response

$$h[n] = \{\underline{1}, 0, 1, 0, 1\}$$

Note that $h[n]$ satisfies the definition (8.31) of a symmetric impulse response with $M = 4$:

$$h[n] = h[4-n], \quad \text{all } n \in \mathbb{Z}$$

Let us verify that the system is linear phase, or equivalently, that its frequency response verifies the condition (8.28). We have

$$\begin{aligned} H(\omega) &= \sum_{n=-\infty}^{\infty} h[n] e^{-j\omega n} \\ &= 1 + e^{-j2\omega} + e^{-j4\omega} \\ &= e^{-j2\omega}(e^{j2\omega} + 1 + e^{-j2\omega} \\ &= e^{-j2\omega}(1 + 2\cos(2\omega)) \\ &= A(\omega) e^{-j(\alpha\omega+\beta)} \end{aligned}$$

where we identify

$$A(\omega) = 1 + 2\cos(2\omega),$$

$$\alpha = 2, \qquad \beta = 0$$

◄

Modified direct form: The symmetry in $h[n]$ can be used advantageously to reduce the number of multiplications in direct form realization:

$$h[n] = \varepsilon h[M-n] \Longrightarrow b_k = \varepsilon b_{M-k} \tag{8.34}$$

In the case M odd, we have

$$\begin{aligned} y[n] &= \sum_{k=0}^{M} b_k x[n-k] \\ &= \sum_{k=0}^{(M-1)/2} (b_k x[n-k] + b_{M-k} x[n-(M-k)]) \\ &= \sum_{k=0}^{(M-1)/2} b_k (x[n-k] + \varepsilon\, x[n-(M-k)]) \end{aligned} \tag{8.35}$$

which only requires $(M+1)/2$ multiplications instead of $M+1$.

Figure 8.16 illustrates a modified DFI structure which enforces linear phase.

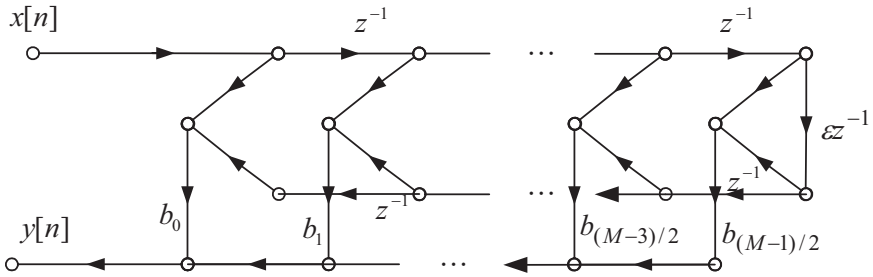


Fig. 8.16 A modified DF structure to enforce linear phase (M odd).

Properties of the zeros of a GLP system: In the z-domain the GLP condition $h[n] = \varepsilon h[M-n]$ is equivalent to

$$H(z) = \varepsilon z^{-M} H(z^{-1}) \tag{8.36}$$

Thus, if z_0 is a zero of $H(z)$, so is $1/z_0$. If in addition $h[n]$ is real, then z_0, z_0^*, $1/z_0$ and $1/z_0^*$ are all zeros of $H(z)$. This is illustrated in Figure 8.17.

As a result, sections of order 4 are often used in the cascade realization of real, linear phase FIR filters:

$$H_k(z) = G_k(1 - z_k z^{-1})(1 - z_k^* z^{-1})(1 - \frac{1}{z_k} z^{-1})(1 - \frac{1}{z_k^*} z^{-1}) \tag{8.37}$$

8.3.4 Lattice realization of FIR systems

Lattice stage: The basic lattice stage is a 2-input 2-output DT processing device. Its signal flow graph is illustrated in Figure 8.18.

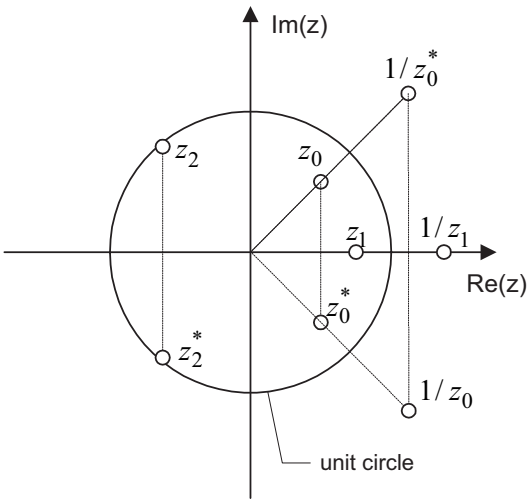


Fig. 8.17 Illustration of the zero locations of FIR real-coefficient GLP systems: a zero always appears with its inverse and complex conjugate.

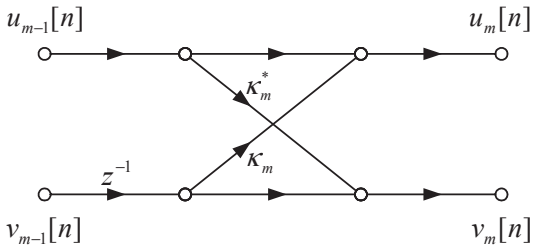


Fig. 8.18 Lattice stage.

The parameter κ_m is called reflection coefficient, while m is an integer index whose meaning will be explained shortly.

Input-output characterization in the time domain:

$$u_m[n] = u_{m-1}[n] + \kappa_m v_{m-1}[n-1] \tag{8.38}$$
$$v_m[n] = \kappa_m^* u_{m-1}[n] + v_{m-1}[n-1] \tag{8.39}$$

Equivalent z-domain matrix representation:

$$\begin{bmatrix} U_m(z) \\ V_m(v) \end{bmatrix} = K_m(z) \begin{bmatrix} U_{m-1}(z) \\ V_{m-1}(v) \end{bmatrix} \tag{8.40}$$

where $K_m(z)$ is a 2×2 transfer matrix defined as

$$K_m(z) \triangleq \begin{bmatrix} 1 & \kappa_m z^{-1} \\ \kappa_m^* & z^{-1} \end{bmatrix} \tag{8.41}$$

Lattice filter: A lattice filter is made up of a cascade of M lattice stages, with index m running from 1 to M. This is illustrated in figure 8.19.

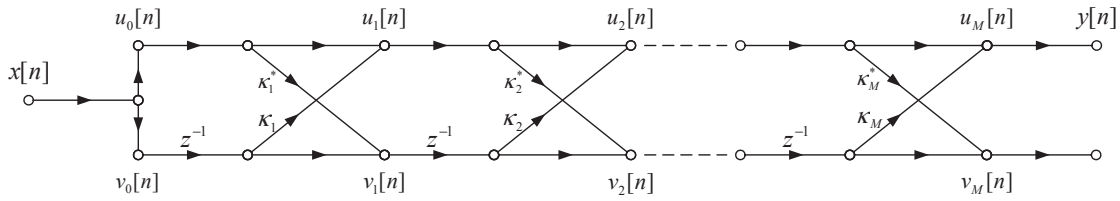


Fig. 8.19 A lattice filter as a series of lattice stages.

Corresponding set of difference equations characterizing the above SFG in the time domain:

$u_0[n] = v_0[n] = x[n]$
for $m = 1, 2, \ldots, M$
$\quad u_m[n] = u_{m-1}[n] + \kappa_m v_{m-1}[n-1]$
$\quad v_m[n] = \kappa_m^* u_{m-1}[n] + v_{m-1}[n-1]$
end
$y[n] = u_M[n]$

Corresponding matrix representation in the z-domain:

$$U_0(z) = V_0(z) = X(z) \tag{8.42}$$
$$\begin{bmatrix} U_M(z) \\ V_M(v) \end{bmatrix} = K_M(z) \ldots K_2(z) K_1(z) \begin{bmatrix} U_0(z) \\ V_0(v) \end{bmatrix} \tag{8.43}$$
$$Y(z) = U_M(z) \tag{8.44}$$

System function:

- From (8.42)– (8.44), it follows that $Y(z) = H(z)X(z)$ with the equivalent system function $H(z)$ given by

$$H(z) = \begin{bmatrix} 1 & 0 \end{bmatrix} K_M(z) \ldots K_2(z) K_1(z) \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{8.45}$$

- Expanding the matrix product in (8.45), it is easy to verify that the system function of the lattice filter is expressible as

$$H(z) = 1 + b_1 z^{-1} + \cdots + b_M z^{-M} \tag{8.46}$$

  where each the coefficients b_k in (8.46) can be expressed algebraically in terms of the reflection coefficients $\{\kappa_m\}_{m=1}^{M}$

- Conclusion: an M-stage lattice filter is indeed FIR filter of length $M+1$.

Minimum-phase property: The lattice realization is guaranteed to be minimum-phase if the reflection coefficients are all less than one in magnitude, that is: if $|\kappa_m| < 1$ for all $m = 1, \ldots, M$.

Note: Thus, by constraining the reflection coefficients as above, we can be sure that a causal and stable inverse exists for the lattice system. The proof of this property is beyond the scope of this course.

Chapter 9

Filter design

9.1 Introduction

9.1.1 Problem statement

Consider the rational system function

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} \tag{9.1}$$

for which several computational realizations have been developed in the preceding Chapter.

In filter design, we seek to find the system coefficients, i.e. $M, N, a_1, \ldots, a_N$, $b_0, \ldots, b_M$, in (9.1) such that the corresponding frequency response

$$H(\omega) = H(z)|_{z=e^{j\omega}}$$

provides a good approximation to a desired response $H_d(\omega)$, i.e.

$$H(\omega) \sim H_d(\omega). \tag{9.2}$$

In order to be practically realizable, the resulting system $H(z)$ must be stable and causal, which imposes strict constraints on the possible values of a_k. Additionally, a linear phase requirement for $H(z)$ may be desirable in certain applications, which further restrict the system parameters.

It is important to realize that the use of an approximation in (9.2) is a fundamental necessity in practical filter design. In the vast majority of problems, the desired response $H_d(\omega)$ does not correspond to a stable and causal system and cannot even be expressed exactly as a rational system function.

To illustrate this point, recall from Chapter 2 that stability of $H(z)$ implies that $H(\omega)$ is continuous, while ideal frequency selective filters $H_d(\omega)$ all have discontinuities.

Example 9.1: Ideal Low-Pass Filter

▶ The ideal low-pass filter has already been studied in Example 3.3. The corresponding frequency response, denoted $H_d(\omega)$ in the present context, is illustrated in Figure 9.1. It is not continuous, and the

corresponding impulse response

$$h_d[n] = \frac{\omega_c}{\pi}\,\mathrm{sinc}(\frac{\omega_c n}{\pi}) \tag{9.3}$$

is easily seen to be non-causal. Furthermore, it is possible to show that

$$\sum_n |h_d[n]| = \infty$$

so that the ideal low-pass filter is unstable. ◄

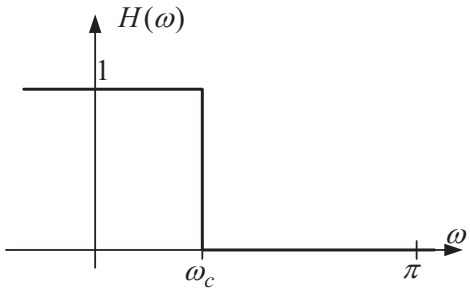


Fig. 9.1 Ideal Low-Pass filter specification in frequency.

9.1.2 Specification of $H_d(\omega)$

For frequency selective filters, the magnitude of the desired response is usually specified in terms of the tolerable

- pass-band distortion,
- stop-band attenuation, and
- width of transition band.

An additional requirement of linear phase (GLP) may be specified.

For other types of filters, such as all-pass equalizers and Hilbert transformers, phase specifications play a more important role.

Typical low-pass specifications: A typical graphical summary of design specifications for a discrete-time low-pass filter is shown in Figure 9.2. The parameters are:

- $[0, \omega_p]$ = pass-band
- $[\omega_s, \pi]$ = stop-band
- $\delta\omega \triangleq \omega_s - \omega_p$ = width of transition band.
- δ_1 = pass-band ripple. Often expressed in dB via $20\log_{10}(1+\delta_1)$.
- δ_2 = stop-band ripple. Usually expressed in dB as $20\log_{10}(\delta_2)$.

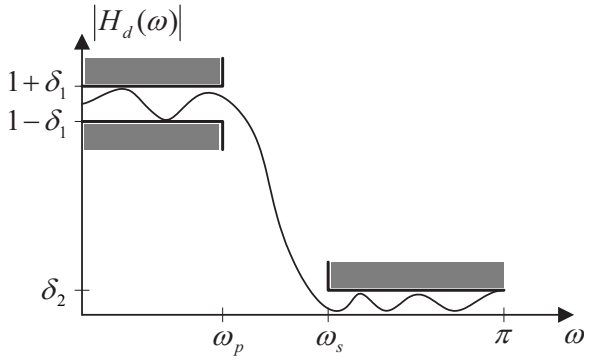


Fig. 9.2 Typical template summarizing the design specifications for a low-pass filter.

The phase response in the pass-band may also be specified, for example by an additional GLP requirement.

Typically, a reduction of δ_1, δ_2 and/or $\delta\omega$ leads to an increase in the required IIR filter order N or FIR filter length M, and thus greater implementation costs. Ideally, one would to use the minimum value of N and/or M for which the filter specifications can be met.

9.1.3 FIR or IIR, That Is The Question

The choice between IIR and FIR is usually based on a consideration of the phase requirements. Recall that a LTI system is GLP iff

$$H(\omega) = A(\omega)e^{-j(\alpha\omega-\beta)} \tag{9.4}$$

where the amplitude function $A(\omega)$ and the phase parameters α and β are real valued.

Below, we explain how the GLP requirement influences the choice between FIR and IIR. In essence, only FIR filter can be at the same time stable, causal and GLP. This is a consequence of the following theorem, stated without proof.

Theorem: A stable LTI system $H(z)$ with real impulse response, i.e. $h[n] \in \mathbb{R}$, is GLP if and only if

$$H(z) = \varepsilon z^{-2\alpha} H(1/z) \tag{9.5}$$

where $\alpha \in \mathbb{R}$ and $\varepsilon \in \{-1,+1\}$.

PZ symmetry:

- Suppose that p is a pole of $H(z)$ with $0 < |p| < 1$.
- According to above theorem, if $H(z)$ is GLP, then:

$$H(1/p) = \varepsilon\, p^{2\alpha} H(p) = \infty \tag{9.6}$$

  which shows that $1/p$ is also a pole of $H(z)$.

- Note that p^* and $1/p^*$ are also poles of $H(z)$ under the assumption of a real system.
- The above symmetry also apply to the zeros of $H(z)$
- The situation is illustrated in the PZ diagram below:

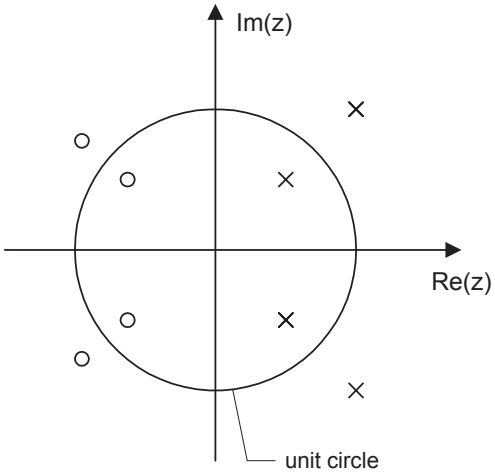


Fig. 9.3 PZ locations of IIR real-coefficient GLP systems: a pole always appears with its inverse (and complex conjugate).

Conclusion: If $H(z)$ is stable and GLP, to any non-trivial pole p inside the unit circle corresponds a pole $1/p$ outside the unit circle, so that $H(z)$ cannot have a causal impulse response. In other words, only FIR filter can be at the same time stable, causal and GLP.

Basic design principle:

- If GLP is essential $\Longrightarrow$ FIR
- If not $\Longrightarrow$ IIR usually preferable (can meet specifications with lower complexity)

9.2 Design of IIR filters

Mathematical setting: We analyze in this section common techniques used to design IIR filters. Of course, we only concentrate on filters with rational system functions, as

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_M z^{-M}}{1 + a_1 z^{-1} + \ldots + a_N z^{-N}}. \tag{9.7}$$

for which practical computational realizations are available.

For $H(z)$ in (9.7) to be the system function of a causal & stable LTI system, all its poles have to be inside the unit circle (U.C.).

Design problem: The goal is to find filter parameters $N, M, \{a_k\}$ and $\{b_k\}$ such that $H(\omega) \sim H_d(\omega)$ to some desired degree of accuracy.

Transformation methods: There exists a huge literature on analog filter design. In particular, several effective approaches do exist for the design of analog IIR filters, say $H_a(\Omega)$. The transformation methods try to take advantage of this literature in the following way:

(1) Map DT specifications $H_d(\omega)$ into analog specifications $H_{ad}(\Omega)$ via proper transformation;

(2) Design analog filter $H_a(\Omega)$ that meets the specifications;

(3) Map $H_a(\Omega)$ back into $H(\omega)$ via a proper inverse transformation.

9.2.1 Review of analog filtering concepts

LTI system: For an analog LTI system, the input/ouput characteristic in the time domain takes the form of a convolution integral:

$$y_a(t) = \int_{-\infty}^{\infty} h_a(u) x_a(t-u) du \tag{9.8}$$

where $h_a(t)$ is the impulse response of the system. Note here that $t \in \mathbb{R}$,

System function: The system function of an analog LTI system is defined as the Laplace transform of its impulse response, i.e.

$$H_a(s) = \mathcal{L}\{h_a(t)\} = \int_{-\infty}^{\infty} h_a(t) e^{-st} dt \tag{9.9}$$

where $\mathcal{L}$ denotes the Laplace operator. The complex variable s is often expressed in terms of its real and imaginary parts as

$$s = \sigma + j\Omega$$

The set of all $s \in \mathbb{C}$ where integral (9.9) converges absolutely defines the region of convergence (ROC) of the Laplace transform. A complete specification of the system funciton $H_a(s)$ must include a description of the ROC.

In the s-domain, the convolution integral (9.8) is equivalent to

$$Y_a(s) = H_a(s) X_a(s) \tag{9.10}$$

where $Y_a(s)$ and $X_a(s)$ are the Laplace transforms of $y_a(t)$ and $x_a(t)$.

Causality and stability: An analog LTI system is causal iff the ROC of its system function $H_a(s)$ is a right-half plane, that is

$$\text{ROC} : \Re(s) > \sigma_0$$

An analog LTI system is stable iff the ROC of $H_a(s)$ contains the $j\Omega$ axis:

$$j\Omega \in \text{ROC for all } \Omega \in \mathbb{R}$$

The ROC of a stable and causal analog LTI system is illustrated in the Fig. 9.4.

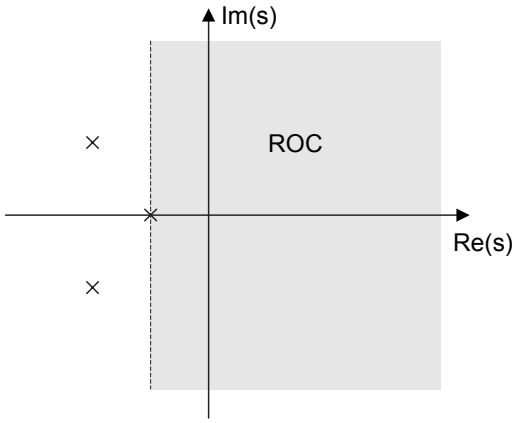


Fig. 9.4 Illustration of a typical ROC for a causal and stable analog LTI system.

Frequency response: The frequency response of an LTI system is defined as the Fourier transform of its impulse response, or equivalently [1]

$$H_a(\Omega) = H_a(s)|_{s=j\Omega} \tag{9.11}$$

In the case an analog system with a real impulse response, we have

$$|H_a(\Omega)|^2 = H_a(s)H_a(-s)|_{s=j\Omega} \tag{9.12}$$

Rational systems: As the name indicates, a rational system is characterized by a system function that is a rational function of s, typically:

$$H_a(s) = \frac{\beta_0 + \beta_1 s + \cdots + \beta_M s^M}{1 + \alpha_1 s + \cdots + \alpha_N s^N} \tag{9.13}$$

The corresponding input-output relationship in the time-domain is an ordinary differential equation of order N, that is:

$$y_a(t) = -\sum_{k=1}^{N} \alpha_k \frac{d^k}{dt^k} y_a(t) + \sum_{k=0}^{M} \beta_k \frac{d^k}{dt^k} x_a(t) \tag{9.14}$$

9.2.2 Basic analog filter types

In this course, we will basically consider only three main types of analog filters in the design of discrete-time IIR filters via transformation methods. These three types will have increasing "complexity" in the sense of being able to design a filter just with a paper and pencil, but will also have an increasing efficiency in meeting a prescribed set of specifications with a limited number of filter parameters.

Butterworth filters: Butterworth filters all have a common shape of squared magnitude response:

$$|H_a(\Omega)|^2 = \frac{1}{1 + (\Omega/\Omega_c)^{2N}}, \tag{9.15}$$

[1] Note the abuse of notation here: we should formally write $H_a(j\Omega)$ but usually drop the j for simplicity.

where Ω_c is the cut-off frequency (-3dB) and N is the filter order. Figure 9.5 illustrates the frequency response of several Butterworth filters of different orders. Note that $|H_a(\Omega)|$ smoothly decays from a maxi-

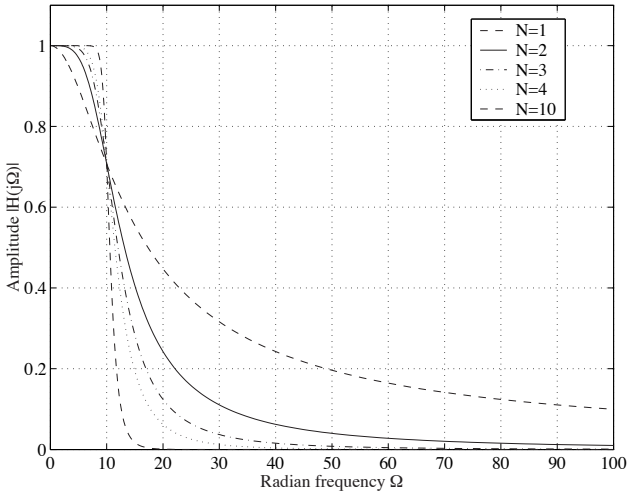


Fig. 9.5 Magnitude response of several Butterworth filters of different orders. ($\Omega_c = 10 rad/s$)

mum of 1 at $\Omega = 0$ to 0 at $\Omega = \infty$. These filters are simple to compute, and can be easily manipulated without the help of a computer. The corresponding transfer function is given by:

$$H_a(s) = \frac{(\Omega_c)^N}{(s-s_0)(s-s_1)\cdots(s-s_{N-1})} \tag{9.16}$$

The poles s_k are uniformly spaced on a circle of radius Ω_c in the s-place, as given by:

$$s_k = \Omega_c \exp(j[\frac{\pi}{2} + (k+\frac{1}{2})\frac{\pi}{N}]) \quad k = 0,1,\ldots,N-1. \tag{9.17}$$

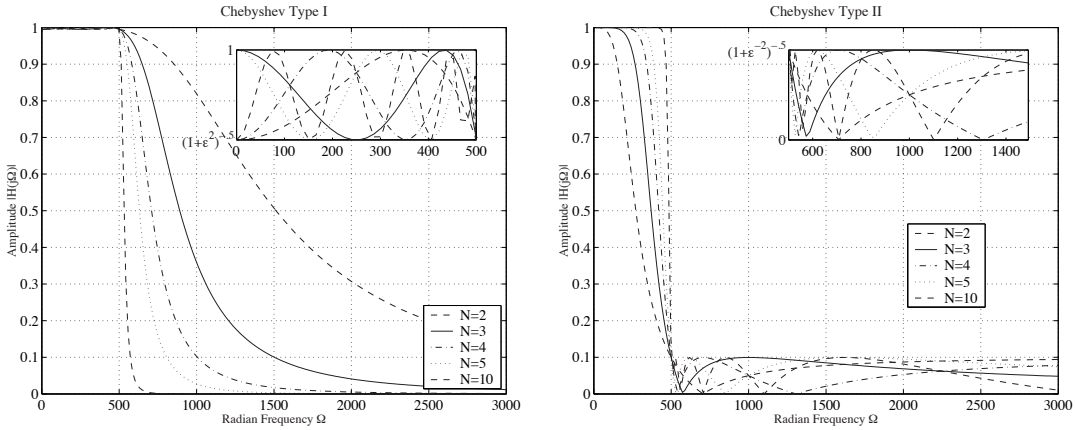
Example 9.2:

► ◄

Chebyshev filters: Chebyshev filters come in two flavours: type I and II. The main difference with the Butterworth filters is that they are not monotonic in their magnitude response: Type I Chebyshev filters have ripple in their passband, while Type II Chebyshev filters have ripple in their stopband. The table below summarizes the features of the two types of Chebyshev filters, and also gives an expression of their squared magnitude response:

	Type I	Type II
Def	$\lvert H_c(\Omega)\rvert^2 = \frac{1}{1+\varepsilon^2 T_N^2(\Omega/\Omega_c)}$	$\lvert H_c(\Omega)\rvert^2 = \frac{1}{1+\left(\varepsilon^2 T_N^2(\Omega_c/\Omega)\right)^{-1}}$
Passband	Ripple from 1 to $\sqrt{1/1+\varepsilon^2}$	Monotonic
Stopband	Monotonic	Ripple $\sqrt{1/(1+1/\varepsilon^2)}$

In the above table, the function $T_N(\cdot)$ represents the N-th order Chebyshev polynomial. The details of the derivation of these polynomials is beyond the scope of this text, but can be obtained from any standard textbook. ε is a variable that determines the amount of ripple, and Ω_c is a cut-off frequency. Figure 9.6 illustrates the frequency response of these filters. Chebyshev filters become more complicated to deal with than Butterworth filters, mainly because of the presence of the Chebyshev polynomials in their definition, and often they will be designed through an ad-hoc computer program.



(a) (b)

Fig. 9.6 Illustration of the magnitude response of analog Chebyshev filters of different orders N: (a) Type I, (b) Type II.

Elliptic filters: Elliptic filters are based on elliptic functions. Their exact expression is beyond the scope of this text, but they can be easily generated by an appropriate computer program. Figure 9.7 illustrates the magnitude response of these filters. One can see that they ripple both in the passband and stop band. They usually require a lower order than Chebyshev and Butterworth filters to meet the same specifications.

9.2.3 Impulse invariance method

Principle: In this method of transformation, the impulse response of the discrete-time filter $H(z)$ is obtained by sampling the impulse response of a properly designed analog filter $H_a(s)$:

$$h[n] = h_a(nT_s) \tag{9.18}$$

where T_s is an appropriate sampling period. From sampling theory in Chapter 7, we know that

$$H(\omega) = \frac{1}{T_s}\sum_k H_a\left(\frac{\omega}{T_s} - \frac{2\pi k}{T_s}\right). \tag{9.19}$$

If there is no significant aliasing, then

$$H(\omega) = \frac{1}{T_s}H_a\left(\frac{\omega}{T_s}\right), \quad |\omega| < \pi. \tag{9.20}$$

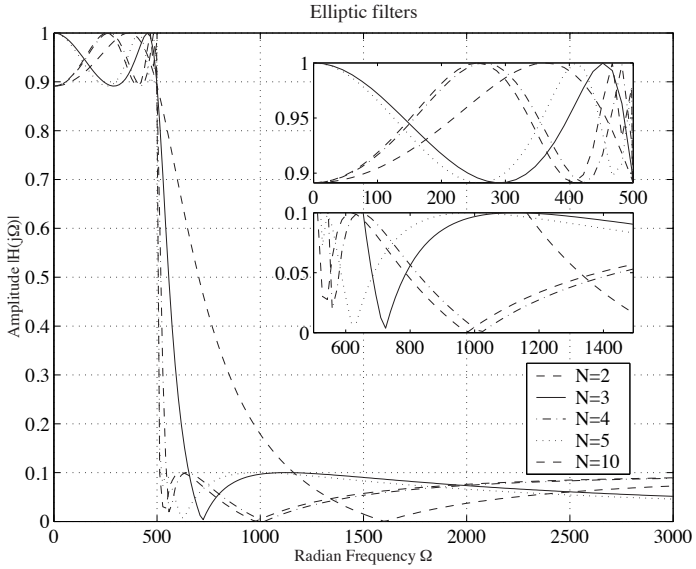


Fig. 9.7 Magnitude Response of low-pass elliptic filters of different orders.

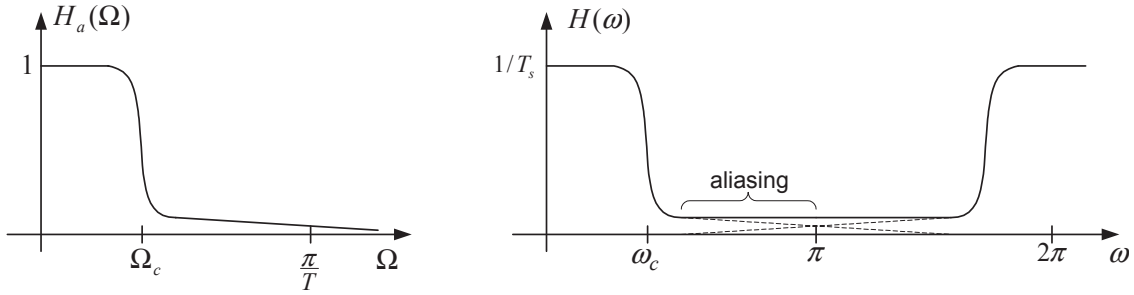


Fig. 9.8 Illustration of aliasing with impulse invariance method.

Thus in the absence of aliasing, the frequency response of the DT filter is the same (up to a scale factor) as the frequency response of the CT filter.

Clearly, this method works only if $|H_a(\Omega)|$ becomes very small for $|\Omega| > \pi/T_s$, so that aliasing can be neglected. Thus, it can be used to design low-pass and band-pass filters, but not high-pass filters.

In the absence of additional specifications on $H_a(\Omega)$, the parameter T_s can be set to 1: it is a *fictitious* sampling period, that does not correspond to any physical sampling.

Design steps:

(1) Given the desired specifications $H_d(\omega)$ for the DT filter, find the corresponding specifications for the

analog filter, $H_{cd}(\Omega)$:

$$H_{ad}(\Omega) = T_s H_d(\omega)|_{\omega=\Omega T_s}, \quad |\Omega| < \frac{\pi}{T_s} \tag{9.21}$$

For example, Figure 9.9 illustrates the conversion of specifications from DT to CT for the case of a low-pass design.

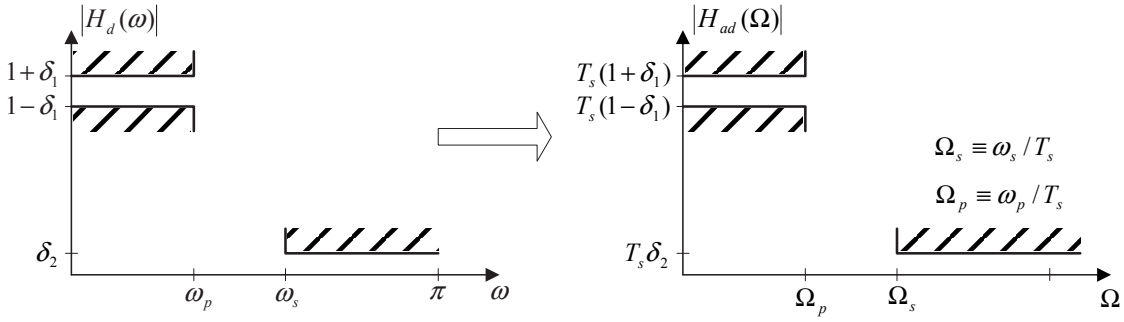


Fig. 9.9 Conversion of specifications from discrete-time to continuous-time for a design by impulse invariance.

(2) Design an analog IIR filter $H_a(\Omega)$ that meets the desired analog specifications $H_{ad}(\Omega)$:
- Chose the filter type (Butterworth, elliptic, etc.)
- Select the filter parameters (filter order N, poles s_k, etc.)

(3) Transform the analog filter $H_a(\Omega)$ into a discrete-time filter $H(z)$ via time-domain sampling of the impulse response:

$$\begin{aligned} h_a(t) &= \mathcal{L}^{-1}\{H_a(\Omega)\} \\ h[n] &= h_a(nT_s) \\ H(z) &= \mathcal{Z}\{h[n]\} \end{aligned} \tag{9.22}$$

In practice, it is not necessary to find $h_a(t)$, as explained below.

Simplification of step (3): Assume $H_a(s)$ has only first order poles (e.g. Butterworth filters). By performing a partial fraction expansion of $H_a(s)$, one gets:

$$H_a(s) = \sum_{k=1}^{N} \frac{A_k}{s - s_k} \tag{9.23}$$

The inverse Laplace transform of $H_a(s)$ is given by

$$h_a(t) = \sum_{k=1}^{N} A_k e^{s_k t} u_c(t) \tag{9.24}$$

where $u_a(t)$ is the analog unit step function, i.e.

$$u_c(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases} \tag{9.25}$$

Now, if we sample $h_a(t)$ in (9.24) uniformly at times $t = nT_s$, we obtain

$$h[n] = \sum_{k=1}^{N} A_k e^{s_k nT_s} u[n] = \sum_{k=1}^{N} A_k (e^{s_k T_s})^n u[n]. \tag{9.26}$$

The Z-transform of $h[n]$ is immediately obtained as:

$$H(z) = \sum_{k=1}^{N} \frac{A_k}{1 - p_k z^{-1}} \quad \text{where} \quad p_k = e^{s_k T_s} \tag{9.27}$$

This clearly shows that there is no need to go through the actual steps of inverse Laplace transform, sampling and Z-transform: one only needs the partial fraction expansion coefficients A_k and the poles s_k from (9.23) and plug them into (9.27).

Remarks: The order of the filter is preserved by impulse invariance, that is: if $H_a(s)$ is of order N, then so is $H(z)$. The correspondence between poles in discrete-time and continuous-time is given by:

$$s_k \text{ is a pole of } H_a(s) \Longrightarrow p_k = e^{s_k T_s} \text{is a pole of } H(z).$$

As a result, stability and causality of the analog filter are preserved by the transformation. Indeed, if $H_a(s)$ is stable and causal, then the real part of all its poles s_k will be negative:

$$s_k = \sigma_k + j\Omega_k, \quad \sigma_k < 0.$$

The corresponding poles p_k in discrete-time can be written as

$$p_k = e^{\sigma_k T_s} e^{j\Omega_k T_s}$$

Therefore,

$$|p_k| = e^{\sigma_k T_s} < 1,$$

which ensures causality and stability of the discrete-time filter $H(z)$.

