Digital Signal and Image Processing

(CSC701)

Semester VII - Computer Engineering
(Mumbai University)

Strictly as per the New Revised Syllabus (Rev- 2016) of Mumbai University w.e.f. academic year 2019-2020

Dhananjay K. Theckedath

M. Tech. - IIT (Mumbai) - Biomedical Engineering,
B.Sc. Tech. - Electronic Instrumentation
B.Sc. - Physics,
Assistant Professor,
Thadomal Shahani Engineering College,
Mumbai University, Maharashtra, India.

TechKnowledge™ Publications

(Book Code : MO80A)

Digital Signal & Image Processing

Dhananjay K. Theckedath

(Semester VII - Computer Engineering) (MU))

First Printed in India : August 2008 **(For UPTU)**

First Edition : July 2019(**TechKnowledge Publications**)

P

ISBN : 978-93-89424-30-0

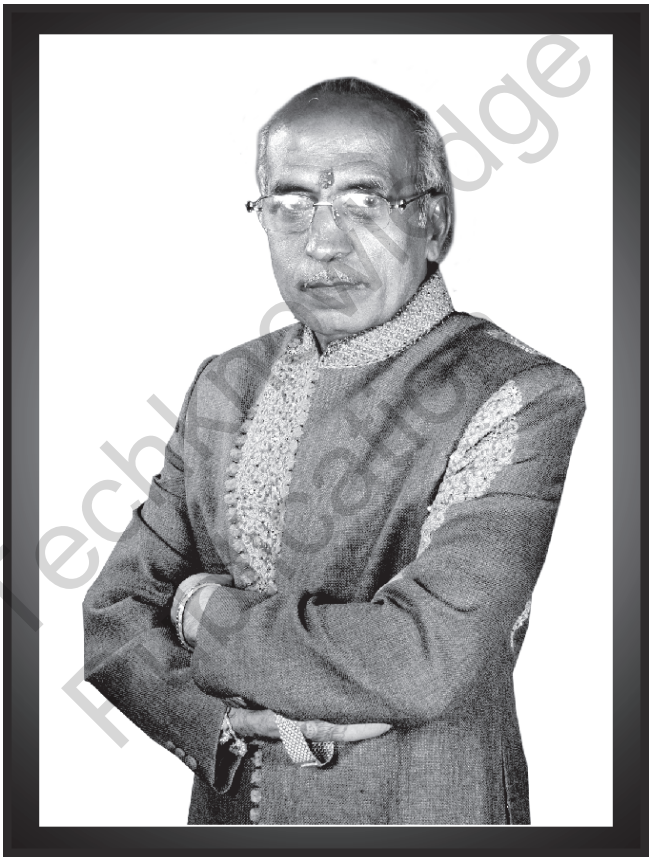
Published by

TechKnowledge Publications

Head Office : B/5, First floor, Maniratna Complex,
Taware Colony, Aranyeshwar Corner,
Pune - 411 009. Maharashtra State, India
Ph : 91-20-24221234, 91-20-24225678.

[CSC701] (FID : MO80) (Book Code : MO80A)

We dedicate this Publication soulfully and wholeheartedly,
in loving memory of our beloved founder director,
Late. Shri. Pradeepsheth Lalchandji Lunawat, *who will always be an inspiration, a positive force and strong support behind us.*



Lt. Shri. Pradeepji L. Lunawat

Soulful Tribute and Gratitude for all Your
Sacrifices, Hardwork and 40 years of Strong Vision.....

(Book Code : MO80A)

Preface

Dear Students,

I am extremely happy to present this book **"Digital Signal & Image Processing"** for you. I have divided the subject into small chapters so that the topics can be arranged and understood properly. The topics within the chapters have been arranged in a proper sequence to ensure smooth flow and understanding of the subject.

A number of solved examples have been included. So, I am sure that this book will cater all your needs for this subject.

I present this book in the loving memory of **Late. Shri. Pradeepji Lunawat**, our source of inspiration and a strong foundation of **"TechKnowledge Publications".** He will always be remembered in our hearts and motivate us to achieve our new milestone.

I am thankful to Prof. Arunoday Kumar, Shri. Shital Bhandari & Shri. Chandroday Kumar for the encouragement and support that they have extended. I am also thankful to the staff members of TechKnowledge Publications and others for their efforts to make this book as good as it is. I have made every possible efforts to eliminate all the errors in this book. However if you find any, please let me know, because that will help me to improve the book quality further.

I am also thankful to my family members and friends for their patience and encouragement.

- Author

❑❑❑

(Book Code : MO80A)

Syllabus

Digital Signal & Image Processing

Course Code	Course Name	Credits
CSC701	Digital Signal & Image Processing	4

Course Objectives :

1. To understand the fundamental concepts of digital signal processing and Image processing.
2. To explore DFT for 1-D and 2-D signal and FFT for 1-Dsignal
3. To apply processing techniques on 1-D and Image signals.
4. To apply digital image processing techniques for edge detection.

Course outcomes :

On successful completion of the course learner will be able to :

1. Apply the concept of DT Signal and DT Systems.
2. Classify and analyze discrete time signals and systems
3. Implement Digital Signal Transform techniques DFT and FFT.
4. Use the enhancement techniques for digital Image Processing
5. Differentiate between the advantages and disadvantages of different edge detection techniques
6. Develop small projects of 1-D and 2-D Digital Signal Processing.

Prerequisite: Applied Mathematics

Module No.	Unit No.	Topic details	Hrs.
1.0	1.1 1.2 1.3	**Discrete-Time Signal and Discrete-Time System** Introduction to Digital Signal Processing, Sampling and Reconstruction, Standard DT Signals, Concept of Digital Frequency, Representation of DT signal using Standard DT Signals, Signal Manipulations(shifting, reversal, scaling, addition, multiplication). Classification of Discrete-Time Signals, Classification of Discrete- Systems Linear Convolution formulation for 1-D and 2-D signal (without mathematical proof), Circular Convolution (without mathematical proof), Linear convolution using Circular Convolution. Auto and Cross Correlation formula evaluation, LTI system, Concept of Impulse Response and Step Response, Output of DT system using Time Domain Linear Convolution. **(Refer Chapters 1, 2 and 3)**	14

2.0		**Discrete Fourier Transform**	08
	2.1	Introduction to DTFT, DFT, Relation between DFT and DTFT, IDFT	
	2.2	Properties of DFT without mathematical proof (Scaling and Linearity, Periodicity, Time Shift and Frequency Shift, Time Reversal, Convolution Property and Parsevals' Energy Theorem). DFT computation using DFT properties.	
	2.3	Transfer function of DT System in frequency domain using DFT. Linear and Circular Convolution using DFT, Convolution of long sequences, Introduction to 2-D DFT **(Refer Chapter 3)**	
3.0		**Fast Fourier Transform**	06
	3.1	Need of FFT, Radix-2 DIT-FFT algorithm,	
	3.2	DIT-FFT Flow graph for N=4 and 8, Inverse FFT algorithm.	
	3.3	Spectral Analysis using FFT **(Refer Chapters 3 and 4)**	
4.0		**Digital Image Fundamentals**	08
	4.1	Introduction to Digital Image, Digital Image Processing System, Sampling and Quantization	
	4.2	Representation of Digital Image, Connectivity	
	4.3	Image File Formats: BMP, TIFF and JPEG. **(Refer Chapters 5 and 6)**	
5.0		**Image Enhancement in Spatial domain**	10
	5.1	Gray Level Transformations, Zero Memory Point Operations,	
	5.2	Histogram Processing, Histogram equalization.	
	5.3	Neighborhood Processing, Spatial Filtering, Smoothing and Sharpening Filters, Median Filter. **(Refer Chapters 7 and 8)**	
6.0		**Image Segmentation**	06
	6.1	Segmentation based on Discontinuities (point, Line, Edge),	
	6.2	Image Edge detection using Robert, Sobel, Previtt masks, Image Edge detection using Laplacian Mask. **(Refer Chapter 9)**	
		Total	52

❑❑❑

Chapter 1 : Introduction to Digital Signal Processing

1-1 to 1-17

Syllabus :

Introduction to Digital Signal Processing, Sampling and Reconstruction, Standard DT Signals, Concept of Digital Frequency, Representation of DT signal using Standard DT Signals, Signal Manipulations(shifting, reversal, scaling, addition, multiplication).

1.1 Introduction 1-1
1.1.1 Digital Signal Processing System 1-2
1.1.2 Advantages of Digital signal Processing over Analog Signal Processing 1-3
1.1.3 Difference between DSP and ASP 1-3
1.1.4 Limitations of Digital signal Processing 1-4
1.2 Signals 1-4
1.2.1 Continuous and Discrete Time Signals 1-4
1.2.2 Continuous Valued and Discrete Valued Signals 1-6
1.2.3 Deterministic and Random Signals 1-6
1.3 Concept of Frequency in the Continuous and Discrete Time Signals 1-6
1.3.1 Concept of Frequency in Continuous Time Signals 1-6
1.3.2 Concept of Frequency in Discrete Time Signals 1-7
1.3.3 Converting a Analog Signal to a Discrete Time Signal 1-8
1.4 Aliasing and Nyquist Rate 1-9
1.4.1 Anti Aliasing Filter 1-13
1.5 Basic Elements of a DSP System 1-13
1.5.1 Solved Examples 1-14
1.6 Applications of DSP 1-17

Chapter 2 : Discrete Time Signal and System

2-1 to 2-29

Syllabus :

Classification of Discrete-Time Signals, Classification of Discrete- Systems Linear Convolution formulation for 1-D and 2-D signal (without mathematical proof), Auto and Cross Correlation formula evaluation, LTI system, Concept of Impulse Response and Step Response, Output of DT system using Time Domain Linear Convolution

2.1 Introduction 2-1
2.1.1 Representation of Discrete Time Signals 2-1
2.1.2 Elementary Discrete Time Signals 2-2
2.2 Classification of Discrete Time Signals 2-4
2.2.1 Even and Odd Signals 2-5
2.2.2 Periodic and Aperiodic Signals 2-7
2.2.3 Energy and Power Signals 2-8
2.3 Operation on Signals 2-10
2.3.1 Time Shifting 2-10
2.3.2 Time Reversal 2-12
2.3.3 Time Scaling 2-12
2.3.4 Scalar Multiplication 2-13
2.3.5 Signal Addition and Multiplication 2-14
2.3.6 Solved Examples on Signals 2-14
2.4 Discrete Time Systems 2-17
2.4.1 Representation of a Signal in Terms of Weighted Sum of Shifted Discrete Impulse 2-20
2.5 Impulse Response and Convolution 2-21
2.5.1 Computation of Linear Convolution 2-22
2.5.2 Linear Convolution using Graphical Method 2-22
2.5.2(A) Solved Examples on Graphical Method 2-22
2.5.3 Linear Convolution using Tabular Method 2-26
2.5.3(A) Solved Examples on Tabular Method 2-27
2.6 Correlation 2-27
2.6.1 Solved Examples on Correlation 2-28

Chapter 3 : Discrete Fourier Transform (DFT)

3-1 to 3-68

Syllabus :

Introduction to DTFT, DFT, Relation between DFT and DTFT, IDFT, Properties of DFT without mathematical proof (Scaling and Linearity, Periodicity, Time Shift and Frequency Shift, Time, Reversal, Convolution Property and Parsevals' Energy Theorem). DFT computation using DFT properties. Transfer function of DT System in frequency domain using DFT. Linear and Circular Convolution using DFT, Convolution of long sequences, Introduction to 2-D DFT Circular Convolution (without mathematical proof), Linear convolution using Circular Convolution. Spectral Analysis using FFT

3.1 The Fourier Transform 3-1

3.2 Discrete Time Fourier Transform 3-2
3.2.1 Solved Examples on DTFT of Standard Signals 3-2
3.3 Relationship between DTFT and DFT 3-4
3.4 Discrete Fourier Transform (DFT) 3-4
3.4.1 Solved Examples on DFT of Simple Signals 3-5
3.5 The Fourier Spectrum 3-13
3.6 Inverse Discrete Fourier Transform (IDFT) 3-14
3.7 Computing DFT by Matrix Method 3-15
3.7.1 Solved Examples of DFT 3-16
3.8 Discrete Fourier Transform (DFT) Properties 3-29
3.8.1 Linearity 3-29
3.8.2 Periodicity 3-30
3.8.3 Circular Time Shift 3-31
3.8.4 Circular Frequency Shift 3-38
3.8.5 Time Reversal 3-39
3.8.6 Symmetry Property 3-40
3.8.7 Complex Conjugate Property 3-44
3.8.8 Parseval's Theorem 3-45
3.8.9 Multiplication of Two Sequences 3-46
3.8.10 Circular Convolution Property 3-47
3.9 Circular Convolution 3-47
3.9.1 Concentric Circle Method 3-47
3.9.1(A) Solved Example on Concentric Circle Method 3-47
3.9.2 Matrix Method 3-51
3.9.2(A) Solved Example on Matrix Method 3-51
3.9.3 Getting Linear Convolution from Circular Convolution 3-54
3.9.4 Solved Examples 3-54
3.10 Filtering of Long Duration Signals 3-60
3.10.1 Overlap - Save Method 3-60
3.10.2 Overlap - Add Method 3-62
3.10.3 Solved Examples on Method 3-64
3.11 Summary of DFT properties 3-66
3.12 2 D- Discrete Fourier Transform (2-DFT) 3-66
3.12.1 The Separability Property 3-66
3.12.2 Solved Examples on 2- DFT 3-67

Chapter 4 : Fast Fourier Transform 4-1 to 4-41

Syllabus :

Need of FFT, Radix-2 DIT-FFT algorithm, DIT-FFT Flow graph for N=4 and 8, Inverse FFT algorithm.

4.1 Introduction 4-1
4.2 Computational Complexity of DFT 4-1
4.3 Decimation In Time FFT (DIT-FFT) 4-2
4.4 Bit Reversal Format 4-6
4.4.1 Solved Examples on Bit Reversal Format 4-6
4.5 Decimation in Frequency FFT (DIT-FFT) 4-17
4.5.1 Solved Examples on DIT-FFT 4-20
4.6 Computational Complexity Comparison between DFT and FFT 4-30
4.6.1 Solved Examples on Computation of Inverse DFT using FFT Algorithms 4-31
4.6.2 Filtering using FFT Algorithms 4-37

Chapter 5 : Introduction to Image Processing 5-1 to 5-14

Syllabus :

Introduction to Digital Image, Digital Image Processing System, Image File Formats : BMP, TIFF and JPEG.