Example 9.3:

- ▶ Apply the impulse invariance method to an appropriate analog Butterworth filter in order to design a low-pass digital filter $H(z)$ that meets the specifications outlined in Figure 9.10.
  - Step 1: We set the sampling period to $T_s = 1$ (i.e. $\Omega = \omega$), so that the desired specifications for the analog filter are the same as stated in Figure 9.10, or equivalently:

$$1 - \delta_1 \le |H_{ad}(\Omega)| \le 1 \quad \text{for} \quad 0 \le |\Omega| \le \Omega_1 = .25\pi \tag{9.28}$$

$$|H_{ad}(\Omega)| \le \delta_2, \quad \text{for} \quad \Omega_2 = .4\pi \le |\Omega| \le \infty \tag{9.29}$$

  - Step 2: We need to design an analog Butterworth filter $H_c(\Omega)$ that meets the above specifications. For the Butterworth family, we have

$$|H_a(\Omega)|^2 = \frac{1}{1 + (\Omega/\Omega_c)^{2N}} \tag{9.30}$$

    Thus, we need to determine the filter order N and the cut-off frequency Ω_c so that the above inequalities are satisfied. This is explained below:

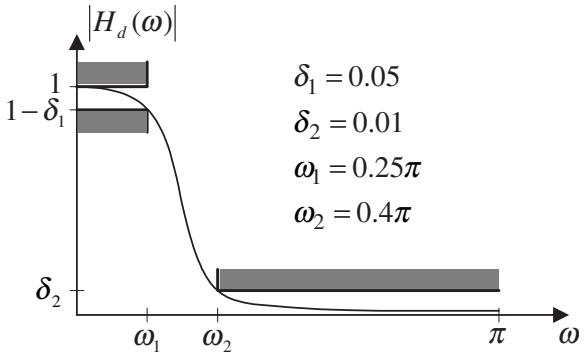


Fig. 9.10 Specifications for the low-pass design of example 9.3

- We first try to an find analog Butterworth filter that meets the band edge conditions exactly, that is:

$$|H_a(\Omega_1)| = 1-\delta_1 \quad \Rightarrow \quad \frac{1}{1+(\Omega_1/\Omega_c)^{2N}} = (1-\delta_1)^2$$
$$\Rightarrow \quad (\Omega_1/\Omega_c)^{2N} = \frac{1}{(1-\delta_1)^2} - 1 \equiv \alpha \tag{9.31}$$

$$|H_a(\Omega_2)| = \delta_2 \quad \Rightarrow \quad \frac{1}{1+(\Omega_2/\Omega_c)^{2N}} = (\delta_2)^2$$
$$\Rightarrow \quad (\Omega_2/\Omega_c)^{2N} = \frac{1}{\delta_2^2} - 1 \equiv \beta \tag{9.32}$$

Combining these two equations, we obtain:

$$\left(\frac{\Omega_1/\Omega_c}{\Omega_2/\Omega_c}\right)^{2N} = \frac{\alpha}{\beta}$$
$$\Rightarrow 2N\ln(\Omega_1/\Omega_2) = \ln(\alpha/\beta)$$
$$\Rightarrow N = \frac{1}{2}\frac{\ln(\alpha/\beta)}{\ln(\Omega_1/\Omega_2)} = 12.16$$

Since N must be an integer, we take $N = 13$.

- The value of Ω_c is then determined by requiring that:

$$|H_c(\Omega_1)| = 1-\delta_1 \quad \Rightarrow \quad \frac{1}{1+(\Omega_1/\Omega_c)^{26}} = (1-\delta_1)^2$$
$$\Rightarrow \quad \Omega_c = 0.856 \tag{9.33}$$

With this choice, the specifications are met exactly at Ω_1 and are exceeded at Ω_2, which contribute to minimize aliasing effects.

- An analog Butterworth filter meeting the desired specifications is obtained as follows:

$$H_c(s) = \frac{(\Omega_c)^{13}}{(s-s_0)(s-s_1)\cdots(s-s_{12})} \tag{9.34}$$

$$s_k = \Omega_c e^{j[\frac{\pi}{2}+(k+\frac{1}{2})\frac{\pi}{13}]} \quad k = 0,1,\ldots,12 \tag{9.35}$$

- Step 3: The desired digital filter is finally obtained by applying the impulse invariance method:
  - Partial fraction expansion:

$$H_c(s) = \frac{A_0}{s - s_0} + \cdots + \frac{A_{12}}{s - s_{12}} \tag{9.36}$$

  - Direct mapping to the z-domain:

$$H(z) = \frac{A_0}{1 - p_0 z^{-1}} + \cdots + \frac{A_{12}}{1 - p_{12} z^{-1}} \tag{9.37}$$

  with

$$p_k = e^{s_k}, \quad k = 0, 1, \ldots, 12 \tag{9.38}$$

  Table 9.1 lists the values of the poles s_k, the partial fraction expansion coefficients A_k and the discrete-time poles p_k for this example.

k	s_k	A_k	p_k
0	$-0.1031 + 0.8493\,j$	$-0.0286 + 0.2356\,j$	$+0.5958 + 0.6773\,j$
1	$-0.3034 + 0.8000\,j$	$-1.8273 - 0.6930\,j$	$+0.5144 + 0.5296\,j$
2	$-0.4860 + 0.7041\,j$	$+4.5041 - 6.5254\,j$	$+0.4688 + 0.3982\,j$
3	$-0.6404 + 0.5674\,j$	$+13.8638 + 15.6490\,j$	$+0.4445 + 0.2833\,j$
4	$-0.7576 + 0.3976\,j$	$-35.2718 + 18.5121\,j$	$+0.4322 + 0.1815\,j$
5	$-0.8307 + 0.2048\,j$	$-13.8110 - 56.0335\,j$	$+0.4266 + 0.0886\,j$
6	$-0.8556 + 0.0000\,j$	$+65.1416 + 0.0000\,j$	$+0.4250 + 0.0000\,j$
7	$-0.8307 - 0.2048\,j$	$-13.8110 + 56.0335\,j$	$+0.4266 - 0.0886\,j$
8	$-0.7576 - 0.3976\,j$	$-35.2718 - 18.5121\,j$	$+0.4322 - 0.1815\,j$
9	$-0.6404 - 0.5674\,j$	$+13.8638 - 15.6490\,j$	$+0.4445 - 0.2833\,j$
10	$-0.4860 - 0.7041\,j$	$+4.5041 + 6.5254\,j$	$+0.4688 - 0.3982\,j$
11	$-0.3034 - 0.8000\,j$	$-1.8273 + 0.6930\,j$	$+0.5144 - 0.5296\,j$
12	$-0.1031 - 0.8493\,j$	$-0.0286 - 0.2356\,j$	$+0.5958 - 0.6773\,j$

Table 9.1 Values of the coefficients used in example 9.3.

  - At this point, it is easy to manipulate the function $H(z)$ to derive various filter realizations (e.g. direct forms, cascade form, etc.). Figure 9.11 illustrates the magnitude response of the filter so designed.

◀

9.2.4 Bilinear transformation

Principle: Suppose we are given an analog IIR filter $H_a(s)$. A corresponding digital filter $H(z)$ can be obtained by applying the bilinear transformation (BT), defined as follows:

$$H(z) = H_a(s) \quad \text{at} \quad s = \phi(z) \triangleq \alpha \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right), \tag{9.39}$$

where α is a scaling parameter introduced for flexibility. In the absence of other requirements, it is often set to 1.

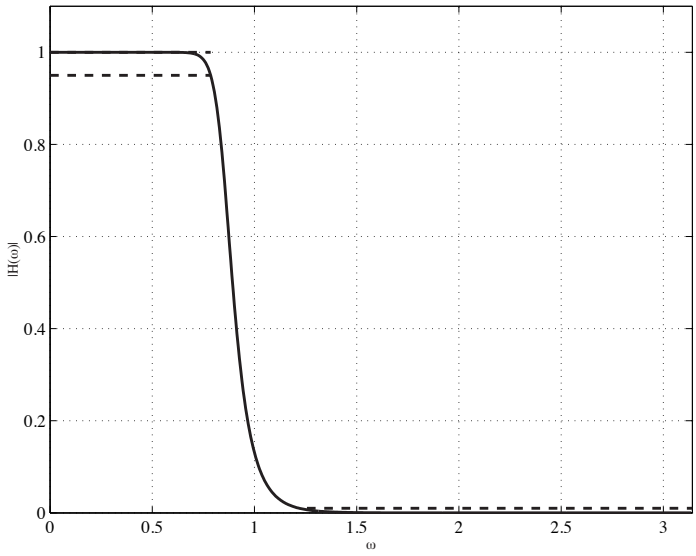


Fig. 9.11 Magnitude Response of the design of example 9.3.

Example 9.4:

► Consider the analog filter

$$H_a(s) = \frac{1}{s+b}, \quad b > 0 \tag{9.40}$$

Up to a scaling factor, this is a Butterworth filter of order $N = 1$ with cut-off frequency $\Omega_c = b$. This filter has only one simple pole at $s = -b$ in the s-plane; it is stable and causal with a low-pass behavior. The corresponding PZ diagram in the s-plane and magnitude response versus Ω are illustrated Figure 9.14 (left) and Figure 9.13 (top), respectively. Note the -3dB point at $\Omega = b$ in this example.

Applying the BT with $\alpha = 1$, we obtain

$$\begin{aligned} H(z) &= \frac{1}{\frac{1-z^{-1}}{1+z^{-1}} + b} \\ &= \frac{1}{1+b} \frac{1+z^{-1}}{1-\frac{1-b}{1+b}z^{-1}} \end{aligned} \tag{9.41}$$

The resulting digital filter has one simple pole at

$$z = \frac{1-b}{1+b} \triangleq \rho$$

inside the U.C.; it is stable and causal with a low-pass behavior The corresponding PZ diagram in the Z-plane and magnitude response versus ω are illustrated Figure 9.12 (right) and Figure 9.13 (bottom), respectively.

◄

Properties of bilinear transformation: The inverse mapping is given by:

$$s = \phi(z) = \alpha\left(\frac{1-z^{-1}}{1+z^{-1}}\right) \Longleftrightarrow z = \phi^{-1}(s) = \frac{1+s/\alpha}{1-s/\alpha} \tag{9.42}$$

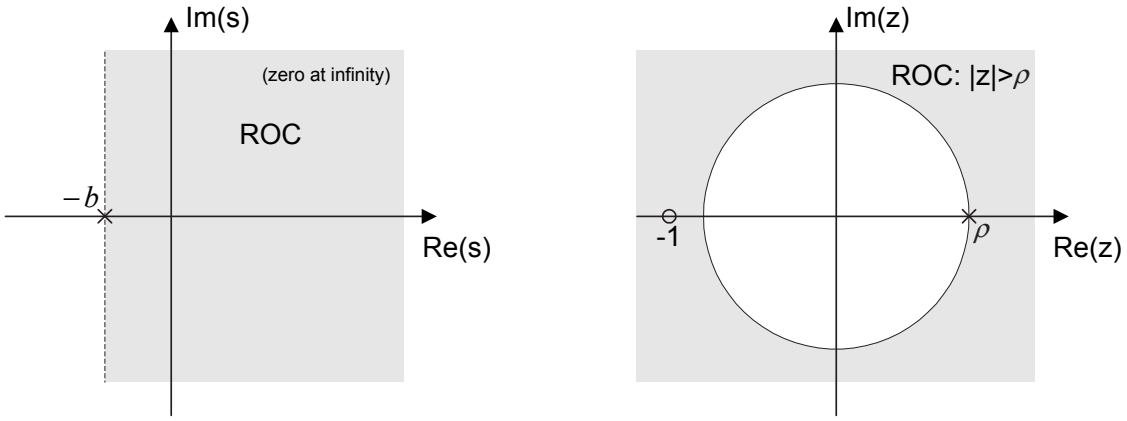


Fig. 9.12 PZ diagrams of the filters used in example 9.4. Left: analog filter ($b = .5$); Right: discrete-time filter.

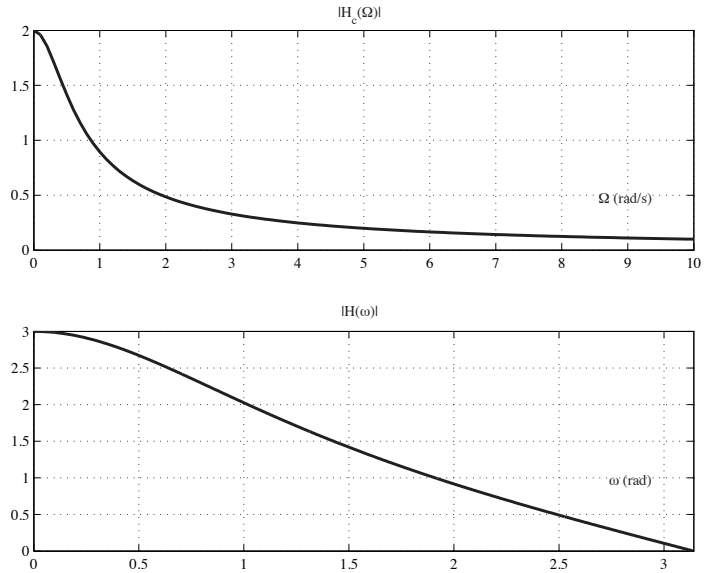


Fig. 9.13 Magnitude responses of the filters used in example 9.4. Top: analog filter ($b = .5$); Bottom: discrete-time filter.

It can be verified from (9.42) that the BT maps the left-half part of the s-plane into the interior of the unit circle in the z-plane, that is:

$$s = \sigma + j\Omega \text{ with } \sigma < 0 \Longleftrightarrow |z| < 1 \tag{9.43}$$

If $H_a(s)$ has a pole at s_k, then $H(z)$ has a corresponding pole at

$$p_k = \phi^{-1}(s_k) \tag{9.44}$$

Therefore, if $H_a(s)$ is stable and causal, so is $H(z)$. This is illustrated in Figure 9.14.

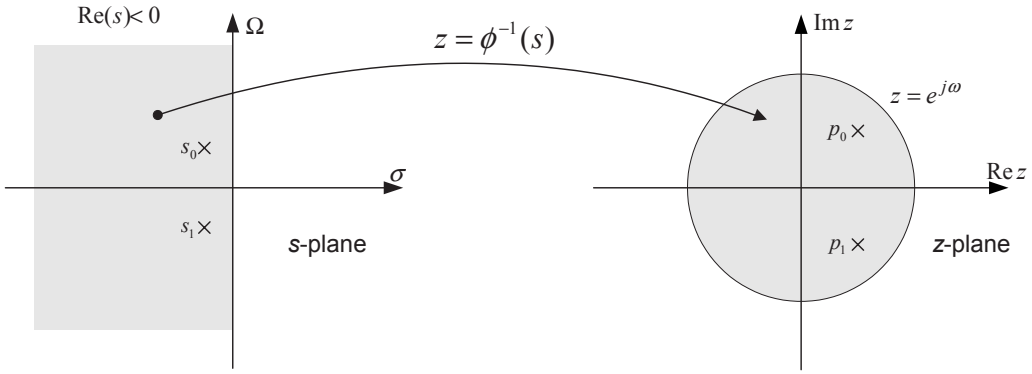


Fig. 9.14 Inverse BT mapping from s-plane to z-plane operated.

The BT uniquely maps the $j\Omega$ axis in the s-plane onto the unit circle in the z-plane (i.e. 1-to-1 mapping):

$$s = j\Omega \Longleftrightarrow z = \frac{1 + j\Omega/\alpha}{1 - j\Omega/\alpha} = \frac{A}{A^*} \tag{9.45}$$

where $A = 1 + j\Omega/\alpha$. Clearly

$$|z| = |A|/|A^*| = 1 \tag{9.46}$$

The relationship between the physical frequency Ω and the normalized frequency ω can be further developed as follows:

$$s = j\Omega \Longleftrightarrow z = e^{j\omega} \tag{9.47}$$

where $\omega = \measuredangle A - \measuredangle A^* = 2\measuredangle A$, that is

$$\omega = 2\tan^{-1}(\frac{\Omega}{\alpha}) \tag{9.48}$$

or equivalently

$$\Omega = \alpha\tan(\frac{\omega}{2}) \tag{9.49}$$

This is illustrated in Figure 9.15, from which it is clear that the mapping between Ω and ω is non-linear. In particular, the semi-infinite Ω axis $[0,\infty)$ is compressed into the finite ω interval $[0,\pi]$. This non-linear compression, also called *frequency warping*, particularly affects the high-frequencies. It is important to understand this effect and take it into account when designing a filter.

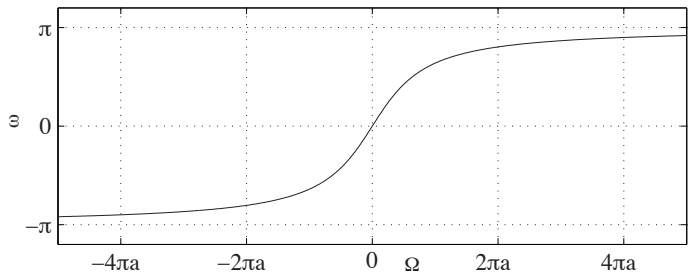


Fig. 9.15 Illustration of the frequency mapping operated by the bilinear transformation.

Design steps:

(1) Given desired specifications $H_d(\omega)$ for the DT filter, find the corresponding specifications for the analog filter, $H_{ad}(\Omega)$:

$$H_{ad}(\Omega) = H_d(\omega) \qquad \Omega = \alpha \tan(\frac{\omega}{2}) \tag{9.50}$$

(2) Design an analog IIR filter $H_a(\Omega)$ that meets these spec.

(3) Apply the BT to obtain the desired digital filter:

$$H(z) = H_a(s) \quad \text{at} \quad s = \phi(z) \triangleq \alpha \left(\frac{1-z^{-1}}{1+z^{-1}} \right) \tag{9.51}$$

Example 9.5:

► Apply the BT to an appropriate analog Butterworth filter in order to design a low-pass digital filter $H(z)$ that meets the specifications described in Figure 9.10.

- Step 1: In this example, we set the parameter α to 1. The desired specifications for the analog filter are similar to the above except for the band-edge frequencies. That is

$$1-\delta_1 \le |H_{ad}(\Omega)| = 1 \quad \text{for} \quad 0 \le |\Omega| \le \Omega_1 \tag{9.52}$$

$$|H_{ad}(\Omega)| \le \delta_2, \quad \text{for} \quad \Omega_2 =\le |\Omega| \le \infty \tag{9.53}$$

where now

$$\begin{aligned} \Omega_1 &= \tan(\frac{\omega_1}{2}) \approx 0.414214 \\ \Omega_2 &= \tan(\frac{\omega_1}{2}) \approx 0.72654 \end{aligned} \tag{9.54}$$

- Step 2: We need to design an analog Butterworth filter $H_a(\Omega)$ that meets the above specifications.
  - Proceeding exactly as we did in Example 9.3, we find that the required order of the Butterworth filter is $N = 11$ (or precisely $N \ge 10.1756$).
  - Since aliasing is not an issue with the BT method, either one of the stop-band edge specifications may be used to determine the cut-off frequency Ω_c. For example:

$$|H_a(\Omega_2)| = \frac{1}{1+(\Omega_2/\Omega_c)^{22}} = \delta_2 \Rightarrow \Omega_c = 0.478019 \tag{9.55}$$

- The desired analog Butterworth filter is specified as

$$H_c(s) = \frac{(\Omega_c)^{11}}{(s-s_0)(s-s_1)\cdots(s-s_{10})} \tag{9.56}$$

$$s_k = \Omega_c e^{j[\frac{\pi}{2}+(k+\frac{1}{2})\frac{\pi}{11}]} \quad k = 0,1,\ldots,10 \tag{9.57}$$

- Step 3: The desired digital filter is finally obtained by applying the bilinear transformation:

$$\begin{aligned} H(z) &= \frac{(\Omega_c)^{11}}{(2\left(\frac{1-z^{-1}}{1+z^{-1}}\right)-s_0)(2\left(\frac{1-z^{-1}}{1+z^{-1}}\right)-s_1)\cdots(2\left(\frac{1-z^{-1}}{1+z^{-1}}\right)-s_{10})} \\ &= \cdots \\ &= \frac{b_0+b_1z^{-1}+\cdots b_{11}z^{-11}}{1+a_1z^{-1}+\cdots a_{11}z^{-11}} \end{aligned} \tag{9.58}$$

The corresponding values of the coefficients b_i and a_i are shown in Table 9.2.

◀

k	b_k	a_k
0	+0.0003	+26.5231
1	+0.0033	−125.8049
2	+0.0164	+296.9188
3	+0.0491	−446.8507
4	+0.0983	+470.4378
5	+0.1376	−360.6849
6	+0.1376	+204.3014
7	+0.0983	−85.1157
8	+0.0491	+25.4732
9	+0.0164	−5.2012
10	+0.0033	+0.6506
11	+0.0003	−0.0377

Table 9.2 Coefficients of the filter designed in example 9.5.

9.3 Design of FIR filters

Design problem: The general FIR system function is obtained by setting $A(z) = 1$ in (9.1), that is

$$H(z) = B(z) = \sum_{k=0}^{M} b_k z^{-k} \tag{9.59}$$

The relationship between coefficients b_k and the filter impulse response is

$$h[n] = \begin{cases} b_n & 0 \le n \le M \\ 0 & \text{otherwise} \end{cases} \tag{9.60}$$

In the context of FIR filter design, it is convenient to work out the analysis directly in terms of $h[n]$. Accordingly, we shall express $H(z)$ as:

$$H(z) = h[0] + h[1]z^{-1} + \cdots h[M]z^{-M} \tag{9.61}$$

It is further assumed throughout that $h[0] \neq 0$ and $h[M] \neq 0$.

The design problem then consists in finding the degree M and the filter coefficients $\{h[k]\}$ in (9.61) such that the resulting frequency response approximates a desired response, $H_d(\omega)$, that is

$$H(\omega) \sim H_d(\omega)$$

In the design of FIR filters:

- stability is not an issue since FIR filters are always stable;
- considerable emphasis is usually given to the issue of linear phase (one of the main reasons for selecting an FIR filter instead of an IIR filter).

In this section, we will review a classification of FIR GLP systems, then explain the principles of design by windowing and design by minmax optimization.

9.3.1 Classification of GLP FIR filters

We recall the following important definition and result from Section 8.3.3:

Definition: A discrete-time LTI system is GLP iff

$$H(\omega) = A(\omega)e^{-j(\alpha\omega-\beta)} \tag{9.62}$$

where the $A(\omega)$, α and β are real valued.

Property: A real FIR filter with system function $H(z)$ as in (9.61) is GLP if and only if

$$h[k] = \varepsilon\, h[M-k] \quad k = 0,1,...,M \tag{9.63}$$

where ε is either equal to $+1$ or -1.

Example 9.6:

▶ Consider the FIR filter with impulse response given by

$$h[n] = \{\underline{1},0,-1,0,1,0,-1\} \tag{9.64}$$

Note that here, $h[n] = h[M-n]$ with $M = 6$ and $\varepsilon = 1$. Let us verify that the frequency response $H(\omega)$ effectively satisfy the condition (9.62):

$$\begin{aligned} H(e^{j\omega}) &= 1 - e^{-j\omega 2} + e^{-j\omega 4} - e^{-j\omega 6} \\ &= e^{-j\omega 3}(e^{j\omega 3} - e^{j\omega} + e^{-j\omega} - e^{-j\omega 3}) \\ &= 2je^{-j\omega 3}(\sin(3\omega) - \sin(\omega)) \\ &= A(\omega)e^{-j(\alpha\omega-\beta)} \end{aligned}$$

where we define the real-valued quantities

$$A(\omega) = 2\sin(\omega) - 2\sin(3\omega), \quad \alpha = 3, \quad \beta = \frac{3\pi}{2}$$

The frequency response is illustrated in Figure 9.16. ◄

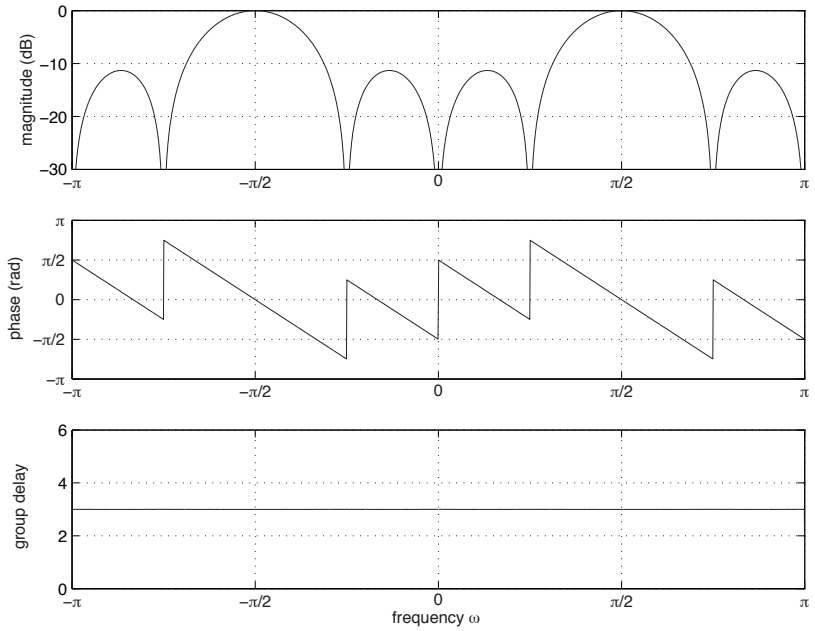


Fig. 9.16 Example frequency response of a generalized linear-phase filter.

Classification of GLP FIR filters: Depending on the values of the parameters ε and M, we distinguish 4 types of FIR GLP filters:

	M even	M odd
$\varepsilon = +1$	Type I	Type II
$\varepsilon = -1$	Type III	Type IV

A good understanding of the basic frequency characteristics of these four filter Types is important for practical FIR filter design. Indeed, not all filter Types, as listed above, can be used say for the design of a low-pass or high-pass filter. This issue is further explored below:

- Type I ($\varepsilon = +1, M$ even):

$$H(\omega) = e^{-j\omega M/2}\{\sum_k \alpha_k \cos(\omega k)\} \tag{9.65}$$

- Type II ($\varepsilon = +1, M$ odd):

$$H(\omega) = e^{-j\omega M/2}\{\sum_k \beta_k \cos(\omega(k-\frac{1}{2}))\} \tag{9.66}$$

- $H(\omega) = 0$ at $\omega = \pi$
- Cannot be used to design high-pass filter

- Type III ($\varepsilon = -1, M$ even):

$$H(\omega) = je^{-j\omega M/2}\{\sum_k \gamma_k \sin(\omega k)\} \tag{9.67}$$

- $H(\omega) = 0$ at $\omega = 0$ and at $\omega = \pi$
- Cannot be used to design either low-pass or high-pass filter

- Type IV ($\varepsilon = -1, M$ odd):

$$H(\omega) = je^{-j\omega M/2}\{\sum_k \delta_k \sin(\omega(k - \frac{1}{2}))\} \tag{9.68}$$

- $H(\omega) = 0$ at $\omega = 0$
- Cannot be used to design low-pass filter

9.3.2 Design of FIR filter via windowing

Basic principle: Desired frequency responses $H_d(\omega)$ usually correspond to non-causal and non-stable filters, with infinite impulse responses $h_d[n]$ extending from $-\infty$ to ∞. Since these filters have finite energy, it can be shown that

$$|h[n]| \to 0 \text{ as } n \to \pm\infty. \tag{9.69}$$

Based on these considerations, it is tempting to approximate $h_d[n]$ by an FIR response $g[n]$ in the following way:

$$g[n] = \begin{cases} h_d[n], & |n| \le K \\ 0, & \text{otherwise} \end{cases} \tag{9.70}$$

The FIR $g[n]$ can then be made causal by a simple shift, i.e.

$$h[n] = g[n - K] \tag{9.71}$$

Example 9.7:

▶ Consider an ideal low-pass filter:

$$H_d(\omega) = \begin{cases} 1, & |\omega| \le \omega_c = \pi/2 \\ 0, & \omega_c < |\omega| \le \pi \end{cases}$$

The corresponding impulse response is given by

$$h_d[n] = \frac{\omega_c}{\pi}\text{sinc}(\frac{n\omega_c}{\pi}) = \frac{1}{2}\text{sinc}(\frac{n}{2})$$

These desired characteristics are outlined in Figure 9.17.

Assuming a value of $K = 5$, the truncated FIR filter $g[n]$ is obtained as

$$g[n] = \begin{cases} h_d[n], & -5 \le n \le 5 \\ 0, & \text{otherwise} \end{cases}$$

Finally, a causal FIR with $M = 2K$ is obtained from $g[n]$ via a simple shift:

$$h[n] = g[n-5]$$

These two impulse responses are illustrated in Figure 9.18. The frequency response of the resulting filter, $G(\omega)$ is illustrated in Figure 9.19. ◀

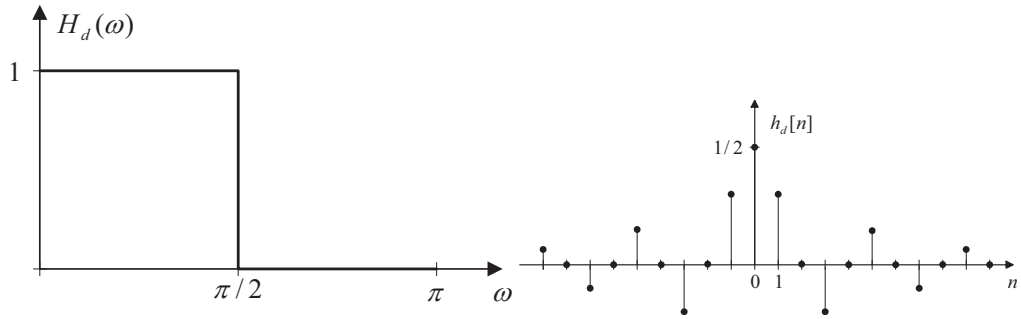


Fig. 9.17 Desired (a) frequency response and (b) impulse response for example 9.7

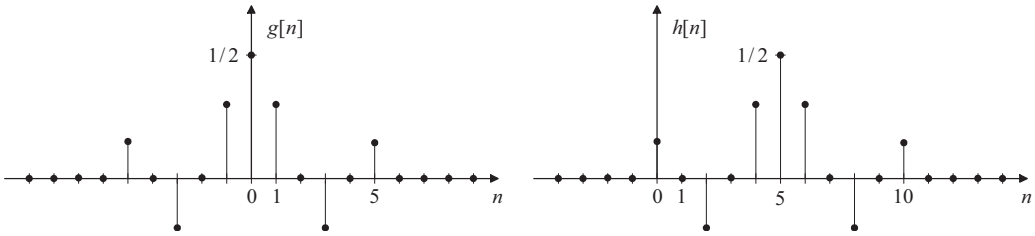


Fig. 9.18 (a) Windowed impulse response $g[n]$ and (b) shifted windowed impulse response $h[n]$ for example 9.7

Observation: The above design steps (9.70)– (9.71) can be combined into a single equation

$$h[n] = w_R[n]\, h_d[n - \frac{M}{2}] \tag{9.72}$$

where

$$w_R[n] \triangleq \begin{cases} 1, & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases} \tag{9.73}$$

According to (9.72), the desired impulse response $h_d[n]$ is first shifted by $K = M/2$ and then multiplied by $w_R[n]$.

Remarks on the window: The function $w_R[n]$ is called a rectangular window. More generally, in the context of FIR filter design, we define a window as a DT function $w[n]$ such that

$$w[n] = \begin{cases} w[M-n] \ge 0, & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases} \tag{9.74}$$

As we explain later, several windows have been developed and analysed for the purpose of FIR filter design. These windows usually lead to better properties of the designed filter.

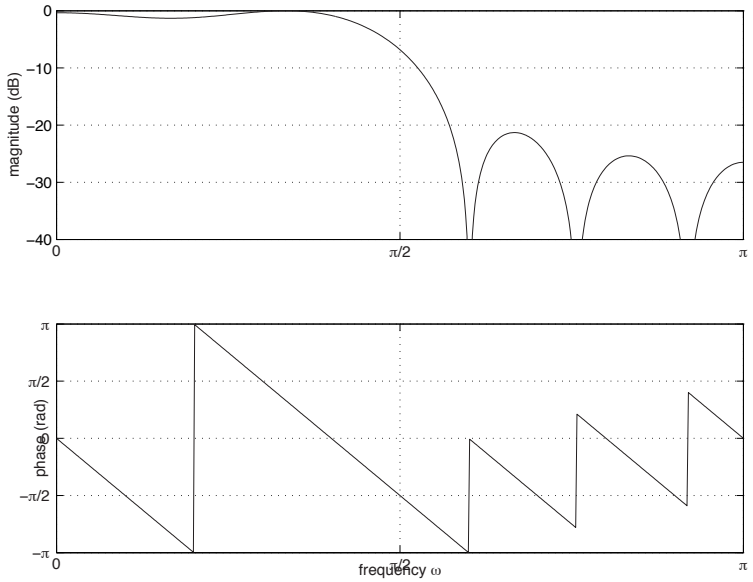


Fig. 9.19 Example frequency response of an FIR filter designed by windowing (refer to example 9.7).

Remarks on time shift: Note that the same magnitude response $|H(\omega)|$ would be obtained if the shift operation in (9.72) was omitted. In practice, the shift operation is used to center $h_d[n]$ within the window interval $0 \le n \le M$, so that symmetries originally present in $h_d[n]$ with respect to $n = 0$ translate into corresponding symmetries of $h[n]$ with respect to $n = K = M/2$ after windowing. In this way, a desired response $H_d(\omega)$ with a GLP property, that is

$$h_d[n] = \varepsilon h_d[-n] \quad n \in \mathbb{Z} \tag{9.75}$$

is mapped into an FIR filter $H(\omega)$ with GLP, i.e.

$$h[n] \triangleq w[n]h_d[n-K] = \varepsilon h[M-n] \tag{9.76}$$

as required in most applications of FIR filtering.

The above approach leads to an FIR filter with $M = 2K$ even. To accommodate the case M odd, or equivalently $K = M/2$ non-integer, the following definition of a non-integer shift is used:

$$h_d[n-K] = \frac{1}{2\pi}\int_{-\pi}^{\pi}\{H_d(\omega)e^{-j\omega K}\}e^{j\omega n}d\omega \tag{9.77}$$

Design steps: To approximate a desired frequency response $H_d(\omega)$

- Step (1): Choose window $w[n]$ and related parameters (including M).
- Step (2): Compute the shifted version of desired impulse response:

$$h_d[n-K] = \frac{1}{2\pi}\int_{-\pi}^{\pi}\{H_d(\omega)e^{-j\omega K}\}e^{j\omega n}d\omega \tag{9.78}$$

  where $K = M/2$.