5.1 Introduction 5-1
5.2 What Do We Mean by Image Processing ? 5-1
5.3 Images 5-2
5.4 The Electromagnetic Spectrum 5-3
5.5 Units of Intensity 5-3
5.6 Basic Elements/Steps of an Image Processing System 5-4
5.7 Image Types 5-9
5.8 Image File Formats 5-12
5.9 Image Processing, Computer Graphics and Computer Vision 5-13

Chapter 6 : Sampling and Quantization 6-1 to 6-10

Syllabus :

Sampling and Quantization. Representation of Digital Image

6.1 Introduction 6-1
6.2 Sampling 6-1
6.3 Quantization 6-5
6.4 Isopreference Curves 6-8
6.5 Non-uniform Sampling 6-9
6.6 Physical Resolution 6-9
6.6.1 Solved Examples on Physical Resolution 6-9

Chapter 7 : Image Enhancement in Spatial Domain 7-1 to 7-35

Syllabus :

Gray Level Transformations, Zero Memory Point Operations, Neighborhood Processing, Spatial Filtering, Smoothing and Sharpening Filters, Median Filter.

7.1 Introduction 7-1
7.2 Spatial Domain Methods 7-1
7.3 Point Processing / Zero Memory Operation 7-2
7.4 Neighbourhood Processing 7-9
7.4.1 Low Pass Filtering (Smoothing) 7-10
7.4.2 Noise 7-10
7.4.3 Low Pass Averaging Filter 7-11
7.4.4 Low Pass Median Filtering 7-14
7.5 Highpass Filtering 7-16
7.6 High-Boost Filtering 7-18
7.7 Zooming 7-19
7.7.1 Replication 7-20
7.7.2 Linear Interpolation 7-21
7.8 Solved Examples 7-22
7.9 Comparison of Contrast Stretching and Thresholding 7-34

Chapter 8 : Histogram Modelling 8-1 to 8-20

Syllabus :

Histogram Processing, Histogram equalization.

8.1 Introduction 8-1
8.1.1 Mean and Standard Deviation of Histogram 8-3
8.2 Linear Stretching 8-3
8.2.1 Solved Examples on Linear Stretching 8-3
8.3 Histogram Equalization 8-6
8.3.1 Solved Examples on Histogram Equalization 8-7
8.4 Histogram Specification 8-12
8.4.1 Solved Example Histogram 8-12
8.5 Solved Examples on Histogram Modelling 8-13

Chapter 9 : Segmentation 9-1 to 9-23

Syllabus :

Segmentation based on Discontinuities (point, Line, Edge), Image Edge detection using Robert, Sobel, Previtt masks, Image Edge detection using Laplacian Mask. Connectivity

9.1 Introduction 9-1
9.2 Image Segmentation based on Discontinuities 9-2
9.2.1 Point Detection 9-2
9.2.2 Line Detection 9-2
9.2.3 Edge Detection 9-3
9.3 Detection of Edges 9-5
9.3.1 Computing the Gradient 9-5
9.4 Finding Gradients using Masks 9-6
9.4.1 Roberts Mask 9-7
9.4.2 Prewitts and Sobel Operators 9-8
9.4.3 Compass Operators 9-11
9.5 Image Segmentation using the Second Derivative - the Laplacian 9-13
9.5.1 Laplacian of Gaussian Operator 9-14
9.5.2 Canny Edge Detector 9-16
9.6 Connectivity 9-16
9.6.1 Solved Examples on Connectivity 9-18
9.7 Distance Transform 9-18
9.7.1 Solved Examples on Distance Transform 9-19
9.8 Additional Solved Examples 9-19

❑❑❑

Introduction to Digital Signal Processing

Syllabus :

Introduction to Digital Signal Processing, Sampling and Reconstruction, Standard DT Signals, Concept of Digital Frequency, Representation of DT signal using Standard DT Signals, Signal Manipulations(shifting, reversal, scaling, addition, multiplication).

1.1 Introduction

- A signal is defined as any physical quantity that varies with time, space or any other independent variable. Anything that carries some information can be called a signal.
- Examples of signals that we encounter frequently in our daily life are speech, music, image, video etc. Some other examples of signals are Electrocardiograms (ECG) which provides us information about the health of a person's heart, Seismic signals which are used to measure the intensity of an earthquake, AC power supply signals that we have at our homes and offices etc.
- Mathematically, we define a signal as a function of one or more independent variables. Consider a voltage signal and a pressure signal,

$$v(t) = 15t$$
$$p(t) = 10e^{-2t} \quad \ldots(1.1.1)$$

MATLAB program for displaying 1-dimensional signals

```
clc
clear all
for t=1:1:10
    v(t)=5*t;
    p(t)=10*exp(–0.2*t);
end
subplot (2, 1, 1)
plot (v)
xlabel ('Time')
ylabel ('voltage v(t)')
grid on
subplot (2, 1, 2)
plot (p)
xlabel ('Time')
ylabel ('Pressure p(t) ')
grid on
```

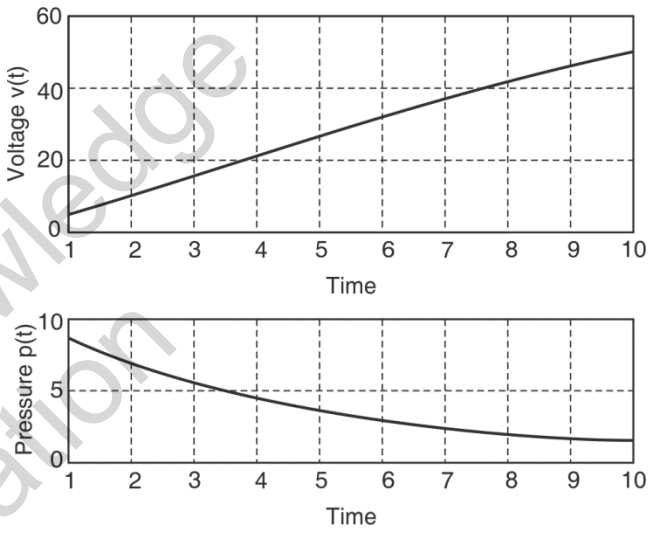


Fig. 1.1.1

- Both these signals in the given example depend on an independent variable t i.e. both voltage and pressure change with respect to time. Such signals are known as 1-Dimensional signals. In the Fig. 1.1.1, the x-axis is the time axis (independent variable) while the y-axis is the voltage (dependent variable).
- Another example of a signal is an image signal that we see on the computer monitor. A pixel on the computer could be represented as,

$$f(x, y) = 10x + 12y \quad \ldots(1.1.2)$$

Here, f depends on two independent variables x and y and hence are known as 2-Dimensional signals. This subject deals with only 1-Dimensional signals.

- The Equation (1.1.2) represent a class of signals that are precisely defined. However there are many such signals that cannot be directly represented by a mathematical formula. For example a speech signal shown in Fig. 1.1.1 cannot be represented by a mathematical equation of the kind shown in Equations (1.1.1) and (1.1.2).

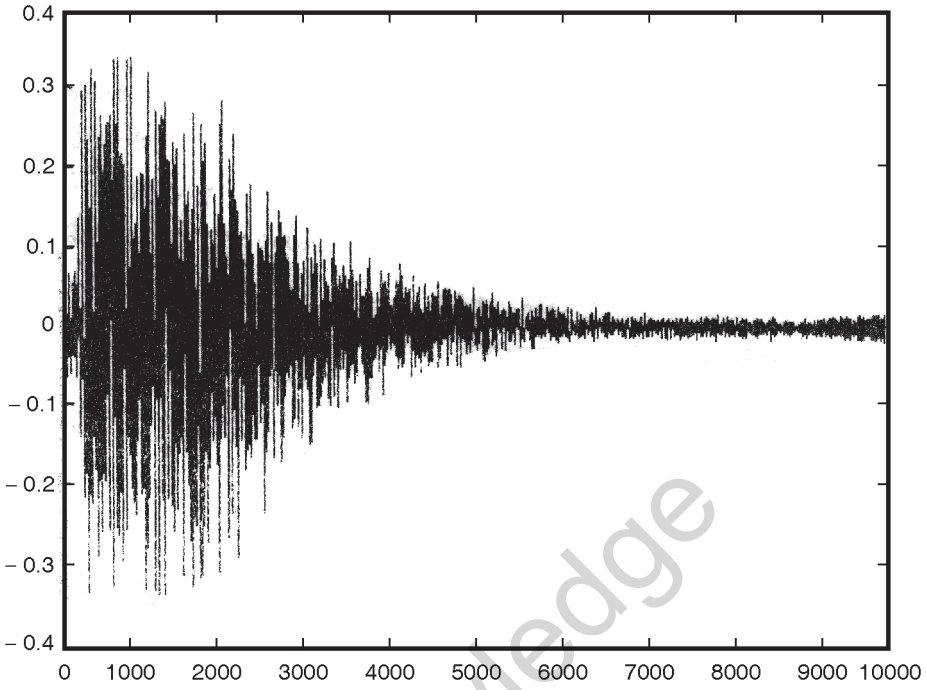


Fig. 1.1.2 : Speech signal

- Associated with all the above mentioned signals are the means by which these signals are generated. For example speech signals are generated by the vocal chords; the ECG signal is generated by the polarization and depolarization of the ventricles of the heat.
- Hence signals can be said to be generated by a **system.**
- A system is a device that produces a signal and also performs an operation on the signal to give the desired output.
- Suppose we want to amplify a signal. In this case the amplifier is a system. Similarly, if we want to filter out some noise, we pass it through a filter. In this case, the filter is also a system. Hence a system could be an interconnection of components that performs an operation on an input signal and produces an output signal.
- This production of a desired output signal from an input signal is known as **signal processing.**

  Let us define the three important terms discussed.

  a) Signal : A signal is defined as any physical quantity that varies with time, space or any other independent variable. Anything that carries some information can be called a signal.

  b) System : A system is defined as a device or a combination of devices which produces signals and also performs operation on signals to get the desired output.

  c) Signal processing : Production of a desired output signal from an input signal is known as signal processing.

This chapter provides an overview of signals, systems and signal processing methods.

1.1.1 Digital Signal Processing System

- Most of the signals that occur in nature are continuous (analog) i.e., they are a function of a continuous variable such as time (e.g. speech and ECG)
- In many cases, this continuous input signal can be directly processed in the continuous form itself.

  Let us take an example,
- Suppose we want to amplify our voice. The set up would be as shown in Fig. 1.1.3.

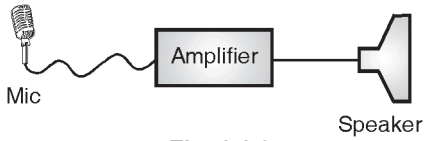


Fig. 1.1.3

- The amplifier which is made up resistors, capacitors, transistors etc., takes the input from the microphone which is a continuous signal and amplifies it to gives us a continuous output signal.
- Such an operation is called Analog Signal Processing. A general analog processing system is shown in Fig. 1.1.4. In this, the signal is directly processed in its continuous (analog) form.

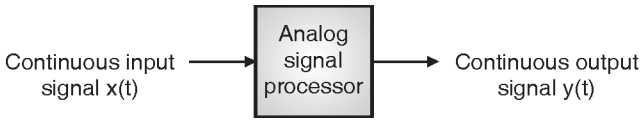


Fig. 1.1.4

- Digital signal processing, which this chapter is all about, provides an alternate method of processing the same signal. In this method we convert the continuous signal to a discrete time signal, also known as a digital signal, using an Analog to Digital Converter (ADC).
- The output of the ADC is fed to a digital processor (could also be a computer) which processes the discrete time signal. The output of the digital processor is again in digital form. This needs to be converted back to the continuous signal for any output device to understand it. This is done by using a Digital to Analog Converter (DAC). Let us take the same example of trying to amplify a voice signal. If we do it using a digital signal processor, we would have the following block diagram as shown in Fig. 1.1.5.

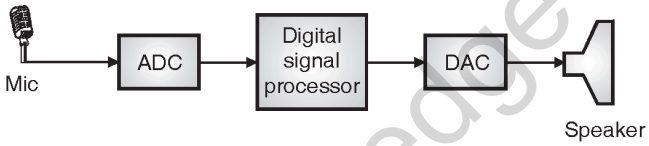


Fig. 1.1.5

In general we have,

Fig. 1.1.6

Here, $x(t) \rightarrow$ Continuous time input signal. $x(n) \rightarrow$ Discrete time input signal.

$y(n) \rightarrow$ Discrete time output signal. $y(t) \rightarrow$ Continuous time output signal.

1.1.2 Advantages of Digital Signal Processing over Analog Signal Processing

Though a DSP system has more blocks than an ASP system, there are many advantages of working in the digital domain. We will list down the advantages here.

1. Digital Signal Processing (DSP) systems are flexible i.e., they can be reconfigured simply by changing the program. Hence these systems are extremely versatile.
2. DSP systems can be easily replicated (duplicated) as they do not depend on component tolerances.
3. Accuracy also plays a pivotal role in determining the form of the signal processor. To design an analog system, we need components like resistors, capacitors and inductors. The tolerances of these components reduce accuracy of the analog system. On the other hand, DSP systems provide much better control of the accuracy.
4. Digital signal can be stored without deterioration. This is not true with analog signals.
5. Complex mathematical algorithms can be easily implemented using DSP systems while it is extremely difficult to build these algorithms using circuits.
6. Because of the use of software, DSP systems can be easily upgraded compared to Analog Signal Processing systems.
7. In most cases, DSP systems are cheaper compared to ASP systems.

Based on the above discussion we shall now form a table for a quick reference.

1.1.3 Difference between DSP and ASP

Table 1.1.1

Sr. No.	DSP	ASP
1.	More flexible	Less flexible
2.	Better accuracy	Less accuracy
3.	Digital signals can be easily stored.	It is difficult to store analog signals.

Sr. No.	DSP	ASP
4.	Digital systems can be easily replicated.	Tedious to replicate analog systems.
5.	Complex mathematical algorithms can be easily implemented.	It is extremely difficult to implement complex mathematical algorithms.
6.	Easily upgradable	Difficult to upgrade

1.1.4 Limitations of Digital Signal Processing

In spite of the obvious advantages of DSP systems over the ASP systems, it would be unfair if we do not highlight the disadvantages of the DSP systems.

1. A DSP system makes use of an ADC's and DAC's. These devices slow down the process. Hence a DSP system is not as fast as an ASP system.
2. A typical DSP chip contains thousands of transistors and hence the power consumption is more compared to ASP systems.
3. The most important limitation of DSP systems is that they are not suitable for signals which have a huge bandwidth. The ADC needs to sample the input signals at twice the bandwidth of the input signal to avoid aliasing. If the frequency content in the input signal is very larger, the ADC will need to sample it at least twice that frequency. There is a practical limitation on the speed of the ADC. Hence it is advisable to use ASP systems when the bandwidth of the input signal is very large.

With this background, we now move to discuss signals and systems.

We shall first study the various types of signals and then study the different systems.

1.2 Signals

Signals can be broadly classified as,

1. Continuous and discrete time signals.
2. Continuous valued and discrete valued signals.
3. Deterministic and random signals.

1.2.1 Continuous and Discrete Time Signals

– A signal that exists for all time in a given interval is said to be a continuous signal (also called analog signal). Most of the naturally occurring signals are continuous in nature. Some of the continuous signals are shown in Fig. 1.2.1.

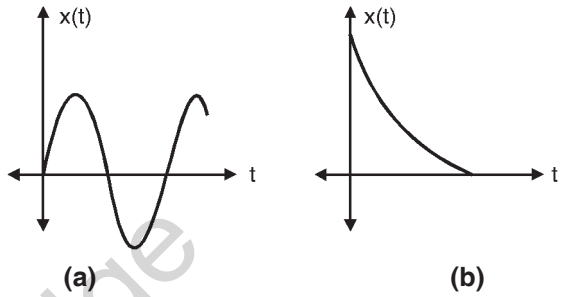


Fig. 1.2.1

These signals can be represented as,

a) $x(t) = A \sin(\Omega t + \theta)$

b) $x(t) = e^{-t/RC}$

– As can be seen in Fig.1.2.1, continuous time signals are defined for every instance of time.

– Some of the examples of continuous time signals are speech signals, vibrations, ECG, seismic waves etc.

– A discrete time signal is one that is defined only at specific values of time. Two discrete time signals are shown in Fig. 1.2.2.

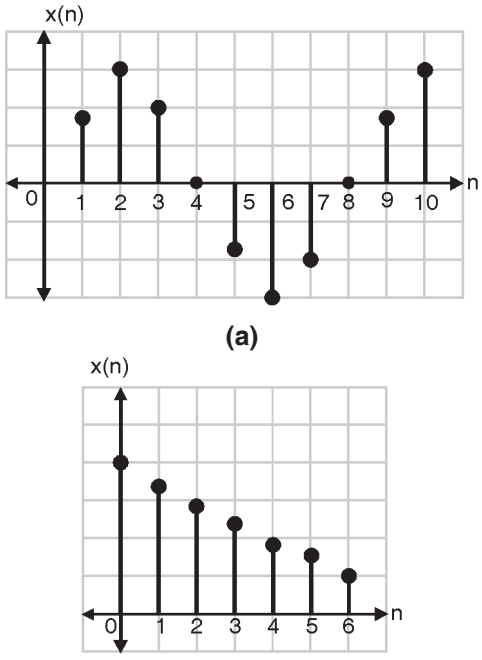


Fig. 1.2.2

- We do not know what is the value between 0 and 1 or between 1 and 2. Hence a discrete time signal exists only at specific values of time.
- A discrete time signal is obtained from a continuous time signal using the sampling operation.

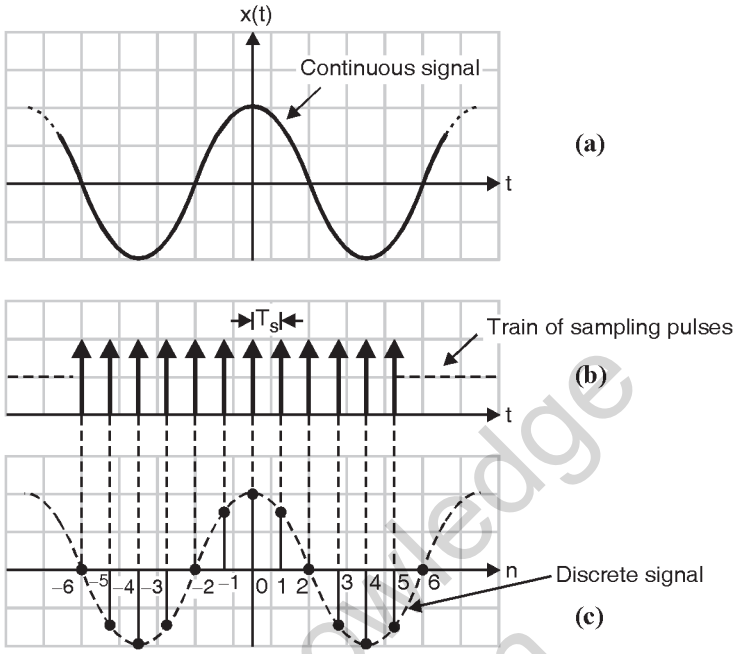


Fig. 1.2.3 : Sampling of a Continuous time signal to obtain a Discrete time signal

- Hence a discrete time signal is an approximation of the continuous time signal. To get a good approximation we need to take more samples in an interval i.e., reduce the sampling period (T_s) (increase sampling frequency). A typical discrete time signal can be represented by the equation.

$$x(n) = A \cos \omega n$$

We will discuss more about sampling in the next section.

- There are some signals that are inherently discrete in time and do not require the sampling of a continuous time signal. One such example is the stock market data. In this the value of a stock is studied at every hour or every minute. Nobody studies the stock value at every millisecond. Hence the data obtained would be discrete in time. Fig.1.2.4 represents a stock at different instances of time.

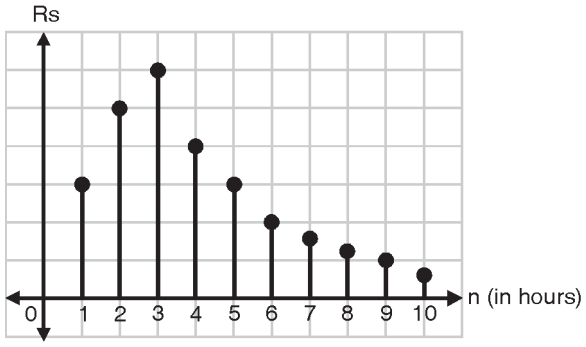


Fig. 1.2.4

1.2.2 Continuous Valued and Discrete Valued Signals

- The values of a continuous time or discrete time signal can be continuous or discrete. If the signal (whether continuous or discrete) can take all possible values, it is said to be continuous valued.
- On the other hand if the signal can take only a finite set of values, it is called a discrete valued signal. To put it simply, if the y-axis can be divided into infinite number of levels, it is called a continuous valued signal. If the y-axis can be divided into only a finite number of levels, it is called as a discrete valued signal. One such discrete valued signal is shown in Fig. 1.2.5. A discrete time signal having a set of discrete values is called a digital signal.

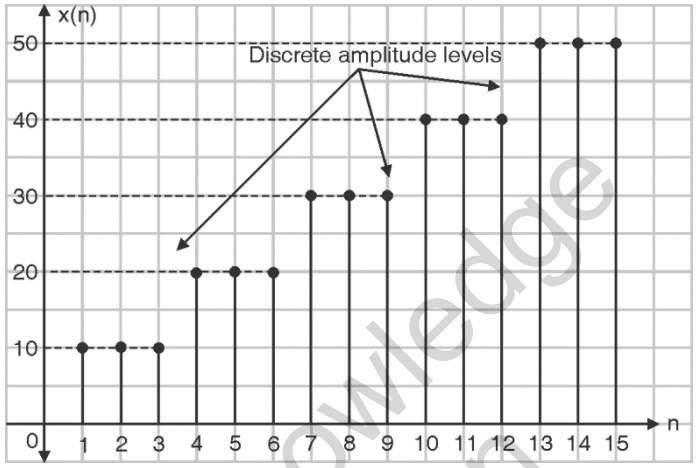


Fig. 1.2.5 : Discrete valued signal which is also discrete in nature

1.2.3 Deterministic and Random Signals

A signal which can be uniquely represented by a mathematical formula is called a deterministic signal. Since there is a mathematical formula, all past, present and future values are known and hence the name deterministic.

A signal which cannot be represented by a mathematical formula is called a Random signal. Such signals evolve over time and are unpredictable. Seismic waves, voice signals etc. are examples of random signals.

1.3 Concept of Frequency in the Continuous and Discrete Time Signals

1.3.1 Concept of Frequency in Continuous Time Signals

A continuous time domain signal is represented as,

$$x(t) = A \cos(2\pi Ft + \theta) \quad \ldots(1.3.1)$$

Where F is the frequency in Hz and has a range $-\infty \leq F \leq +\infty$.

One such continuous time signal is shown in Fig.1.3.1.

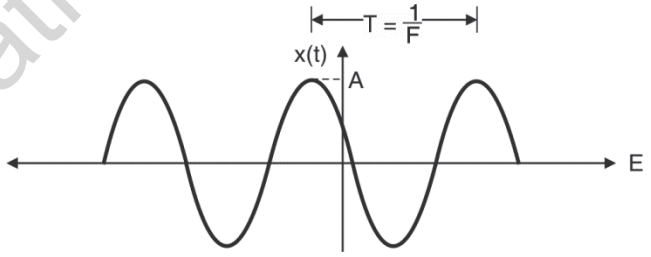


Fig. 1.3.1

This analog signal has three very nice properties associated with it.

1) For every fixed value of F, x(t) is periodic i.e., x(t + T) = x(t) (10 Hz, 20 Hz, 100 Hz … etc. are all periodic signals.
2) All continuous time signals with distinct frequencies are distinct i.e., 10 Hz looks different from 11 Hz which looks different from 100 Hz. Every frequency looks different.
3) Range of frequency, F is $-\infty \leq F \leq +\infty$.

Though frequency is a positive quantity, we consider negative frequency as well from Euler's formula.

Let us understand the concept of Negative frequencies

- A continuous time signal is represented as,

$$x(t) = A \cos(2\pi Ft + \theta)$$

We know $\cos\theta = \frac{e^{j\theta} + e^{-j\theta}}{2}$

$\therefore \quad x(t) = A\left[\frac{e^{j(2\pi Ft + \theta)} + e^{-j(2\pi Ft + \theta)}}{2}\right]$

$\therefore \quad x(t) = \frac{A}{2} e^{-j(2\pi Ft + \theta)} + \frac{A}{2} e^{-j(2\pi Ft + \theta)}$

- Hence x(t) can be considered to be made up of two phases moving in opposite directions.

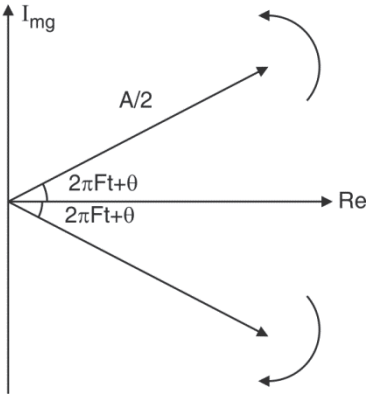


Fig. 1.3.2

- As time progresses one phasor moves in the counter clockwise direction while the other moves in the clockwise direction.
- A positive frequency corresponds to a counter clockwise uniform angular motion and negative frequency corresponds to negative clockwise uniform angular motion. Hence the range of Analog frequency is $-\infty \le F \le +\infty$.
- Remember, like negative numbers, negative frequencies do not exist and are used for mathematical convenience.
- When we pass this x(t), having the three properties just discussed, through an Analog to Digital Converter (ADC) we obtain x(n).
- We will now see if these 4 properties of x(t) are retained in x(n).

1.3.2 Concept of Frequency in Discrete Time Signals

A discrete time domain signal in represented as,

$$x(n) = A\cos(2\pi fn + \theta) \quad -\infty \le n \le +\infty \quad \ldots(1.3.2)$$

One such discrete time signal is shown in Fig. 1.3.3.

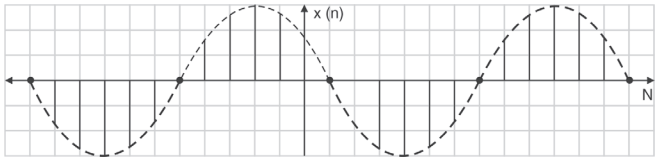


Fig. 1.3.3

1. A discrete time signal is periodic only if its frequency is a rational number.

Let us prove it

For periodicity x (n + N) = x(n)

Hence, for our x(n) to be periodic, we should have

A cos (2πf(n + N) + θ) = A cos (2πfn + θ)

$\therefore$ A cos (2πfn + 2πfN) + θ) = A cos (2πfn + θ)

This relation will be true only if

$$2\pi fN = 2k\pi$$

$$\therefore \quad f = \frac{k}{N}$$

Hence only if $f = \frac{k}{N}$ (Rational number)

A cos (2πf (n + N) + θ) = A cos (2π fn + θ)

Therefore x(t), which was always periodic, when passed through a ADC remains periodic only if ' f ' is a rational number.

2. Discrete time signals whose frequencies are separated by 2πk are identical.

Consider $x_1(n) = A\cos(2\pi f_0 n + \theta)$

We know $\omega_0 = 2\pi f_0$.

$\therefore \quad x_1(n) = A\cos(\omega_0 n + \theta)$

This signal has a frequency ω_0

Consider another discrete time signal $x_2(n)$ which as a frequency $(\omega_0 + 2\pi)$

$$x_2(n) = A\cos((\omega_0 n + 2\pi)n + \theta)$$

$$\therefore \quad x_2(n) = A\cos(\omega_0 n + 2\pi n + \theta)$$

$$= A\cos(\omega_0 n + \theta)$$

$$= x_1(n)$$

Hence $x_2(n)$ looks identical to $x_1(n)$ even though $x_1(n)$ has a frequency ω_0 and $x_2(n)$ has a frequency $(\omega_0 + 2\pi)$.

This can be generalized to state that all time signals

$$x_k(n) = A\cos(\omega_k n + \theta) \qquad k = 0, 1, 2, 3$$

where, $\omega_k = \omega_0 + 2\pi k \qquad -\pi \le \omega_0 \le +\pi$

are identical

Therefore while all x(t)'s were distinct, their discrete time domain counterparts, x(n)'s are not

3. The range of discrete frequency is $-\frac{1}{2} \le f \le +\frac{1}{2}$

From point number (2) we write,

All discrete time signals

$$x_k(n) = A \cos(\omega_k n + \theta); \qquad k = 0, 1, 2, 3\ldots$$

where, $\omega_k = \omega_0 + 2K\pi$; $-\pi \le \omega_0 \le \pi$

are identical.