- Step(3): Apply the selected window $w[n]$:

$$h[n] = w[n]h_d[n-K] \tag{9.79}$$

Remarks: Several standard window functions exist that can be used in this method (e.g. Bartlet, Hanning, etc.).

For many of these windows, empirical formulas have been developed that can be used as guidelines in the choice of a suitable M.

To select the proper window, it is important to understand the effects of and trade-offs involved in the windowing operation (9.79) in the frequency domain.

Spectral effects of windowing: Consider the window design formula (9.79), with shift parameter K set to zero in order to simplify the discussion, i.e.

$$h[n] = w[n]h_d[n] \tag{9.80}$$

Taking the DTFT on both sides, we obtain:

$$H(\omega) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(\theta)W(\omega-\theta)d\theta \tag{9.81}$$

where

$$W(\omega) = \sum_{n=0}^{M} w[n]e^{-j\omega n} \tag{9.82}$$

is the DTFT of the window function $w[n]$.

Thus, application of the window function $w[n]$ in the time-domain is equivalent to periodic convolution with $W(\omega)$ in the frequency domain.

For an infinitely long rectangular window, i.e. $w[n] = 1$ for all $n \in \mathbb{Z}$, we have $W(\omega) = 2\pi\sum_k \delta_c(\omega - 2\pi k)$ with the result that $H(\omega) = H_d(\omega)$. In this limiting case, windowing does not introduce any distortion in the desired frequency response.

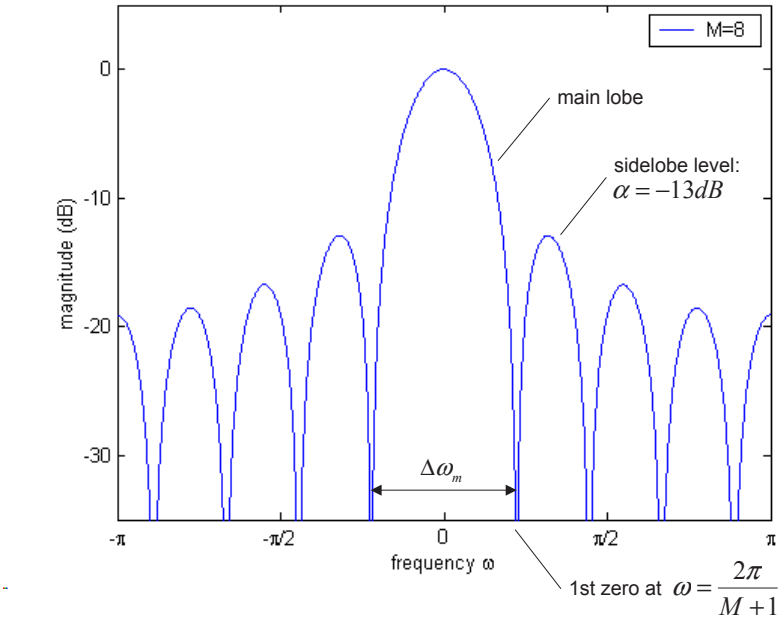
For a practical finite-length window, i.e. time limited to $0 \le n \le M$, the DTFT $W(\omega)$ is spread around $\omega = 0$ and the convolution (9.81) leads to smearing of the desired response $H_d(\omega)$.

Spectral characteristics of the rectangular window: For the rectangular window $w_R[n] = 1, 0 \le n \le M$, we find:

$$\begin{aligned} W_R(\omega) &= \sum_{n=0}^{M} e^{-j\omega n} \\ &= e^{-j\omega M/2}\left(\frac{\sin(\omega(M+1)/2)}{\sin(\omega/2)}\right) \end{aligned} \tag{9.83}$$

The corresponding magnitude spectrum is illustrated below for the $M = 8$:

$|W_R(\omega)|$ is characterized by a main lobe, centered at $\omega = 0$, and secondary lobes of lower amplitudes, also called sidelobes: For $w_R[n]$, we have:



- mainlobe width: $\Delta\omega_m = \frac{4\pi}{M+1}$
- peak sidelobe level: $\alpha = -13$dB (independent of M)

Typical LP design with window method: Figure 9.20 illustrates the typical look of the magnitude response of a low-pass FIR filter designed by windowing. Characteristic features include:

- Ripples in $|H(\omega)|$ due to the window sidelobes in the convolution (Gibb's phenomenon);
- δ, which denotes the peak approximation error in $[0,\omega_p] \cup [\omega_s,\pi]$.
  δ, strongly depends on the peak sidelobe level α of the window.
- Width of transition band: $\Delta\omega = \omega_s - \omega_p < \Delta\omega_m$. Note that $\Delta\omega \neq 0$ due to main lobe width of the window;
- $|H(\omega)| \neq 0$ in stopband $\Rightarrow$ leakage (also an effect of sidelobes)

For a rectangular window:

- $\delta \approx -21$dB (independent of M)
- $\Delta\omega \approx \frac{2\pi}{M}$

To reduce δ, one must use an other type of window (Kaiser, Hanning, Hamming,...). These other windows have wider main lobes (resulting in wider transition bands) but in general lower sidelobes (resulting in lower ripple).

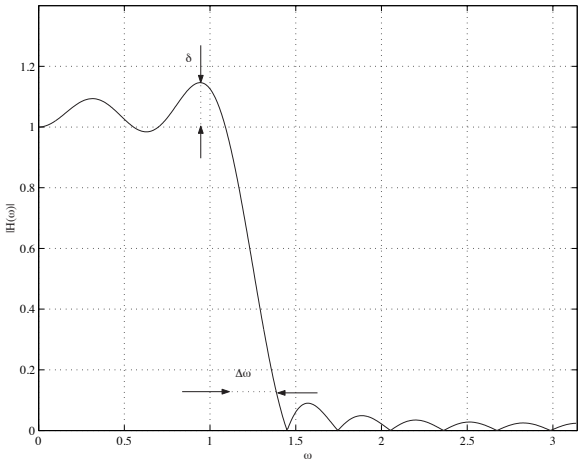


Fig. 9.20 Typical magnitude response of an FIR filter designed by windowing.

9.3.3 Overview of some standard windows

Windowing trade-offs: Ideally, one would like to use a window function $w[n]$ that has finite duration in the time-domain and such that its DTFT $W(\omega)$ is perfectly concentrated at $\omega = 0$. This way, the application of $w[n]$ to a signal $x[n]$ in the time-domain would not not introduce spectral smearing, as predicted by (9.81). Unfortunately, such a window function does not exist: it is a fundamental property of the DTFT that a signal with finite duration has a spectrum that extends from $-\infty$ to $+\infty$ in the ω-domain.

For a fixed value of the window length M, it is possible however to vary the window coefficients $w[n]$ so as to trade-off main-lobe width for sidelobe level attenuation. In particular, it can be observed that by tapering off the values of the window coefficients near its extremity, it is possible to reduce sidelobe level α at the expense of increasing the width of the main lobe, $\Delta\omega_m$. This is illustrated in Figure 9.21: We refer to this phenomenon as the fundamental windowing trade-off.

Bartlett or triangular window:

$$w[n] = (1 - |\frac{2n}{M} - 1|)\, w_R[n] \tag{9.84}$$

Hanning Window:

$$w[n] = \frac{1}{2}(1 - \cos\frac{2\pi n}{M})\, w_R[n] \tag{9.85}$$

Hamming Window:

$$w[n] = (0.54 - 0.46\cos\frac{2\pi n}{M})\, w_R[n] \tag{9.86}$$

Blackman Window:

$$w[n] = (0.42 - 0.5\cos\frac{2\pi n}{M} + 0.08\cos\frac{4\pi n}{M})\, w_R[n] \tag{9.87}$$

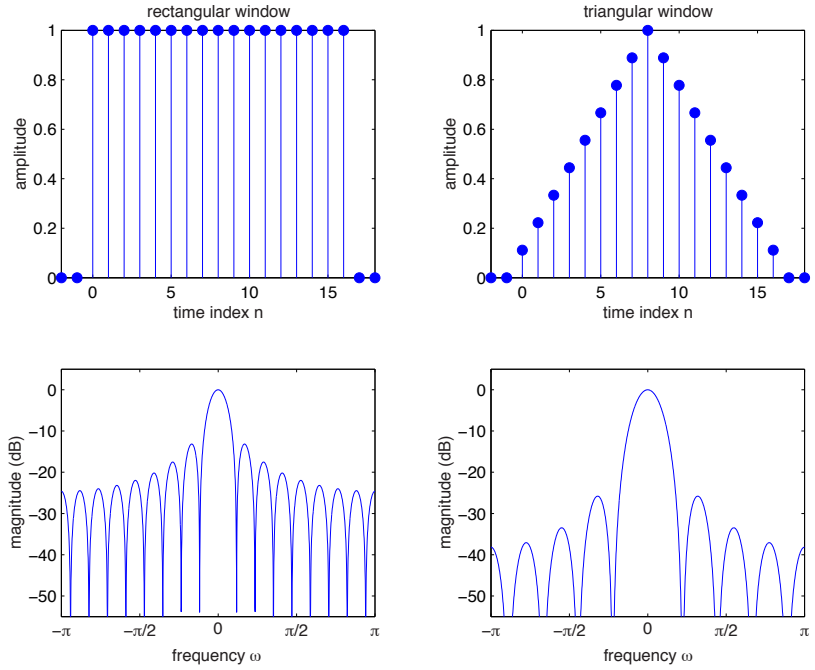


Fig. 9.21 Illustration of the fundamental windowing tradeoff.

A generic form for the Hanning, Hamming and Blackman windows is

$$w[n] = (A + B\cos\frac{2\pi n}{M} + C\cos\frac{4\pi n}{M})\, w_R[n]$$

where coefficients A, B and C determine the window type.

Kaiser window family:

$$w[n] = \frac{1}{I_0(\beta)} I_0(\beta\sqrt{1-(\frac{2n}{M}-1)^2})\, w_R[n] \tag{9.88}$$

where $I_o(.)$ denotes the Bessel function of the 1st kind and $\beta \geq 0$ is an adjustable design parameter. By varying α, it is possible to trade-off sidelobe level α for mainlobe width: $\Delta\omega_m$:

- $\beta = 0 \Rightarrow$ rectangular window;
- $\beta > 0 \Rightarrow \alpha \downarrow$ and $\Delta\omega_m \uparrow$.

The so-called Kaiser design formulae enable one to find necessary values of β and M to meet specific low-pass filter requirements. Let $\Delta\omega$ denote the transition band of the filter i.e.

$$\Delta\omega = \omega_s - \omega_p \tag{9.89}$$

and let A denote the stopband attenuation in positive dB:

$$A = -20\log_{10}\delta \tag{9.90}$$

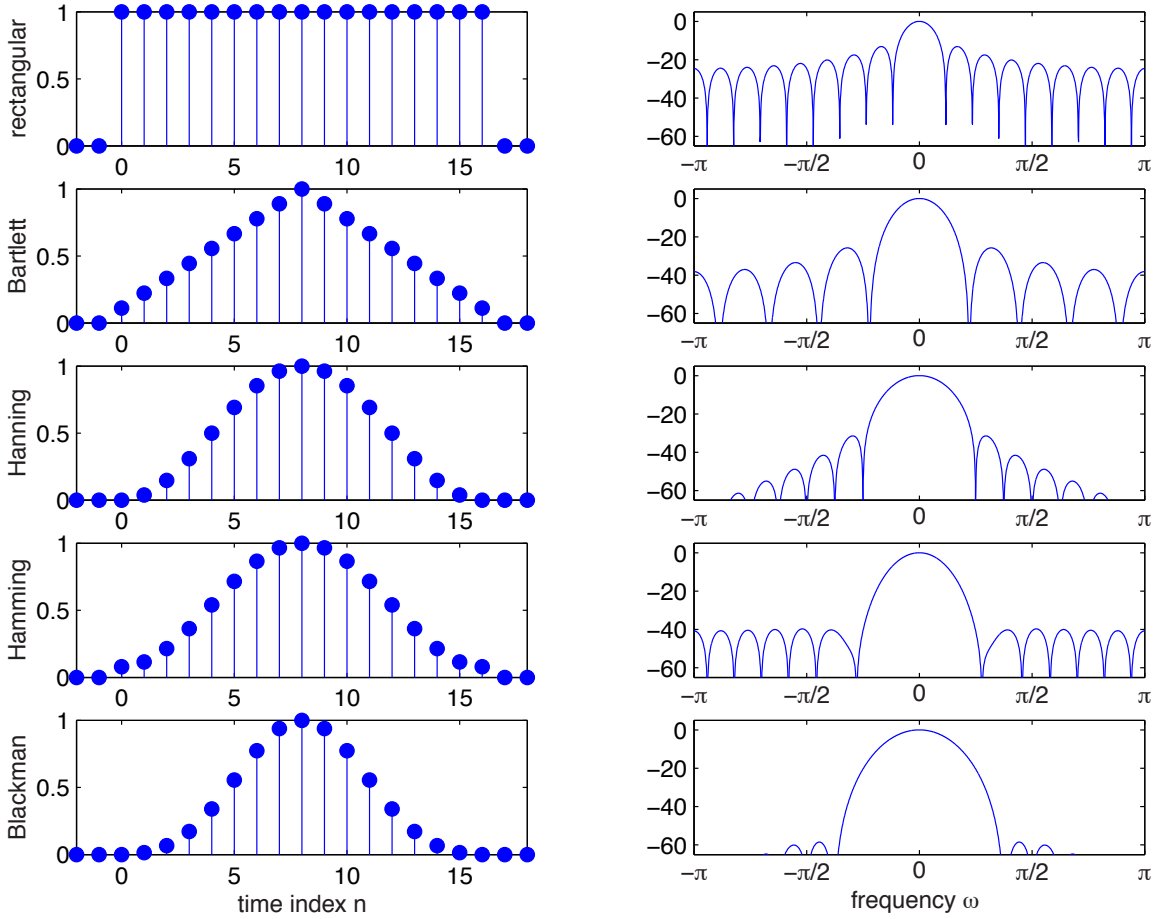


Fig. 9.22 .

The following formulae [2] may be used to determine β and M:

$$M = \frac{A-8}{2.285\Delta\omega} \tag{9.91}$$

$$\beta = \begin{cases} 0.1102(A-8.7) & A > 50 \\ 0.5842(A-21)^{0.4} + 0.07886(A-21) & 21 \le A \le 50 \\ 0.0 & A < 21 \end{cases} \tag{9.92}$$

[2] J. F. Kaiser, "Nonrecursive digital filter design using the I_o – sinh window function," *IEEE Int. Symp. on Circuits and Systems*, April 1974, pp. 20-23.

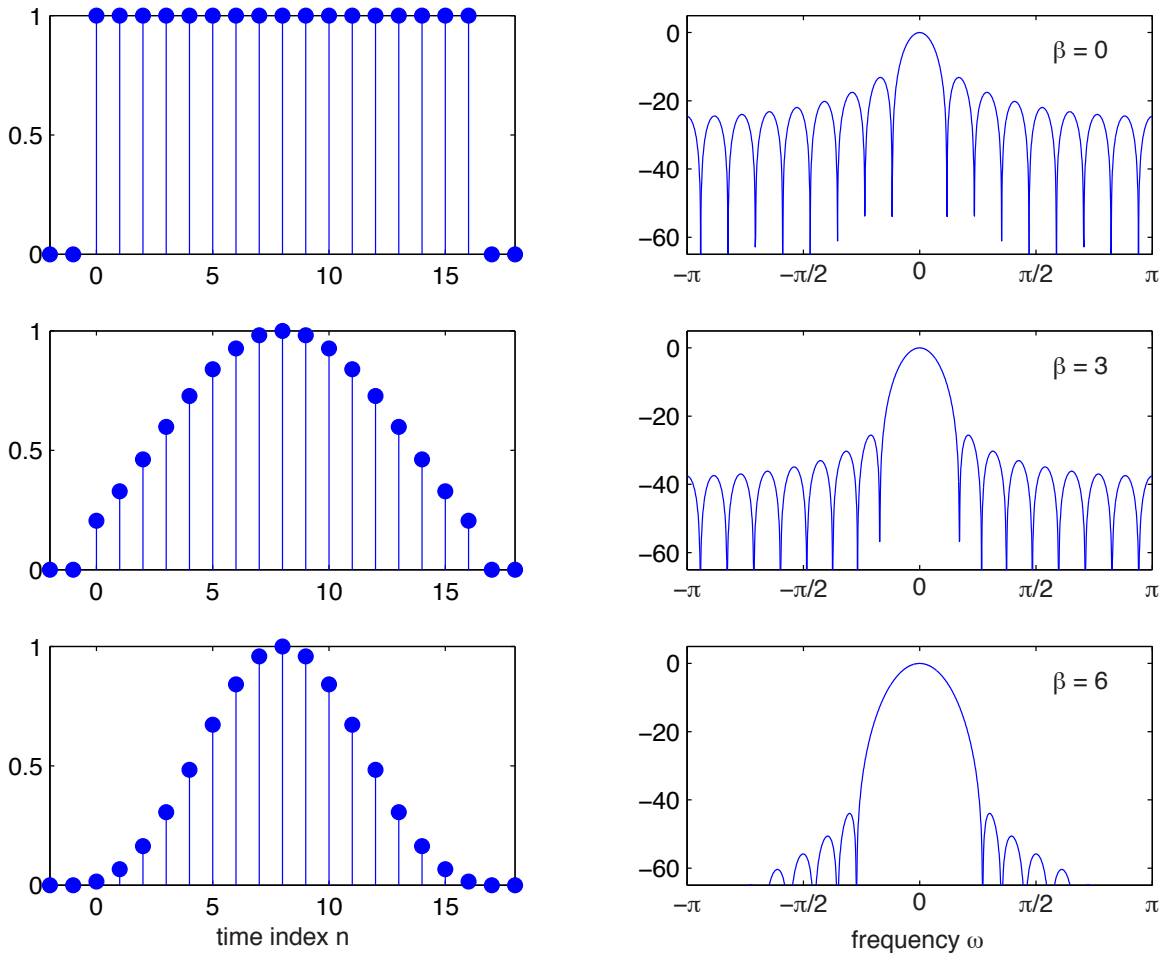


Fig. 9.23 .

9.3.4 Parks-McClellan method

Introduction: Consider a Type-I GLP FIR filter (i.e., M even and $\varepsilon = +1$):

$$H(\omega) = e^{-j\omega M/2}A(\omega) \tag{9.93}$$

$$A(\omega) = \sum_{k=0}^{L} \alpha_k \cos(k\omega) \tag{9.94}$$

where $L = M/2$, $\alpha_0 = h[L]$, and $\alpha_k = 2h[L-k]$ for $k = 1, ..., L$. Let $H_d(\omega)$ be the desired frequency response (e.g., ideal low-pass filter), specified in terms of tolerances in frequency bands of interest.

The Parks-McClellan (PM) method of FIR filter design attempts to find the best set of coefficients α_k's, in the sense of minimizing the maximum weighted approximation error

$$E(\omega) \triangleq W(\omega)[H_d(\omega) - A(\omega)] \tag{9.95}$$

over the frequency bands of interest (e.g., passband and stopband). $W(\omega) \geq 0$ is a frequency weighting introduced to penalize differently the errors made in different bands.

Specifying the desired response: The desired frequency response $H_d(\omega)$ must be accompanied by a tolerance scheme or template which specifies:

- a set of disjoint frequency bands $\mathcal{B}_i = [\omega_{i1}, \omega_{i2}]$
- and for each band $\mathcal{B}_i$, a corresponding tolerance δ_i, so that

$$|H_d(\omega) - A(\omega)| \leq \delta_i, \quad \omega \in \mathcal{B}_i \tag{9.96}$$

The frequency intervals between the bands $\mathcal{B}_i$ are called transition bands. The behavior of $A(\omega)$ in the transition bands cannot be specified.

Example 9.8:

▶ For example, in the case of a low-pass design:

- Passband: $\mathcal{B}_1 = [0, \omega_p]$ with tolerance δ_1, i.e.

$$|1 - A(\omega)| \leq \delta_1, \quad 0 \leq \omega \leq \omega_p \tag{9.97}$$

- Stopband: $\mathcal{B}_2 = [\omega_s, \pi]$ with tolerance δ_2, i.e.

$$|A(\omega)| \leq \delta_2, \quad \omega_s \leq \omega \leq \pi \tag{9.98}$$

- Transition band: $[\omega_p, \omega_s]$

◀

The weighting mechanism: In the Parks-McClellan method, a weighting function $W(\omega) \geq 0$ is applied to the approximation error $H_d(\omega) - A(\omega)$ prior to its optimization, resulting in a weighted error

$$E(\omega) = W(\omega)[H_d(\omega) - A(\omega)]. \tag{9.99}$$

The purpose of $W(\omega)$ is to scale the error $H_d(\omega) - A(\omega)$, so that an error in a band $\mathcal{B}_i$ with a small tolerance δ_i is penalized more than an error in a band $\mathcal{B}_j$ with a large tolerance $\delta_j > \delta_i$. This may be achieved by constructing $W(\omega)$ in the following way:

$$W(\omega) = \frac{1}{\delta_i}, \quad \omega \in \mathcal{B}_i \tag{9.100}$$

so that the cost $W(\omega)$ is larger in those bands $\mathcal{B}_i$ where δ_i is smaller. In this way, the non-uniform tolerance specifications δ_i over the individual bands $\mathcal{B}_i$ translate into a single tolerance specification of over all the bands of interest, i.e.

$$|E(\omega)| \leq 1, \quad \omega \in \mathcal{B}_1 \cup \mathcal{B}_2 \cup \ldots \tag{9.101}$$

Properties of the min-max solution: In the Parks-McClellan method, one seeks an optimal filter $H(\omega)$ that minimizes the maximum approximation error over the frequency bands of interest. Specifically, we seek

$$\min_{\text{all } \alpha_k} \{ \max_{\omega \in \mathcal{B}} |E(\omega)| \} \tag{9.102}$$

where

$$E(\omega) = W(\omega)[H_d(\omega) - A(\omega)] \tag{9.103}$$

$$A(\omega) = \sum_{k=0}^{L} \alpha_k \cos(k\omega) \tag{9.104}$$

$$\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \cup \ldots \tag{9.105}$$

The optimal solution $A_o(\omega)$ satisfies the following properties:

- *Alternation theorem:* $A_o(\omega)$ is the unique solution to the above min-max problem iff it has at least $L+2$ alternations over $\mathcal{B}$.
- That is, there must exist at least $L+2$ frequency points $\omega_i \in \mathcal{B}$ such that $\omega_1 < \omega_2 < \cdots < \omega_{L+2}$ and that

$$E(\omega_i) = -E(\omega_{i+1}) = \pm E_{\max} \tag{9.106}$$

$$E_{\max} \triangleq \max_{\omega \in \mathcal{B}} |E(\omega)| \tag{9.107}$$

- All interior points of $\mathcal{B}$ where $A(\omega)$ has zero slope correspond to alternations.

This is illustrated in Figure 9.24 for a low-pass design.

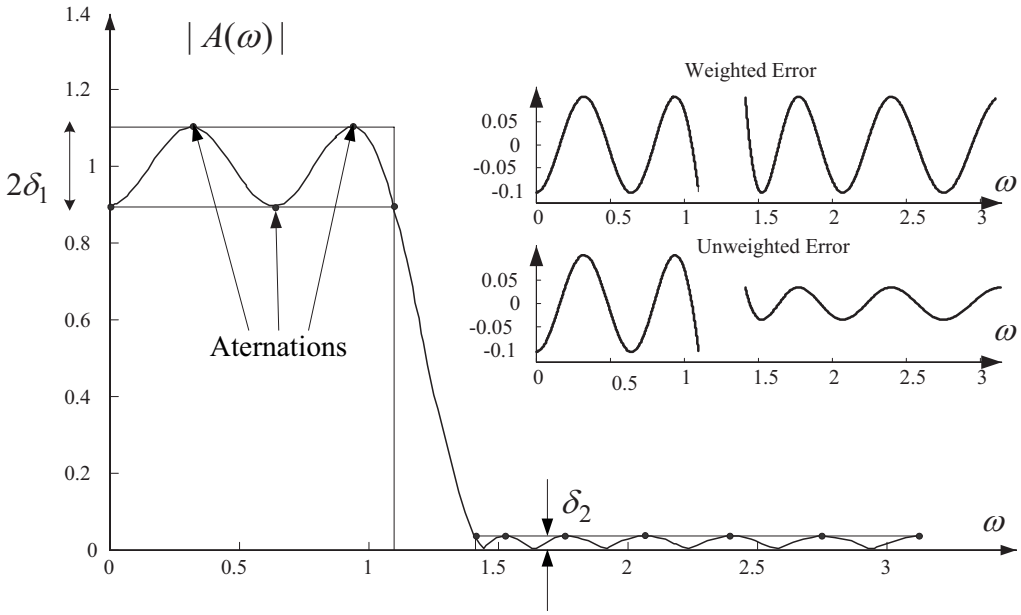


Fig. 9.24 Illustration of the characteristics of a low-pass filter designed using the Parks-McClellan method.

Remarks: Because of the alternations, these filters are also called equiripple. Due to the use of the weighting function $W(\omega)$, an equiripple behavior of $E(\omega)$ over the band $\mathcal{B}$ with amplitude $E_{\max}$ translates into equiripple behavior of $H_d(\omega) - A(\omega)$ with amplitude $\delta_i E_{\max}$ over band $\mathcal{B}_i$.

Observe that the initial specifications in (9.96) will be satisfied if the final error level $E_{\max} \leq 1$. One practical problem with the PM method is that the required value of M to achieve this is not known a priori. In other words, the method only ensures that the relative sizes of the desired tolerances are attained (i.e. the ratio δ_i/δ_j).

In the case of a low-pass design, the following empirical formula can be used to obtain an initial guess for M:

$$M \approx \frac{-10\log_{10}(\delta_1\delta_2) - 13}{2.324(\omega_s - \omega_p)} \tag{9.108}$$

Although we have presented the method for Type-I GLP FIR filter, it is also applicable with appropriate modifications to Types II, III and IV GLP FIR filters. Generalizations to non-linear phase complex FIR filters also exist.

Use of computer program: Computer programs are available for carrying out the minimax optimization numerically. They are based on the so-called Remez exchange algorithm. In the Matlab signal processing toolbox, the function `remez` can be used to design minimax GLP FIR filters of Type I, II, III and IV. The function `cremez` can be used to design complex valued FIR filters with arbitrary phase.

Example 9.9:

- Suppose we want to design a low-pass filter with
  - Passband: $|1 - H(\omega)| \leq \delta_1 = .05, \quad 0 \leq \omega \leq \omega_p = 0.4\pi$
  - Stopband: $|H(\omega)| \leq \delta_2 = .01, \quad 0.6\pi = \omega_s \leq \omega \leq \pi$

  - Initial guess for M:

$$M \approx \frac{-10\log_{10}(0.05 \times 0.01) - 13}{2.324(0.6\pi - 0.4\pi)} \approx 10.24 \tag{9.109}$$

  - Set $M = 11$ and use the commands:

$$\begin{aligned} F &= [0, 0.4, 0.6, 1.0] \\ Hd &= [1, 1, 0, 0] \\ W &= [1/0.05, 1/0.01] \\ B &= \texttt{remez}(M, F, Hd, W) \end{aligned} \tag{9.110}$$

    The output vector B will contain the desired filter coefficients, i.e.

$$B = [h[0], \ldots, h[M]] \tag{9.111}$$

    The corresponding impulse response and magnitude response ($M = 11$) are shown in Figure 9.25.

  - The filter so designed has tolerances δ_1' and δ_2' that are different from δ_1 and δ_2. However:

$$\frac{\delta_1'}{\delta_2'} = \frac{\delta_1}{\delta_2} \tag{9.112}$$

- By trial and error,we find that the required value of M necessary to meet the spec is $M = 15$. The corresponding impulse response and magnitude response are shown in Figure 9.26.

◀

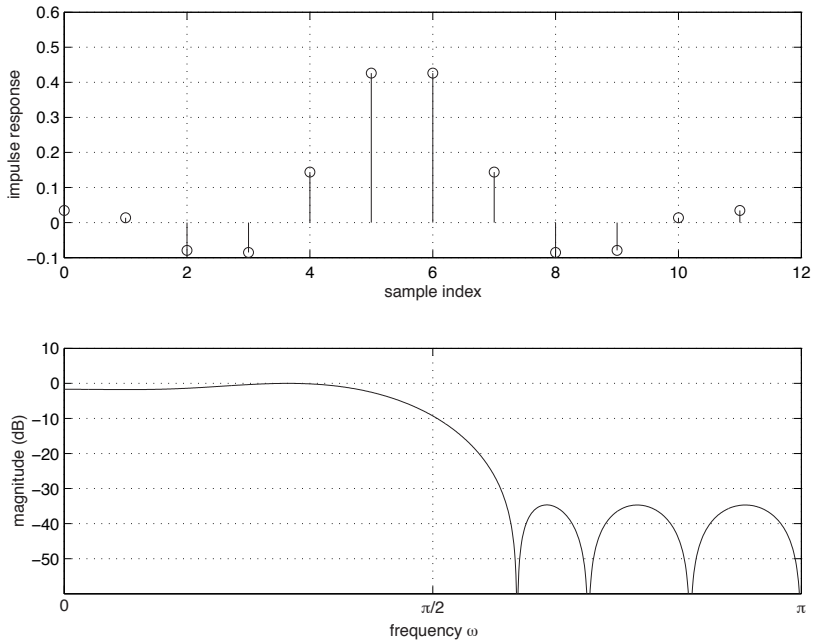


Fig. 9.25 Illustration of trial design in example 9.9 ($M = 11$).

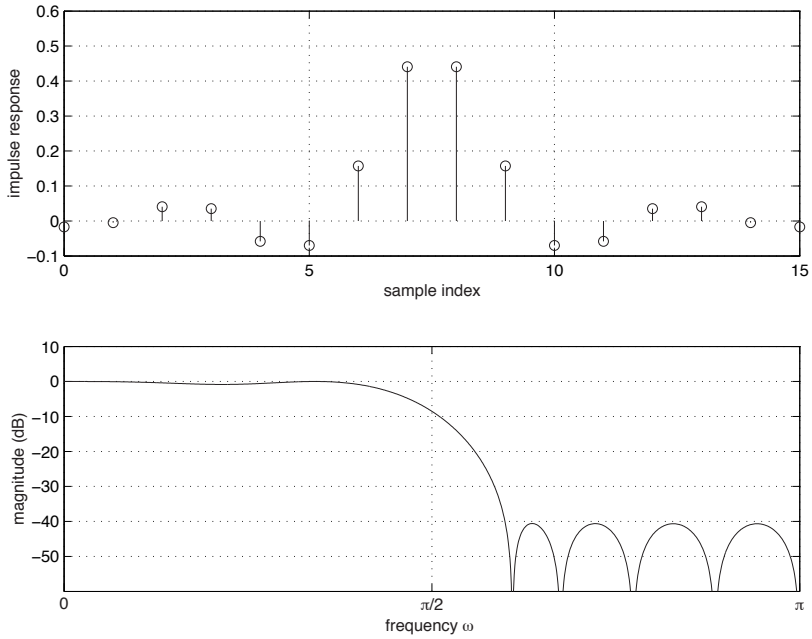


Fig. 9.26 Illustration of the final design in example 9.9 ($M = 15$)

Chapter 10

Quantization effects

Introduction

Practical DSP systems use finite-precision (FP) number representations and arithmetic. The implementation in FP of a given LTI system with rational system function

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{M} a_k z^{-k}}$$

leads to deviations in its theoretically predicted performance:

- Due to quantization of system coefficients a_k's and b_k's:

$$H(\omega) \Rightarrow \hat{H}(\omega) \neq H(\omega) \tag{10.1}$$

- Due to round-off errors in finite precision arithmetic, there is quantization noise at the system output.
- Other round-off effects such as limit cycles, etc.

Different structures or realizations (e.g. direct form, cascade, etc.) will be affected differently by these effects.

10.1 Binary number representation and arithmetic

In a binary number representation system, a real number $x \in \mathbb{R}$ is represented by a sequence of binary digits (i.e. bits) $b_i \in \{0,1\}$:

$$x \Longleftrightarrow \ldots b_2 b_1 b_0 b_{-1} \ldots$$

In practice, the number of bits available for the representation of numbers is finite (e.g. 16, 32, etc.). This means that only certain numbers $x \in \mathbb{R}$ can be represented exactly.

Several approaches are available for the representation of real numbers with finite sequences of bits, some of which are discussed below. The choice of a specific type of representation in a given application involves the consideration of several factors, including: desired accuracy, system costs, etc.

10.1.1 Binary representations

Signed fixed-point representations: In this type of representation, a real number x in the range $[-2^K, 2^K]$ is represented as

$$x = b_K b_{K-1} \cdots b_0 . b_{-1} \cdots b_{-L}, \quad b_i \in \{0,1\} \tag{10.2}$$

where b_K is called sign bit or sometimes most significant bit (MSB), b_{-L} is called the least significant bit (LSB), the K bits $b_{K-1} \cdots b_0$ represent the integer part of x and the L bits $b_{-1} \cdots b_{-L}$ represent the fractional part of x. The binary dot "." is used to separate the fractional part from the integer part. It is convenient to define $B \triangleq K + L$ so that the total number of bits is $B+1$, where the 1 reflects the presence of a sign bit.

Below, we discuss two special types of signed fixed-point representations, that is: the *sign and magnitude* (SM) and the *2's complement* (2C) representations.

SM representation: The simplest type of fixed-point representation is the SM representation. Its general format is:

$$x = (b_K \cdots b_0 . b_{-1} \cdots b_{-L})|_{\text{SM}} = (-1)^{b_K} \times \sum_{i=-L}^{K-1} b_i 2^i \tag{10.3}$$

2's complement representation: The most common type of signed fixed-point representation is 2's complement representation (2C). The general format is

$$x = (b_K \cdots b_0 . b_{-1} \cdots b_{-L})_{\text{2C}} = -b_K 2^K + \sum_{i=-L}^{K-1} b_i 2^i \tag{10.4}$$

Example 10.1:

▶ Using (10.3) and (10.4), we find that

$$(0.11)_{\text{SM}} = (0.11)_{\text{2C}} = 1 \times 2^{-1} + 1 \times 2^{-2} = 3/4$$

while

$$(1.11)_{\text{SM}} = (-1) \times 3/4 = -3/4$$

$$(1.11)_{\text{2C}} = (-1) + 3/4 = -1/4$$

◀

Observation: For fixed-point representation, the distance between two consecutive numbers, also called resolution of step size, is a constant equal to

$$\Delta = 2^{-L}$$

Thus, the available dynamic range $[-2^K, 2^K]$ is quantized into 2^{B+1} uniformly spaced representation levels, where $B = K + L$. In scientific applications characterized by a large dynamic range, the use of such an uniform number representation grid is not the most efficient way of using the available $B+1$ bits.