This means only those frequencies which are in the range $-\pi \le \omega_0 \le +\pi$ are distinct.

$$-\pi \le \omega \le +\pi$$

$\because \omega = 2\pi f$, we can write

$$-\pi \le 2\pi f \le +\pi$$

$$\therefore \quad -\frac{1}{2} \le f \le +\frac{1}{2}$$

Hence the distinct range of discrete time signals is only $-\frac{1}{2} \le f \le +\frac{1}{2}$

Therefore while x(t)'s had a range of $-\infty \le F \le +\infty$, their discrete time domain counterparts, x(n)'s, have a range of $\frac{1}{2} \le f \le +\frac{1}{2}$

Hence when x(t), which has a range $-\infty \le F \le +\infty$, is passed through a ADC, it becomes x(n) and has a frequency range of $-\frac{1}{2} \le f \le +\frac{1}{2}$

Note : The range of digital frequency is,

$$-\pi \le \omega \le +\pi \quad \text{OR} \quad -\frac{1}{2} \le f \le +\frac{1}{2}$$

How did a large range $-\infty \le F \le +\infty$ get reduced to $-\frac{1}{2} \le f \le +\frac{1}{2}$? What does this mean ? How can discrete time signals have such low frequencies ?

These questions will be answered if we know what 'f' is.

This is what the next section is all about.

1.3.3 Converting a Analog Signal to a Discrete Time Signal

- The process of converting an analog signal (continuous signal) to a discrete time signal is known as sampling.
- Sampling converts x(t) to x(n) and is performed by an Analog to Digital Converter (ADC).

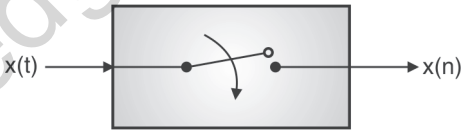


Fig. 1.3.4

- The sampling process can be visualized as a simple switch which opens and closes at fixed intervals of time, T_s.

Consider a switch which closes every T_s sec.

Let, x(t) be a pure sinusoidal function given by the formula.

$$x(t) = A \cos(2\pi F t + \theta) \qquad \ldots (1.3.3)$$

Here, F is the frequency in Hz.

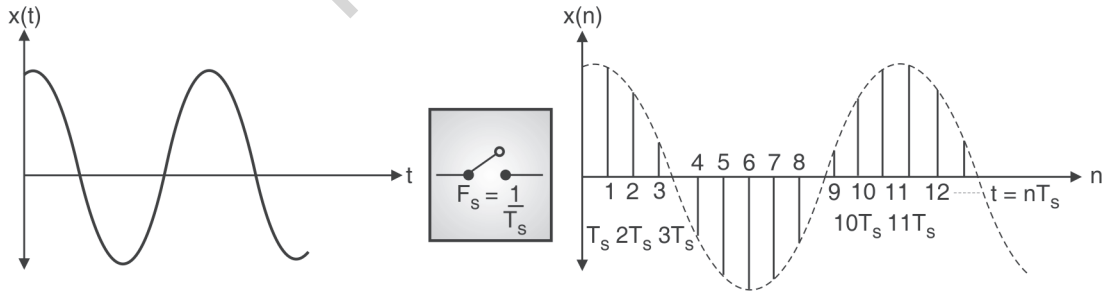


Fig. 1.3.5

- Since the switch closes every T seconds, we get the output only at fixed intervals of time.

  The sampling rate $F_s = \frac{1}{T_s}$

- From Fig. 1.3.5, it is clear that the x-axis which represents time gets replaced by nT_s in the discrete domain.

  $\therefore$ $t = nT_s$ in the discrete domain

- We can have thus convert an analog signal to a discrete time signal by replacing t=nT, where T is the Sampling period.

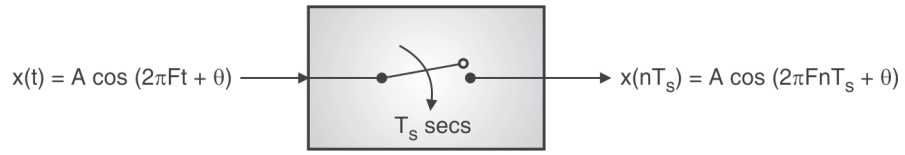


Fig.1.3.6

– Hence we obtain a digital signal that is represented as,

$$x(nT) = x(n) = A \cos (2 \pi F n T_s + \theta) \quad \ldots(1.3.4)$$

– We know that T_s is the sampling period i.e, $T_s = \frac{1}{F_s}$, where, F_s is the sampling frequency.

– Substituting $T_s = \frac{1}{F_s}$ in Equation (1.3.4) we get,

$$\therefore \quad x(n) = A \cos \left(2\pi \frac{F}{F_s} n + \theta \right) \quad \ldots(1.3.5)$$

– Hence, an Analog signal $x(t) = A \cos (2\pi F t + \theta)$ can be converted to a digital signal by replacing $t = nT = \frac{n}{F_s}$.

$$\text{Let} \quad f = \frac{F}{F_s}$$

– Substituting this value of f in Equation (1.3.5), we get the standard representation of a discrete time signal.

$$x(n) = A \cos (2\pi fn + \theta) \quad \ldots(1.3.6)$$

$$\text{where,} \quad f = \frac{F}{F_s}$$

We know that the range of f is $-\frac{1}{2} \le f \le \frac{1}{2}$

$$\text{Since} \quad f = \frac{F}{F_s}$$

$$\therefore \quad -\frac{1}{2} \le \frac{F}{F_s} \le \frac{1}{2}$$

$$\therefore \quad \frac{-F_s}{2} \le F \le +\frac{F_s}{2}$$

Hence $-\frac{1}{2} \le f \le +\frac{1}{2}$ basically means $\frac{-F_s}{2} \le F \le +\frac{F_s}{2}$

– To understand this range we consider the positive frequency.

$$F \le \frac{F_s}{2}$$

$$\text{i.e.,} \quad F_s \ge 2F \quad \ldots(1.3.7)$$

Here F is the frequency of input signal (in Hz) while F_s is the sampling frequency (in Hz).

– From Equation (1.3.7) we realize that sampling frequency F_s should be greater than or equal to twice the maximum frequency of the signal being sampled.

Only if $F_s \ge 2F$, will f remain in the distinct range of $-\frac{1}{2} \le f \le +\frac{1}{2}$

This is known as the Nyquist criteria.

If $F_s \le 2F$, f will no longer remain in the distinct range and will give rise to aliasing.

Therefore a range of $-\frac{1}{2} \le f \le +\frac{1}{2}$ implies that $F_s \ge 2F$

1.4 Aliasing and Nyquist Rate

– Aliasing is a phenomenon where high frequencies look identical to low frequencies because of low sampling frequency.

– In order to represent a continuous signal faithfully, we need to take as many samples as possible. This is done by an Analog to Digital Converter having a high sampling.

What is the ideal sampling required ?

– Sampling theorem was introduced in 1949 by Shannon and is also called as Shannon sampling's theorem.

– It is stated as *'A continuous time signal x(t) can be completely represented in its sampled form and recovered back from the sampled form if the sampling frequency $F_s \ge 2F$ where F is the maximum frequency in the continuous signal'.*

– A minimum sampling rate of 2F is called the Nyquist rate. Low sampling frequency gives rise to aliasing.

– We will see the effect of aliasing by solving a couple of examples.

Solved Example

Ex. 1.4.1 : Consider the give continuous time signals having frequencies 5 Hz, 10 Hz, 15 Hz, 45 Hz, 50 Hz and 130 Hz.

$x_1 (t) = 5 \cos 2 \pi (5) t$

$x_2 (t) = 5 \cos 2\pi (10) t$

$x_3 (t) = 5 \cos 2\pi (15) t$

$x_4 (t) = 5 \cos 2\pi (45) t$

$x_5 (t) = 5 \cos 2\pi (50) t$

$x_6 (t) = 5 \cos 2\pi (130) t$

Obtain the discrete time signals by using a sampling frequency of 40 Hz.

Soln. :

We will first solve this example mechanically and then analyze the results obtained.

The given signals have the following frequencies.

$x_1(t) = 5$ Hz, $x_2(t) = 10$ Hz, $x_3(t) = 15$ Hz, $x_4(t) = 45$ Hz, $x_5(t) = 50$ Hz and $x_6(t) = 130$ Hz,

A continuous time signal is given by the equation,

$$x(t) = A\cos(2\pi Ft + \theta)$$

We convert this to a discrete time signal by replacing $t = nT_s = \frac{n}{F_s}$

$$\therefore \quad x(n) = A\cos\left(2\pi\frac{F}{F_s}n + \theta\right) \quad \ldots(1)$$

$$\therefore \quad x(n) = A(\cos 2\pi fn + \theta) \quad \ldots(2)$$

Here $\frac{F}{F_s} = f$

In our case $F_s = 40$ Hz

We will convert each of the given continuous signals into discrete time signals using Equation (1).

$\mathbf{x_1(t)} \rightarrow x_1(t) = 5\cos 2\pi\, 5t$

$$\therefore \quad x_1(n) = 5\cos 2\pi\left(\frac{5}{40}\right)n$$

$$\therefore \quad x_1(n) = 5\cos 2\pi\left(\frac{1}{8}\right)n$$

Comparing this with Equation (1) we have

$$f_1 = \frac{1}{8} = 0.125 \quad \ldots(3)$$

$\mathbf{x_2(t)} \rightarrow x_2(t) = 5\cos 2\pi 10t$

$$\therefore \quad x_2(n) = 5\cos 2\pi\left(\frac{10}{40}\right)n$$

$$\therefore \quad x_2(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$$

$$\therefore \quad f_2 = \frac{1}{4} = 0.25 \quad \ldots(4)$$

$\mathbf{x_3(t)} \rightarrow x_3(t) = 5\cos 2\pi 15t$

$$\therefore \quad x_3(n) = 5\cos 2\pi\left(\frac{15}{40}\right)n$$

$$\therefore \quad x_3(n) = 5\cos 2\pi\left(\frac{3}{8}\right)n$$

$$\therefore \quad f_3 = \frac{3}{8} = 0.325 \quad \ldots(5)$$

$\mathbf{x_4(t)} \rightarrow x_4(t) = 5\cos 2\pi 45t$

$$\therefore \quad x_4(n) = 5\cos 2\pi\left(\frac{45}{40}\right)n$$

$$\therefore \quad x_4(n) = 5\cos 2\pi\left(\frac{9}{8}\right)n$$

Since $\frac{9}{8} > 1$, it can be written as,

$$x_4(n) = 5\cos 2\pi\left(1 + \frac{1}{8}\right)n \qquad \left(\because \frac{9}{8} = 1 + \frac{1}{8}\right)$$

$$\therefore \quad x_4(n) = 5\cos\left(2\pi + 2\pi\cdot\frac{1}{8}\right)n$$

Since $\cos(2\pi + \theta) = \cos\theta$ we write,

$$x_4(n) = 5\cos 2\pi\left(\frac{1}{8}\right)n$$

$$\therefore \quad f_4 = \frac{1}{8} = 0.125 \quad \ldots(6)$$

$\mathbf{x_5(t)} \rightarrow x_5(t) = 5\cos 2\pi\, 50t$

$$\therefore \quad x_5(n) = 5\cos 2\pi\left(\frac{50}{40}\right)n$$

$$\therefore \quad x_5(n) = 5\cos 2\pi\left(\frac{5}{4}\right)n$$

Since $\frac{5}{4} > 1$, it can be written as,

$$x_5(n) = 5\cos 2\pi\left(1 + \frac{1}{4}\right)n \qquad \left(\because \frac{5}{4} = 1 + \frac{1}{4}\right)$$

$$\therefore \quad x_5(n) = 5\cos\left(2\pi + 2\pi\cdot\frac{1}{4}\right)n$$

Since $\cos(2\pi + \theta) = \cos\theta$ we write,

$$x_5(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$$

$$\therefore \quad f_5 = \frac{1}{4} = 0.25 \quad \ldots(7)$$

$\mathbf{x_6(t)} \rightarrow x_6(t) = 5\cos 2\pi 130t$

$$\therefore \quad x_6(n) = 5\cos 2\pi\left(\frac{130}{40}\right)n = 5\cos 2\pi\left(\frac{13}{4}\right)n$$

Since $\frac{13}{4} > 1$, it can be written as,

$$= 5\cos 2\pi\left(3 + \frac{1}{4}\right)n \qquad \left(\because \frac{5}{4} = 3 + \frac{1}{4}\right)$$

$$\therefore \quad x_6(n) = 5\cos\left(6\pi + 2\pi\cdot\frac{1}{4}\right)n$$

Since $\cos(6\pi + \theta) = \cos\theta$ we write,

$$x_6(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$$

$$\therefore \quad f_6 = \frac{1}{4} = 0.25 \quad \ldots(8)$$

We have thus been successful in converting the 6 continuous time signals into discrete time signals.

Let us now analyze what we have got. We form a table of the results obtained.

Sr. No.	Continuous time signal	Value of analog frequency	Is the Nyquist condition satisfied ? $F_s \geq 2F$ (Fs = 40 Hz)	Discrete time signal	Value of Discrete frequency(f)	Is f distinct
1.	$x_1(t) = 5\cos 2\pi 5t$	$F_1 = 5$ Hz	Yes	$x_1(n) = 5\cos 2\pi\left(\frac{1}{8}\right)n$	$f_1 = \frac{1}{8} = 0.125$	Yes
2.	$x_2(t) = 5\cos 2\pi 10t$	$F_2 = 10$ Hz	Yes	$x_2(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$	$f_2 = \frac{1}{4} = 0.25$	Yes
3.	$x_3(t) = 5\cos 2\pi 15t$	$F_3 = 15$ Hz	Yes	$x_3(n) = 5\cos 2\pi\left(\frac{3}{8}\right)n$	$f_3 = \frac{3}{8} = 0.375$	Yes
4.	$x_4(t) = 5\cos 2\pi 45t$	$F_4 = 45$ Hz	No	$x_4(n) = 5\cos 2\pi\left(\frac{1}{8}\right)n$	$f_4 = \frac{1}{8} = 0.125$	No (f_4 same as f_1)
5.	$x_5(t) = 5\cos 2\pi 50t$	$F_5 = 50$ Hz	No	$x_5(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$	$f_5 = \frac{1}{4} = 0.25$	No (f_5 same as f_2)
6.	$x_6(t) = 5\cos 2\pi 130t$	$F_6 = 130$ Hz	No	$x_6(n) = 5\cos 2\pi\left(\frac{1}{4}\right)n$	$f_6 = \frac{1}{4} = 0.25$	No (f_6 same as f_2)

We observe that, $x_4(n)$ looks identical to $x_1(n)$. Similarly $x_5(n)$ and $x_6(n)$ are identical to $x_2(n)$. This is known as Aliasing. It is important to note that the continuous time signals $x_1(t)$, $x_2(t)$, $x_3(t)$, $x_4(t)$, $x_5(t)$ and $x_6(t)$ were all distinct having different frequencies. However on sampling, $x_4(n)$ becomes identical to $x_1(n)$. Similarly $x_5(n)$ and $x_6(n)$ resemble $x_2(n)$.

On performing digital to analog conversion again, we would not be able to distinguish between $x_4(t)$ and $x_1(t)$ similarly we would not be able to distinguish between $x_5(t)$, $x_6(t)$ and $x_2(t)$.

In other words, while converting the discrete time signal back to a continuous time signal, 45 Hz signal ($x_4(t)$) will look the same as a 5 Hz signal ($x_1(t)$). Similarly 50 Hz signal($x_5(t)$) and 130 Hz signal ($x_6(t)$) will look the same as 10 Hz signal ($x_2(t)$). This distortion has taken place because the Nyquist condition was not satisfied while sampling $x_4(t)$, $x_5(n)$ and $x_6(n)$. (Look at the 4th column on the table).

This phenomena of higher frequencies resembling low frequencies is known as aliasing. A simple way to avoid this is to ensure that the sampling frequency F_s is always greater than or equal to twice the maximum frequency that needs to be sampled.

$$F_s \geq 2F_{max}$$

In the above example, $x_6(t)$ had the maximum frequency of 130 Hz. To avoid aliasing, we should have sampled all the signals at a sampling frequency of $F_s \geq 2 \times 130$ Hz

$\therefore F_s \geq 260$ Hz

Try solving the earlier example with F_s = 300 Hz and you will observe that $x_1(n)$, $x_2(n)$, $x_3(n)$, $x_4(n)$, $x_5(n)$ and $x_6(n)$ are all distinct.

Given below is the MATLAB code for the above example.

Aliasing

```
clc
clear all

Fs=40; %sampling Frequency
F1=5;F2=10;F3=15;F4=45;F5=50;F6=130 ;
% various frequencies present
For n=1:1:50;
x1(n)=5*cos(2*pi*(F1/Fs)*n);
x2(n)=5*cos(2*pi*(F2/Fs)*n);
x3(n)=5*cos(2*pi*(F3/Fs)*n);
x4(n)=5*cos(2*pi*(F4/Fs)*n);
x5(n)=5*cos(2*pi*(F5/Fs)*n);
x6(n)=5*cos(2*pi*(F6/Fs)*n);
```

```
end
subplot(6,1,1)
stem(x1)
ylabel('x1(n)')
subplot (6,1,2)
stem(x2)
ylabel ('x2(n)')
subplot(6,1,3)
stem(x3)
ylabel('x3(n)')
subplot(6,1,4)
stem(x4)
ylabel('x4(n)')
subplot(6,1,5)
stem(x5)
ylabel('x5(n)')
subplot(6,1,6)
stem(x6)
ylael('x6(n)')
```

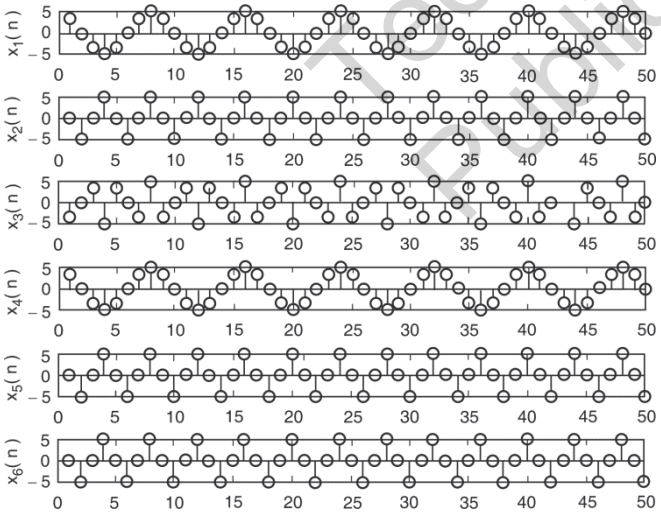


Fig. P. 1.4.1

We observe that, $x_4(n)$ is identical to $x_1(n)$ where as $x_5(n)$ and $x_6(n)$ are identical to $x_2(n)$.

If we simply change the sampling frequency (F_s) to 300 Hz in the program, each discrete time signal will be distinct. This is shown in Fig. P.1.4.1(a).

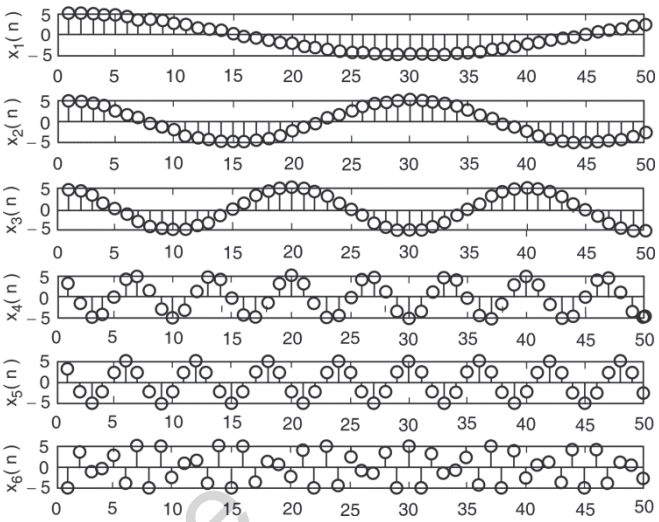


Fig. P.1.4.1(a)

Ex.1.4.2 : Consider the analog sinusoidal signal

$x(t) = 5 \sin (500 \pi t)$

If the signal is sampled at F_s = 1500 Hz,

(i) What is the discrete time signal obtained after sampling.

(ii) Find the frequency of discrete time signal.

(iii) If sampling frequency F_s = 300 Hz then obtain discrete time signal.

Soln. :

Given : $x(t) = 5 \sin (500 \pi t)$

(i) F_s = 1500 Hz

Now discrete time signal is obtained by replacing

$$t = nT_s = \frac{n}{F_s}$$

$$\therefore \quad x(n) = 5 \sin \left(\frac{500 \pi n}{1500}\right)$$

$$\therefore \quad x(n) = 5 \sin \left(\frac{\pi}{3}\right) n \qquad \ldots(1)$$

This is the discrete time signal obtained after sampling.

(ii) The standard equation of the Discrete time signal is,

$$x(n) = A \sin \omega_n \qquad \ldots(2)$$

Comparing Equations (1) and (2) we get,

$$\omega = \frac{\pi}{3}$$

$$\because \quad \omega = 2\pi f$$

$$\therefore \quad f = \frac{1}{6}$$

Hence the frequency of discrete time signal is $\omega = \frac{\pi}{3}$ or $f = \frac{1}{6}$

(iii) Here F_s = 300 Hz

Discrete time signal is obtained by replacing

$$t = nT_s = \frac{n}{F_s}$$

$$x(n) = 5 \sin\left(\frac{500\,\pi \cdot n}{300}\right)$$

$$\therefore \quad x(n) = 5 \sin\left(\frac{5\pi n}{3}\right)$$

Now $$\frac{5\pi}{3} = \left(2\pi - \frac{\pi}{3}\right)$$

$$\therefore \quad x(n) = 5 \sin\left(2\pi - \frac{\pi}{3}\right) n$$

$$\therefore \quad x(n) = 5 \sin\left(-\frac{\pi}{3} n\right)$$

$$x(n) = -5 \sin\left(\frac{\pi}{3}\right) n$$

This is the discrete time signal.

1.4.1 Anti Aliasing Filter

The minimum sampling frequency which is essential is 2F. where F is the maximum frequency. This $F_s = 2F_{max}$ is called the Nyquist rate.

Note : Most of the naturally occurring signals are not band limited. This means that they have infinite frequencies present in them (The extremely high frequencies have very small amplitudes, but they exist).

Nyquist criteria of $F_s \geq 2\ F_{max}$ will never be satisfied in such cases and we will always encounter aliasing. These Band unlimited signals are converted to Band limited signals by passing them through a low pass filter. The sole purpose of this low pass filter is to remove very high frequencies thereby making the signal band limited and hence removing the affects of aliasing. These low pass filters are called anti-aliasing filters.

1.5 Basic Elements of a DSP System

We have discussed the basics DSP systems in this chapter. We had drawn a basic block diagram of a DSP system in Fig. 1.5.1. We will now draw a modified block diagram of a DSP system and explain each block.

Fig.1.5.1

1. **Input signal :** A signal is defined as any physical quantity that varies with time, space or any other independent variable. Anything that carries some information can be called a signal.
2. **Anti-Aliasing Filter :** Most of the naturally occurring signals are not band limited. This means that they have infinite frequencies present in them. Anti-Aliasing filters are basically low pass filters which are used to remove the high frequencies from the input signal thereby making them band limited.
3. **Analog to Digital Converter (ADC) :** As the name suggests, ADC converts a continuous time signal x(t) to a discrete time signal x(n).
4. **Digital Signal Processor (DSP) :** This is the heart of the DSP system. It processes the discrete time input signal x(n) and gives us the required discrete time output signal y(n). One could either use a computer or dedicated DSP's manufactured by companies such as Motorola and Texas Instruments.
5. **Digital to Analog Converter (DAC) :** The output of the DSP is in digital form but most of the output devices only work in continuous time signals. DAC converts a discrete time signal y(n) to a continuous time signal y(t) which can be then fed to the out device.
6. **Reconstruction filter:** The output of the DAC might contain high frequencies due to the interpolation of sinc functions. These spurious high frequencies are eliminated using the reconstruction filter which is basically a low pass filter.

We will now solve a few examples to understand the sampling concepts.

1.5.1 Solved Examples

Ex. 1.5.1 : An analog signal is given as :

$x(t) = \sin(10\pi t) + 2\sin(20\pi t) + 2\cos(30\pi t)$.

(i) What is the Nyquist rate of this signal ?

(ii) If the signal is sampled with sampling frequency of 20 Hz, what is the discrete time signal obtained after sampling ?

Soln. :

Given :

$x(t) = \sin(10\pi t) + 2\sin(20\pi t) + 2\cos(30\pi t)$

(i) This signal consists of following frequencies,

$2\pi F_1 = 10\pi,\ 2\pi F_2 = 20\pi$ and $2\pi F_3 = 30\pi$

$\therefore F_1 = 5$ Hz, $F_2 = 10$ Hz, and $F_3 = 15$ Hz

Thus the maximum frequency present in the signal is,

$F_{max} = W = 15$ Hz

$\therefore$ Nyquist rate $= 2F_{max} = 30$ Hz

(ii) The discrete time signal is obtained by substituting

$t = \frac{n}{F_s} = \frac{n}{20}$ in the given equation.

$$\therefore x(n) = \sin\left(10\pi\frac{n}{20}\right) + 2\sin\left(20\pi\frac{n}{20}\right) + 2\cos\left(30\pi\frac{n}{20}\right)$$

$$= \sin\left(\frac{\pi}{2}\right)n + 2\sin(\pi)n + 2\cos\left(\frac{3\pi}{2}\right)n$$

but, $\sin(\pi n) = 0$

also $\frac{3\pi}{2} = 2\pi - \frac{\pi}{2}$

$$\therefore x(n) = \sin\left(\frac{\pi}{2}\right)n + 2\cos\left(2\pi - \frac{\pi}{2}\right)n$$

$$\therefore x(n) = \sin\left(\frac{\pi}{2}\right)n + \cos\left(\frac{-\pi}{2}\right)n$$

but,

$$\cos\left(\frac{\pi}{2}\right)n = \cos\left(\frac{-\pi}{2}\right)n = 0$$

$$\therefore x(n) = \sin\left(\frac{\pi}{2}\right)n$$

Ex.1.5.2 : A signal $x(t) = \sin(\omega t)$ of frequency 50 Hz is sampled using a sampling frequency of 80 Hz. Obtain the recovered signal if ideal reconstruction is used.

Soln. : Given :

$x(t) = \sin \omega t$ of frequency 50 Hz

$\therefore x(t) = \sin(2\pi \times 50t)$

Now, F_s = Sampling frequency = 80 Hz

Thus discrete signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{80}$$

$$\therefore x(n) = \sin\left(2\pi 50\frac{n}{80}\right) = \sin\left(\frac{5\pi}{4}\right)n$$

Now, $\frac{5\pi}{4} = \left(2\pi - \frac{3\pi}{4}\right)$

$$\therefore x(n) = \sin\left(2\pi - \frac{3\pi}{4}\right)$$

$$x(n) = -\sin\left(\frac{3\pi}{4}\right)n$$

Since Ideal interpolation is used, analog signal is obtained by replacing n by $t F_s$.

$\therefore n = 80t$

$$\therefore x(t) = -\sin\left(\frac{3\pi}{4}\ 80\ t\right)$$

$$\therefore x(t) = -\sin(60\pi t)$$

Hence a 50 Hz signal gets reconstructed as a 60 Hz signal.

Ex. 1.5.3 : What should be the sampling frequency to avoid aliasing for an analog signal represented as :

$x(t) = \cos(150\pi t) + 2\sin(300\pi t) - 4\cos(600\pi t)$

Obtain the discrete time signal if this sequence is sampled at $F_S = 400$ Hz.

Does aliasing occur ? If yes, calculate the aliased frequencies from the original frequencies.

Soln. :

Given :

$$x(t) = \cos(150\pi t) + 2\sin(300\pi t) - 4\cos(600\pi t) \quad ...(1)$$

Here $2\pi F_1 t = 150\pi t$, $2\pi F_2 t = 300\pi t$ and $2\pi F_3 t = 600\pi t$

$\therefore F_1 = 75$ Hz, $F_2 = 150$ Hz, and $F_3 = 300$ Hz

$\therefore$ Maximum frequency $F_{max} = 300$ Hz

Thus the sampling frequency, required to avoid aliasing is,

$F_s \geq 2F_{max}$

$\therefore F_s \geq 600$ Hz

The given signal is sampled at $F_s = 400$ Hz.

Thus discrete signal is obtained by substituting

$t = \frac{n}{F_s} = \frac{n}{400}$ in Equation (1)

$\therefore t = \frac{n}{400}$

$$\therefore \quad x(n) = \cos\left(150\pi\frac{n}{400}\right) + 2\sin\left(300\pi\frac{n}{400}\right) - 4\cos\left(600\pi\frac{n}{400}\right)$$

$$x(n) = \cos\left(\frac{3\pi}{8}\right)n + 2\sin\left(\frac{3\pi}{4}\right)n - 4\cos\left(\frac{3\pi}{2}\right)n \quad ...(2)$$

The third terms of $\frac{3\pi}{2}$ can be written as $-4\cos\left(2\pi-\frac{\pi}{2}\right)n$.

$$= -4\cos\left(-\frac{\pi}{2}\right)n$$

but, $\cos\left(\frac{\pi}{2}\right)n = \cos\left(\frac{-\pi}{2}\right)n = 0$

$$\therefore \quad x(n) = \cos\left(\frac{3\pi}{8}\right)n + 2\sin\left(\frac{3\pi}{4}\right)n$$

Hence aliasing does occur.