Floating-point representations: In this type of representations, a number $x \in \mathbb{R}$ is represented as

$$x = \underbrace{b_0 b_1 \dots b_l}_{M} \underbrace{b_{l+1} \dots b_{l+k}}_{E} = M \times 2^E \tag{10.5}$$

where M is a signed mantissa and E is an exponent. Several formats do exist for binary encoding of M and E.

Example 10.2:

▶ THe IEEE 753 floating-point format uses 32 bvits in total. 1 is reserved for the sign, 23 for the mantissa, and 8 for the exponent (see Figure 10.1). The corresponding binary-to-decimal mapping is

$$x = (-1)^S \times (0.1M) \times 2^{E-126}$$

◀

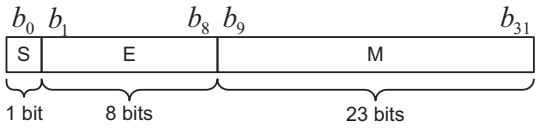


Fig. 10.1 Floating-point representation as per format IEEE753.

Observations: Floating-point representations offer several advantages over fixed-point:

- Larger dynamic range
- Variable resolution (depends on value of E)

However, floating-point hardware is usually more expensive, so that fixed-point is often used when system cost is a primary factor.

In the sequel, we focus on fixed-point representations with small number of bits (say 16 or less), where quantization effects are the most significant. For simplicity, the emphasis in give to fractional representations, for which

$$K = 0 \quad \text{and} \quad L = B.$$

However, generalizations to mixed (i.e. integer/fractional) representations are immediate.

10.1.2 Quantization errors in fixed-point arithmetic

Sources of errors: The two basic arithmetic operations in any DSP system are multiplication and addition, as exemplified by

$$y[n] = 0.5(x[n] + x[n-1]).$$

Because only a finite number of bits is available to represent the result of these operations, internal errors will occur. Multiplication and addition in fixed-point lead to very different types of errors.

Multiplication: Consider two numbers, say a and b, each expressed in fixed-point fractional representation with $(B+1)$-bit words:

$$\begin{aligned} a &= \alpha_0.\alpha_{-1}\cdots\alpha_{-B} \\ b &= \beta_0.\beta_{-1}\cdots\beta_{-B} \end{aligned}$$

Multiplication of these two $(B+1)$-bit words in general leads to a $(B'+1)$-bit word, where here $B'=2B$. This is illustrated below:

$$0.101 \times 0.001 = 0.000101$$

In practice, only $B+1$ bits are available to store the result, generally leading to a so-called round-off error. There are two basic ways in which number $c = a \times b$ with $B'+1$ bits can be represented with $B+1$ bits where $B < B'$:

- *Truncation*: The truncation of x with $B'+1$ bits into $\hat{x}$ with $B+1$ bits, where $B < B'$, is expressed as follows:

$$x = (b_0.b_{-1}\ldots b_{-B'}) \rightarrow \hat{x} = Q_{\text{tr}}[x] = (b_0.b_{-1}\ldots b_{-B}). \tag{10.6}$$

  In other words, the unwanted bits are simply dropped.

- *Rounding*: The operation of rounding may be expressed as follows:

$$x = (b_0.b_{-1}\ldots b_{-B'}) \rightarrow \hat{x} = Q_{\text{rn}}[x] = (b'_0.b'_{-1}\ldots b'_{-B}). \tag{10.7}$$

  so that $|x-\hat{x}|$ is minimized. That is, the available number $\hat{x}$ that is closest to x is used for its representation. The bits b'_i $(i=0,\ldots,B)$ may be different from the original bits b_i.

Example 10.3:

► Consider the representation of 5-bit number $x = (0.0011) = 3/16$, using only a 3-bit word (i.e. $B'=4$, $B=2$). For truncation, we have $x \rightarrow \hat{x} = (0.00) = 0$ For rounding, we have $x \rightarrow \hat{x} = (0.01) = 1/4$ ◄

Quantization error: Both rounding and truncation lead to a so-called quantization error, defined as

$$e \triangleq Q[x] - x \tag{10.8}$$

The value of e depends on the type of representation being used (e.g. Sign-Magnitude, 2C), as well as whether rounding or truncation is being applied. For the 2C representation (most often used), it can be generally verified that for truncation:

$$-\Delta < e \leq 0 \quad \text{where} \quad \Delta \triangleq 2^{-B}$$

while for rounding:

$$-\Delta/2 < e \leq \Delta/2$$

Example 10.4:

▶ Consider the truncation of $(1.0111)_{2C}$ to 2+1 bits (i.e. $B = 2$):

$$x = (1.0111)_{2C} = -9/16 \rightarrow \hat{x} = Q_{tr}[x] = (1.01)_{2C} = -3/4$$

$$e = \hat{x} - x = -3/4 + 9/16 = -3/16$$

Now consider the case of rounding:

$$x = (1.0111)_{2C} = -9/16 \rightarrow \hat{x} = Q_{rn}[x] = (1.10)_{2C} = -1/2$$

$$e = \hat{x} - x = -1/2 + 9/16 = 1/16$$

◀

Addition: Consider again two numbers, say a and b, each expressed in fixed-point fractional representation with $B+1$ bits. In theory, the sum of these two numbers may require up to $B+2$ bits for its correct representation. That is:

$$c = a + b = (c_1 c_0 . c_{-1} \ldots c_{-B})$$

where c_1 is a sign bit and the binary point has been shifted by one bit to the right. In practice, only $B+1$ bits are available to store the result c, leading to a type of error called overflow.

Example 10.5:

▶ Assuming 2C representation, we have:

$$(1.01) + (1.10) = -3/4 - 1/2 = -5/4$$

which cannot be represented exactly in a $(2+1)$-bit fractional 2C format. In a practical DSP system, the above operation would be realized using modulo-2 addition, leading to an erroneous result of 3/4 (i.e. left carry bit being lost). ◀

In conventional applications of digital filters, overflow should by all mean be avoided, as it introduces significant distortions in the system output. In practice, two basic means are available to avoid overflow:

- Scaling of signals at various point in a DSP system (corresponds to a left shift of the binary point; resolution is lost)
- Use of temporary guard bits to the left of the binary point.

10.2 Effects of coefficient quantization

In our previous study of filter structures and design techniques, the filter coefficients were implicitly assumed to be represented in infinite precision. In practice, the filter coefficients must be quantized to a finite precision representation (e.g. fixed-point, 16 bits) prior to the system's implementation. For example, consider a DFII realization of an IIR filter with system function

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{N} a_k z^{-k}} \tag{10.9}$$

Due to quantization of the filter coefficients $a_k \to \hat{a}_k$ and $b_k \to \hat{b}_k$, the corresponding system function is no longer $H(z)$ but instead

$$\begin{aligned} \hat{H}(z) &= \frac{\sum_{k=0}^{M} \hat{b}_k z^{-k}}{1 - \sum_{k=1}^{N} \hat{a}_k z^{-k}} \\ &= H(z) + \Delta H(z) \end{aligned} \tag{10.10}$$

Equivalently, one may think of the poles p_k and zeros z_k of $H(z)$ being displaced to new locations:

$$\hat{p}_k = p_k + \Delta p_k \tag{10.11}$$

$$\hat{z}_k = z_k + \Delta z_k \tag{10.12}$$

Small changes in the filter coefficients due to quantization may lead to very significant displacements of the poles and zeros that dramatically affect $H(z)$, i.e. very large $\Delta H(z)$.

It should be clear that quantization effects may be different for two different filter realizations with the same system function $H(z)$. Indeed, even tough the two realizations have the same transfer characteristics, the actual system coefficients in the two realizations may be different, leading to different errors in the presence of quantization.

In this section, we investigate such quantization effects.

10.2.1 Sensitivity analysis for direct form IIR filters

Consider an IIR filter with corresponding system function

$$H(z) = \frac{B(z)}{A(z)} \tag{10.13}$$

where $B(z)$ and $A(z)$ are polynomial in z^{-1}:

$$\begin{aligned} B(z) &= b_0 + b_1 z^{-1} + \cdots + b_M z^{-M} \\ &= b_0 \prod_{k=1}^{M} (1 - z_k z^{-1}) \end{aligned} \tag{10.14}$$

$$\begin{aligned} A(z) &= 1 - a_1 z^{-1} - \cdots - a_N z^{-N} \\ &= \prod_{k=1}^{M} (1 - p_k z^{-1}) \end{aligned} \tag{10.15}$$

Note that the filter coefficients b_k and a_k uniquely define the poles and zeros of the system, and vice versa. Thus, any change in the a_ks and b_ks will be accompanied by corresponding changes in p_ks and z_ks.

Below, we investigate the sensitivity of the pole-zero locations to small quantization errors in a_ks and b_ks via a first order perturbation analysis. Note that in a direct form realization of $H(z)$, the filter coefficients are precisely the a_ks and b_ks. Thus the results of this analysis will be immediately applicable to IIR filters realized in DFI, DFII and DFII-transposed.

PZ sensitivity: In Suppose the filter coefficients in (10.13)– (10.15) are quantized to

$$\hat{a} = Q[a_k] = a_k + \Delta a_k \tag{10.16}$$
$$\hat{b} = Q[b_k] = b_k + \Delta b_k \tag{10.17}$$

The resulting system function is now $\hat{H}(z) = \hat{B}(z)/\hat{A}(z)$ where

$$\begin{aligned}\hat{B}(z) &= \hat{b}_0 + \hat{b}_1 z^{-1} + \cdots + \hat{b}_M z^{-M} \\ &= \hat{b}_0 \prod_{k=1}^{M} (1 - \hat{z}_k z^{-1}) \end{aligned} \tag{10.18}$$

$$\begin{aligned}\hat{A}(z) &= 1 - \hat{a}_1 z^{-1} - \cdots - \hat{a}_N z^{-N} \\ &= \prod_{k=1}^{M} (1 - \hat{p}_k z^{-1}) \end{aligned} \tag{10.19}$$

Using the chain rule of calculus, it can be shown that to the first order in Δa_k, the poles of the quantized system become

$$\hat{p}_k = p_k + \Delta p_k \tag{10.20}$$
$$\Delta p_k = \frac{\sum_{l=1}^{N} p_k^{N-l} \Delta a_l}{\prod_{l \neq k} (p_k - p_l)} \tag{10.21}$$

A similar formula can be derived for the variation in the zeros of the quantized system as a function of Δb_k.

Remarks: For a system with a cluster of two or more closely spaced poles, we have

$$\begin{aligned} p_k \approx p_l \text{ for some } l \neq k &\Rightarrow \Delta p_k \text{ very large} \\ &\Rightarrow \Delta H(z) \text{ very large} \end{aligned} \tag{10.22}$$

The above problem posed by a cluster of poles becomes worse as N increases (higher probability of having closely spaced poles). To avoid the pole cluster problem with IIR filters:

- do not use direct form realization for large N;
- instead, use cascade or parallel realizations with low order sections (usually in direct form II)

As indicated above, the quantization effects on the zeros of the system are described by similar equations.

- However, the zeros z_ks are usually not as clustered as the poles p_ks in practical filter designs.
- Also, $\Delta H(z)$ is typically less sensitive to Δz_k than to Δp_k.

10.2.2 Poles of quantized 2nd order system

Intro: Consider a 2nd order all-pole filter section with

$$H(z) = \frac{1}{1 - 2\alpha_1 z^{-1} + \alpha_2 z^{-2}} \tag{10.23}$$

Observe that the corresponding poles, say p_1 and p_2, are functions of the coefficients α_1 and α_2. Thus, if α_1 and α_2 are quantized to $B+1$ bits, only a finite number of pole locations are possible. Furthermore, only a subset of these locations will correspond to a stable and causal DT system.

System poles: The poles of H(z) in (10.23) are obtained by solving for the roots of $z^2 - 2\alpha_1 z + \alpha_2 = 0$ or equivalently:

$$z = \alpha_1 \pm \sqrt{\Delta} \quad \text{where} \quad \Delta = \alpha_1^2 - \alpha_2$$

We need to distinguish three cases,

- $\Delta = 0$: Real double pole at
$$p_1 = p_2 = \alpha_1$$
- $\Delta > 0$: Distinct real poles at
$$p_{1,2} = \alpha_1 \pm \sqrt{\Delta}$$
- $\Delta < 0$: Complex conjugate poles at
$$p_{1,2} = \alpha_1 \pm j\sqrt{|\Delta|} = re^{\pm j\theta}$$
where the parameters r and θ satisfy
$$r = \sqrt{\alpha_2} \qquad r\cos\theta = \alpha_1 \tag{10.24}$$

Coefficient quantization: Suppose that a $B+1$-bit, fractional sign-magnitude representation is used for the storage of the filter coefficients a_1 and a_2, i.e.:

$$a_i \in \{0.b_1 \cdots b_B : b_i = 0 \text{ or } 1\}$$

Accordingly, only certain locations are possible for the corresponding poles p_1 and p_2. This is illustrated in Figure 10.2in the case $B = 3$ where we make the following observation:

- The top part shows the quantized (a_1, a_2)-plane, where circles correspond to $\Delta < 0$ (complex conjugate pole locations), bullets to $\Delta = 0$ and x's to $\Delta > 0$.
- The bottom part illustrates the corresponding pole locations in the z-plane in the case $\Delta < 0$ and $\Delta = 0$ only (distinct real poles not shown to simplify presentation).
- Each circle in the top figure corresponds to a pair of complex conjugate location in the bottom figure (see e.g. circles labelled $1, 2$, etc.)
- According to (10.24), the complex conjugate poles are at the intersections of circles with radius $\sqrt{\alpha_2}$ and vertical lines with ordinates $r\cos\theta = \alpha_1$. That is, distance from the origin and projection on real-axis are quantized in this scheme.

10.3 Quantization noise in digital filters

In a digital filter, each multiplier introduces a round-off error signal, also known as quantization noise. The error signal from each multiplier propagates through the system, sometimes in a recursive manner (i.e. via feedback loops). At the system output, the round-off errors contributed from each multiplier build up in a cumulative way, as represented by a total output quantization noise signal.

In this Section, we present a systematic approach for the evaluation of the total quantization noise power at the output of a digitally implemented LTI system. This approach is characterized by the use of an equivalent random linear model for (non-linear) digital filters. Basic properties of LTI systems excited by random sequences are invoked in the derivation.

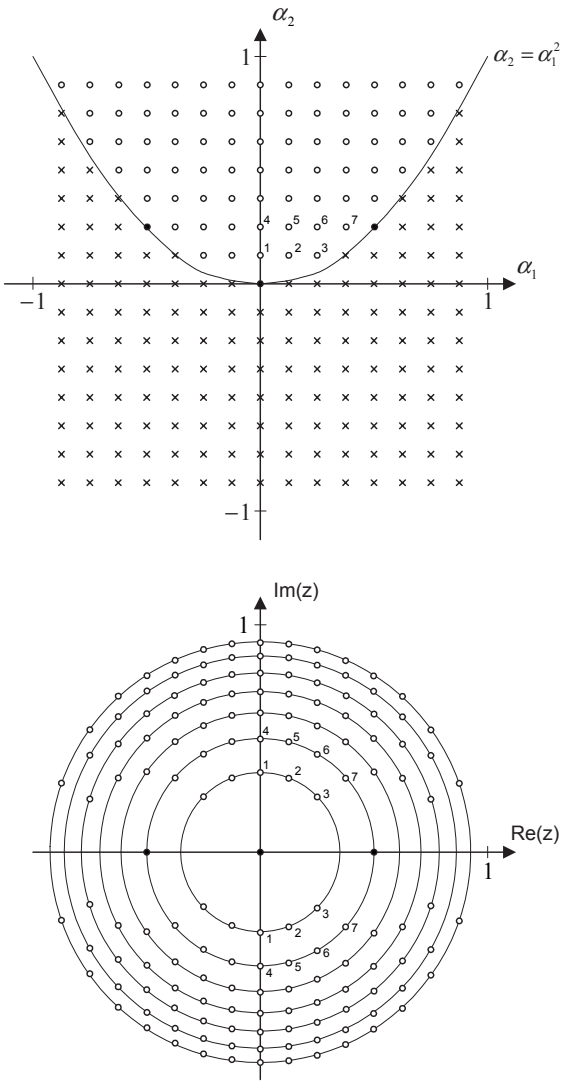


Fig. 10.2 Illustration of coefficient quantization in a second-order system..

Non-linear model of digital filter: Let $\mathcal{H}$ denote an LTI filter structure with L multiplicative branches. Let $u_i[n]$, α_i and $v_i[n]$ respectively denote the input, multiplicative gain and output of the ith branch ($i = 1, \ldots, L$), as shown in Figure 10.3 (left).

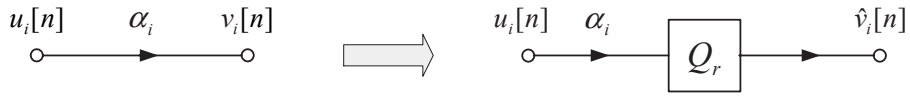


Fig. 10.3 Non-linear model of digital multiplier.

Let $\hat{\mathcal{H}}$ denote the $(B+1)$-bit fixed-point implementation of $\mathcal{H}$, where it is assumed that each multiplier output is rounded to $B+1$ bits:

$$v_i[n] = \underbrace{\alpha_i u_i[n]}_{2B+1} \Longrightarrow \hat{v}_i[n] = \underbrace{Q_r(\alpha_i u_i[n])}_{B+1} \tag{10.25}$$

In the deterministic non-linear model of $\hat{\mathcal{H}}$, each multiplicative branch α_i in $\mathcal{H}$ is modified as shown in Figure 10.3 (right).

Equivalent linear-noise model: In terms of the quantization error $e_i[n]$, we have

$$\begin{aligned} \hat{v}_i[n] &= Q_r(\alpha_i u_i[n]) \\ &= \alpha_i u_i[n] + e_i[n] \end{aligned} \tag{10.26}$$

In the equivalent linear-noise model of $\hat{\mathcal{H}}$, each multiplicative branch α_i with quantizer $Q_r(.)$ is replaced as shown in Figure 10.4.

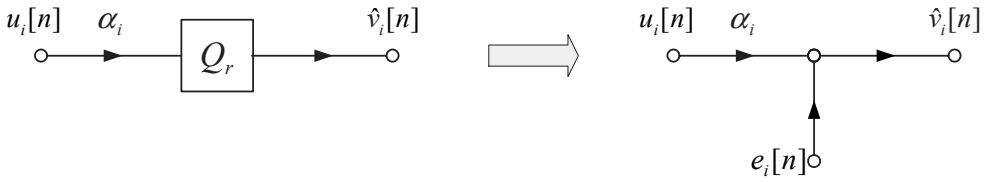


Fig. 10.4 Linear-noise model of digital multiplier.

The quantization error signal $e_i[n]$ ($n \in \mathbb{Z}$) is modelled as a white noise sequence, uncorrelated with the system input $x[n]$ and other quantization error signals $e_j[n]$ for $j \neq i$. For a fixed n, we assume that $e_i[n]$ is uniformly distributed within the interval $\pm\frac{1}{2}2^{-B}$, so that

- its mean value (or DC value) is zero:

$$E\{e_i[n]\} = 0 \tag{10.27}$$

- its variance (or noise power) is given by:

$$E\{e_i[n]^2\} = \frac{2^{-2B}}{12} \triangleq \sigma_e^2 \tag{10.28}$$

Property 1: Consider an LTI system with system function $K(z)$. Suppose that a zero mean white noise sequence $e[n]$ with variance σ_e^2 is applied to its input and let $f[n]$ denote the corresponding output (see Figure 10.5). Then, the sequence $f[n]$ has zero-mean and its variance σ_f^2 is given by

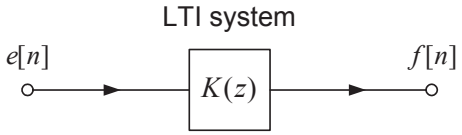


Fig. 10.5 Illustration of LTI system $K(z)$ with stochastic input $e[n]$.

$$\sigma_f^2 = \sigma_e^2 \sum_{n=-\infty}^{\infty} |k[n]|^2 = \frac{\sigma_e^2}{2\pi} \int_{-\pi}^{\pi} |K(\omega)|^2 d\omega \tag{10.29}$$

where $k[n]$ denotes the impulse response of LTI system $K(z)$.

Property 2: Let $e_1[n], \ldots, e_L[n]$ be zero-mean, uncorrelated white noise sequences with variance σ_e^2. Suppose that each sequence $e_i[n]$ is applied to the input of an LTI system $K_i(z)$ and let $f_i[n]$ denote the corresponding output. Finally, let $f[n] = f_1[n] + \cdots f_L[n]$ (see Figure 10.6). Then, the sequence $f[n]$ has zero-mean and variance

$$\sigma_f^2 = \sigma_{f_1}^2 + \cdots + \sigma_{f_L}^2 \tag{10.30}$$

where the $\sigma_{f_i}^2$ are computed as in Property 1.

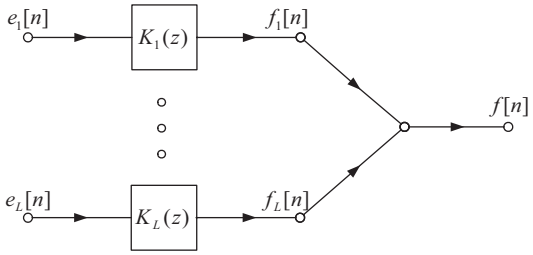


Fig. 10.6 LTI systems $K_i(z)$ with stochastic inputs $e_i[n]$ and combined output $f[n]$.

Linear superposition of noise sources: Consider an LTI filter structure $\mathcal{H}$ with L multiplicative branches. Assume that $\mathcal{H}$ is implemented in fixed-point arithmetic and let $\hat{\mathcal{H}}$ denote the corresponding linear-noise model: where the noise sources $e_i[n]$ are modelled as uncorrelated zero-mean white noise sources, as previously described.

$\hat{\mathcal{H}}$ may be interpreted as a linear system with a main input $x[n]$, L secondary inputs $e_i[n]$ $(i = 1, \ldots, L)$ and a main output $\hat{y}[n]$. Invoking the principle of superposition for linear systems, the output may be expressed as

$$\hat{y}[n] = y[n] + f_1[n] + \cdots + f_L[n] \tag{10.31}$$

In this equation:

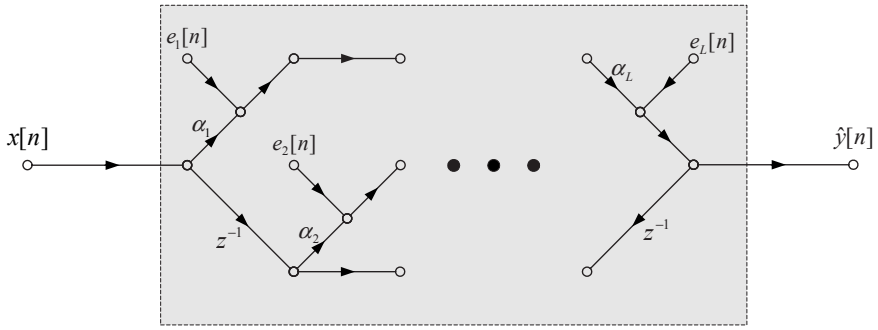


Fig. 10.7 Linear-noise model of LTI system with L multipliers.

- $y[n]$ is the desired response in the absence of quantization noise:

$$y[n] \triangleq H(z)x[n]$$

  where $H(z)$ is the system function in infinite precision.

- $f_i[n]$ is the individual contribution of noise source $e_i[n]$ to the system output in the absence of other noise sources and of main input $x[n]$:

$$f_i[n] = K_i(z)e_i[n]$$

  where $K_i(z)$ is defined as the transfer function between the injection point of $e_i[n]$ and the system output $\hat{y}[n]$ when $x[n] = 0$ and $e_j[n] = 0$ for all $j \neq i$.

Total output noise power: According to (10.31), the total quantization noise at the system output is given by

$$f[n] = f_1[n] + \cdots + f_L[n]$$

Invoking Properties 1 and 2, we conclude that the quantization noise $f[n]$ has zero-mean and its variance is given by

$$\sigma_f^2 = \sum_{i=1}^{L} \sigma_e^2 \sum_{n=-\infty}^{\infty} |k_i[n]|^2 \tag{10.32}$$

$$= \frac{2^{-B}}{12} \sum_{i=1}^{L} \sum_{n=-\infty}^{\infty} |k_i[n]|^2 \tag{10.33}$$

where (10.28) has been used. The variance σ_f^2 obtained in this way provides a measure of the quantization noise power at the system output.

Note on the computation of (10.33): To compute the quantization noise power σ_f^2 as given by (10.33), it is first necessary to determine the individual transfer functions $K_i(z)$ seen by each of the noise sources $e_i[n]$. This information can be obtained from the flowgraph of the linear-noise model $\hat{\mathcal{H}}$ by setting $x[n] = 0$ and $e_j[n] = 0$ for all $j \neq i$. For simple system functions $K_i(z)$, it may be possible to determine the associated impulse response $k_i[n]$ via inverse z-transform and then compute a numerical value for the sum of squared magnitudes in (10.33). This is the approach taken in the following example.

Example 10.6:

► Consider the system function

$$H(z) = \frac{1}{(1-a_1 z^{-1})(1-a_2 z^{-2})}$$

where $a_1 = 1/2$ and $a_2 = 1/4$. Let $\mathcal{H}$ denote a cascade realization of $H(z)$ using 1st order sections, as illustrated in Figure 10.8

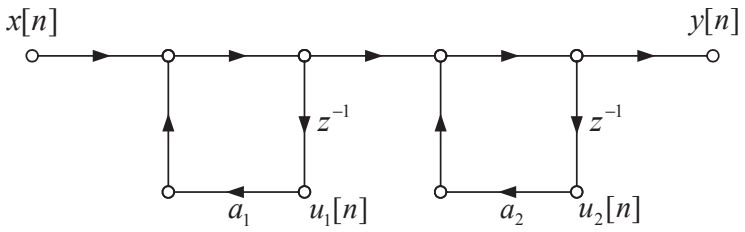


Fig. 10.8 Cascade realization of a 2nd order system.

Consider the fixed-point implementation of $\mathcal{H}$ using a $B+1$ bits fractional two's complement (2C) number representation in which multiplications are rounded. In this implementation of $\mathcal{H}$, each product $a_i u_i[n]$ in Fig. 10.8, which requires $2B+1$ for its exact representation, is rounded to $B+1$ bits:

$$\underbrace{a_i u_i[n]}_{2B+1 \text{ bits}} \rightarrow \underbrace{Q_r(a_i u_i[n])}_{B+1 \text{ bits}} = a_i u_i[n] + e_i[n]$$

where $e_i[n]$ denotes round-off error. The equivalent linear-noise model for $\mathcal{H}$, shown in Fig. 10.9, may be viewed as a multiple input-single output LTI system.

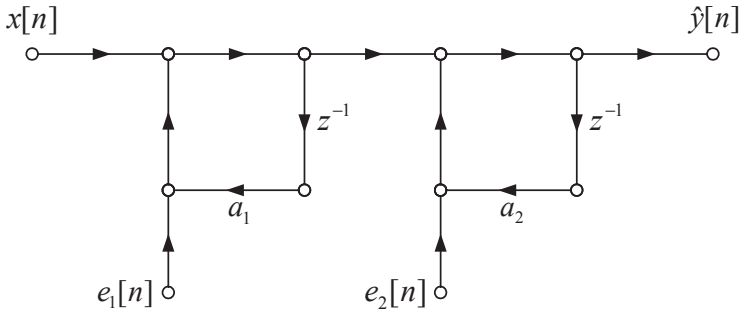


Fig. 10.9 Equivalent linear noise model.

The output signal $\hat{y}[n]$ in Fig. 10.9 may be expressed as

$$\hat{y}[n] = y[n] + f[n]$$

where $y[n] = H(z)x[n]$ is the desired output and $f[n]$ represents the cumulative effect of the quantization noise at the system's output. In turns, $f[n]$ may be expressed as

$$f[n] = f_1[n] + f_2[n]$$

where

$$f_i[n] = K_i(z)e_i[n]$$

represents the individual contribution of noise source $e_i[n]$ and $K_i(z)$ is the system function between the injection point of $e_i[n]$ and the system output, obtained by setting the input as well as all other noise sources to zero. To compute the noise power contributed by source $e_i[n]$, we need to find $K_i(z)$ and compute the energy in its impulse response:

- To obtain $K_1(z)$, set $x[n] = e_2[n] = 0$:

$$\begin{aligned} K_1(z) &= H(z) \\ &= \frac{1}{(1-a_1z^{-1})(1-a_2z^{-2})} \\ &= \frac{2}{1-\frac{1}{2}z^{-1}} - \frac{1}{1-\frac{1}{4}z^{-1}} \end{aligned}$$

  The impulse response is obtained via inverse z-transform (hence the partial fraction expansion) assuming a causal system:

$$k_1[n] = \left[2\left(\frac{1}{2}\right)^n - \left(\frac{1}{4}\right)^n\right]u[n]$$

  from which we compute the energy as follows:

$$\sum_{\text{all } n} |k_1[n]|^2 = \sum_{n=0}^{\infty}\left[2\left(\frac{1}{2}\right)^n - \left(\frac{1}{4}\right)^n\right]^2 = \cdots = 1.83$$

- To obtain $K_2(z)$, set $x[n] = e_1[n] = 0$:

$$K_1(z) = \frac{1}{1-a_2z^{-2}}$$

  The impulse response is

$$k_1[n] = \left(\frac{1}{4}\right)^n u[n]$$

  with corresponding energy

$$\sum_{\text{all } n} |k_2[n]|^2 = \sum_{n=0}^{\infty}\left(\frac{1}{4}\right)^{2n} = \cdots = 1.07$$

Finally, according to (10.33), the total quantization noise power at the system output is obtained as :

$$\begin{aligned} \sigma_f^2 &= \sigma_e^2\left(\sum_{\text{all } n}|k_1[n]|^2 + \sum_{\text{all } n}|k_2[n]|^2\right) \\ &= \frac{2^{-2B}}{12}(1.83+1.07) = 2.90\frac{2^{-2B}}{12} \end{aligned} \tag{10.34}$$

We leave it as an exercise for the student to verify that if the computation is repeated with $a_1 = 1/4$ and $a_2 = 1/2$ (i.e. reversing the order of the 1st order sections), the result will be

$$\sigma_f^2 = 3.16\frac{2^{-2B}}{12}$$

Why...? ◄

Signal-to-quantization noise ratio (SQNR): We define the SQNR at the output of a digital system as follows:

$$\text{SQNR} = 10\log_{10}\frac{P_y}{\sigma_f^2} \quad \text{in dB} \tag{10.35}$$

where

- P_y denotes the average power of the output signal $y[n]$ in infinite precision (see below).
- σ_f^2 denotes the average power of the total quantization noise $f[n]$ at the system output. This is computed as explained previously.

Output signal power: The computation of the output signal power P_y depends on the model used for the input signal $x[n]$. Two important cases are considered below:

- Random input: Suppose $x[n]$ is a zero-mean white noise sequence with variance σ_x^2. Then, from Property 1, we have
$$P_y = \sigma_x^2 \sum_{n=-\infty}^{\infty} |h[n]|^2 = \frac{\sigma_x^2}{2\pi}\int_{-\pi}^{\pi} |H(\omega)|^2 d\omega \tag{10.36}$$
- Sinusoidal input: Suppose $x[n] = A\sin(\omega_o n + \phi)$. Then, the average power of $y[n]$ is given by
$$P_y = \frac{A^2}{2}|H(\omega_o)|^2 \tag{10.37}$$

Example 10.7: (Ex. 10.6 cont.)

▶ Suppose that in Example 10.6, the input signal is a white noise sequence with sample values $x[n]$ uniformly distributed within $\pm X_{\max}$, where the maximum permissible value of $|x[n]|$, represented by $X_{\max}$, may be less than 1 to avoid potential internal overflow problems. From the above, it follows that $x[n]$ has zero-mean and

$$\sigma_x^2 = \frac{X_{\max}^2}{12}$$

The corresponding output power in infinite precision is computed from (10.35):

$$P_y = \sigma_x^2 \sum_{n=-\infty}^{\infty} |h[n]|^2 = 1.83\sigma_x^2 = 1.83\frac{X_{\max}^2}{12}$$

Finally, the SQNR is obtained as

$$\begin{aligned} \text{SQNR} &= 10\log_{10}\frac{P_y}{\sigma_f^2} \\ &= 10\log_{10}\frac{1.83X_{\max}^2/12}{2.902^{-2B}/12} \\ &\approx 6.02B + 20\log_{10}X_{\max} - 2.01 \quad \text{(dB)} \end{aligned}$$

Note the dependance on the peak signal value $X_{\max}$ and the number of bits B. Each additional bit contributes a 6dB increase in SQNR. In practice, the choice of $X_{\max}$ is limited by overflow considerations. ◀

Further note on the computation of (10.33): The approach used in the above example for the computation of $\sum_n |k_i[n]|^2$ may become cumbersome if $K_i(z)$ is too complex. For rational IIR transfer functions $K_i(z)$, the computation of the infinite sum can be avoided, based on the fact that

$$\sum_{n=-\infty}^{\infty} |k[n]|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} |K(\omega)|^2 d\omega$$

It has been shown earlier that the transfer function $C(z)$ with frequency response $C(\omega) = |K_i(\omega)|^2$ has poles given by d_k and $1/d_k^*$, where d_k is a (simple) pole of $K_i(z)$. Thus a partial fraction expansion of C(z) would yield a result in the form:

$$C(z) = C_1(z) + \sum_{k=1}^{N} \frac{A_k}{1-d_k z^{-1}} - \frac{A_k^*}{1-z^{-1}/d_k^*},$$

where $C_1(z)$ is a possible FIR component, and the A_k's are partial fraction expansion coefficients. It is easily shown by the definition of $C(z)$ as $K_i(z)K_i^*(1/z^*)$ that the partial fraction expansion corresponding to $1/d_k^*$ is indeed $-A_k^*$. Evaluating the above PFE on the unit circle and integrating over one period gives:

$$\sum_n |k[n]|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} C_1(\omega)d\omega + \sum_{k=1}^{N} \frac{1}{2\pi}\int_{-\pi}^{\pi} \{\frac{A_k}{1-d_k e^{-j\omega}} - \frac{A_k^*}{1-e^{-j\omega}/d_k^*}\} d\omega.$$

All the above terms are easily seen to be inverse discrete-time Fourier transforms evaluated at $n = 0$, so that

$$\sum_n |k[n]|^2 = c_1[0] + \sum_{k=1}^{N} A_k \tag{10.38}$$

Note that the term corresponding to A_k^* is zero at $n = 0$ since the corresponding sequence has to be anti-causal, starting at $n = -1$.