Ex.1.5.4 : An analog signal contains frequencies upto 10 kHz

(i) What range of sampling frequencies allows exact reconstruction of this signal from its samples.

(ii) If the signal is sampled with a sampling frequency F_s = 8 kHz

What is the folding frequency ?

Examine what happens to the frequency F_1 = 5 kHz

Examine what happens to the frequency F_2 = 9 kHz

Soln. :

(i) From the Nyquist criteria we know that,

$$F_s \geq 2W$$

$$\therefore \quad F_s \geq 2 \times 10$$

$$\therefore \quad F_s \geq 20 \text{ kHz}$$

This is the range of sampling frequencies which allows exact reconstruction.

(ii) The signal is sampled at 8 kHz.

Hence the folding frequency is,

$$F_{fold} = \frac{F_s}{2} = \frac{8 \text{ kHz}}{2} = 4 \text{ kHz}$$

$$F_1 = 5 \text{ kHz (given)}$$

Since F_{fold} is less than F_1 aliasing takes place. 5 KHz will look like 1kHz (5 – 4) = 1.

$$F_2 = 9 \text{ kHz (given)}$$

Since F_{fold} is less than F_1 aliasing takes place. 9KHz will look like 1 kHz (9 – (4+4)) = 1 kHz.

Ex. 1.5.5 : Consider the analog signal

$x_a(t) = 3\cos 2000\pi t + 5\sin 6000\pi t + 10\cos 12{,}000\pi t$.

(i) What is Nyquist rate for this signal ?

(ii) If this analog signal is sampled at

F_s = 5000 samples/sec, what is the discrete-time signal obtained after sampling ?

(iii) What is the analog signal $y_a(t)$ we can reconstruct from the samples if we use ideal interpolation ?

Soln. :

Given :

$$x_a(t) = 3\cos 2000\pi t + 5\sin 6000\pi t + 10\cos 12000\pi t \quad ...(1)$$

(i) The given signal consists of following frequencies,

$$2\pi F_1 = 2000\pi t, \; 2\pi F_2 = 6000\pi t, \; 2\pi F_3 = 12000\pi t$$

$$\therefore \; F_1 = 1000 \text{ Hz}, F_2 = 3000 \text{ Hz and } F_3 = 6000 \text{ Hz}$$

Thus $F_{max} = 6000 \text{ Hz} = 6 \text{ kHz}$

$\therefore$ Nyquist rate = $2F_{max} = 12$ kHz

(ii) A discrete signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{5000} \text{ in Equation (1)}$$

$$\therefore \quad x(n) = 3\cos\left(\frac{2000\,\pi\, n}{5000}\right) + 5\sin\left(\frac{6000\,\pi n}{5000}\right) + 10\cos\left(\frac{12000\,\pi n}{5000}\right)$$

$$\therefore \quad x(n) = 3\cos\left(\frac{2\pi}{5}\right)n + 5\sin\left(\frac{6\pi}{5}\right)n + 10\cos\left(\frac{12\pi}{5}\right)n$$

The second term $\left(\frac{6\pi}{5}\right)$ can be written as,

$$\left(\frac{6\pi}{5}\right) = \left(2\pi - \frac{4\pi}{5}\right)$$

$$\therefore \quad 5\sin\left(\frac{6\pi}{5}\right)n = 5\sin\left(2\pi - \frac{4\pi}{5}\right)n$$

Since $\sin(2\pi) = 0$, we obtain

$$5\sin\left(\frac{6\pi}{5}\right)n = 5\sin\left(-\frac{4\pi}{5}\right) = -5\sin\left(\frac{4\pi}{5}\right)$$

The third term, $\frac{12\pi}{5}$ can be written as,

$$\frac{12\pi}{5} = \left(2\pi + \frac{2\pi}{5}\right)$$

$$\therefore \quad 10\cos\left(2\pi + \frac{2\pi}{5}\right) = 10\cos\left(\frac{2\pi}{5}\right)$$

$$\therefore \quad x(n) = 3\cos\left(\frac{2\pi}{5}\right)n - 5\sin\left(\frac{4\pi}{5}\right)n + 10\cos\left(\frac{2\pi}{5}\right)$$

$$\therefore \quad x(n) = 13\cos\left(\frac{2\pi}{5}\right)n - 5\sin\left(\frac{6\pi}{5}\right)n$$

(iii) The reconstructed signal is obtained by substituting $n = t\,F_s$ in the discrete time signal.

Since $F_s = 5000$

$\therefore \quad n = 5000\,t$

$$\therefore \quad x(t) = 13\cos\left(\frac{2\pi}{5}\right)5000\,t - 5\sin\left(\frac{4\pi}{5}\right)5000\,t$$

$$\therefore \quad x(t) = 13\cos 2000\,\pi t - 5\sin 4000\pi t.$$

Ex. 1.5.6 : An analog signal is given by :

$x(t) = 3\cos(100\,\pi t) + 2\sin(300\,\pi t) - 4\cos(100\,\pi t)$

(1) What is the Nyquist rate of the sigal.

(2) Write the equation of the sampled signal

(3) If this analog signal is sampled at

F_s = 200 samples/sec, what is the discrete-time signal obtained after sampling ?

Soln. : Given :

$$x(t) = 3\cos(100\,\pi t) + 2\sin(300\,\pi t) - 4\cos(100\,\pi t)$$

This can be written as,

$$x(t) = -\cos(100\,\pi t) + 2\sin(300\,\pi t) \quad ...(1)$$

Here $2\pi F_1 t = 100\pi t$ and $2\pi F_2 t = 300\pi t$

$\therefore \quad F_1 = 50$ Hz and $F_2 = 150$ Hz,

$\therefore$ Maximum frequency $F_{max} = 150$ Hz

$\therefore$ Nyquist rate $= 2\,F_{max} = 300$ Hz

(2) The discrete signal is obtained by substituting

$$T = \frac{n}{F_s} = \frac{n}{300} \text{ in Equation (1)}$$

$$\therefore \quad x(n) = -\cos\left(100\pi\frac{n}{300}\right) + 2\sin\left(300\pi\frac{n}{300}\right)$$

$$\therefore \quad x(n) = -\cos\left(\frac{\pi}{3}\right)n + 2\sin(\pi n)$$

Since $\sin(\pi n) = 0$, we get

$$x(n) = -\cos\left(\frac{\pi}{3}\right)n$$

(3) **Given :** $F_s = 200$ Hz.

The discrete signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{200} \text{ in Equation (1)}$$

$$\therefore \quad x(n) = -\cos\left(100\pi\frac{n}{200}\right) + 2\sin\left(300\pi\frac{n}{200}\right)$$

$$x(n) = -\cos\left(\frac{\pi}{2}\right)n + 2\sin\left(\frac{3\pi}{2}\right)n$$

Now $\frac{3\pi}{2} = 2\pi - \frac{\pi}{2}$

$$x(n) = -\cos\left(\frac{\pi}{2}\right)n + 2\sin\left(2\pi - \frac{\pi}{2}\right)n$$

Since $\sin(2\pi) = 0$, we obtain

$$x(n) = -\cos\left(\frac{\pi}{2}\right)n + 2\sin\left(-\frac{\pi}{2}\right)n$$

Ex. 1.5.7 : An analog signal is given as :

$x_a(t) = 15\cos(1250\,\pi t) + 17\cos(2170\,\pi t) + 33\cos(4750\,\pi t)$. is converted to a discrete time signal. Determine the Nyquist rate, Folding frequency, resulting discrete time signal x(n) if the sampling frequency is 625 Hz. Also write the discrete frequencies in radians

Soln. :

Given

$x_a(t) = 15\cos(1250\,\pi t) + 17\cos(2170\,\pi t) + 33\cos(4750\,\pi t)$.

This signal consists of following frequencies,

$2\pi F_1 = 1250\pi$, $2\pi F_2 = 2170\pi$ and $2\pi F_3 = 4750$

$\therefore \quad F_1 = 625$ Hz, $F_2 = 1085$ Hz, and $F_3 = 2375$ Hz

Thus the maximum frequency present in the signal is,

$$F_{max} = W = 2375 \text{ Hz}$$

$\therefore$ Nyquist rate $= 2\,F_{max} = 4750$ Hz

Given : The signal is being sampled at 625 Hz

$$\therefore \text{ The folding frequency} = F_{fold} = \frac{F_s}{2} = \frac{625}{2} = 312.5 \text{ Hz}.$$

The discrete time signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{625} \text{ in the given equation.}$$

$$\therefore \quad x(n) = 15\cos\left(1250\,\pi\frac{n}{625}\right) + 17\cos\left(2170\pi\frac{n}{625}\right) + 33\cos\left(4750\,\pi\frac{n}{625}\right)$$

$$\therefore \quad x(n) = 15\cos(2\pi)n + 17\cos(2\pi\ 1.736)\,n + 33\cos(2\pi\ 3.8)n$$

This equation is of the form $x(n) = A\cos(\omega_0 n)$

$$\therefore \quad x(n) = 15\cos(2\pi)n + 17\cos(3.47\,\pi)n + 33\cos(7.6\,\pi)n$$

Hence the discrete frequencies in terms of ω are,

$$\omega_1 = 2\pi$$
$$\omega_2 = 3.47\,\pi$$
$$\omega_3 = 7.6\,\pi$$

Ex. 1.5.8 : Consider the following analog signal

$x(t) = 2\sin(100\pi t)$

The signal is sampled at x(t) is sampled with a sampling rate F_S = 50 Hz. determine the discrete time signal.

MU - Dec. 2016, 10 Marks

Soln. : Given : $x(t) = 2 \sin (100\pi t)$ …(1)

The discrete signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{50} \text{ in Equation (1)}$$

$$\therefore \quad x(n) = 2 \sin \left(100\pi \frac{n}{50} \right)$$

$$\therefore \quad x(n) = 2 \sin 2\pi n$$

We vary n and realize that x(n) = 0 for all values of n

$$\therefore \quad x(n) = \{ 0,0,0,\ldots\ldots \}$$

Ex. 1.5.9 : Consider the analog signal

$x(t) = 5 \cos 2\pi(1000)t + 10 \cos 2\pi(5000)t$ is to be sampled.

(i) What is Nyquist rate for this signal ?

(ii) If this analog signal is sampled at 4 kHz, will the signal be recovered from its samples ? **MU - May 2018, 10 Marks**

Soln. : (i)Nyquist rate

Given : $x(t) = 5 \cos 2\pi(1000)t + 10 \cos 2\pi(5000)t$ …(1)

The given signal consists of following frequencies,

$$F_1 = 1000 \text{ Hz and}$$

$$F_2 = 5000 \text{ Hz}$$

$$\text{Thus } F_{max} = 5000 \text{ Hz} = 5 \text{ kHz}$$

$$\therefore \text{ Nyquist rate } = 2 F_{max} = 10 \text{ kHz}$$

(ii) Analog signal is sampled at 4 kHz .

A discrete signal is obtained by substituting

$$t = \frac{n}{F_s} = \frac{n}{4000} \text{ in Equation (1)}$$

$$\therefore \quad x(n) = 5 \cos 2\pi \left(\frac{1000\, n}{4000} \right) + 10 \cos 2\pi \left(\frac{5000\, n}{4000} \right)$$

$$\therefore \quad x(n) = 5 \cos \left(\frac{\pi}{2} \right) n + 10 \cos \left(\frac{5\pi}{2} \right) n$$

Now, in the second term $\left(\frac{5\pi}{2} \right)$ can be written as,

$$\left(\frac{5\pi}{2} \right) = \left(2\pi + \frac{\pi}{2} \right)$$

$$\therefore \quad 10 \cos \left(\frac{5\pi}{2} \right) n = 10 \cos \left(2\pi + \frac{\pi}{2} \right) n = 10 \cos \left(\frac{\pi}{2} \right) n$$

Hence we observe that $10 \cos \left(\frac{5\pi}{2} \right) n$ is identical to $10 \cos \left(\frac{\pi}{2} \right) n$. This is called aliasing.

$$\therefore \quad x(n) = 15 \cos \left(\frac{\pi}{2} \right) n$$

The continuous time signal had two different frequencies while the discrete time signal has only one.

(iii) The reconstructed signal is obtained by substituting $n = t\, F_s$ in the discrete time signal

Since $F_s = 4000$

$$\therefore \quad n = 4000\, t$$

$$\therefore \quad x(t) = 15 \cos \left(\frac{\pi}{2} \right) 4000t$$

$$\therefore \quad x(t) = 15 \cos 2\pi\, 1000t$$

We observe that the reconstructed signal has only one frequency of 1000 Hz.

1.6 Applications of DSP

Digital signal processing systems are used in a variety of applications. Look around and we realize that every electronic system has a DSP in it. Some of the applications are listed as follows :

1. Speech Recognition, Synthesis, Analysis.
2. Image processing applications.
3. Robotic vision.
4. Biomedical signal application like processing and analyzing ECG, EMG, CT, MRI, etc.
5. Data compression.
6. Military application like Radar signal processing, secure communications.
7. Industrial application like robotics, CNN.
8. Communication application like voice commands, cellular telephones.
9. Robotic vision, vibration analysis.

Summary

This chapter deals with the basics of Digital signal processing. A basic classification of signals is done. We have discussed the concept of frequency and explained the Nyquist criteria. Examples have been solved to understand Aliasing. A block diagram of a DSP system is discussed. The chapter ends with listing down the applications of DSP.

Review Questions

Q. 1 Explain advantages of Digital signal Processing over Analog Signal Processing

Q. 2 Explain aliasing effect.

Q. 3 Define : Aliasing

Q. 4 What is aliasing observed in sampling process ? How this can be avoided ?

Q. 5 Define a Nyquist rate.

Q. 6 Draw and explain basic elements of DSP.

Q. 7 Explain Concept of Frequency in Discrete Time Signals.

❑❑❑

Discrete Time Signal and System

Syllabus :

Classification of Discrete-Time Signals, Classification of Discrete- Systems

Linear Convolution formulation for 1-D and 2-D signal (without mathematical proof), Auto and Cross Correlation formula evaluation, LTI system, Concept of Impulse Response and Step Response, Output of DT system using Time Domain Linear Convolution

2.1 Introduction

In chapter 1, we discussed the basics of digital signal processing. We also understood the method of obtaining a discrete time signal form an analog signal. In this chapter we discuss various aspects of discrete time signals and systems. We will also learn how to perform operations on discrete time signals.

2.1.1 Representation of Discrete Time Signals

There are various methods of representing a discrete time signal.

(1) Graphical representation

Consider a signal x(n) with values x(– 2) = 2, x(– 1) = 1, x(0) = 1.5, x(1) = 1, x(2) = 2, x(3) = 3, x(4) = 1.5. This signal can be represented graphically as shown in Fig. 2.1.1.