Finally, it often occurs that two or more, noise sources $e_i[n]$ have the same system function $K_i(z)$. In this case, according to (10.33), these noise sources may be replaced by a single noise source with a scaled variance of $q\sigma_e^2$, where q is the number of noise sources being merged.

Example 10.8:

▶ Let us study the actual effect of round-off noise on the DFII implementation of the filter designed by bilinear transformation in example 9.5. The coefficients of the transfer function are shown in table 9.2. Figure 10.10(a) shows the equivalent linear model to take quantization noise into account. The different $2N+1$ noise sources can be regrouped as in Figure 10.10(b) into two composite noise sources $e_1'[n]$ and $e_2'[n]$ with variances $N\sigma_e^2$ and $(N+1)\sigma_e^2$ respectively. The transfer function from $e_2'[n]$ to the output of the structure is just 1, whereas the noise source $e_1'[n]$ goes through the direct part of the transfer function $H(z)$. The corresponding output noise variances can therefore be computed as:

$$\sigma_{f_1'}^2 = \sigma_{e_1'}^2 \sum_{n=0}^{\infty} h[n]^2 = N\sigma_e^2 \sum_{n=0}^{\infty} h[n]^2 \tag{10.39}$$

$$\sigma_{f_2'}^2 = \sigma_{e_2'}^2 = (N+1)\sigma_e^2, \tag{10.40}$$

for a total noise power of

$$\sigma_f^2 = \frac{2^{-2B}}{12}(N+1+N\sum_{n=0}^{\infty} h[n]^2).$$

In the case of the coefficients shown in table 9.2, we will use the formula (10.38) in order to compute the sum over $h[n]^2$. The first step is to create the numerator and denominator coefficients of the transfer function $C(z) = H(z)H^*(1/z^*)$. We will use Matlab to do so. After grouping the coefficients of the numerator polynomial b_k in a column vector `b` in Matlab and doing the same for the a_k's in a vector `a`, one can easily find the numerator by `numc=conv(b,flipup(conj(b))`, and the denominator by `denc=conv(a,flipud(conj(a))`. To compute the partial fraction expansion, the function `residuez` is used, yielding:

$$\begin{aligned} C_1(z) &= 9.97\ 10^{-6} \\ \sum_{k=1}^{6} A_k &= 0.23463, \end{aligned}$$

so that the actual sum

$$\sum_{n=0}^{\infty} h[n]^2 = 0.23464.$$

Finally, the output noise variance is given by:

$$\sigma_f^2 = 0.70065\ 2^{-2B}.$$

For instance, assuming a 12-bit representation and an input signal uniformly distributed between $-\frac{1}{2}$ and $\frac{1}{2}$, we would get an SQNR of 50.7 dB. Though this value might sound high, it will not be sufficient for many applications (e.g. "CD quality" audio processing requires a SNR of the order of 95 dB).

Note that the example is not complete:

- the effect of coefficient quantization has not been taken into account. In particular, some low values of the number $B+1$ of bits used might push the poles outside of the unit circle and make the system unstable.
- the effect of overflows has not been taken into account; in practise, overflow in key nodes is avoided by a proper scaling of the input signal by a factor $s < 1$ such that the resulting signals in the nodes of interest do not overflow. The nodes to consider are final results of accumulated additions, since overflow in intermediate nodes is harmless in two's complement arithmetics. These nodes are, in the case of Figure 10.10(b), the two nodes where the noise components are injected. By looking at the transfer function from the input to these nodes, it is easy to find a conservative value of s that will prevent overflow. Note that this will also reduce the SQNR, since the power P_y will be scaled by a factor s^2.

Finally, let us point out that if we have at our disposal a double-length accumulator (as is available in many DSP chips, see chapter **??**), then the rounding or truncation only has to take place before storage in memory. Consequently, the only locations in the DFII signal flow graph where quantization will occur are right before the delay line, and before the output, as illustrated in Figure 10.11. Carrying out a study similar to the one outline above, we end up with a total output noise variance of

$$\sigma_f^2 = \frac{2^{-2B}}{12}(1 + \sum_{n=0}^{\infty} h[n]^2),$$

where the factors N have disappeared. With 12 bits available, this leads to an SQNR value of 59.02 dB. ◀

10.4 Scaling to avoid overflow

Problem statement: Consider a filter structure $\mathcal{H}$ with K adders and corresponding node values $w_i[n]$, as illustrated in Fig. 10.12. Note that whenever $|w_i[n]| > 1$, for a given adder at a given time, overflow

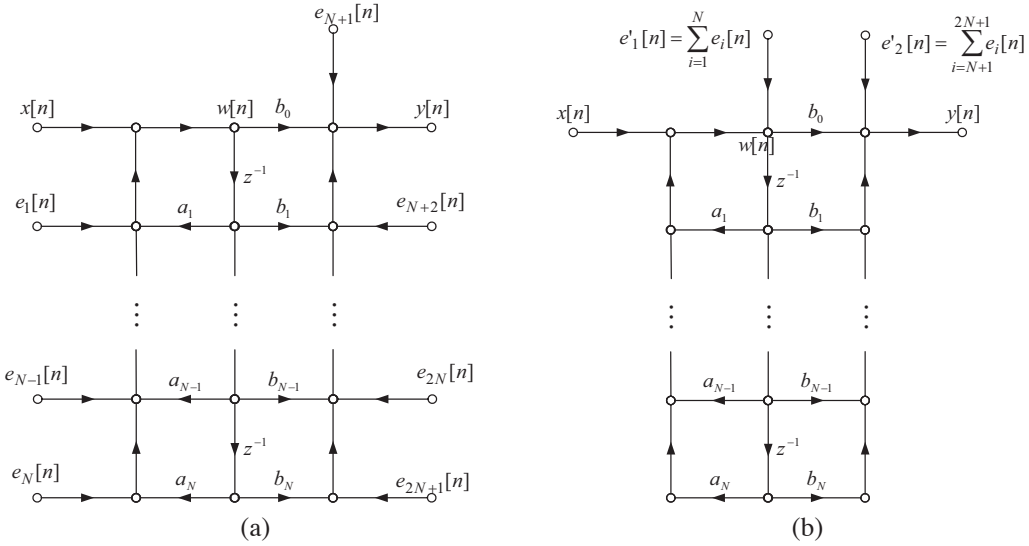


Fig. 10.10 DFII modified to take quantization effects into account (a) each multiplier contains a noise source and (b) a simplified version with composite noise sources.

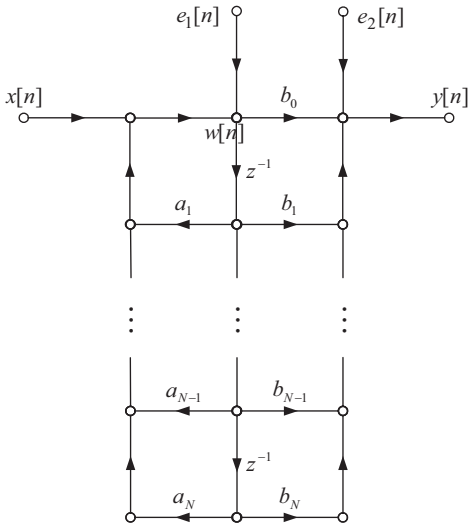


Fig. 10.11 Round-off noise model when a double-length accumulator is available.

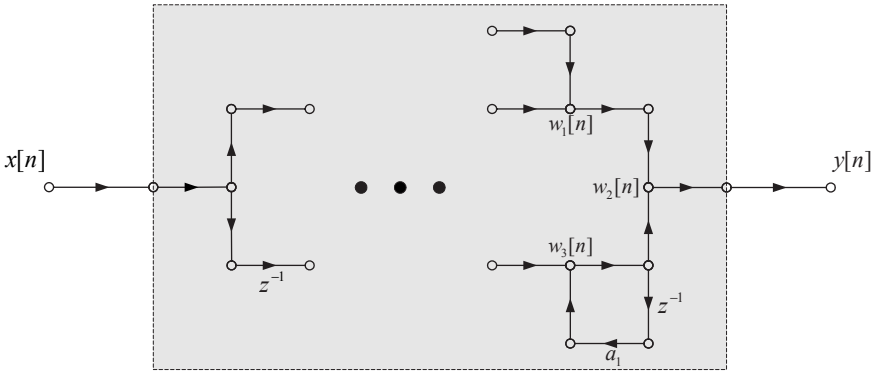


Fig. 10.12 Linear-noise model of LTI system with L multipliers.

will result in the corresponding fixed-point implementation. Overflow usually produces large, unpredictable distortion in the system output and must be avoided at all cost.

Principles of scaling: The basic idea is very simple: properly scale the input signal $x[n]$ to a range $|x[n]| < X_{\max}$ so that the condition

$$|w_i[n]| < 1, \quad i = 1, \ldots, K \tag{10.41}$$

is enforced at all time n. Several approaches exist for choosing $X_{\max}$, the maximum permissible value of $x[n]$. The best approach usually depends on the type of input signals being considered. Two such approaches are presented below:

Wideband signals: For wideband signals, a suitable value of $X_{\max}$ is provided by

$$X_{\max} < \frac{1}{\max_i \sum_n |h_i[n]|} \tag{10.42}$$

where $H_i(z)$ denotes the transfer function from the system input to the ith adder output. The above choice of $X_{\max}$ ensures that overflow will be avoided regardless of the particular type of inputs: it represents a sufficient condition to avoid overflow.

Narrowband signals: For narrowband signals, the above bound on $X_{\max}$ may be too conservative. An often better choice of $X_{\max}$ if provided by the bound

$$X_{\max} < \frac{1}{\max_{i,\omega} |H_i(\omega)|} \tag{10.43}$$

Chapter 11

Fast Fourier transform (FFT)

Computation of the discrete Fourier transform (DFT) is an essential step in many signal processing algorithms. Even for data sequences of moderate size N, a direct computation of the N-point DFT requires on the order of N^2 arithmetic operations. Even for moderate values of N, this usually entails significant computational costs.

In this Chapter, we investigate so-called *fast* Fourier transform (FFT) algorithms for the efficient computation of the DFT. These algorithms achieves significant savings in the DFT computation by exploiting certain symmetries of the Fourier basis functions and only require on the order of $N\log_2 N$ arithmetic operations. They are at the origin of major advances in the field of signal processing, particularly in the 60's and 70's. Nowadays, FFT algorithms find applications in numerous commercial DSP-based products.

11.1 Direct computation of the DFT

Recall the definition of the N-point DFT of a discrete-time signal $x[n]$ defined for $0 \leq n \leq N-1$:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad k = 0,1,...,N-1 \tag{11.1}$$

where for convenience, we have introduced

$$W_N \triangleq e^{-j2\pi/N}, \tag{11.2}$$

Based on (11.1), one can easily come up with the following algorithmic steps for its direct evaluation, also known as the direct approach:

Step 1: Compute W_N^l and store it in a table:

$$\begin{aligned} W_N^l &= e^{-j2\pi l/N} \\ &= \cos(2\pi l/N) + j\sin(2\pi l/N), \quad l = 0,1,...,N-1 \end{aligned} \tag{11.3}$$

Note: this only needs to be evaluated for $l = 0,1,...,N-1$ because of the periodicity.

Step 2: Compute the DFT $X[k]$ using stored values of W_N^l and input data $x[n]$:

$$\begin{aligned}
&\text{for } k = 0 : N-1 \\
&\quad X[k] \leftarrow x[0] \\
&\quad \text{for } n = 1 : N-1 \\
&\qquad l = (kn)_N \\
&\qquad X[k] \leftarrow X[k] + x[n]W_N^l \\
&\quad \text{end} \\
&\text{end}
\end{aligned} \tag{11.4}$$

The complexity of this approach can be broken up as follows:

- In step (1): N evaluation of the trigonometric functions sin and cos (actually less than N because of the symmetries). These values are usually computed once and stored for later use. We refer to this approach as *table look-up*.
- In step (2):
  - There are $N(N-1) \approx N^2$ complex multiplications ($\otimes_c$)
  - There are $N(N-1) \approx N^2$ complex additions ($\oplus_c$)
  - One must also take into account the overhead: indexing, addressing, etc.

Because of the figures above, we say that direct computation of the DFT is $O(N^2)$, i.e. of order N^2. The same is true for the IDFT.

In itself, the computation of the N-point DFT of a signal is a costly operation. Indeed, even for moderate values of N, the $O(N^2)$ complexity will usually require a considerable amount of computational resources.

The rest of this Chapter will be devoted to the description of efficient algorithms for the computation of the DFT. These so-called *Fast Fourier Transform* or FFT algorithms can achieve the same DFT computation in only $O(N\log_2 N)$.

11.2 Overview of FFT algorithms

In its most general sense, the term FFT refers to a family of *computationally* fast algorithms used to compute the DFT. FFT should not be thought of as a new transform: FFTs are merely algorithms.

Typically, FFT algorithms require $O(N\log_2 N)$ complex multiplications ($\otimes_c$), while the direct evaluation of the DFT requires $O(N^2)$ complex multiplications. For $N \gg 1$, $N\log_2 N \ll N^2$ so that there is a significant gain with the FFT.

Basic principle: The FFT relies on the concept of *divide and conquer*. It is obtained by breaking the DFT of size N into a cascade of smaller size DFTs. To achieve this, two essential ingredients are needed:

- N must be a composite number (e.g. $N = 6 = 2 \times 3$).

- The periodicity and symmetry properties of the factor W_N in (11.2) must be exploited, e.g.:

$$W_N^k = W_N^{k+N} \tag{11.5}$$
$$W_N^{Lk} = W_{N/L}^k. \tag{11.6}$$

Different types of FFTs: There are several FFT algorithms, e.g.:

- $N = 2^\nu \Rightarrow$ radix-2 FFTs. These are the most commonly used algorithms. Even then, there are many variations:
  - Decimation in time (DIT)
  - Decimation in frequency (DIF)
- $N = r^\nu \Rightarrow$ radix-r FFTs. The special cases $r = 3$ and $r = 4$ are not uncommon.
- More generally, $N = p_1 p_2 ... p_l$ where the p_is are prime numbers lead to so-called mixed-radix FFTs.

Radix-2 FFTs: In these algorithms, applicable when $N = 2^\nu$:

- the DFT_N is decomposed into a cascade of ν stages;
- each stage is made up of $\frac{N}{2}$ 2-point DFTs (DFT_2).

Radix-2 FFTs are possibly the most important ones. Only in very specialized situations will it be more advantageous to use other radix type FFTs. In these notes, we shall mainly focus on the radix-2 FFTs. However, the students should be able to extend the basic principles used in the derivation of radix-2 FFT algorithms to other radix type FFTs.

11.3 Radix-2 FFT via decimation-in-time

The basic idea behind decimation-in-time (DIT) is to partition the input sequence $x[n]$, of length $N = 2^\nu$, into two subsequences, i.e. $x[2r]$ and $x[2r+1]$, $r = 0, 1, ..., (N/2) - 1$, corresponding to even and odd values of time, respectively. It will be shown that the N-point DFT of $x[n]$ can be computed by properly combining the $(N/2)$-point DFTs of each subsequences. In turn, the same principle can be applied in the computation of the $(N/2)$-point DFT of each subsequence, which can be reduced to DFTs of size $N/4$. This basic principle is repeated until only 2-point DFTs are involved. The final result is an FFT algorithm of complexity $O(N \log_2 N)$

We will first review the 2-point DFT, which is the basic building block of radix-2 FFT algorithms. We will then explain how and why the above idea of decimation-in-time can be implemented mathematically. Finally, we discuss particularities of the resulting radix-2 DIT FFT algorithms.

The 2-point DFT: In the case $N = 2$, (11.1) specializes to

$$X[k] = x[0] + x[1]W_2^k, \quad k = 0, 1. \tag{11.7}$$

Since $W_2 = e^{-j\pi} = -1$, this can be further simplified to

$$X[0] = x[0] + x[1] \tag{11.8}$$
$$X[1] = x[0] - x[1], \tag{11.9}$$

which leads to a very simple realization of the 2-point DFT, as illustrated by the signal flowgraph in Figure 11.1.

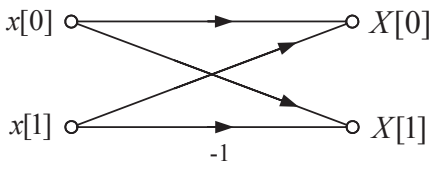


Fig. 11.1 Signal flow-graph of a 2-point DFT.

Main steps of DIT:

(1) Split the $\sum_n$ in (11.1) into $\sum_{n\,\text{even}} + \sum_{n\,\text{odd}}$

(2) Express the sums $\sum_{n\,\text{even}}$ and $\sum_{n\,\text{odd}}$ as $(N/2)$-point DFTs.

(3) If $N/2 = 2$ stop; else, repeat the above steps for each of the individual $(N/2)$-point DFTs.

Case $N = 4 = 2^2$:

- Step (1):

$$\begin{aligned} X[k] &= x[0] + x[1]W_4^k + x[2]W_4^{2k} + x[3]W_4^{3k} \\ &= (x[0] + x[2]W_4^{2k}) + W_4^k(x[1] + x[3]W_4^{2k}) \end{aligned} \tag{11.10}$$

- Step (2): Using the property $W_4^{2k} = W_2^k$, we can write

$$\begin{aligned} X[k] &= (x[0] + x[2]W_2^k) + W_4^k(x[1] + x[3]W_2^k) \\ &= G[k] + W_4^k H[k] \end{aligned} \tag{11.11}$$
$$G[k] \triangleq \text{DFT}_2\{\text{even samples}\} \tag{11.12}$$
$$H[k] \triangleq \text{DFT}_2\{\text{odd samples}\} \tag{11.13}$$

  Note that $G[k]$ and $H[k]$ are 2-periodic, i.e.

$$G[k+2] = G[k], \quad H[k+2] = H[k] \tag{11.14}$$

- Step (3): Since $N/2 = 2$, we simply stop; that is, the 2-point DFTs $G[k]$ and $H[k]$ cannot be further simplified via DIT.

The 4-point DFT can thus be computed by properly combining the 2-point DFTs of the even and odd samples, i.e. $G[k]$ and $H[k]$, respectively:

$$X[k] = G[k] + W_4^k H[k], \quad k = 0,1,2,3 \tag{11.15}$$

Since $G[k]$ and $H[k]$ are 2-periodic, they only need to be computed for $k = 0,1$, hence the equations:

$$\begin{aligned} X_0[k] &= G[0] + W_4^0 H[0] \\ X_1[k] &= G[1] + W_4^1 H[1] \\ X_2[k] &= G[2] + W_4^2 H[2] = G[0] + W_4^2 H[0] \\ X_3[k] &= G[3] + W_4^3 H[3] = G[1] + W_4^3 H[1] \end{aligned}$$

The flow graph corresponding to this realization is shown in Figure 11.2. Note the special ordering of the input data. Here further simplifications of the factors W_4^k are possible here: $W_4^0 = 1$, $W_4^1 = -j$, $W_4^2 = -1$ and $W_4^3 = j$.

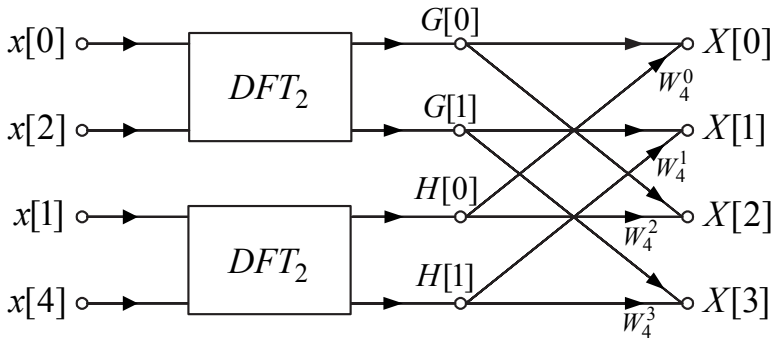


Fig. 11.2 Decimation in time implementation of a 4-point DFT. The DFT_2 blocks are as shown in figure 11.1.

General case:

- Step (1): Note that the even and odd samples of $x[n]$ can be represented respectively by the sequences $x[2r]$ and $x[2r+1]$, where the index r now runs from 0 to $\frac{N}{2} - 1$. Therefore

$$\begin{aligned} X[k] &= \sum_{n=0}^{N-1} x[n] W_N^{kn} \\ &= \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_N^{2kr} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_N^{2kr} \end{aligned} \tag{11.16}$$

- Step (2): Using the property $W_N^{2kr} = W_{N/2}^{kr}$, we obtain

$$\begin{aligned} X[k] &= \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_{N/2}^{kr} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_{N/2}^{kr} \\ &= G[k] + W_N^k H[k] \end{aligned} \tag{11.17}$$

where we define

$$G[k] \triangleq \text{DFT}_{N/2}\{x[2r], r = 0, 1..., \frac{N}{2} - 1\} \tag{11.18}$$

$$H[k] \triangleq \text{DFT}_{N/2}\{x[2r+1], r = 0, ..., \frac{N}{2} - 1\} \tag{11.19}$$

Note that $G[k]$ and $H[k]$ are $\frac{N}{2}$-periodic and need only be computed for $k = 0$ up to $\frac{N}{2} - 1$. Thus, for $k \geq N/2$, (11.17) is equivalent to

$$X[k] = G[k - \frac{N}{2}] + W_N^k H[k - \frac{N}{2}] \tag{11.20}$$

- Step (3): If $N/2 \geq 4$, apply these steps to each DFT of size $N/2$.

Example 11.1: Case $N = 8 = 2^3$

► The application of the general decimation in time principle to the case $N = 8$ is described by Figures 11.5 through 11.5, where, for the sake of uniformity, all phase factors have been expressed as W_8^k (e.g. $W_4^1 = W_8^2$).

Figure 11.3 illustrates the result of a first pass through DIT steps (1) and (2). The outputs of the top (bottom) 4-point DFT box are the coefficients $G[k]$ (resp. $H[k]$) for $k = 0, 1, 2, 3$. Note the ordering of the input data to accommodate the 4-point DFTs over even and odd samples. In DIT step (3), since

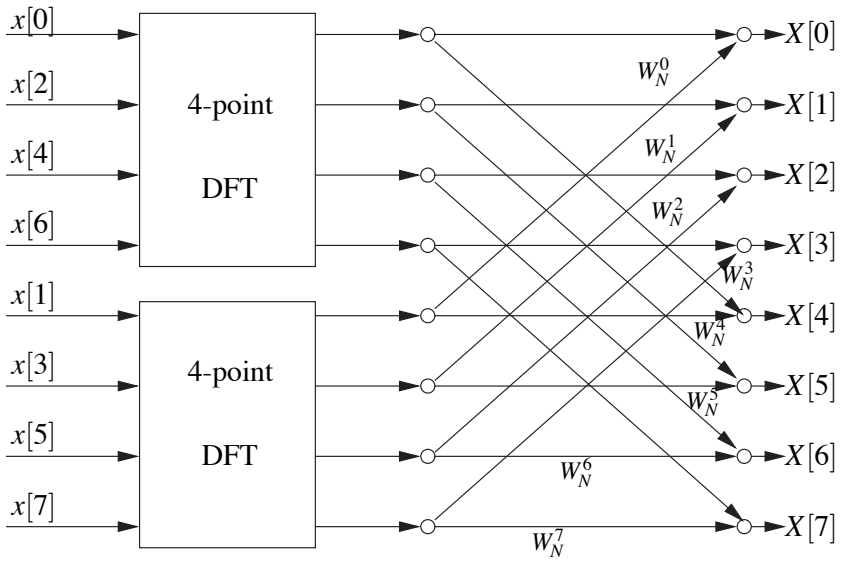


Fig. 11.3 Decomposition of an 8-point DFT in 2 4-point DFTs.

$N/2 = 4$, the DIT procedure is repeated for the computation of the 4-point DFTs $G[k]$ nd $H[k]$. This amounts to replacing the 4-point DFT boxes in Figure 11.3 by the flowgraph of the 4-point DIT FFT, as derived previously. In doing so, the order of the input sequence must be modified accordingly. The result is illustrated in Figure 11.4. The final step amount to replacing each of the 2-point DFTs in Figure 11.4 by their corresponding signal flowgraph. The result is illustrated in Figure 11.5. ◄

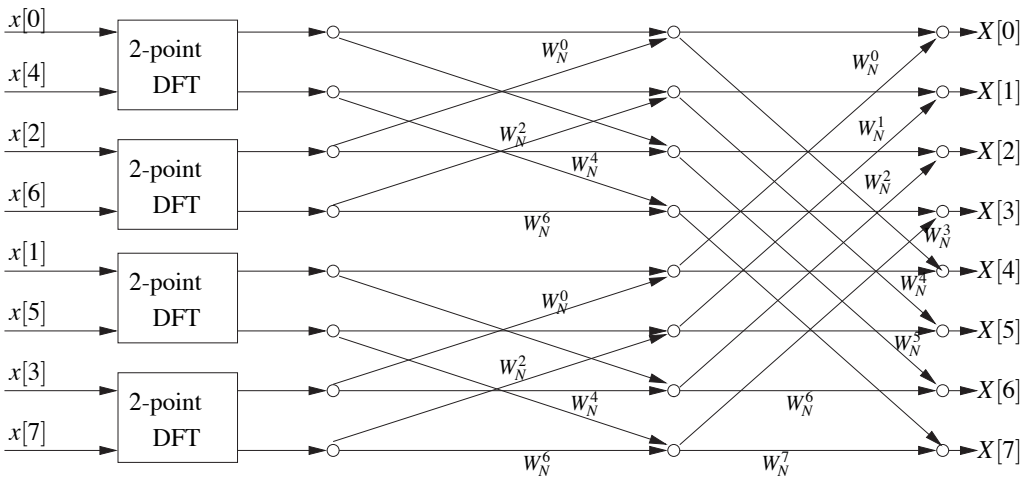


Fig. 11.4 Decomposition of an 8-point DFT in 4 2-point DFTs.

Computational complexity: The DIT FFT flowchart obtained for $N = 8$ is easily generalized to $N = 2^\nu$, i.e. an arbitrary power of 2. In the general case, the signal flow graph consists of a cascade of $\nu = \log_2 N$ stages. Each stage is made up of $N/2$ basic binary flow graphs called *butterflies*, as shown in Figure 11.6. In practice, a simplified butterfly structure is used instead of the one in Figure 11.6. Realizing that

$$W_N^{r+N/2} = W_N^r W_N^{N/2} = W_N^r e^{-j\pi} = -W_N^r,$$

an equivalent butterfly with only one complex multiplication can be implemented, as shown in Figure 11.7. By using the modified butterfly structure, the computational complexity of the DIT FFT algorithm can be determined based on the following:

- there are $\nu = \log_2 N$ stages;
- there are $N/2$ butterflies per stage;
- there is 1 $\otimes_c$ and 2 $\oplus_c$ per butterfly.

Hence, the total complexity:

$$\frac{N}{2}\log_2 N \otimes_c, \quad N\log_2 N \oplus_c \tag{11.21}$$

Ordering of input data: The input data $x[n]$ on the LHS of the FFT flow graph is not listed in standard sequential order (refer to Figures 11.2 or 11.5). Rather, the data is arranged is a so-called bit-reversed order. Indeed, if one looks at the example in Figure 11.5 and compares the indices of the samples $x[n]$ on the left of the structure with a natural ordering, one sees that the 3-bit binary representation of the actual index is the bit-reversed version of the 3-bit binary representation of the natural ordering:

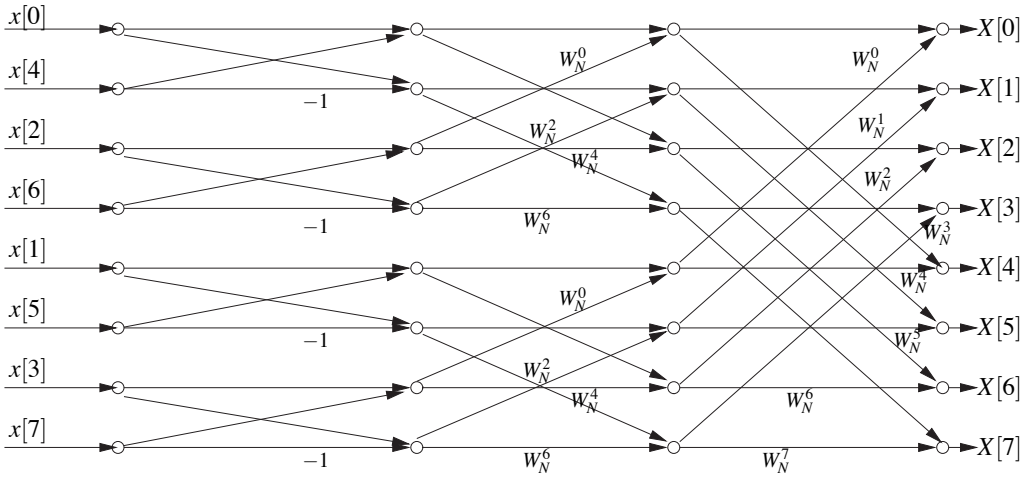


Fig. 11.5 An 8-point DFT implemented with the decimation-in-time algorithm.

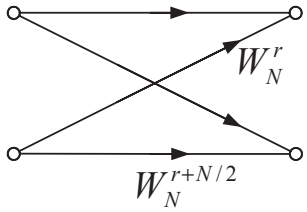


Fig. 11.6 A *butterfly*, the basic building block of an FFT algorithm.

Sample index		Actual offset in memory	
decim.	binary	decim.	binary
0	000	0	000
4	100	1	001
2	010	2	010
6	110	3	011
1	001	4	100
5	101	5	101
3	011	6	110
7	111	7	111

Practical FFT routines contain bit-reversing instructions, either prior or after the FFT computation, depending upon the specific nature of the algorithm. Some programmable DSP chips contain also an option for bit-reversed addressing of data in memory.

We note that the DIT FFT flowgraph can be modified so that the inputs are in sequential order (e.g. by properly re-ordering the vertical line segments in Fig. 11.3); but then the output $X[k]$ will be listed in bit-reversed order.

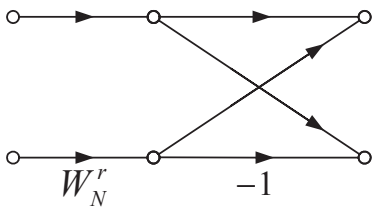


Fig. 11.7 A modified butterfly, with only one multiplication.

Storage requirement: A complex array of length N, say $A[q]$ $(q = 0, ..., N-1)$, is needed to store the input data $x[n]$. The computation can be done *in-place*, meaning that the same array $A[q]$ is used to store the results of intermediate computations. This is made possible by the very structure of the FFT flow graph, which is entirely made up of independent binary butterfly computations. We note that once the outputs of a particular butterfly have been computed, the corresponding inputs are no longer needed and can be overwritten. Thus, provided a single additional complex register is made available, it is possible to overwrite the inputs of each butterfly computation with the corresponding outputs. Referring to Figure 11.7 and denoting by A_1 and A_2 the storage locations of the two inputs to the butterfly, we can write the pseudo-code of the butterfly:

$$\begin{aligned} \text{tmp} &\leftarrow A_2 * W_N^r \\ A_2 &\leftarrow A_1 - \text{tmp} \\ A_1 &\leftarrow A_1 + \text{tmp}, \end{aligned}$$

after which the input has been overwritten by the output, and use of the temporary storage location tmp has been made. This is so because the inputs of each butterfly are only used once.

FFT program: Based on the above consideration, it should be clear that a properly written program for the DIT FFT algorithm contains the following ingredients:

- Pre-processing to address the data in bit-reversed order;
- An outer loop over the stage number, say from $l = 1$ to $\nu = \log_2 N$.
- An inner loop over the butterfly number, say from $k = 1$ to $N/2$. Note that the specific indices of the butterfly inputs depends on the stage number l.

11.4 Decimation-in-frequency

Decimation in frequency is another way of decomposing the DFT computation so that the resulting algorithm has complexity $O(N \log_2 N)$. The principles are very similar to the decimation in time algorithm.

Basic steps: Assume $N = 2^\nu$, where ν integer ≥ 1.

(1) Partition DFT samples $X[k]$ $(k = 0, 1, \ldots, N-1)$ into two subsequences:

- even-indexed samples: $X[2r], r = 0, 1, ..., \frac{N}{2} - 1$

- odd-indexed samples: $X[2r+1], r = 0, 1, ..., \frac{N}{2} - 1$

(2) Express each subsequence as a DFT of size $N/2$.

(3) If $N/2 = 2$, stop; else, apply steps (1)-(3) to each DFT of size $N/2$.

Case $N = 4$:

- For even-indexed DFT samples, we have

$$\begin{aligned} X[2r] &= \sum_{n=0}^{3} x[n]W_4^{2rn}, \quad r = 0, 1 \\ &= x[0]W_4^0 + x[1]W_4^{2r} + x[2]W_4^{4r} + x[3]W_4^{6r} \end{aligned}$$

  Now, observe that:

$$\begin{aligned} W_4^0 &= 1 & W_4^{2r} &= W_2^r \\ W_4^{4r} &= 1 & W_4^{6r} &= W_4^{4r}W_4^{2r} = W_2^r \end{aligned}$$

  Therefore

$$\begin{aligned} X[2r] &= (x[0] + x[2]) + (x[1] + x[3])W_2^r \\ &= u[0] + u[1]W_2^r \\ &= \mathrm{DFT}_2\{u[0], u[1]\} \end{aligned}$$

$$u[r] \triangleq x[r] + x[r+2], \quad r = 0, 1$$

- In the same way, it can be verified that for the odd-indexed DFT samples,

$$X[2r+1] = \mathrm{DFT}_2\{v[0], v[1]\}$$

$$v[r] \triangleq W_4^r(x[r] - x[r+2]), \quad r = 0, 1$$

- The resulting signal flow graph is shown in Figure 11.8, where each of the 2-point DFT have been realized using the basic SFG in Figure 11.7.

General case: Following the same steps as in the case $N = 4$, the following mathematical expressions can be derived for the general case $N = 2^\nu$:

$$X[2r] = \mathrm{DFT}_{N/2}\{u[r]\} \tag{11.22}$$

$$X[2r+1] = \mathrm{DFT}_{N/2}\{v[r]\} \tag{11.23}$$

where $r = 0, 1, \ldots, N/2 - 1$ and

$$u[r] \triangleq x[r] + x[r + N/2] \tag{11.24}$$

$$v[r] \triangleq W_N^r(x[r] - x[r+2]) \tag{11.25}$$

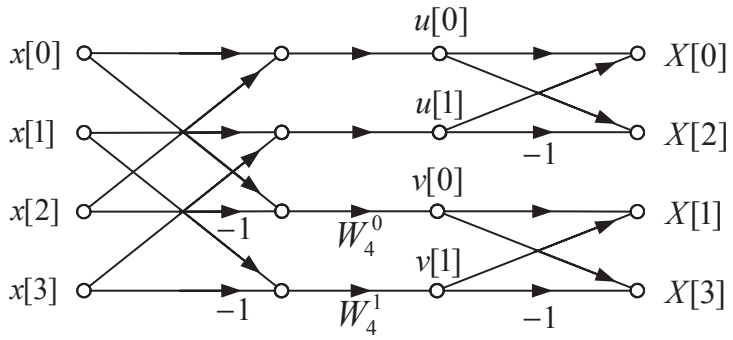


Fig. 11.8 Decimation in frequency realization of a 4-point DFT.

Remarks on the DIF FFT: Much like the DIT, the DIF radix-2 FFT algorithms have the following characteristics:

- computational complexity of $O(N\log_2 N)$;
- output $X[k]$ in bit-reversed order;
- in-place computation is also possible.

A program for FFT DIF contains components similar to those described previously for the FFT DIT.

11.5 Final remarks

Simplifications: The FFT algorithms discussed above can be further simplified in the following special cases:

- $x[n]$ is a real signal
- only need to compute selected values of $X[k]$ ($\Rightarrow$ FFT pruning)
- $x[n]$ containing large number of zero samples (e.g. as a result of zero-padding)

Generalizations: The DIT and DIF approaches can be generalized to:

- arbitrary radix (e.g. $N = 3^\nu$ and $N = 4^\nu$)
- mixed radix (e.g. $N = p_1 p_2 \cdots p_K$ where p_i are prime numbers)

Chapter 12

An introduction to Digital Signal Processors

12.1 Introduction

Digital Signal Processing Chips are specialized microprocessors aimed at performing DSP tasks. They are often called Digital Signal Processors — "DSPs" for hardware engineers. These DSP chips are sometimes referred to as PDSPs (Programmable Digital Signal Processor), as opposed to hard-wired solutions (ASICs) and reconfigurable solutions (FPGAs).

The first DSP chips appeared in the early 80's, mainly produced by Texas Instruments (TI). Nowadays, their power has dramatically increased, and the four big players in this game are now TI, Agere (formerly Lucent), Analog Devices and Motorola. All these companies provide families of DSP processors with different target applications – from low-cost to high performance.

As the processing capabilities of DSPs have increased, they are being used in more and more applications. Table 12.1 lists a series of application areas, together with the DSP operations they require, as well as a series of examples of devices or appliances where a DSP chip can be used.

The very need for specialized microprocessors to accomplish DSP-related tasks comes from a series of requirements of classic DSP algorithms, that could not be met by general purpose processors in the early 80's: DSPs have to support high-performance, repetitive and numerically intensive tasks.

In this chapter, we will review the reasons why a DSP chip should be different from an other microprocessor, through motivating examples of DSP algorithms. Based on this, we will analyze how DSP manufacturers have designed their chips so as to meet the DSP constraints, and review a series of features common to most DSP chips. Finally, we will briefly discuss the performance measures that can be used to compare different DSP chips.

12.2 Why PDSPs ?

To motivate the need for specialized DSP processors, we will study in this section the series of operations required to accomplish two basic DSP operations, namely FIR filtering and FFT computation.

Application Area	DSP task	Device
Speech & Audio Signal Processing	Effects (Reverb, Tone Control, Echo) , Filtering, Audio Compression, Speech Synthesis, Recognition & Compression, Frequency Equalization, Pitch , Surround Sound	Musical instruments & Amplifiers, Audio Mixing Consoles & Recording Equipment, Audio Equipment & Boards for PCs, Toys & Games, Automotive Sound Systems, DAT & CD Players, HDTV Equipment, Digital Tapeless Recorders, Cellular phones
Instrumentation and Measurement	Fast Fourier Transform (FFT), Filtering, Waveform Synthesis, Adaptive Filtering, High Speed Numeric Calculations	Test & Measurement Equipment, I/O Cards for PCs, Power Meters, Signal Analyzers and Generators
Communications	Modulation & Transmission, Demodulation & Reception, Speech Compression, Data Encryption, Echo Cancellation	Modems, Fax Machines, Broadcast Equipment, Mobile Phones, Digital Pagers, Global Positioning Systems, Digital Answering Machines
Medical Electronics	Filtering, Echo Cancellation, Fast Fourier Transform (FFT), Beam Forming	Respiration, Heart & Fetal Monitoring Equipment, Ultra Sound Equipment, Medical Imaging Equipment, Hearing Aides
Optical and Image Processing	2-Dimensional Filtering, Fast Fourier Transform (FFT), Pattern Recognition, Image Smoothing	Bar Code Scanners, Automatic Inspection Systems, Fingerprint Recognition, Digital Televisions, Sonar/Radar Systems, Robotic Vision

Table 12.1 A non-exhaustive list of applications of DSP chips.

FIR Filtering: Filtering a signal $x[n]$ through an FIR filter with impulse response $h[n]$ amounts to computing the convolution

$$y[n] = h[n] * x[n].$$

At each sample time n, the DSP processor has to compute

$$y[n] = \sum_{k=0}^{N-1} h[k]x[n-k],$$

where N is the length of the FIR. From an algorithmic point of view, this means that the processor has to fetch N data samples at each sample time, multiply them with corresponding filter coefficients stored in memory, and accumulate the sum of products to yield the final answer. Figure 12.1 illustrates these operations at two consecutive times n and $n+1$.

It is clear from this drawing that the basic DSP operation in this case is a multiplication followed by an addition, also know as a **Multiply and Accumulate** operation, or MAC. Such a MAC has to be implemented N times for each output sample.

This also shows that the FIR filtering operation is both **data intensive** (N data needed per output sample) and **repetitive** (the same MAC operation repeated N times with different operands).

An other point that appears from this graphic is that, although the filter coefficients remain the same from one time instant to the next, the input coefficients change. They do not all change: the last one is discarded, the newest input sample appears, and all other samples are shifted one location in memory. This is known as a ***First In-First Out* (FIFO) structure** or queue.

A final feature that is common to a lot of DSP applications is the need for **real time operation**: the output samples $y[n]$ must be computed by the processor between the moment $x[n]$ becomes available and the time $x[n+1]$ enters the FIFO, which leaves T_s seconds to carry out the whole processing.

FFT Computation: As explained in section 11.3, a classical FFT computation by a decimation-in-time algorithm uses a simple building block known as a *butterfly* (see Figure 11.7). This simple operation on 2 values involves again **multiplication and addition**, as suggested by the pseudo-code in section 11.3. An other point mentioned in section 11.3 is that the FFT algorithm yields DFT coefficient in **bit-reversed order**, so that once they are computed, the processor has to restore the original order.

Summary of requirements: We have seen with the two above examples that DSP algorithms involve:

1. Real time requirements with intensive data flow through the processor;
2. Efficient implementation of loops, and MAC operations;
3. Efficient addressing schemes to access FIFOs and/or bit-reversed ordered data.

In the next section we will describe the features common to many DSPs that address the above list of requirements.

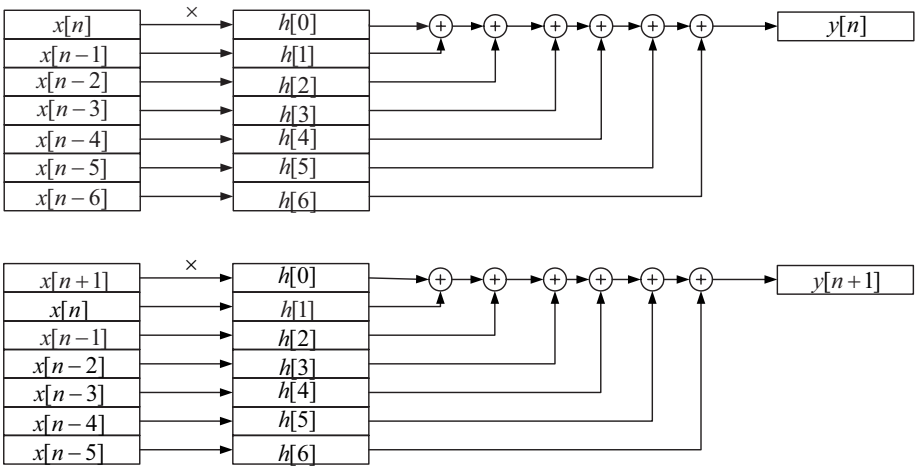


Fig. 12.1 Operations required to compute the output of a 7-tap FIR filter at times n and $n+1$.

12.3 Characterization of PDSPs

12.3.1 Fixed-Point vs. Floating-Point

Programmable DSPs can be characterized by the type of arithmetic they implement. Fixed-point processors are in general much cheaper than their floating-point counterparts. However, the dynamic range offered by the floating-point processors is much higher. An other advantage of floating-point processors is that they do not require the programmer to worry too much about overflow and scaling issues, which are of paramount importance in fixed-point arithmetics (see section 10.1.2). Some applications require a very low cost processor with low power dissipation (like cell phones for instance), so that fixed-point processors have the advantage on these markets.

12.3.2 Some Examples

We present here a few examples of existing DSPs, to which we will refer later to illustrate the different concepts.

Texas Instrument has been successful with its TMS320 series for nearly 20 years. The latest generation of these processors is the TMS320C6xxx family, which represent very high performance DSPs. This family can be grossly divided in the TMS320C62xx, which are fixed-point processors, and the TMS320C67xx, which use floating-point arithmetics.

An other example of a slightly older DSP is the DSP56300 from Motorola. The newer, state-of-the-art, DSPs from this company are based on the very high performance StarCore DSP Core, which yield unprecedented processing power. This is the result of a common initiative with Agere systems.

Analog Devices produces one of the most versatile low-cost DSPs, namely the ADSP21xx family. The next generation of Analog Devices DSPs are called SHARC DSPs, and contain the ADSP21xxx family of 32-bit floating-point processors. Finally, their latest generation is called TigerSHARC, and provides both fixed

and floating point operation, on variable word lengths. The TigerSHARC is a very high performance DSP targeting mainly very demanding applications, such as telecommunication networks equipment.

12.3.3 Structural features of DSPs

We will now review several features of Digital Signal Processors that are designed to meet the demanding constraints of real-time DSP.

Memory Access: Harvard architecture

Most DSPs are created according to the so-called *Harvard Architecture*. In classic microprocessors, there is only one memory bank, one address bus and one data bus. Both program instructions and data are fetched from the same memory through the same buses, as illustrated in Figure 12.2(a). On the other hand, in order to increase data throughput, DSP chips have in general two separate memory blocks, one for program instructions and one for data. This is the essence of the Harvard architecture, where each memory block also has a separate set of buses, as illustrated in Figure 12.2(b). Combined with pipelining (see below), this architecture enables simultaneous fetching of the next program instruction while fetching the data for the current instruction, thus increasing data throughput. This architecture enables two memory accesses during one clock cycle.

Many DSP chips have an architecture similar to the one in Figure 12.2(b), or an enhanced version of it. A common enhancement is the duplication of the data memory: many common DSP operations use two operands, like e.g. the MAC operation. It is thus normal to imagine fetching two data words from memory while fetching one instruction. This is realized in practise either with two separate data memory banks with their own buses (like in the Motorola DSP563xx family— see Figure 12.3), or by using two-way accessible RAM (like in Analog Device's SHARC Family), in which case only the buses have to be duplicated. Finally, other enhancements found in some DSPs (such as SHARC processors) is a program instruction cache.

Other critical components that enable an efficient flow of data on and off the DSP chip are Direct Memory Access (DMA) units, which enable direct storage of data in memory without processor intervention. (See e.g. the architecture of the Motorola DSP56300 in Figure 12.3).

Specialized hardware units and instruction set

In order to accommodate the constraints of DSP algorithms, DSP chips usually comprise specialized hardware units not found in many other processors.

Most if not all DSP chips contain one or several dedicated *hardware multipliers*, or even MAC units. On conventional processors without hardware multipliers, binary multiplication can take several clock cycles; in a DSP, it only takes one clock cycle. The multiplier is associated with an accumulator register (in a MAC unit), which normally is wider than a normal register. For instance, the Motorola 563xx family (see Figure 12.3) contains a 24-bit MAC unit, and a 56-bit accumulator. The extra bits provided by the accumulator are *guard bits* that prevent overflow from occurring during repetitive accumulate operations, such as the computation of the output of an FIR filter. In order to store the accumulated value into a normal register or in memory, the accumulator value must be shifted right by the number of guard bits used, so that resolution is lost, but no overflow occurs. Figure 12.4 illustrates the operation of a MAC unit, as well as the

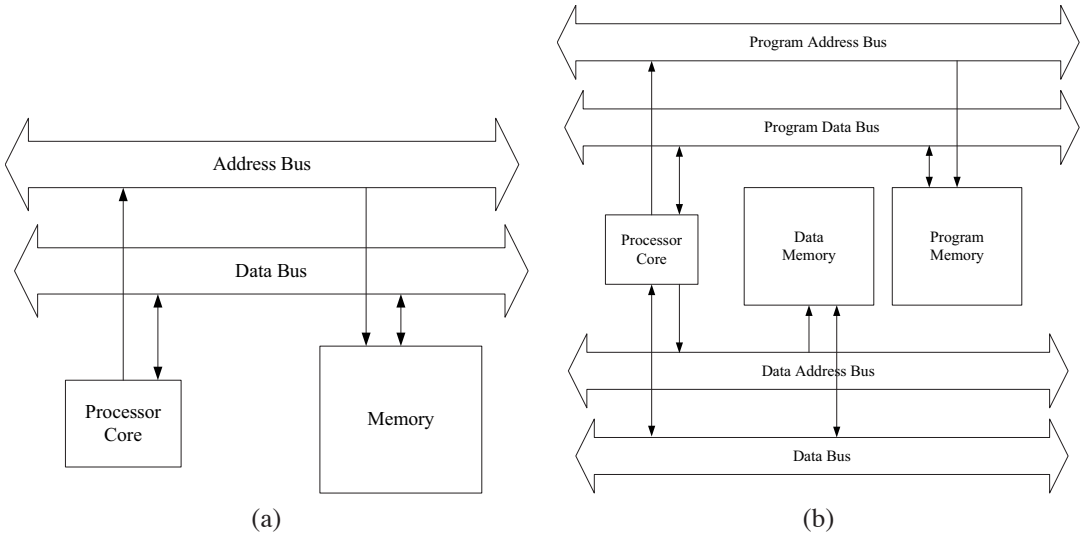


Fig. 12.2 Illustration of the difference between a typical microprocessor memory access (a) and a typical Harvard architecture (b).

need for guard bits. Note that G guard bits prevent overflow in the worst case of the addition of 2^G words. The issue of shifting the result stored in the accumulator is very important: the programmer has to keep track of the implied binary point, depending on the type of binary representation used (integer or fractional or mixed). The instruction set of DSP chips often contains a MAC instruction that not only does the MAC operation, but also increments pointers in memory.

We have also seen that DSP algorithms often contain repetitive instructions. Remember that to compute one output sample of an N-tap FIR filter, you need to repeat N MAC operations. This is why DSPs contain a provision for efficient looping (called *zero-overhead looping*), which may be a special assembly language instruction — so that the programmer does not have to care about checking and decrementing loop counters — , or even a dedicated hardware unit.

The third component that is necessary to DSP algorithms implementation is an efficient addressing mechanism. This is often taken care of by a specialized hardware *Address Generation Unit*. This unit works in the background, generating addresses in parallel with the main processor computations. Some common features of a DSP's address generation unit are:

- register-indirect addressing with post-increment, where a register points to an address in memory, from which the operand of the current instruction is fetched, and the pointer is incremented automatically at the end of the instruction. All of this happens in one instruction cycle. This is a low-level equivalent of a C language `*(var_name++)` instruction. It is obvious that such a function is very useful when many contiguous locations in memory have to be used, which is common when processing signal samples.
- circular addressing, where addresses are incremented modulo a given buffer size. This is an easy way to implement a FIFO or queue, which is the best way of creating a window sliding over a series of samples, as in the FIR filtering example given above. The only overhead for the programmer is to set up the circular addressing mode and the buffer size in some control register.

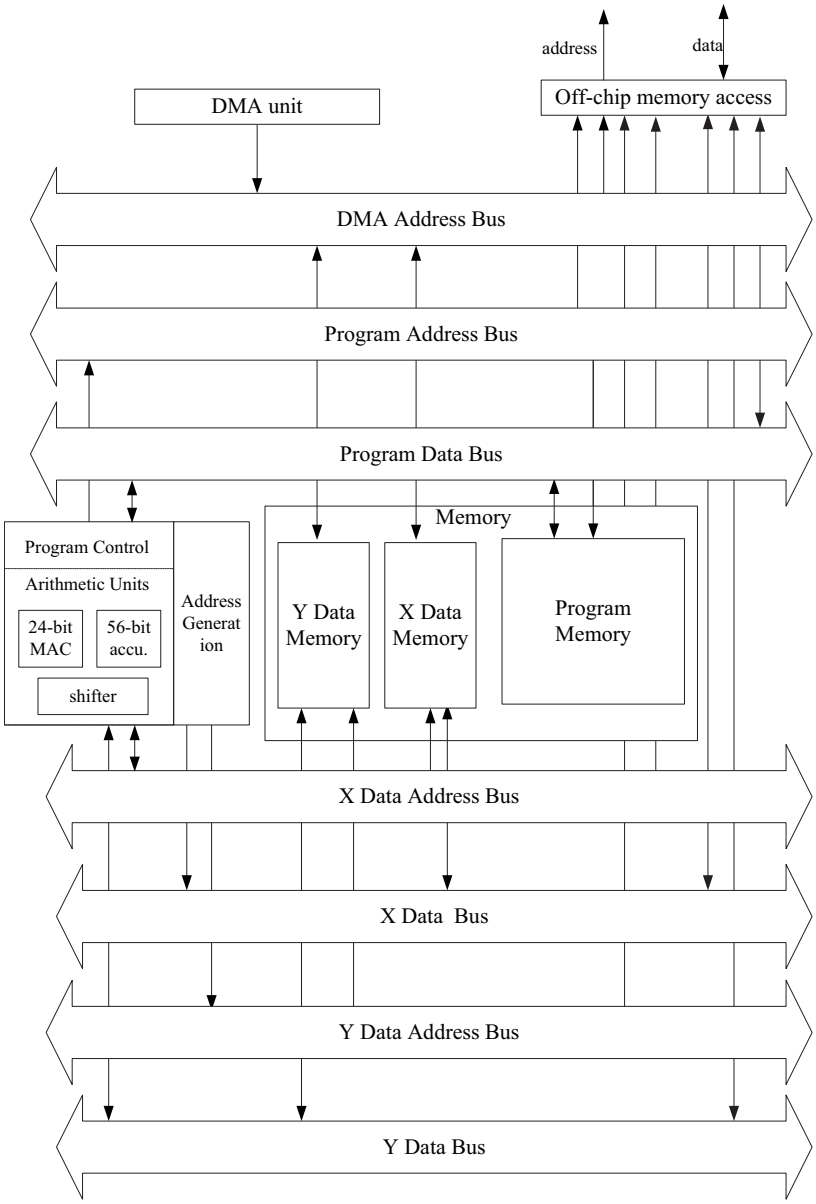


Fig. 12.3 Simplified architecture of the Motorola DSP563xx family. (*Source: Motorola[9]*). The DSP563xx is a family of 24-bit fixed-point processors. Notice the 3 separate memory banks with their associated sets of buses. Also shown is the DMA unit with its dedicated bus linked directly to memory banks and off-chip memory.

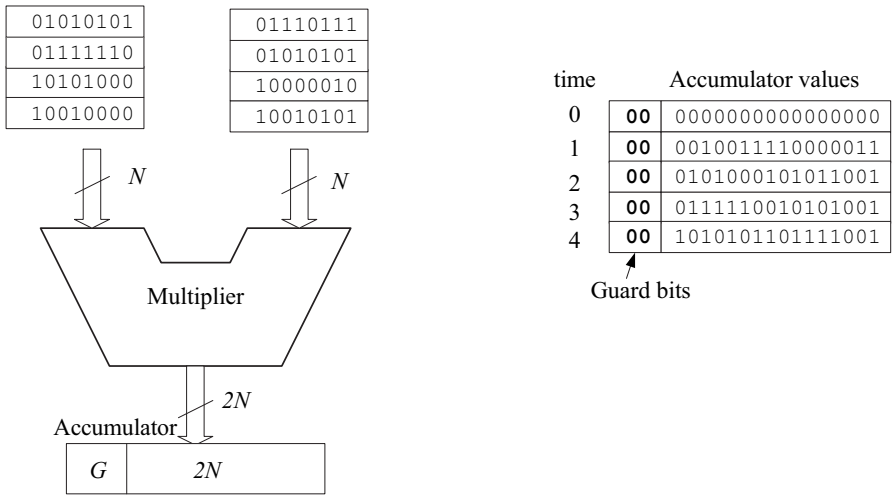


Fig. 12.4 Operation of a MAC unit with guard bits. Two N-bit operands are multiplied, leading to a $2N$-bit result, which is accumulated in a $(2N+G)$-bit accumulator register. The need for guard bits is illustrated with the given operands: the values of the accumulator at different times are shown on the right of the figure, showing that the fourth operation would generate an overflow without the guard bits: the positive result would appear as a negative value. It is the programmer's task in this case to keep track of the implied binary point when shifting back the accumulator value to an N-bit value.

- bit-reversed addressing, where address offsets are bit-reversed. This has been shown to be useful for the implementation of FFT algorithms.

Parallelism and Pipelining

At one moment, it becomes difficult to increase clock speed enough to satisfy the ever more demanding DSP algorithms. So, in order to meet real-time constraints in more and more sophisticated DSP applications, other means of increasing the number of operations carried out by the processor have been implemented: parallelism and pipelining.

Pipelining: Pipelining is a principle that is in application in any modern processor, and is not limited to DSPs. Its goal is to make better use of the processor resources by avoiding leaving some components idle while others are working. Basically, the execution of an instruction requires three main steps, each carried out by a different part of the processor: a fetching step, during which the instruction is fetched from memory; a decoding step, during which the instruction is decoded and an execution step during which it is actually executed. Often, each of these three steps can also be decomposed in sub-step: for example, the execution may require fetching operands, computing a result and storing the result. The idea behind pipelining is just to begin fetching the next instruction as soon as the current one enters the decode stage. When the current instruction reaches the execution stage, the next one is a the decode stage, and the instruction after that is in the fetching stage. In this way, the processor resources are used more efficiently and the actual

instruction throughput is increased. Figure 12.5 illustrates the pipelining principle. In practise, care must be taken because instruction do not all require the same time (mostly during the execution phase), so that NOPs should sometimes be inserted. Also, branching instructions might ruin the gain of pipelining for a few cycles, since some instructions already in the pipeline might have to be dropped. These issues are far beyond the scope of this text and the interested reader is referred to the references for details.

	Fetch unit	Decode unit	Execute unit
t=0	Instruction1	idle	idle
t=1	idle	Instruction1	idle
t=2	idle	idle	Instruction1
t=3	Instruction 2	idle	idle
t=4	idle	Instruction 2	idle
t=5	idle	idle	Instruction 2

	Fetch unit	Decode unit	Execute unit
t=0	Instruction1	idle	idle
t=1	Instruction 2	Instruction1	idle
t=2	Instruction 3	Instruction 2	Instruction1
t=3	Instruction 4	Instruction 3	Instruction 2
t=4	Instruction 5	Instruction 4	Instruction 3
t=5	Instruction 6	Instruction 5	Instruction 4

Fig. 12.5 Schematic illustration of the difference in throughput between a pipelined (right) and non pipelined (left) architecture. Columns represent processor functional units, lines represent different time instants. Idle units are shaded.

Data level parallelism: Data level parallelism is achieved in many DSPs by duplication of hardware units. It is not uncommon in this case to have two or more ALUs (Arithmetic and Logic Unit), two or more MACs and Accumulators, each with their own data path, so that they can be used simultaneously. *Single Instruction-Multiple Data* (SIMD) processors can then apply the same operation to several data operands at the same time (two simultaneous MACs for instance). An other way of implementing this, is to have the processor accommodate multiple date lengths, so that for instance a 16-bit multiplier can accomplish 2 simultaneous 8-bit multiplications. A good example of this idea is the TigerSHARC processor from Analog Devices, which is illustrated in Figure 12.6. Figure 12.6(a) illustrates the general architecture of the processor core, with multiple buses and memory banks, multiple address generation units, and two parallel 32-bit computational units. Figure 12.6(b) shows how theses two computational units can be used in parallel, together with sub-word parallelism to enhance the throughput on sub-word operations: in the case illustrated, 8 16-bit MAC operations can be carried out in just one cycle.

SIMD is efficient only for algorithms that lend themselves to this kind of parallel execution, and for instance not where data are treated in series or where a low-delay feedback loop exists. But it is very efficient for vector-intensive operations like multimedia processing.

Instruction level parallelism: Instruction level parallelism is achieved by having different units with different roles running concurrently. Two main families of parallel DSPs exist, namely those with VLIW (Very Long Instruction Word) architecture and those with super-scalar architectures. Super-scalar architectures are used heavily in general purpose processors (e.g. Intel's Pentium), and contain a special scheduling unit which determines at run-time which instructions could run in parallel based e.g. on data dependencies. As is, this kind of behaviour is not desirable in DSPs, because the run-time determination of parallelism makes the predictability of execution time difficult — though it is a factor of paramount importance in real time operation. This is why there are few super-scalar DSPs available.

VLIW architectures [6] are characterized by long instruction words, which are in fact concatenations of

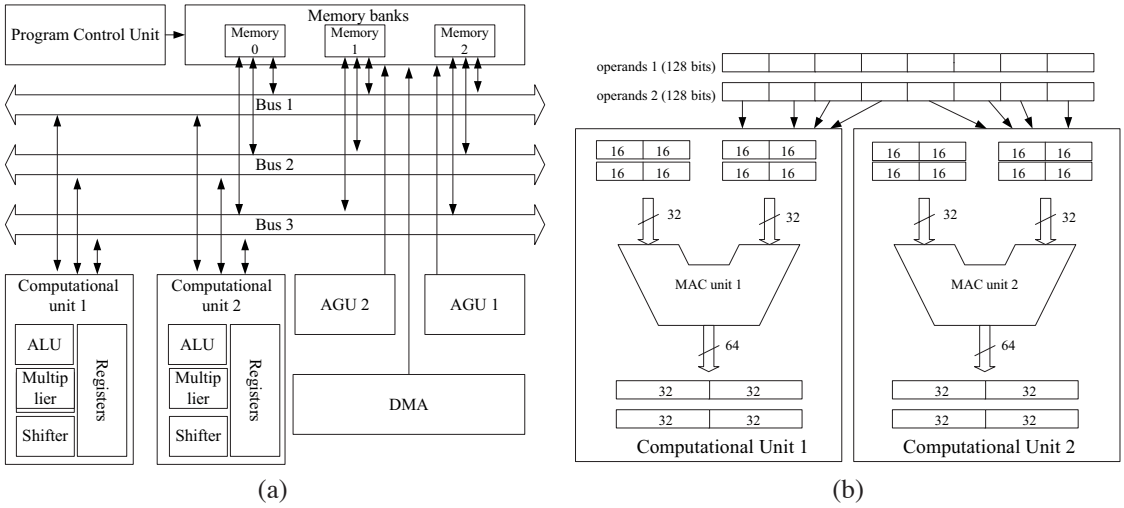


Fig. 12.6 Simplified view of an Analog Devices *TigerSHARC* architecture. (a) The figure shows the 3 128-bit buses and 3 memory banks as well as 2 computational units and 2 address generation units. The programmer can assign program and/or data to each bus/memory bank. (b) Illustration of the data-level parallelism by SIMD operation: each computational unit can execute two 32-bit $\times$ 32-bit fixed-point MACs per cycle, but can also work in sub-word parallelism, executing up to four 16-bit $\times$ 16-bit MACs, for a total of eight simultaneous 16-bit $\times$ 16-bit MACs per cycle. (*Source: ADI [1]*)

several shorter instructions. Each short instruction will ideally execute in parallel with the others on a different hardware unit. This principle is illustrated in Figure 12.7. This heavy parallelism normally provides an enormous increase in instruction throughput,but in practise, some limitations will prevent the hardware units from working at full rate in parallel all the time. For instance, data dependencies between the short instructions might prevent their parallel execution; also, if some of the short instructions in a VLIW instruction use the same resources (e.g. registers), they will not be able to execute in parallel.

A good illustration of these limitations can be given by looking at a practical example of VLIW processor: the TMS320C6xxx family from Texas Instruments. The simplified architecture of this processor is shown in Figure 12.8, where the address buses have been omitted for clarity. In the TMS320C6xxx, the instruction words are 128-bit wide, each containing eight 32-bit instructions. Each of these 32-bit instructions is to be executed by one of the 8 different computational units. These 8 units are constituted by 4 duplicate units, grouped in so-called data paths A and B (see Figure 12.8): ALU, ALU and shifter, address generation unit and multiplier. Each data path contains a set of registers, and both data paths share the same data memory. It is clear in this case that in order for all the 8 instructions of a VLIW instruction to be executed in parallel, one instruction must be executed on each computational unit, i.e. for instance, the VLIW instruction must contain two multiplications. The actual 32-bit instructions from the same 128-bit VLIW that are executed in parallel are in fact chosen at assembly time, by a special directive in the TMS320C6xxx assembly language. Of course, if the program is written in some high-level language such as C, the compiler takes care of this instructions organization and grouping.

Another example of processor using VLIW is the TigerSHARC from Analog Devices, described earlier (see Figure 12.6). This DSP combines SIMD and VLIW to attain a very high level of parallelism. VLIW means

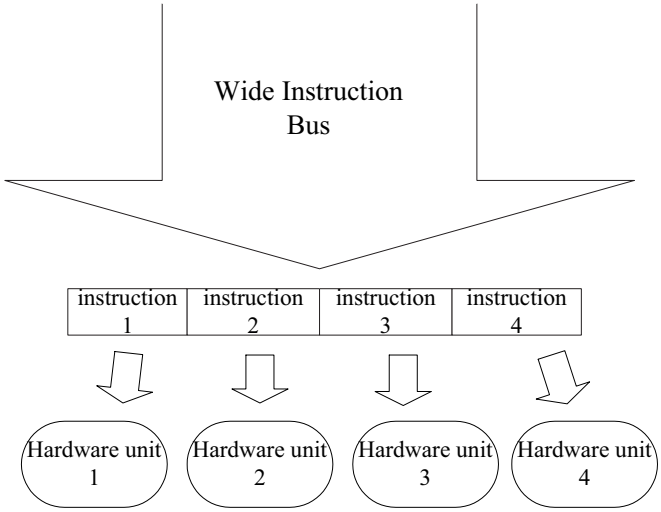


Fig. 12.7 Principle of VLIW architecture: long instructions are split in short instructions executed in parallel by different hardware units.

that each of the two computational units in Figure 12.6(b) can be issued a different instruction.

Other Implementations

We have reviewed the characteristics of DSPs, and shown why they are suited to the implementation of DSP algorithms. However, it is obvious that some other choices are possible when a DSP algorithm has to be implemented in an end-product. These choices are listed below, and compared with a PDSP-based solution.

General Purpose Processor: Nowadays, general purpose processors, such as Intel's Pentium, show extremely high clock rates. They also have very efficient caching, pipelining and make extensive use of super-scalar architectures. Most of them also have a SIMD extension to their instruction set (e.g. MMX for Intel). However, they suffer some drawbacks for DSP implementations: high power consumption and need for proper cooling; higher cost (at least than most fixed-point DSPs); and lack of execution-time predictability because of multiple caching and dynamic super-scalar architectures, which make them less suited to hard real-time requirements.

ASIC: An Application Specific Integrated Circuit will execute the required algorithm much faster than a programmable DSP processor, but will of course lack any reconfigurability. So upgrade and maintenance of products in the field are impossible. The design costs are also prohibitive except for very high volumes of units deployed.

FPGA: Field-Programmable Gate Arrays are hardware reconfigurable circuits that represent some middle way between a programmable processor and and ASIC. They are faster than PDSPs, but in general have

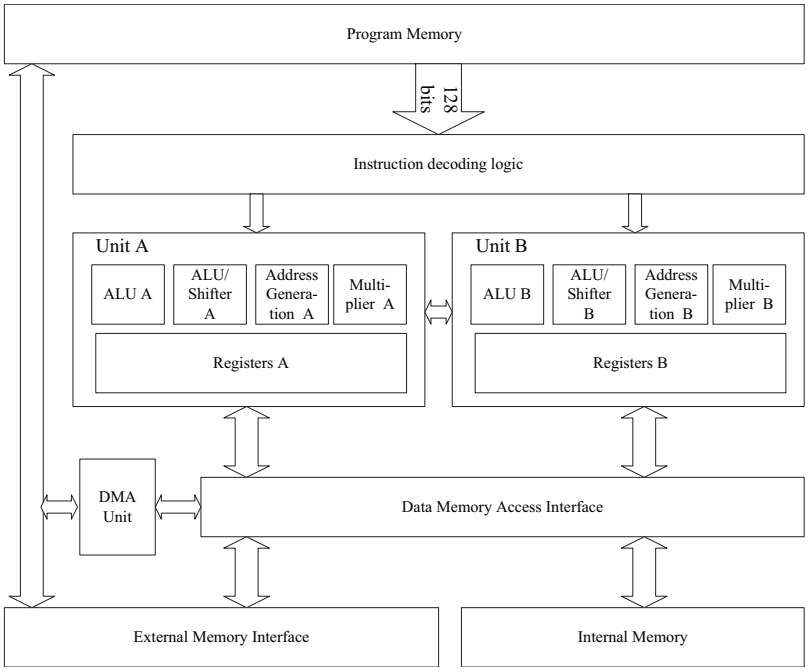


Fig. 12.8 Simplified architecture of VLIW processor: the Texas Instruments TMS320C6xxx VLIW Processor. Each VLIW instruction is made of eight 32-bit instructions, for up to eight parallel instructions issued to the eight processing units divided in operational Units A and B. (*Source:TI [13]*)

higher power consumption. They are more expensive than PDSPs. The design work tends to be also longer than on a PDSP.

12.4 Benchmarking PDSPs

Once the decision has been made to implement an algorithm on a PDSP, the time comes to choose a model amongst the many available brands and families. Many factors will enter into account, and we try to review some of these here.