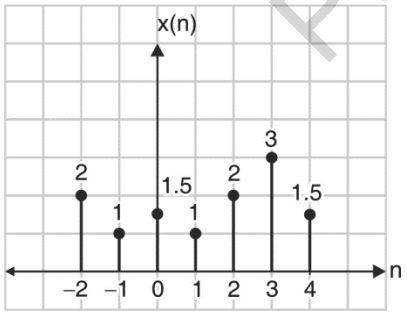


Fig. 2.1.1

(2) Sequence representation

A more concise way of representing the same signal is by representing it as a sequence shown as follows,

$$x(n) = \{2, 1, 1.5, 1, 2, 3, 1.5\}$$
$$\uparrow$$

The arrow at the bottom indicates the origin.

If the arrow is not mentioned, then the first element is the origin.

Solved Example

Ex. 2.1.1 : Draw the graphical representation of the given signal x(n) = {1, 2, 2, 2, 3}

Soln. : Since there is no arrow, the first element is the origin.

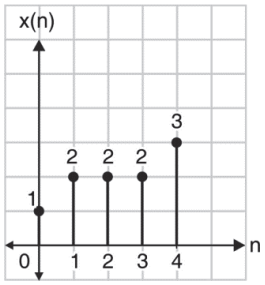


Fig. P. 2.1.1

An infinite sequence is represented by dots on both sides.

$$x(n) = \{\ldots, 1, 2, 4, 3, \ldots\}$$

(3) Functional representation :

The discrete time signal can also be represented as,

$$x(n) = \begin{cases} 1 & n = 1,3 \\ -4 & n = 0 \\ 0 & \text{else where} \end{cases}$$

This can be graphically drawn as shown in Fig. 2.1.2.

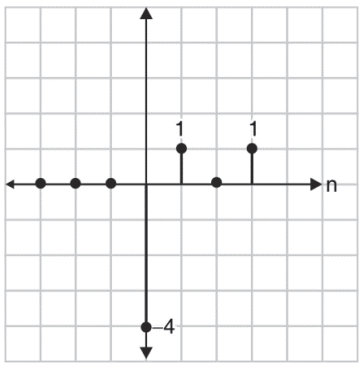


Fig. 2.1.2

Solved Example

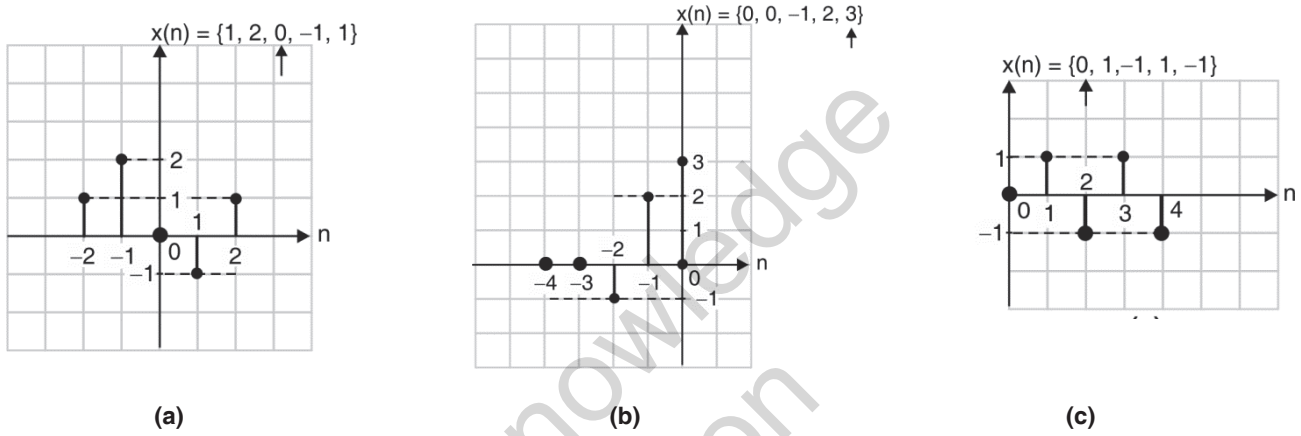
Ex. 2.1.2 : Represent the following signals graphically :

(i) $x(n) = \{1, 2, \underset{\uparrow}{0}, -1, 1\}$

(ii) $x(n) = \{0, 0, -1, 2, \underset{\uparrow}{3}\}$

(iii) $x(n) = \{\underset{\uparrow}{0}, 1, -1, 1, -1)\}$

Soln. : These signals are as shown in Figs. P.2.1.2(a), (b) and (c) respectively.



(a) (b) (c)

Fig. P. 2.1.2 : Graphical representation of given sequences

2.1.2 Elementary Discrete Time Signals

There are a number of basic signals that we encounter regularly and which play an important role in signal processing.

1. Delta or a unit impulse function

A unit impulse signal is denoted by $\delta(n)$ and is defined as,

$$\delta(n) = \begin{cases} 1 \; ; & n = 0 \\ 0 \; ; & \text{otherwise} \end{cases}$$

The graphical representation of the unit impulse signal is shown in Fig. 2.1.3.

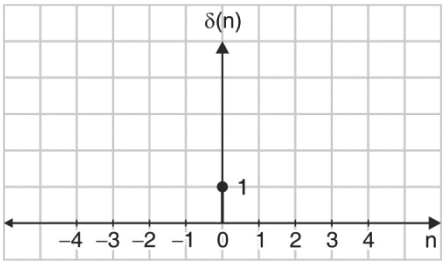


Fig. 2.1.3

A unit impulse function is a signal that is zero everywhere except at n = 0 where its value is unity.

2. Unit step signal

A unit step signal is denoted by u(n) and is defined as,

$$u(n) = \begin{cases} 1 \; ; & n \geq 0 \\ 0 \; ; & \text{otherwise} \end{cases}$$

The graphical representation of the unit step signal is shown in Fig. 2.1.4.

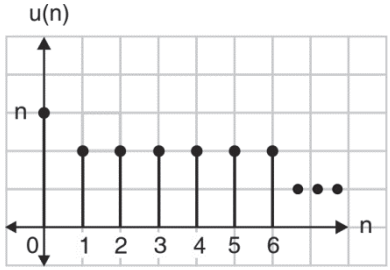


Fig. 2.1.4

3. Unit ramp signal

A unit ramp signal is denoted by $u_r(n)$ and is defined as,

$$u_r(n) = \begin{cases} n & ; \quad n \geq 0 \\ 0 & ; \quad \text{otherwise} \end{cases}$$

The graphical representation of the unit ramp signal is shown in Fig. 2.1.5.

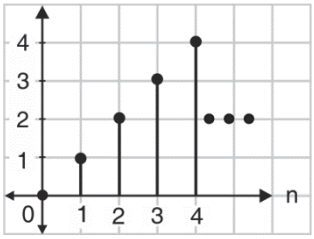


Fig. 2.1.5

4. Exponential signal

A exponential signal is of the form

$$x(n) = a^n \qquad \text{for all } n$$

The graphical representation of the exponential signal is shown in Fig. 2.1.6. Four different ranges of 'a' are taken to study the change in the exponential pattern.

Fig. 2.1.6 illustrates different type of discrete-time exponential signals. When the value is 0 < a < 1, the sequence decays exponentially, when value of a > 1, the sequence grows exponentially. Note that when a < 0, the discrete-time exponential signal takes alternating signs.

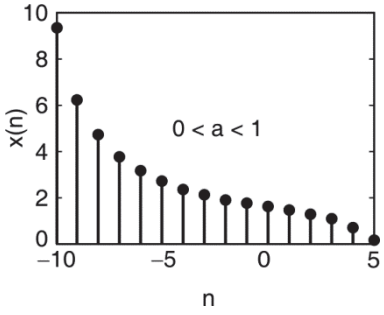
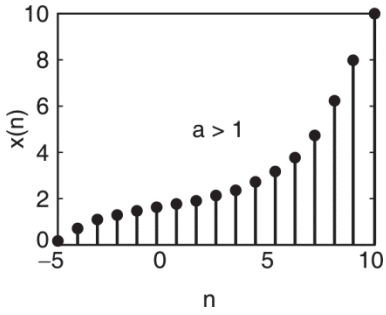
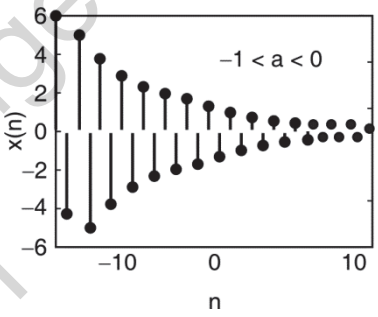


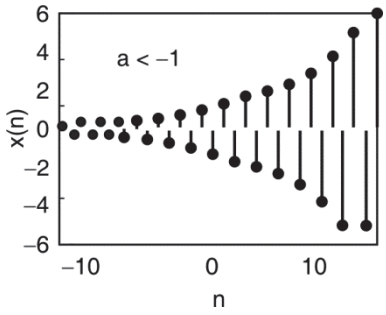
Fig. 2.1.6 (a)



(b)



(c)



(d)

Fig. 2.1.6 : Exponential sequences

5. Sinusoidal signal

A discrete time sinusoidal signal is given by the formula

$$x(n) = A \cos(\omega n + \phi)$$

Where A is the amplitude, ω is the frequency in radians and ϕ is the phase.

An example of discrete-time sinusoidal signal is shown in Fig. 2.1.7.

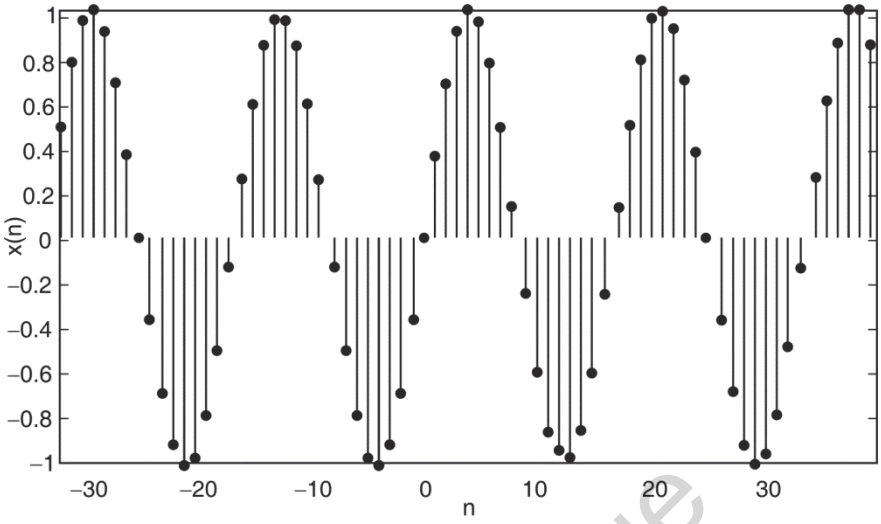


Fig. 2.1.7 : A sinusoidal sequence

Given below is the MATLAB code for generating various elementary discrete time signals. Run the above program and see the results.

```
%Plot Unit Step, Unit Ramp, Exponential and Sinusoidal Functions

clc
clear all

%UNIT STEP
for n = 1 : 1 : 20% We take the length of the sequence to be 20
        x1(n) = 1;
end
figure(1)
stem(x1)

%UNIT RAMP
for n = 1 : 1 : 20
        x2 (n) = n ;
end
figure(2)
stem(x2)

% EXPONENTIAL SIGNAL
a = input ('enter the value of a for exponential signal :')
for n = 1 : 1 : 20
        x3(n) = a ^ n;
end
figure (3)
stem (x3)

%SINUSOIDAL SIGNAL
A = input ('enter the amplitude for sinusoidal function :')
w = input ('enter the value of frequency for sinusoidal function :')
%Take w = 0.1*pi, 0.2*pi,....0.9* pi, pi
for n = 1 : 1 : 20
        x4(n) = A*cos (w*n) ;
end
figure(4)
stem(x4)
```

2.2 Classification of Discrete Time Signals

Discrete time signals have different characteristics. We now classify the discrete time signals based on these characteristics. The discrete time signals are classified as follows :

1. Even and odd signals (Symmetric and Anti-symmetric signals)
2. Periodic and Aperiodic signals
3. Energy and power signals

2.2.1 Even and Odd Signals

A discrete time signal is said to be even (symmetric) if it satisfies the following condition.

$$x(n) = x(-n) \quad \ldots(2.2.1)$$

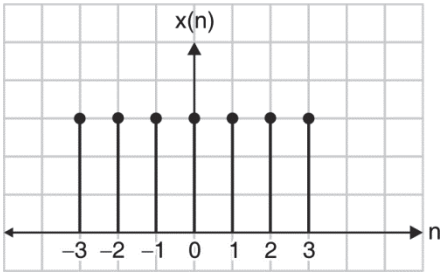
A discrete time signal is said to be odd (Anti-symmetric) if it satisfies the following condition.

$$x(n) = -x(-n) \quad \ldots(2.2.2)$$

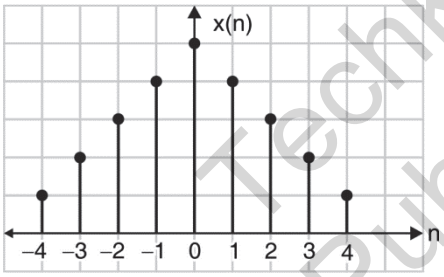
For an even signal, $x(1) = x(-1), x(2) = x(-2)\ldots$

While for a odd signal, $x(1) = -x(-1), x(2) = -x(-2)\ldots$

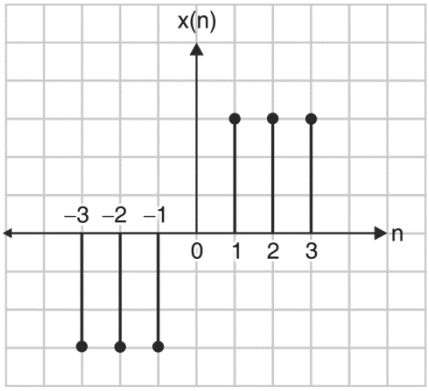
Given below are examples of even and odd signals.