Fixed or Floating Point: As stated earlier, floating point DSPs have higher dynamic range and do not require the programmer to worry as much about overflow and round-off errors. These comes at the expense of a higher cost and a higher power consumption. So, most of the time, the application will decide: applications where low-cost and/or low power consumption are an issue (large volume, mobile,...) will call for a fixed-point DSP. Once the type of arithmetic is chosen, one still has to choose the word width (in bits). This is crucial for fixed-point application, since the number of bits available directly defines the dynamic range attainable.

Speed: Constructors publish the speed of their devices as MIPS, MOPS or MFLOPS. MIPS (Millions of instructions per second) gives simply a hypothetical peak instruction rate through the DSP. It is simply determined by

$$\frac{N_I}{T_I},$$

where T_I is the processor's instruction cycle time in μs, and N_I is the maximum number of instructions executed per cycle ($N_I > 1$ for processors with parallelism). Due to the big differences between the instruction sets, architectures and operational units of different DSPs, it is difficult to conclude from MIPS measurements. Instructions on one DSP can be much more powerful than on another one, so that a lower instruction count will do the same job.

MOPS or MFLOPS (Millions of Operations per Second or Millions of Floating-Point Operations per Second) have no consensual definition: no two device manufacturers do agree on what an "operation" might be.

The only real way of comparing different DSPs in terms of speed is to run highly optimized benchmark programs on the devices to be compared, including typical DSP operations: FIR filtering, FFT,... Figure 12.9 shows an example of such a benchmark, the BDTIMark2000, produced by Berkeley Design Technologies.

Design issues: Time-to-market is an important factor that will perhaps be the main driving force towards the choice of any DSP. In order to ease the work of design engineers, the DSP manufacturer must provide development tools (compiler, assembler). Since most new DSPs have their own instruction set and architecture, if prior knowledge of one DSP is available, it may orient the choice of a new DSP to one that is code-compatible. Also, in order to reuse legacy code, designers may opt for a DSP that is code-compatible with one they already use.

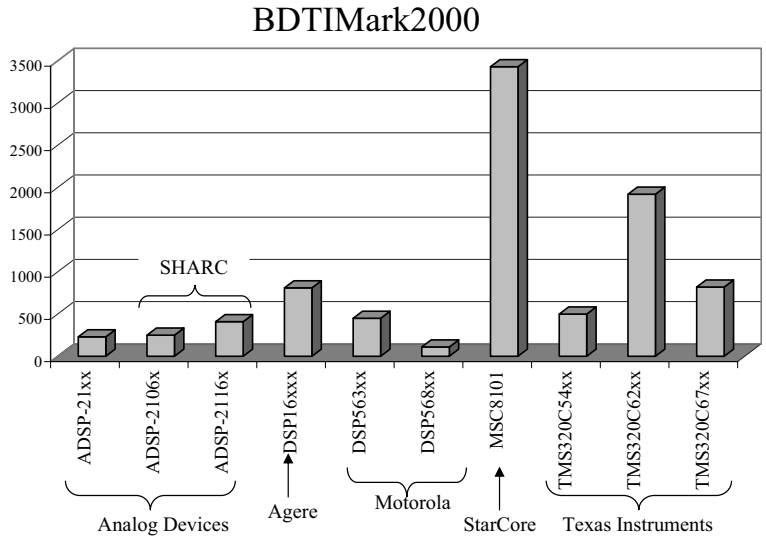


Fig. 12.9 Speed benchmark for several available DSPs. The benchmark is the BDTIMark2000, by *Berkeley Design Technologies* (higher is faster). BDTIMark2000 is an aggregate speed benchmark computed from the speed of the processor in accomplishing common DSP tasks such as FFT, FIR filtering, IIR filtering,...(*source: BDTI*)

New high-level development tools should ease the development of DSP algorithms and their implementation. Some manufacturers have a collaboration e.g. with Mathworks to develop a MATLAB/SIMULINK toolbox enabling direct design and/or simulation of software for the DSP chip in MATLAB and SIMULINK.

12.5 Current Trends

Multiple DSPs: Numerous new applications cannot be tackled by one DSP processor, even the most heavily parallel ones. As a consequence, new DSP chips are developed that are easily grouped in parallel or in clusters to yield heavily parallel computing engines [2]. Of course, new challenges appear in the parallelisation of algorithms, shared resources management, and resolution of data dependencies.

FPGAs coming on fast: FPGA manufacturers are aggressively entering the DSP market, due mainly to the very high densities attained by their latest products and to the availability of high-level development tools (directly from a block-level vision of the algorithm to the FPGA). The claimed advantages of FPGAs over PDSPs are speed and a completely flexible structure that allows for complete customization and highly parallel implementations [12]. For example, one manufacturer now promises the capability of implementing 192 eighteen-bit multiplications in parallel.

Chapter 13

Multirate systems

13.1 Introduction

Multi-rate system: A multi-rate DSP system is characterized by the use of multiple sampling rates in its realization, that is: signal samples at various points in the systems may not correspond to the same physical sampling frequency.

Some applications of multi-rate processing include:

- Sampling rate conversion between different digital audio standards.
- Digital anti-aliasing filtering (used in commercial CD players).
- Subband coding of speech and video signals
- Subband adaptive filtering.

In some applications, the need for multi-rate processing comes naturally from the problem definition (e.g. sampling rate conversion). In other applications, multi-rate processing is used to achieve improved performance of a DSP function (e.g. lower binary rate in speech compression).

Sampling rate modification: The problem of sampling rate modification is central to multi-rate signal processing. Suppose we are given the samples of a continuous-time signal $x_a(t)$, that is

$$x[n] = x_a(nT_s), \quad n \in \mathbb{Z} \tag{13.1}$$

with sampling period T_s and corresponding sampling frequency $F_s = 1/T_s$. Then, how can we efficiently generate a new set of samples

$$x'[n] = x_a(nT'_s), \quad n \in \mathbb{Z} \tag{13.2}$$

corresponding to a different sampling period, i.e. $T'_s \neq T_s$?

Assuming that the original sampling rate $F_s > 2F_{max}$, where F_{max} represent the maximum frequency contained in analog signal $x_a(t)$, one possible approach is to reconstruct $x_a(t)$ from the set of samples $x[n]$ (see Sampling Theorem, Section 7.1.2) and then to resample $x_a(t)$ at the desired rate $F'_s = 1/T'_s$. This approach is expensive as it requires the use of D/A and A/D devices.

In this Chapter, we seek a cheaper, purely discrete-time solution to the above problem, i.e. performing the resampling by direct manipulation of the discrete-time signal samples $x[n]$, *without* going back in the analog domain.

The following special terminology will be used here:

- $T_s' > T_s$ (i.e. $F_s' < F_s$) $\Rightarrow$ downsampling
- $T_s' < T_s$ (i.e. $F_s' > F_s$) $\Rightarrow$ upsampling or interpolation

13.2 Downsampling by an integer factor

Let $T_s' = MT_s$ where M is an integer ≥ 1. In this case, we have:

$$\begin{aligned} x'[n] &= x_a(nT_s') \\ &= x_a(nMT_s) = x[nM] \end{aligned} \tag{13.3}$$

The above operation is so important in multi-rate signal processing that it is given the special name of *decimation*. Accordingly, a device that implements 13.3 is referred to as a *decimator*. The basic properties of the decimator are studied in greater details below.

M-fold decimation:

Decimation by an integer factor $M \geq 1$, or simply M-fold decimation, is defined by the following operation:

$$x_D[n] = \mathcal{D}_M\{x[n]\} \triangleq x[nM], \quad n \in \mathbb{Z} \tag{13.4}$$

That is, the output signal $x_D[n]$ is obtained by retaining only one out of every M input samples $x[n]$.

The block diagram form of the M-fold decimator is illustrated in Figure 13.1. The decimator is sometimes

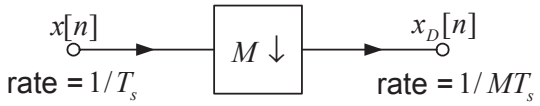


Fig. 13.1 M-fold decimator

called a sampling rate compressor or subsampler. It should be clear from (13.4) that the decimator is a linear system, albeit a time-varying one, as shown by the following example.

Example 13.1:

► Consider the signal

$$\begin{aligned} x[n] &= \cos(\pi n/2)\, u[n] \\ &= \{\ldots, 0, 0, \underline{1}, 0, -1, 0, 1, 0, -1, \ldots\} \end{aligned}$$

If $x[n]$ is applied at the input of a 2-fold decimator (i.e. $M = 2$), the resulting signal will be

$$\begin{aligned} x_D[n] &= x[2n] \\ &= \cos(\pi n)\, u[2n] \\ &= \{\ldots, 0, 0, \underline{1}, -1, 1, -1, \ldots\} \end{aligned}$$

Only those samples of $x[n]$ corresponding to even values of n are retained in $x_D[n]$. This is illustrated in Figure 13.2.

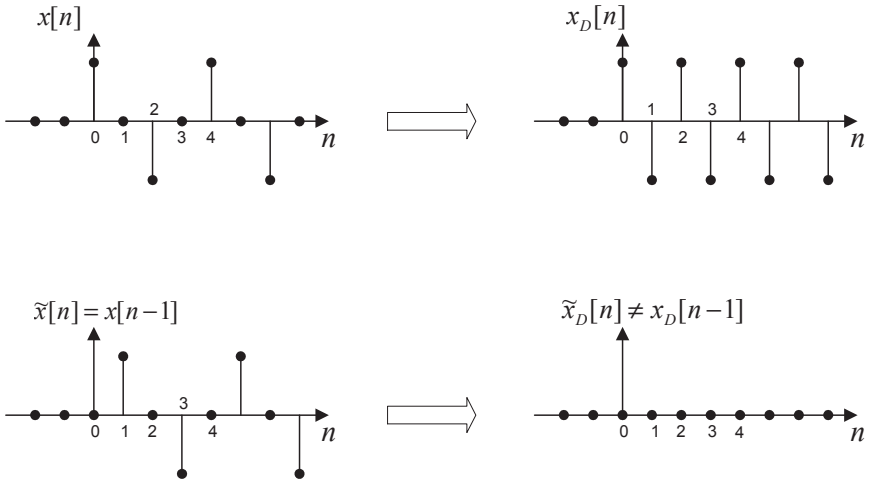


Fig. 13.2 Example of 2-fold decimation

Now consider the following shifted version of $x[n]$:

$$\begin{aligned} \tilde{x}[n] &= x[n-1] \\ &= \{\ldots, 0, \underline{0}, 1, 0, -1, 0, 1, 0, -1, \ldots\} \end{aligned}$$

If $\tilde{x}[n]$ is applied at the input of a 2-fold decimator, the output signal will be

$$\begin{aligned} \tilde{x}_D[n] &= \tilde{x}[2n] \\ &= \{\ldots, 0, \underline{0}, 0, 0, 0, 0, \ldots\} \end{aligned}$$

Note that $\tilde{x}_D[n] \neq x_D[n-1]$. This shows that the 2-fold decimator is time-varying. ◄

Property: Let $x_D[n] = x[nM]$ where M is a positive integer. The z-transforms of the sequences $x_D[n]$ and $x[n]$, denoted $X_D(z)$ and $X(z)$, respectively, are related as follows:

$$X_D(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(z^{1/M} W_M^k) \tag{13.5}$$

where $W_M = e^{-j2\pi/M}$.

Proof: Introduce the sequence

$$p[l] \triangleq \frac{1}{M}\sum_{k=0}^{M-1} e^{j2\pi kl/M} = \begin{cases} 1, & l = 0, \pm M, \pm 2M, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{13.6}$$

Making use of $p[n]$, we have:

$$\begin{aligned} X_D(z) &\triangleq \sum_{n=-\infty}^{\infty} x_D[n]z^{-n} \\ &= \sum_{n=-\infty}^{\infty} x[nM]z^{-(nM)/M} \\ &= \sum_{l=-\infty}^{\infty} p[l]x[l]z^{-l/M} \\ &= \sum_{l=-\infty}^{\infty} \{\frac{1}{M}\sum_{k=0}^{M-1} e^{j2\pi kl/M}\}x[l]z^{-l/M} \\ &= \frac{1}{M}\sum_{k=0}^{M-1}\sum_{l=-\infty}^{\infty} x[l](z^{1/M}e^{-j2\pi k/M})^{-l} \\ &= \frac{1}{M}\sum_{k=0}^{M-1} X(z^{1/M}W_M^k)\square \end{aligned} \tag{13.7}$$

Remarks:

- In the special case $M = 2$, we have $W_2 = e^{-j\pi} = -1$, so that

$$X_D(z) = \frac{1}{2}(X(z^{1/2}) + X(-z^{1/2})) \tag{13.8}$$

- Setting $z = e^{j\omega}$ in (13.5), an equivalent expression is obtained that relates the DTFTs of $x_d[n]$ and $x[n]$, namely:

$$X_D(\omega) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(\frac{\omega - 2\pi k}{M}\right) \tag{13.9}$$

- According to (13.9), the spectrum $X_D(\omega)$ is made up of properly shifted, expanded images of the original spectrum $X(\omega)$.

Example 13.2:

► In this example, we illustrate the effects of an M-fold decimation in the frequency domain. Consider a discrete-time signal $x[n]$ with DTFT $X(\omega)$ as shown in Figure 13.3(top), where $\omega_{max} = \pi/2$ denotes the maximum frequency contained in $x[n]$. That is, $X(\omega) = 0$ if $\omega_{max} \leq |\omega| \leq \pi$. Let $x_D[n]$ denote the result of a M-fold decimation on $x[n]$.

First consider the case $M = 2$, where (13.9) simplifies to

$$X_D(\omega) = \frac{1}{2}[X(\frac{\omega}{2}) + X(\frac{\omega - 2\pi}{2})] \tag{13.10}$$

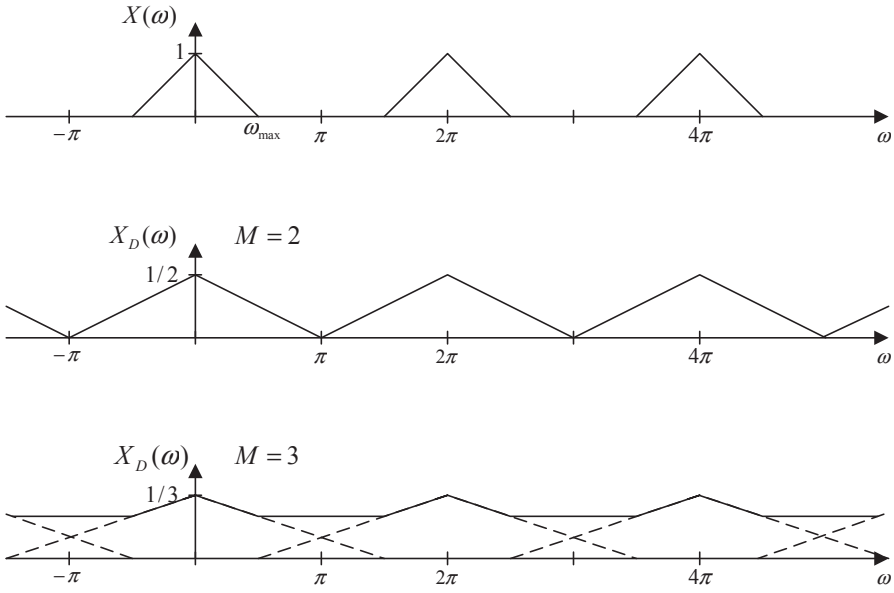


Fig. 13.3

In this equation, $X(\omega/2)$ is a dilated version of $X(\omega)$ by a factor of 2 along the ω-axis. Note that $X(\omega)$ is no longer of period 2π but instead its period is 4π. Besides scaling factor $1/2$, $X_D(\omega)$ is obtained as the sum of $X(\omega/2)$ and its shifted version by 2π, i.e. $X(\frac{\omega-2\pi}{2})$. The introduction of this latter component restores the periodicity to 2π. The resulting DTFT $X_D(\omega)$ is illustrated in Figure 13.3 (middle).

Note that in the above case $M = 2$, the two sets of spectral images $X(\omega/2)$ and $X(\frac{\omega-2\pi}{2})$ do not overlap because $\omega_{max} < pi/2$. Thus, there is no spectral aliasing between the images and, besides the amplitude scaling and the frequency dilation, the original spectral shape $X(\omega)$ is not distorted. In other words, no information about the original $x[n]$ is lost through the decimation process. We will see later how $x[n]$ can be recovered from $x_D[n]$ in this case.

Next consider the case $M = 3$, where (13.9) simplifies to

$$X_D(\omega) = \frac{1}{3}[X(\frac{\omega}{3}) + X(\frac{\omega-2\pi}{3}) + X(\frac{\omega-4\pi}{3})] \tag{13.11}$$

Here, $X_D(\omega)$ is obtained by superposing dilated and shifted versions of $X(\omega)$, as shown in Figure 13.3 (bottom). Here the dilation factor is 3 and since $\omega_{max} > \pi/3$, the various images do overlap. That is, there is spectral aliasing and a loss of information in the decimation process. ◀

Discussion: It can be inferred from the above example that if $X(\omega)$ is properly band-limited, i.e.

$$X(\omega) = 0 \quad \text{for} \quad \frac{\pi}{M} \leq M \leq \pi \tag{13.12}$$

then no loss of information occurs during the decimation process. In this special case, the relationship (13.9) between $X(\omega)$ and $X_D(\omega)$ simplifies to the following if one only considers the fundamental period $[-\pi,\pi]$:

$$X_D(\omega) = \frac{1}{M}X(\frac{\omega}{M}), \quad 0 \leq |\omega| \leq \pi \tag{13.13}$$

Downsampling system: In practice, a low-pass digital filter with cut-off at π/M would be included along the processing chain prior to M-fold decimation. The resulting system, referred to as a *downsampling system* is illustrated in Figure 13.4.

Fig. 13.4

Ideally, $H_D(\omega)$ is a the low-pass anti-aliasing filter $H_D(\omega)$ should have the following desired specifications:

$$H_D(\omega) = \begin{cases} 1 & \text{if } |\omega| < \pi/M \\ 0 & \text{if } \pi/M \leq |\omega| \leq \pi. \end{cases} \tag{13.14}$$

13.3 Upsampling by an integer factor

Here we consider the sampling rate conversion problem (13.1)– (13.2) with $T_s' = T_s/L$ where L is an integer ≥ 1.

Suppose that the underlying analog signal $x_a(t)$ is band-limited to Ω_N, i.e. $|X(\Omega)| = 0$ for $|\Omega| \geq \Omega_N$, and that the sequence $x[n]$ was obtained by uniform sampling of $x_a(t)$ at a rate F_s such that $\Omega_s = 2\pi F_s \geq 2\Omega_N$. In this case, according to the sampling theorem (Section 7.1.2), $x_a(t)$ can be expressed in terms of its samples $x[n]$ as

$$x_a(t) = \sum_{k=-\infty}^{\infty} x[k]\,\mathrm{sinc}(\frac{t}{T_s} - k) \tag{13.15}$$

Invoking (13.2), the new samples $x'[n]$ can therefore be expressed in terms of the original samples $x[n]$ as follows: We have

$$\begin{aligned} x'[n] &= x_a(nT_s') \\ &= \sum_{k=-\infty}^{\infty} x[k]\,\mathrm{sinc}(\frac{nT_s'}{T_s} - k) \\ &= \sum_{k=-\infty}^{\infty} x[k]\,\mathrm{sinc}(\frac{n-kL}{L}) \end{aligned} \tag{13.16}$$

The above equation can be given a simple interpretation if we introduce the concept of sampling rate expansion.

13.3.1 L-fold expansion

Sampling rate expansion by an integer factor $L \geq 1$, or simply L-fold expansion, is defined as follows:

$$x_E[n] = \mathcal{U}_L\{x[n]\} \triangleq \begin{cases} x[k] & \text{if } n = kL, k \in \mathbb{Z} \\ 0 & \text{otherwise} \end{cases} \tag{13.17}$$

That is, $L-1$ zeros are inserted between each consecutive samples of $x[n]$. Real-time implementation of an L-fold expansion requires that the output sample rate be L times larger than the input sample rate.

The block diagram form of an L-fold expander system is illustrated in Figure 13.5. The expander is also sometimes called upsampler in the DSP literature.

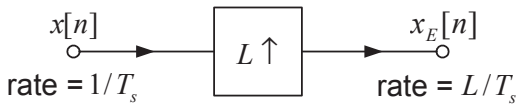


Fig. 13.5 L-fold expander.

Sampling rate expansion is a linear time-varying operation. This is illustrated in the following example.

Example 13.3:

► Consider the signal

$$\begin{aligned} x[n] &= a^n u[n] \\ &= \{\ldots,0,0,\underline{1},a,a^2,a^3,\ldots\} \end{aligned}$$

L-fold expansion applied to $x[n]$, with $L=3$, yields

$$x_E[n] = \{\ldots,0,0,0,\underline{1},0,0,a,0,0,a^2,0,0,a^3,0,0,\ldots\}$$

That is, two zero samples are inserted between every consecutive samples of $x[n]$.
Next, consider the following shifted version of $x[n]$:

$$\begin{aligned} \tilde{x}[n] &= x[n-1] \\ &= \{\ldots,0,0,\underline{0},1,a,a^2,a^3,\ldots\} \end{aligned}$$

The result of a 3-fold expansion on $\tilde{x}[n]$ is

$$\tilde{x}_E[n] = \{\ldots,0,0,0,\underline{0},0,0,1,0,0,a,0,0,a^2,0,0,a^3,0,0,\ldots\}$$

Note that $\tilde{x}_E[n] \neq x_E[n-1]$, showing that the expander is a time-varying system. Actually, in the above example, we have $\tilde{x}_E[n] = x_E[n-3]$. ◄

Property: Let $x_E[n]$ be defined as in (13.17), with L a positive integer. The z-transforms of the sequences $x_E[n]$ and $x[n]$, respectively denoted $X_E(z)$ and $X(z)$, are related as follows:

$$X_E(z) = X(z^L) \tag{13.18}$$

Proof: Since the sequence $x_E[n] = 0$ unless $n = kL$, $k \in \mathbb{Z}$, we immediately obtain

$$\begin{aligned} X_E(z) &\triangleq \sum_{n=-\infty}^{\infty} x_E[n]z^{-n} = \sum_{k=-\infty}^{\infty} x_E[kL]z^{-kL} \\ &= \sum_{k=-\infty}^{\infty} x[k]z^{-kL} = X(z^L) \quad \square \end{aligned} \tag{13.19}$$

Remarks: Setting $z = e^{j\omega}$ in (13.18), we obtain an equivalent relation in terms of the DTFTs of $x_E[n]$ and $x[n]$, namely:

$$X_E(\omega) = X(\omega L) \tag{13.20}$$

According to (13.20), the frequency axis in $X(\omega)$ is compressed by a factor L to yield $X_E(\omega)$. Because of the periodicity of $X(\omega)$ this operation introduces $L-1$ images of the (compressed) original spectrum within the range $[-\pi, \pi]$.

Example 13.4:

► The effects of an L-fold expansion in the frequency domain are illustrated in Figure 13.6. The top portion

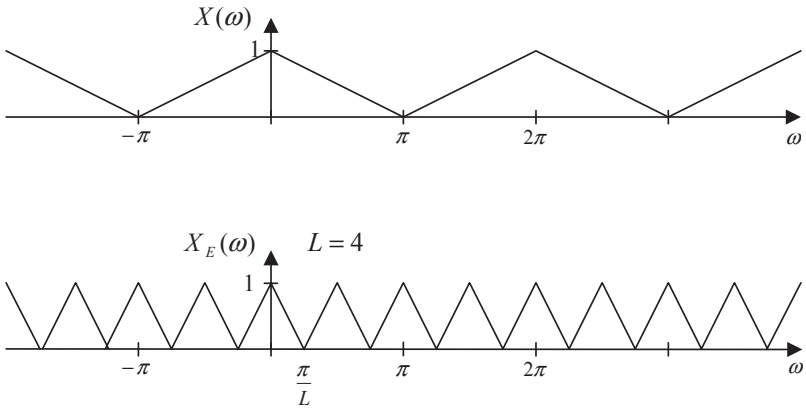


Fig. 13.6 Effects of L-fold expansion in the frequency domain.

shows the DTFT $X(\omega)$ of the original signal $x[n]$. The bottom portion shows the DTFT $X_E(\omega)$ of $x_E[n]$, the L-fold expansion of $x[n]$, in the special case $L = 4$. As a result of the time domain expansion, the original spectrum $X(\omega)$ has been compressed by a factor of $L = 4$ along the ω-axis. ◄

13.3.2 Upsampling (interpolation) system

We are now in a position to provide a simple interpretation for the upsampling formula (13.16).

Suppose $x_a(t)$ is BL with maximum frequency Ω_N. Consider uniform sampling at rate $\Omega_s \geq 2\Omega_N$ followed by L-fold expansion, versus direct uniform sampling at rate $\Omega'_s = L\Omega_s$. The effects of these operations in the frequency domain are shown in Figure 13.7 in the special case $L = 2$ and $\Omega_s = 2\Omega_N$.

Clearly, $X'(\omega)$ can be recovered from $X_E(\omega)$ via low-pass filtering:

$$X'(\omega) = H_I(\omega)X_E(\omega) \tag{13.21}$$

where $H_I(\omega)$ is an ideal low-pass filter with specifications

$$H_I(\omega) = \begin{cases} L, & \text{if } |\omega| < \pi/L \\ 0, & \text{if } \pi/L \leq |\omega| \leq \pi \end{cases} \tag{13.22}$$

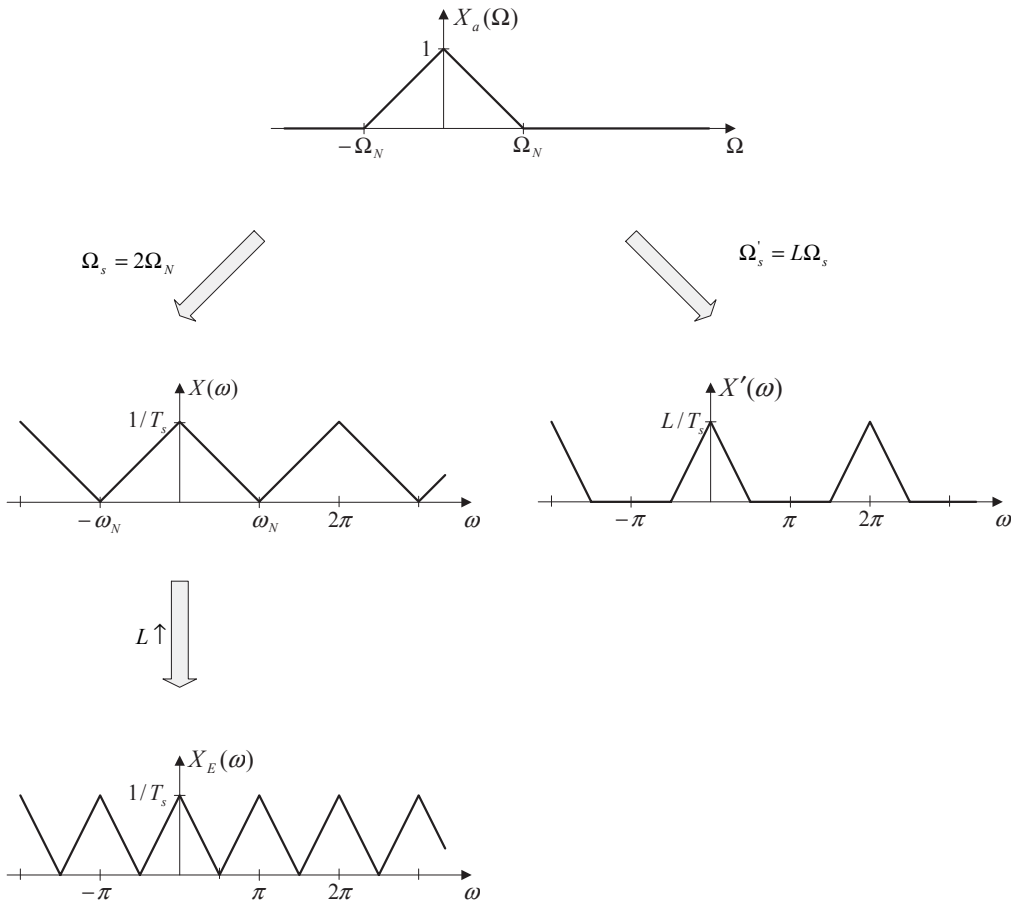


Fig. 13.7 Sampling at rate $\Omega_s = 2\Omega_N$ followed by L-fold expansion (left) versus direct sampling at rate $\Omega_s' = L\Omega_s$ (right).

The equivalent impulse response of this filter is given by

$$h_I[n] = \text{sinc}(\frac{n}{L}) \tag{13.23}$$

Let us verify that the above interpretation of upsampling as the cascade of a sampling rate expander followed by a LP filter is consistent with (13.16):

$$\begin{aligned} x'[n] &= \sum_{k=-\infty}^{\infty} x[k]\,\text{sinc}(\frac{n-kL}{L}) \\ &= \sum_{k=-\infty}^{\infty} x_E[kL]\,\text{sinc}(\frac{n-kL}{L}) \\ &= \sum_{l=-\infty}^{\infty} x_E[l]\,\text{sinc}(\frac{n-l}{L}) \\ &= \sum_{l=-\infty}^{\infty} x_E[l]\,h_i[n-l] \end{aligned} \tag{13.24}$$

A block diagram of a complete upsampling system operating in the discrete-time domain is shown in Figure 13.8. We shall also refer to this system as a digital interpolator. In the context of upsampling, the LP

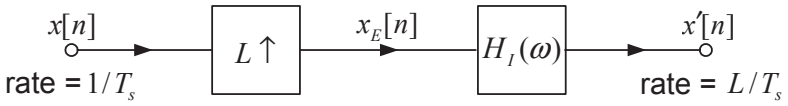


Fig. 13.8 Discrete-Time Upsampling System.

filter $H_I(\omega)$ is called anti-imaging filter, or also interpolation filter.

13.4 Changing sampling rate by a rational factor

Suppose Ω'_s and Ω_s are related as follows:

$$\Omega'_s = \frac{L}{M}\Omega_s \tag{13.25}$$

Such a conversion of the sampling rate may be accomplished by means of upsampling and downsampling by appropriate integer factors:

$$\Omega_s \xrightarrow[\text{by } L]{\text{Upsampling}} L\Omega_s \xrightarrow[\text{by } M]{\text{Downsampling}} \frac{L}{M}\Omega_s \tag{13.26}$$

The corresponding block diagram is shown in Figure 13.9. To avoid unnecessary loss of information when $M > 1$, upsampling is applied first.

Simplified structure: Recall the definitions of $H_I(\omega)$ and $H_D(\omega)$:

$$H_I(\omega) = \begin{cases} L, & |\omega| < \pi/L \\ 0, & \pi/L \le |\omega| \le \pi \end{cases} \tag{13.27}$$

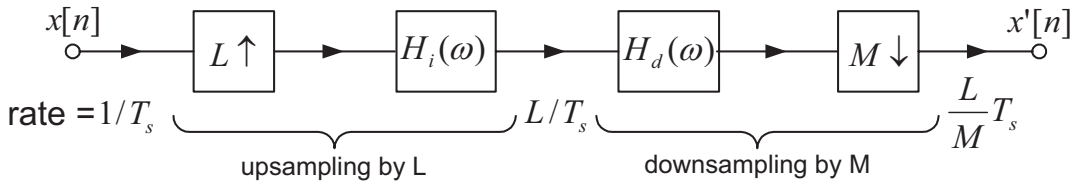


Fig. 13.9 Sampling rate alteration by a rational factor.

$$H_D(\omega) = \begin{cases} 1, & |\omega| < \pi/M \\ 0, & \pi/M \le |\omega| \le \pi \end{cases} \tag{13.28}$$

The cascade of these two filters is equivalent to a single LP filter with frequency response:

$$H_{ID}(\omega) = H_I(\omega)H_D(\omega) = \begin{cases} L, & |\omega| < \omega_c \\ 0, & \omega_c \le |\omega| \le \pi \end{cases} \tag{13.29}$$

where

$$\omega_c \triangleq \min(\pi/M, \pi/L) \tag{13.30}$$

The resulting simplified block diagram is shown in Figure 13.10.

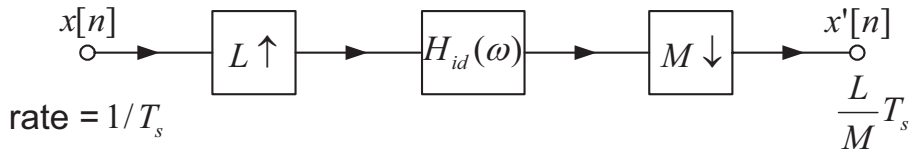


Fig. 13.10 Simplified rational factor sampling rate alteration

13.5 Polyphase decomposition

Introduction: The polyphase decomposition is a mathematical tool used in the analysis and design of multirate systems. It leads to theoretical simplifications as well as computational savings in many important applications.

Consider a system function

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n} \tag{13.31}$$

where $h[n]$ denotes the corresponding impulse response. Consider the following development of $H(z)$ where the summation over n is split into a summations over even odd indices, respectively:

$$\begin{aligned} H(z) &= \sum_{r=-\infty}^{\infty} h[2r]z^{-2r} + \sum_{r=-\infty}^{\infty} h[2r+1]z^{-(2r+1)} \\ &= \sum_{r=-\infty}^{\infty} h[2r]z^{-2r} + z^{-1}\sum_{r=-\infty}^{\infty} h[2r+1]z^{-2r} \\ &= E_0(z^2) + z^{-1}E_1(z^2) \end{aligned} \tag{13.32}$$

where the functions $E_0(z)$ and $E_1(z)$ are defined as

$$E_0(z) = \sum_{r=-\infty}^{\infty} h[2r]z^{-r} \tag{13.33}$$

$$E_1(z) = \sum_{r=-\infty}^{\infty} h[2r+1]z^{-r} \tag{13.34}$$

We refer to (13.32) as the *2-fold polyphase decomposition* of the system function $H(z)$. The functions $E_0(z)$ and $E_1(z)$ are the corresponding polyphase component. The above development can be easily generalized, leading to the following result.

Property: In its region of convergence, any system function $H(z)$ admits a unique M-fold polyphase decomposition

$$H(z) = \sum_{l=0}^{M-1} z^{-l} E_l(z^M) \tag{13.35}$$

where the functions $E_l(z)$ are called polyphase components of $H(z)$.

Proof: The basic idea is to split the summation over n in (13.31) into M individual summation over index $r \in \mathbb{Z}$, such that each summation covers the time indices $n = Mr + l$, where $l = 0, 1, \ldots, M-1$:

$$\begin{aligned} H(z) &= \sum_{l=0}^{M-1} \sum_{r=-\infty}^{\infty} h[Mr+l]z^{-(Mr+l)} \\ &= \sum_{l=0}^{M-1} z^{-l} \sum_{r=-\infty}^{\infty} h[Mr+l]z^{-(Mr)} \\ &= \sum_{l=0}^{M-1} z^{-l} E_l(z^M) \end{aligned} \tag{13.36}$$

where we define

$$E_l(z) = \sum_{r=-\infty}^{\infty} h[Mr+l]z^{-r} \tag{13.37}$$

This shows the existence of the polyphase decomposition. The unicity is proved by involving the unique nature of the series expansion (13.31).

Remarks:

- The proof is constructive in that in suggests a way of deriving the polyphase components $E_l(z)$ via (13.37).
- The use of (13.37)is recommended for deriving the polyphase components in the case of an FIR filter $H(z)$. For IIR filters, there may be simpler approaches.
- In anycase, once a polyphase decomposition as in (13.36) has been obtained, one can be sure that it is *the right one* since it is unique.

Example 13.5:

► Consider the FIR filter

$$H(z) = (1 - z^{-1})^6$$

To find the polyphase components for $M = 2$, we may proceed as follows:

$$\begin{aligned} H(z) &= 1 - 6z^{-1} + 15z^{-2} - 20z^{-3} + 15z^{-4} - 6z^{-5} + z^{-6} \\ &= 1 + 15z^{-2} + 15z^{-4} + z^{-6} + z^{-1}(-6 - 20z^{-2} - 6z^{-4}) \\ &= E_0(z^2) + z^{-1}E_1(z^2) \end{aligned}$$

where

$$E_0(z) = 1 + 15z^{-1} + 15z^{-2} + z^{-3}$$
$$E_1(z) = -6 - 20z^{-1} - 6z^{-2}$$

◄

Noble identities: The first identy can be used to exchange the order of a decimator and an arbitrary filter $G(z)$: The second identity can be used to exchange the order of an expander and filter $G(z)$: The proof of

Fig. 13.11 1st Noble identity.

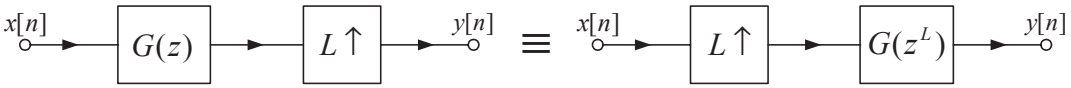


Fig. 13.12 2nd Noble identity.

these identities are beyond the scope of this course.

Efficient downsampling structure: The polyphase decomposition can be used along with Noble identity 1 to derive a computationally efficient structure for downsampling.

- Consider basic block diagram of a downsampling system: Suppose that the low-pass anti-aliasing

Fig. 13.13 Block diagram of a downsampling system.

filter $H(z)$ is FIR with length N, i.e.

$$H(z) = \sum_{n=0}^{N-1} h[n]z^{-n} \tag{13.38}$$

Assuming a direct form (DF) realization for $H(z)$, the above structure requires N multiplications per unit time at the input sampling rate. Note that only 1 out of every M samples computed by the FIR filter is actually output by the M-fold decimator.

- For simplicity, assume that N is a multiple of M. Making use of the polyphase decomposition (13.36), the following equivalent structure for the downsampling system is obtained where the polyphase com-

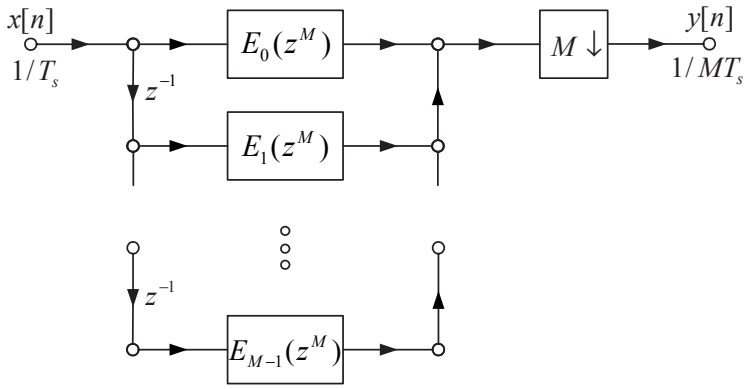


Fig. 13.14 Equivalent structure for the downsampling system.

ponents are given here by

$$E_l(z^M) = \sum_{r=0}^{N/M-1} h[Mr+l]z^{-Mr} \tag{13.39}$$

If realized in direct form, each filter $E_l(z^M)$ requires N/M multiplications per unit time. Since there are M such filters, the computational complexity is the same as above.

- Finally, consider interchanging the order of the filters $E_l(z^M)$ and the decimator in Figure 13.14 by making use of Noble identity 1: Observe that the resulting structure now only requires N/M multipli-

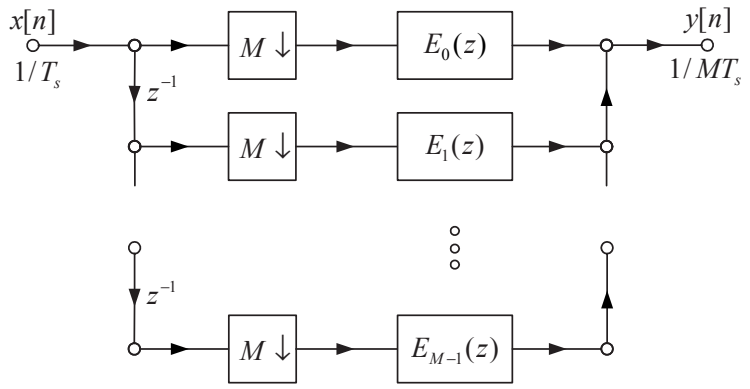


Fig. 13.15 Final equivalent structure for the downsampling system.

cations per unit time.

One could argue that in the case of an FIR filter $H(z)$, the computational saving is artificial since actually, only one out of every M filter outputs in Fig. 13.13 needs to be computed. However, since the input samples are coming in at the higher rate F_s, the use of a traditional DF realization for $H(z)$ still requires that the computation of any particular output be completed before the next input sample $x[n]$ arrives and modifies the delay line. Thus the peak required computational rate is still N multiplies per input sample, even though the filtering operation needs only be activated once every M samples. This may be viewed as an inefficient use of available computational resources. The real advantage of the structure in Fig. 13.15 is that it makes it possible to spread the FIR filter computation over time, so that the required computational rate is uniform and equal to N/M multiplications per input samples.

Chapter 14

Applications of the DFT and FFT

In this Chapter, we discuss two basic applications of the DFT and associated FFT algorithms.

We recall from Chapter 6 that the linear convolution of two time-limited signals can be realized via circular convolution in the DFT domain. The first application describes an efficient way of realizing an FIR filtering via block DFT processing. Essentially, this amounts to decomposing the input signal into consecutive blocks of smaller length, and implementing the filtering in the frequency domain via FFT and IFFT processing. This results in significant computational savings, provided a processing delay is tolerated in the underlying application.

In the second application, we look at the use of the DFT (efficiently computed via FFT) as a frequency analysis or estimation tool. We review the fundamental relationship between the DFT of a finite signal and the DTFT and we investigate the effects of the DFT length and windowing function on the spectral resolution. The effects of measurement noise on the quality of the estimation are also discussed briefly.

14.1 Block FIR filtering

Consider a causal FIR filtering problem as illustrated in Figure 14.1. Assume that the filter's impulse re-

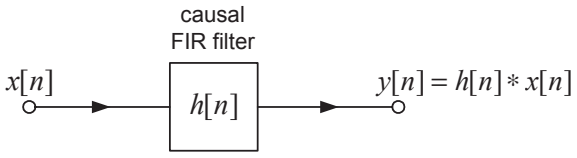


Fig. 14.1 Causal FIR filtering

sponse $h[n]$ is time-limited to $0 \leq n \leq P-1$ and that the input signal $x[n]$ is zero for $n < 0$. Accordingly, the filter output

$$y[n] = h[n] * x[n] = \sum_{k=0}^{P-1} h[k]x[n-k] \tag{14.1}$$

is also zero for $n < 0$. If $x[n]$ was also time-limited, then the DFT approach in Section 6.5.1 could be used to compute $y[n]$, provided the DFT size N is large enough.

Two fundamental problems exist with respect to the assumption that $x[n]$ is time-limited: $x[n]$:

- in certain applications, it is not possible to set a specific limit on the length of $x[n]$ (e.g. voice communications);
- such a limit, if it exists, may be too large, leading to various technical problems (processing delay, memory requirement, etc.).

In practice, it is still possible to compute the linear convolution (14.1) as a circular convolution via the DFT (i.e. FFT), but it is necessary to resort to *block convolution*.

Block convolution: The basic principle of block convolution may be summarized as a series of three steps as follows:

(1) Decompose $x[n]$ into consecutive blocks of finite length L:

$$x[n] \Rightarrow x_r[n], \quad r = 0,1,2,... \tag{14.2}$$

where $x_r[n]$ denotes the samples in the $r$the block.

(2) Filter each block individually using the DFT approach:

$$x_r[n] \Rightarrow y_r[n] = h[n] * x_r[n] \tag{14.3}$$

In practice, the required DFTs and inverse DFT are computed using an FFT algorithm (more on this later).

(3) Reassemble the filtered blocks (as they become available) into an output signal:

$$y_r[n] \Rightarrow y[n] \tag{14.4}$$

The above generic steps may be realized in a different ways. In particular, there exist two basic methods of block convolution: overlap-add and overlap-save. Both will be described below.

14.1.1 Overlap-add method:

In this method, the generic block convolution steps take the following form:

(1) Blocking of input data into *non-overlapping* blocks of length L:

$$x_r[n] = \begin{cases} x[n+rL] & 0 \le n < L, \\ 0 & \text{otherwise.} \end{cases} \tag{14.5}$$

where, for convenience of notation, the shift factor rL is introduced so that the non-zero samples of $x_r[n]$ are all between $0 \le n \le L-1$.

(2) For each input block $x_r[n]$, compute its linear convolution with FIR filter impulse response $h[n]$ (see DFT approach below):

$$y_r[n] = h[n] * x_r[n] \tag{14.6}$$

Note that each block $y_r[n]$ is now of length $L+P-1$.

(3) Form the output signal by adding the shifted filtered blocks:

$$y[n] = \sum_{r=0}^{\infty} y_r[n-rL] \tag{14.7}$$

The method takes its name from the fact that the output blocks $y_r[n]$ need to be properly overlapped and added in the computation of the desired output signal $y[n]$.

Mathematical justification: It is easy to verify that the overlap-add approach produces the desired convolution results in (14.1). Under the assumption that $x[n] = 0$ for $n < 0$, we have from (14.5) that

$$x[n] = \sum_{r=0}^{\infty} x_r[n-rL] \tag{14.8}$$

Then, invoking basic properties of the linear convolution, we have

$$\begin{aligned} y[n] &= h[n] * x[n] = h[n] * \sum_{r=0}^{\infty} x_r[n-rL] \\ &= \sum_{r=0}^{\infty} h[n] * x_r[n-rL] = \sum_{r=0}^{\infty} y_r[n] \end{aligned} \tag{14.9}$$

DFT realization of step (2): Recall that the DFT approach to linear convolution amounts to computing a circular convolution. To ensure that the circular and linear convolutions are equivalent, the DFT length, say N, should be sufficiently large and the data must be appropriately zero-padded. Specifically, based on Section 6.5, we may proceed as follows

- Set DFT length to $N \geq L+P-1$ (usually a power of 2)
- Compute (once and store result)

$$H[k] = \text{FFT}_N\{h[0], ..., h[P-1], 0, ..., 0\} \tag{14.10}$$

- For $r = 0, 1, 2, ...$, compute

$$X_r[k] = \text{FFT}_N\{x_r[0], ..., x_r[L-1], 0, ..., 0\} \tag{14.11}$$

$$y_r[n] = \text{IFFT}_N\{H[k]X_r[k]\} \tag{14.12}$$

Often, the value of N is selected as the smallest power of 2 that meets the above condition.

Computational complexity: The computational complexity of the overlap-add method can be evaluated as follows. For each block of L input samples, we need

- $\frac{N}{2}\log_2 N$ multiplications to compute $X_r[k]$ in (14.11)
- N multiplications to compute $H[k]X_r[k]$ in (14.12)

- $\frac{N}{2}\log_2 N$ multiplications to compute $y_r[n]$ in (14.12)

for a total of $N\log_2 N + N$ multiplications per L input samples. Assuming $P \approx L$ for simplicity, we may set $N = 2L$. The resulting complexity of the overlap-add method is then

$$2\log_2 L + 4$$

multiplications per input sample. Linear filtering based on (14.1) requires P multiplications per input sample. Thus, a significant gain may be expected when $P \approx L$ is large (e.g. $L = 1024 \Rightarrow 2\log_2 L + 4 = 24 \ll L$.

Example 14.1: Overlap-add

▶ The overlap-add process is depicted in Figures 14.2 and 14.3. The top part of Figure 14.2 shows the FIR filter $h[n]$ as well as the input signal $x[n]$, supposed to be semi-infinite (right-sided). The three bottom parts of Figure 14.2 show the three first blocks of the input $x_0[n], x_1[n]$ and $x_2[n]$. Figure 14.3 shows the result of the convolution of these blocks with $h[n]$. Notice that the convolution results are longer than the original blocks. Finally, the bottom plot of Figure 14.3 shows the reconstructed output signal obtained by adding up the overlapping blocks $y_r[n]$. ◀

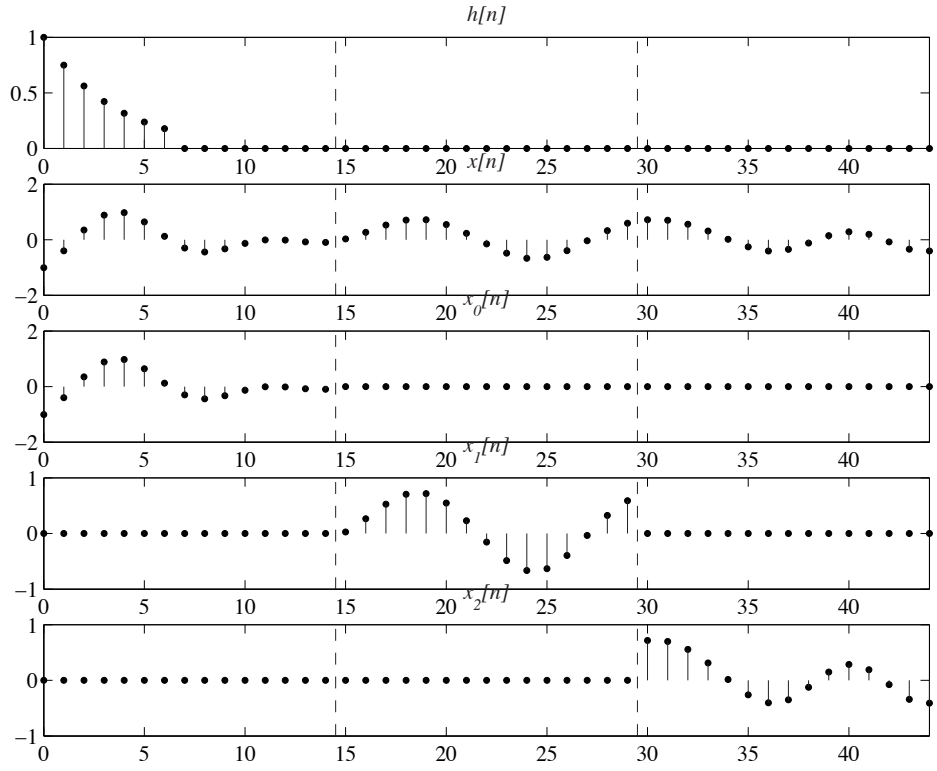


Fig. 14.2 Illustration of blocking in the overlap-add method. Top: the FIR filter $h[n]$; Below: the input signal $x[n]$ and different blocks $x_r[n]$.

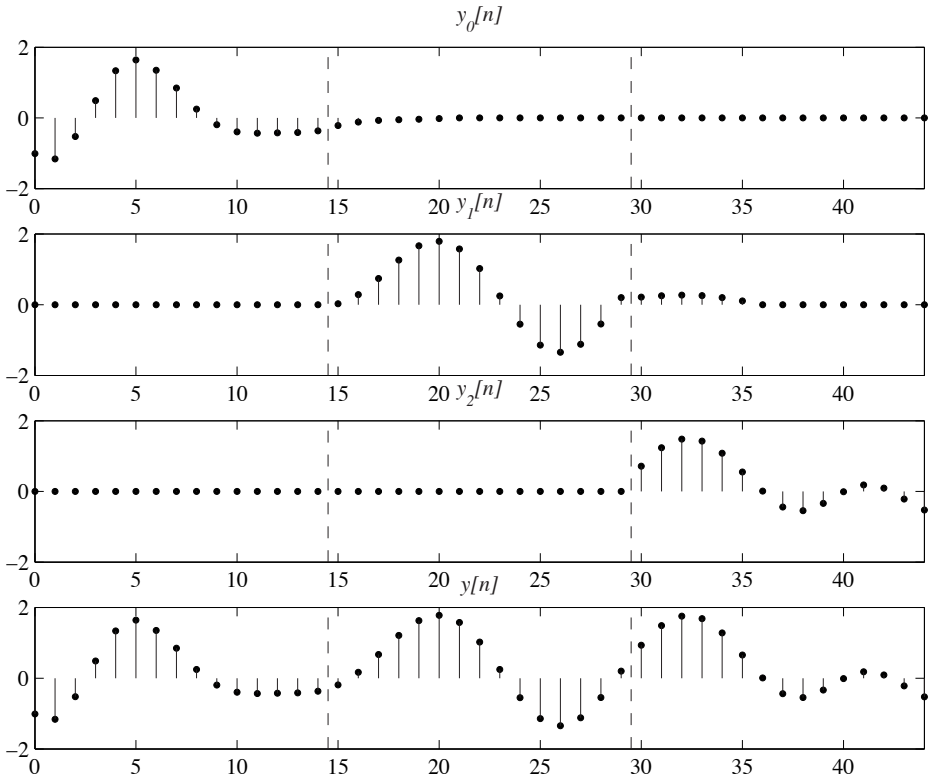


Fig. 14.3 Illustration of reconstruction in the overlap-add method. Top: several blocks $y_r[n]$ after convolution; Bottom: the output signal $y[n]$.

14.1.2 Overlap-save method (optional reading)

This method contains the following steps:

(1) Blocking of input data in *overlapping* blocks of length L, overlapping by $P-1$ samples:

$$x_r[n] = \begin{cases} x[n+r(L-P+1)-P+1] & 0 \le n < L, \\ 0 & \text{otherwise.} \end{cases} \tag{14.13}$$

(2) For each input block, compute the L-point *circular* convolution with the FIR filter impulse response $h[n]$:

$$y_r[n] = h[n] \circledast x_r[n] \tag{14.14}$$

Note that each block $y_r[n]$ is still of length L. Since this circular convolution does not respect the criterion in (6.63), there will be time-aliasing. By referring to the example in Figure 6.12, one sees that the first $P-1$ samples of this circular convolution will be corrupted by aliasing, whereas the remaining $L-P+1$ will remain equal to the true samples of the corresponding linear convolution $x_r[n] * h[n]$.

(3) Form the output signal by adding the shifted filtered blocks after discarding the leading $P-1$ samples of each block:

$$y[n] = \sum_{r=0}^{\infty} y_r[n-r(L-P+1)](u[n-P-r(L-P+1)] - u[n-L-r(L-P+1)]) \tag{14.15}$$

Example 14.2: Overlap-save

▶ The overlap-save process is depicted in Figures 14.4 and 14.5. The top part of Figure 14.4 shows the FIR filter $h[n]$ as well as the input signal $x[n]$. The three bottom parts of Figure 14.4 show the three first blocks of the input $x_0[n], x_1[n]$ and $x_2[n]$. Figure 14.5 shows the result of the circular convolution of these blocks with $h[n]$. Notice that the convolution white dots correspond to samples corrupted by aliasing. Finally, the bottom plot of Figure 14.5 shows the reconstructed output signal obtained by adding up the blocks $y_r[n]$ after dropping the aliased samples. ◀

14.2 Frequency analysis via DFT/FFT

14.2.1 Introduction

The frequency content of a discrete-time signal $x[n]$ is defined by its DTFT:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}, \quad \omega \in [-\pi, \pi], \tag{14.16}$$

As pointed out in Chapter 6, the practical computation of $X(\omega)$ poses the several difficulties:

- an infinite number of mathematical operations is needed (the summation over n is infinite and the variable ω is continuous).

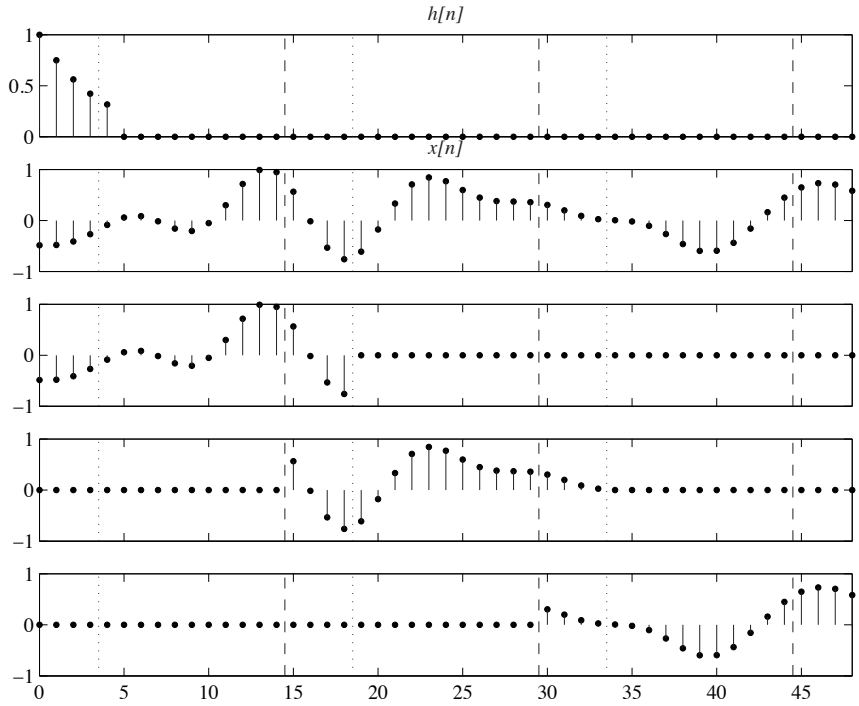


Fig. 14.4 Illustration of blocking in the overlap-save method. Top: the FIR filter $h[n]$; Below: the input signal $x[n]$ and different blocks $x_r[n]$.

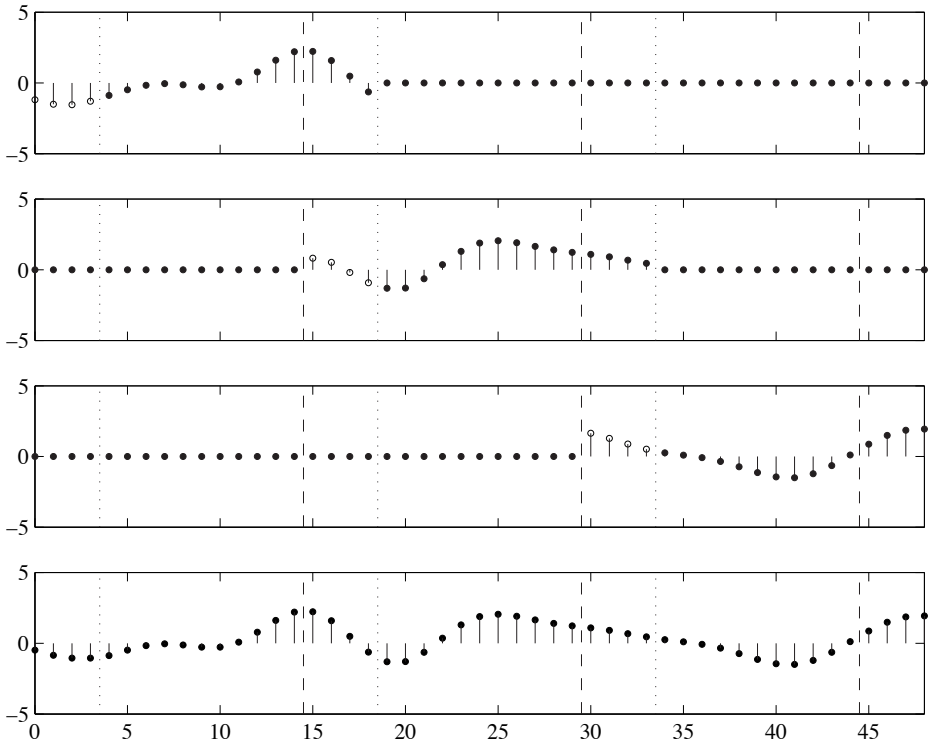


Fig. 14.5 Illustration of reconstruction in the overlap-save method. Top: several blocks $y_r[n]$ after convolution; Bottom: the output signal $y[n]$.

- all the signals samples $x[n]$ from $-\infty$ to ∞ must be available

Recall the definition of the DFT

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n}, \quad k \in \{0,1,\ldots,N-1\} \tag{14.17}$$

where $\omega_k = 2\pi k/N$. In practice, the DFT is often used as an approximation to the DTFT for the purpose of frequency analysis. The main advantages of the DFT over the DTFT are

- finite number of mathematical operation
- only a finite set of signal samples $x[n]$ is needed

If $x[n] = 0$ for $n < 0$ and $n \geq N$, then the DFT and the DTFT are equivalent concept. Indeed, as we have shown in Chapter 6, there is in this case a 1-to-1 relationship between the DFT and the DTFT, namely:

$$X[k] = X(\omega_k) \tag{14.18}$$

$$X(\omega) = \sum_{k=0}^{N-1} X[k]P(\omega - \omega_k) \tag{14.19}$$

where $P(\omega)$ is an interpolation function defined in (6.18).

In general, however, we are interested in analyzing the frequency content of a signal $x[n]$ which is not a priori time-limited. In this case, the DFT only provides an approximation to the DTFT. In this Section:

- we investigate in further detail the nature of this approximation;
- based on this analysis, we present and discuss modifications to the basic DFT computation that lead to improved frequency analysis results.

14.2.2 Windowing effects

Rectangular window: Recall the definition of a rectangular window of length N:

$$w_R[n] \triangleq \begin{cases} 1, & 0 \leq n \leq N-1 \\ 0, & \text{otherwise} \end{cases} \tag{14.20}$$

The DTFT of $w[n]$ will play a central role in our discussion:

$$W_R(\omega) = \sum_{n=0}^{N-1} e^{-j\omega n} = e^{-j\omega(N-1)/2} \frac{\sin(\omega N/2)}{\sin(\omega/2)} \tag{14.21}$$

The corresponding magnitude spectrum is illustrated in Figure 14.6 for the $N = 16$. Note the presence of:

- a main lobe with
$$\Delta\omega = \frac{2\pi}{N} \tag{14.22}$$
- sidelobes with peak a amplitude of about 0.22 (-13dB)

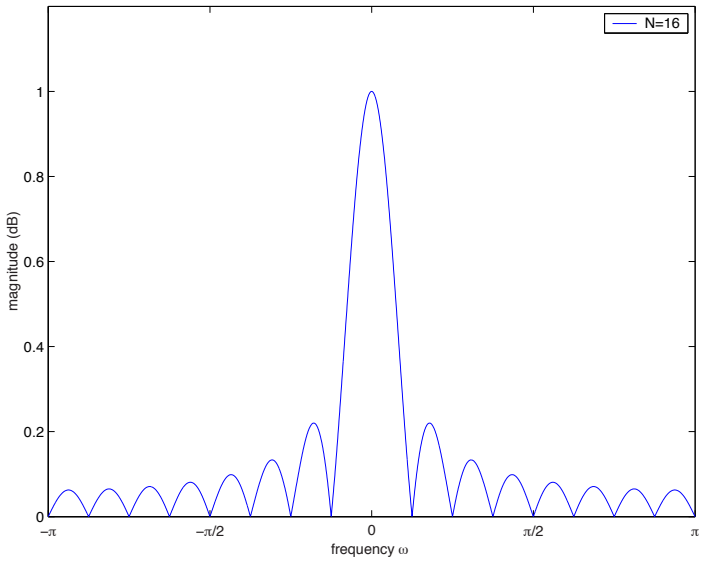


Fig. 14.6 Magnitude spectrum of $N = 16$ point rectangular window (linear scale)

Relation between DFT and DTFT: Define the windowed signal

$$y[n] = w_R[n]x[n] \tag{14.23}$$

The N-point DFT of signal $x[n]$ can be expressed as the DTFT of $y[n]$:

$$\begin{aligned} X[k] &= \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} \\ &= \sum_{n=-\infty}^{\infty} w_R[n]x[n]e^{-j\omega_k n} \\ &= \sum_{n=-\infty}^{\infty} y[n]e^{-j\omega_k n} = Y(\omega_k) \end{aligned} \tag{14.24}$$

where $Y(\omega)$ denotes the DTFT of $y[n]$. The above shows that the DFT can be obtained by uniformly sampling $Y(\omega)$ at the frequencies $\omega = \omega_k$.

Nature of $Y(\omega)$: Since $y[n]$ is obtained as the product of $x[n]$ and $w_R[n]$, we have from the multiplication property of the DTFT (Ch. 3) that

$$\begin{aligned} Y(\omega) &= \text{DTFT}\{w_R[n]x[n]\} \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)W_R(\omega-\phi)d\phi \end{aligned} \tag{14.25}$$

That is, $Y(\omega)$ is equal to the circular convolution of the periodic functions $W_R(\omega)$ and $X(\omega)$. When computing the N-point DFT of the signal $x[n]$, the windowing of $x[n]$ that implicitly takes place in the time-domain is equivalent to the periodic convolution of the desired spectrum $X(\omega)$ with the window spectrum $W_R(\omega)$.

Example 14.3:

► Consider the signal

$$x[n] = e^{j\theta n}, \quad n \in \mathbb{Z}$$

where $0 < \theta < \pi$ for simplicity. The DTFT of $x[n]$ is given by

$$X(\omega) = 2\pi \sum_{k=-\infty}^{\infty} \delta_a(\omega - \theta - 2\pi k)$$

In this special case, equations (14.24)- (14.25) yield

$$X[k] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\phi) W_R(\omega_k - \phi) d\phi = W_R(\omega_k - \theta)$$

Thus, the DFT values $X[k]$ can be obtained by uniformly sampling the shifted window spectrum $W_R(\omega - \theta)$ at the frequencies $\omega = \omega_k$. ◄

Effects of windowing: Based on the above example and taking into consideration the special shape of the window magnitude spectrum in Figure 14.6, two main effects of the time-domain windowing on the original DTFT spectrum $X(\omega)$ may be identified:

- Due to non-zero width of the main lobe in Figure 14.6, local smearing (or spreading) of the spectral details in $X(\omega)$ occur as a result of the convolution with $W_R(\omega)$ in (14.25). This essentially corresponds to a loss of resolution in the spectral analysis. In particular, if say $X(\omega)$ contains two narrow peaks at closely spaced frequencies θ_1 and θ_2, it may be that $Y(\omega)$ in (14.25) only contains a single wider peak at frequency $(\theta_1 + \theta_2)/2$. In other words, the two peaks will merge as a result of spectral spreading. The resolution of the spectral analysis with DFT is basically a function of the window width $\Delta\omega = 2\pi/N$. Thus resolution can be increased (i.e. $\Delta\omega$ smaller) by increasing N.
- Due to the presence of sidelobes in Figure 14.6, spectral leakage in $X(\omega)$ will occur as a result of the convolution with $W_R(\omega)$ in (14.25). For instance, even if $X(\omega) = 0$ in some band of frequency, $Y(\omega)$ will usually not be zero in that band due to the trailing effect of the sidelobes in the convolution process. The amount of leakage depends on the peak sidelobe level. For a rectangular winodw, this is fixed to -13dB.

Improvements: Based on the above discussion, it should be clear that the results of the frequency analysis can be improved by using a different type of window, instead of the rectangular window implicit in (14.24). Specifically:

$$\tilde{X}[k] = \sum_{n=0}^{N-1} w[n] x[n] e^{-j\omega_k n}, \quad k \in \{0, 1, \ldots, N-1\} \tag{14.26}$$

where $w[n]$ denotes an appropriately chosen window. In this respect, all the window functions introduced in Chapter 9 can be used. In selecting a window, the following approach might be taken:

- Select window so that peak sidelobe level is below acceptable level.
- Adjust window length N so that desired resolution $\Delta\omega$ is obtained. For most windows of interest $\Delta\omega = cte/N$ where cte depends on the specific window type.

The trade-offs involved in the selection of a window for spectral analysis are very similar to those use in the window method of FIR filter design (Ch. 9).

References

[1] Analog Devices Inc., Norwood, MA. *TIgerSHARC DSP Microcomputer Preliminary Technical data*, 2002.

[2] Ashok Bindra. Novel architectures pack multiple dsp cores on-chip. *Electronic Design*, December 2001.

[3] Yu Hen Hu, editor. *Programmable Digital Signal Processors: Architecture, Programming and Applications*. Marcel Dekker, 2002.

[4] Emmanuel C. Ifeachor and Barrie W. Jervis. *Digital Signal Processing, a practical approach*. Prentice-Hall, New Jersey, 2nd edition edition, 2002.

[5] Phil Lapsley, Jeff Bier, Amit Shoham, and Edward A. Lee. *DSP Processor Funcamentals: Architectures and Features*. IEEE Press, 1997.

[6] Ravi A. Managuli and Yongmin Kim. VLIW processor architectures and algorithm mappings for DSP applications. In Hu [3], chapter 2.

[7] James H. McClellan, Ronald W. Schafer, and Mark A. Yoder. *DSP First: a multimedia approach*. Prentice-Hall, 1998.

[8] Sanjit K. Mitra. *Digital Signal Processing, a computer-based approach*. McGraw-Hill, 2nd edition edition, 2001.

[9] Motorola, Inc., Denver, CO. *DSP56300 Family Manual*, 2000.

[10] Allan V. Oppehneim, Ronald W. Schfafer, and J. R. Buck. *Discrete-Time Signal Processing*. Prentice-Hall, New Jersey, 2nd edition edition, 1999.

[11] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processing, Principles, algorithms and applications*. Prentice-Hall, 3rd edition edition, 1996.

[12] Russell Tessier and Wayne Burleson. Reconfigurable computing and digital signal processing: Past, present and future. In Hu [3], chapter 4.

[13] Texas Instruments, Houston, TX. *TMS320C6204 Data Sheet*, 2000.