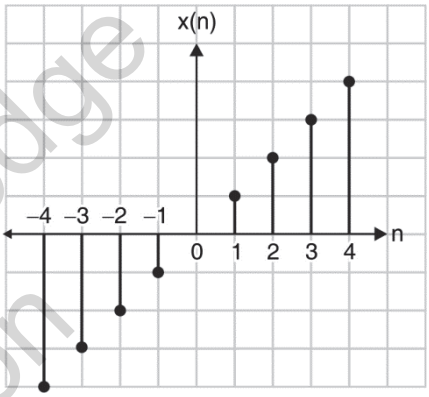
(a) Even signal



(b) Even signal



(c) Odd signal



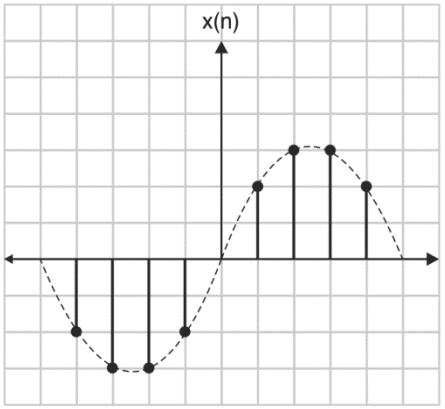
(d) Odd signal

Fig. 2.2.1

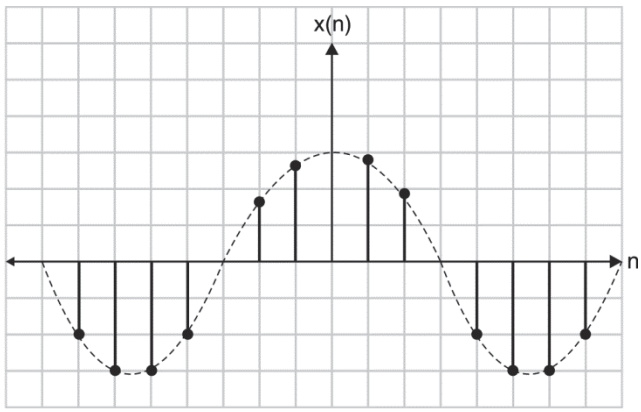
A sine wave is a odd signal while a cos wave is a even signal,

Note :	$\sin(50) = -\sin(-50)$
	$\cos(50) = \cos(-50)$

Most of the signals that we encounter are neither purely even or odd but are a combination of both. In other words any arbitrary signal can be expressed as a sum of even signals and odd signals.



(a) Sine wave (odd signal)



(b) cos wave (even signal)

Fig. 2.2.2

The even component is extracted by adding x(n) to x(– n) and dividing by 2 i.e.

$$x_e(n) = \frac{x(n) + x(-n)}{2} \quad \ldots(2.2.3)$$

Similarly the even component is extracted by adding x(n) to – x(– n) and diving by 2.

$$X_o(n) = \frac{x(n) - x(-n)}{2} \quad \ldots(2.2.4)$$

Clearly Equation (2.2.3) satisfies the symmetric condition in Equation (2.2.1) while Equation (2.2.4) satisfies the antisymmetric condition in Equation (2.2.2).

Let us solve an example to understand this better.

Note : x (– n) is simplify the mirror image of x (n).

Solved Example

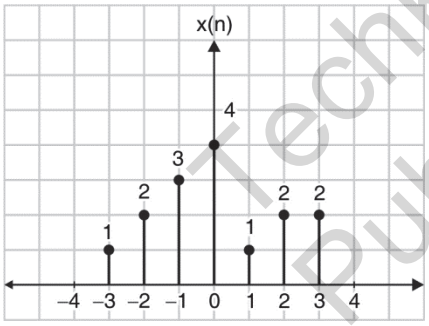
Ex. 2.2.1 : Find the even and odd components of the given discrete time signal

$$x(n) = \{1, 2, 3, 4, 1, 2, 2\}$$
$$\uparrow$$

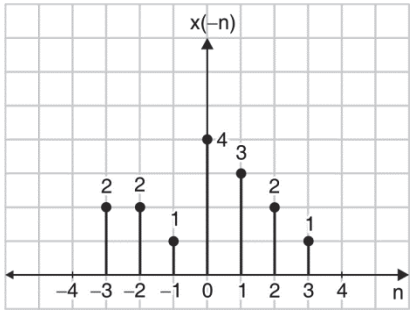
Soln. : We shall solve this example graphically. To find the even and odd components, we need x (n) as well as x (– n).

We draw them one below the other.

As stated earlier, x (– n) is simply the mirror image of x (n).



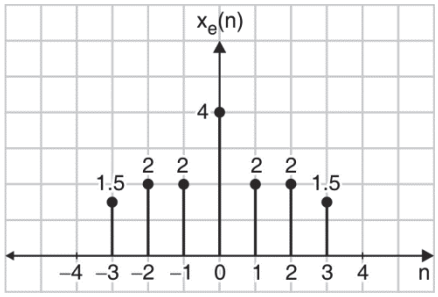
(a)



(b)

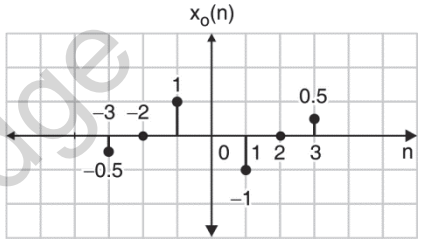
Fig. P. 2.2.1

Now, $x_e(n) = \frac{x(n) + x(-n)}{2}$



(c)

$$x_o(n) = \frac{x(n) - x(-n)}{2}$$



(d)

Fig. P. 2.2.1

Note : If we add $x_e(n)$ and $x_o(n)$, we get back the original signal ! !

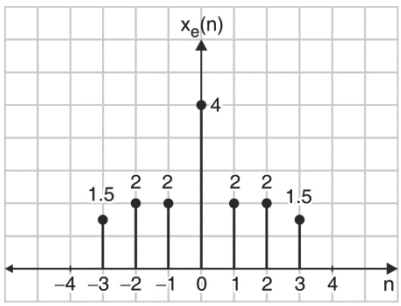


Fig. P. 2.2.1 (e)

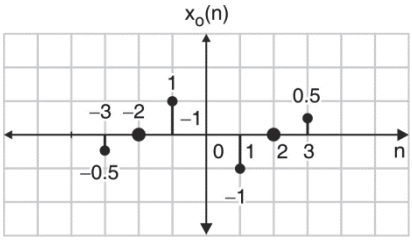


Fig. P. 2.2.1 (f)

$x(n) = x_e(n) + x_o(n)$

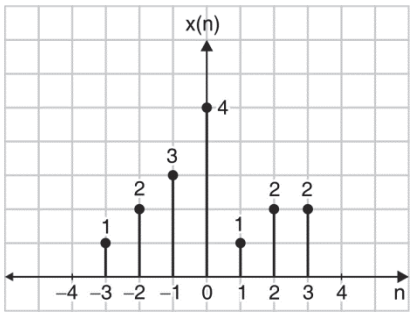


Fig. P.2.2.1 (g)

Take any sequence and this property will always work.

2.2.2 Periodic and Aperiodic Signals

A signal is periodic if there exist N such that

$$x(n+N) = x(n) \quad \ldots(2.2.5)$$

Let us take an example. Check whether

x(n) = A cos (ωn + θ) is periodic.

$$x(n) = A\cos(\omega n + \theta)$$

Here, $\omega = 2\pi f$

$$\therefore \quad x(n) = A\cos(2\pi f n + \theta)$$

Now, $x(n+N) = A\cos(2\pi f(n+N) + \theta)$

For periodicity $x(n+N) = x(n)$

$$\therefore \quad A\cos(2\pi f(n+N) + \theta) = A\cos(2\pi f n + \theta)$$

$$\therefore \quad A\cos(2\pi f n + 2\pi f N + \theta) = A\cos(2\pi f n + \theta)$$

The above equation will be true only if

$$2\pi f N = 2\pi k \quad \text{where, k is an integer.}$$

$$\therefore \quad f = \frac{k}{N}$$

Here, N and k are both integers.

Hence a discrete time signal is periodic only if it's frequency is a rational number (ratio of two integers).

If the input signal is expressed as a summation of two signals i.e., $x(n) = x_1(n) + x_2(n)$, we calculate f_1 and f_2 where, $f_1 = \frac{k_1}{N_1}$ and $f_2 = \frac{k_2}{N_2}$

The resultant signal x (n) is periodic if $\frac{N_1}{N_2}$ is a ratio of two integers.

The period of x (n) will be the LCM of N_1 and N_2.

A few examples would make things clear.

Solved Examples

Ex. 2.2.2 : Few discrete time sequences are given below

(i) cos (0.01 π n) (ii) cos (3 π n)

(iii) sin (3 n) (iv) $\cos\left(\frac{n}{8}\right)\cos\left(\frac{\pi n}{8}\right)$

Determine whether they are periodic or aperiodic. If a sequence is periodic, determine its fundamental period.

Soln. :

(i) Given :

$$x(n) = \cos(0.01\pi n) \quad \ldots(1)$$

We have the standard equation,

$$x(n) = \cos\omega n \quad \ldots(2)$$

Comparing Equations (1) and (2) we get,

$$\omega = 0.01\pi$$

But $\omega = 2\pi f$

$$\therefore \quad 2\pi f = 0.01\pi$$

$$\therefore \quad f = \frac{0.01\pi}{2\pi} = \frac{0.01}{2}$$

$$\therefore \quad f = \frac{1}{200} \text{ cycles per sample} \quad \ldots(3)$$

Since frequency 'f 'is expressed as the ratio of two integers; this sequence is periodic. Now we have the condition of periodicity,

$$f = \frac{k}{N} \quad \ldots(4)$$

Here 'N' indicates, the fundamental period.

Comparing Equations (3) and (4)

Fundamental period = N = 200 samples

(ii) Given :

$$x(n) = \cos(3\pi n) \quad \ldots(5)$$

Comparing with Equation (2) we get,

$$\omega = 3\pi$$

$$\therefore \quad 2\pi f = 3\pi$$

$$\therefore \quad f = \frac{3}{2} \text{ cycles / sample} \quad \ldots(6)$$

Since `f' is ratio of two integers; the given sequence is periodic. Comparing Equations (4) and (6) we get,

Fundamental period = N = 2 samples

(iii) Given sequence is,

$$x(n) = \sin 3n \quad \ldots(7)$$

Comparing with Equation (2) we get,

$$\omega = 3$$

$$\therefore \quad 2\pi f = 3$$

$$\therefore \quad f = \frac{3}{2\pi}$$

Here 2π is not an integer. That means 'f' cannot be expressed as the ratio of two integers. In this case f is irrational. Thus the given sequence is non-periodic.

(iv) Given sequence is,

$$x(n) = \cos\left(\frac{n}{8}\right)\cos\left(\frac{\pi n}{8}\right) \quad \ldots(8)$$

The standard equation can be expressed as,

$$x(n) = \cos\omega_1 n \cos\omega_2 n \quad \ldots(9)$$

Comparing Equations (8) and (9) we get,

$$\omega_1 = \frac{1}{8} \text{ and } \omega_2 = \frac{\pi}{8}$$

but, $\omega = 2\pi f$

$$\therefore \quad 2\pi f_1 = \frac{1}{8} \text{ and } 2\pi f_2 = \frac{\pi}{8}$$

$$\therefore \quad f_1 = \frac{1}{16\pi} \text{ and } f_2 = \frac{1}{16}$$

Hence f_1 is ratio of non-integer values. So it is non-periodic. While f_2 is ratio of two integer values. So it is periodic. But the total signal is multiplication of non-periodic and periodic signals. So the resultant signal x(n) is non-periodic.

Ex. 2.2.3 : Find if the following sequences are periodic or not. If yes find its fundamental time period.

(i) $x_1(n) = e^{j\left(\frac{\pi}{4}\right)^n}$ (ii) $x_2(n) = 3\sin\left(\frac{1}{8}n\right)$

Soln. :

(i) Given : $x_1(n) = e^{j\left(\frac{\pi}{4}\right)^n}$ ...(1)

A discrete time complex exponential is periodic if its relative frequency is a rational number. We have the standard equation of a discrete time complex exponential.

$$x(n) = e^{j\frac{2\pi k n}{N}} \quad \ldots(2)$$

Comparing Equations (1) and (2) we get,

$$\frac{2\pi k n}{N} = \frac{\pi}{4}n \qquad \therefore \frac{2k}{N} = \frac{1}{4}$$

$$\therefore \frac{k}{N} = \frac{1}{8}$$

Since it is ratio of two integers; the given sequence is periodic. It's fundamental period is N= 8 samples.

(ii) Given $x_2(n) = 3\sin\left(\frac{1}{8}n\right)$ ...(3)

We have the standard equation,

$$x(n) = A\sin\omega n \quad \ldots(4)$$

Comparing Equations (3) and (4) we get,

$$\omega = \frac{1}{8} \qquad \therefore 2\pi f = \frac{1}{8}$$

$$\therefore \quad f = \frac{1}{16\pi}$$

Here π is an irrational number hence f is not the ratio of two integers. Thus given sequence is non-periodic.

2.2.3 Energy and Power Signals

For a discrete time signal x (n), the energy is defined as,

$$E = \sum_{n=-\infty}^{+\infty} |x(n)|^2$$

The reason for using magnitude square is so that the definition applies to real valued as well as complex valued signals. If E is finite, then x (n) is called an **energy signal**. Many signals that possess infinite energy have finite power. The average power of a discrete time signal x (n) is defined as

$$P = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x(n)|^2$$

A signal x (n) is said to be a **power signal** if the average power is finite.

Solved Example

Ex. 2.2.4 : Obtain energy for the signal $x(n) = a^n u(n)$ where $|a| < 1$.

Soln. : For D.T. signal, energy is given by,

$$E = \sum_{n=-\infty}^{\infty} |x(n)|^2 \quad ...(1)$$

The given equation of x (n) is,

$$x(n) = a^n u(n) \quad ...(2)$$

Here u(n) represents unit step. It's value is one for the range 0 to ∞. Hence multiplication by u (n) does not change the amplitude of a^n; but it indicates that a^n is present only from n = 0 to n = ∞.

Thus Equation (1) reduces to,

$$E = \sum_{n=0}^{\infty} |a^n|^2 \quad ...(3)$$

Rearranging the term we get,

$$E = \sum_{n=0}^{\infty} [a^2]^n \quad ...(4)$$

We use the geometric series formula,

$$\sum_{n=0}^{\infty} A^n = 1 + A + A^2 + A^3 + ... = \frac{1}{1-A} \text{ if } |A| < 1.$$

Here $A = a^2$. Thus Equation (4) reduces to,

$$E = \frac{1}{1-a^2} \text{ if } |a^2| < 1$$

Ex. 2.2.5 : $x(n) = (0.5)^n u(n)$. State whether it is an energy or power signal. Justify.

Soln. : First we will calculate the energy of signal x (n). It is given by,

$$E = \sum_{n=-\infty}^{\infty} |x(n)|^2 \quad \ldots (1)$$

The given signal is,

$$x(n) = (0.5)^n u(n) \quad \ldots (2)$$

Since it is multiplied by unit step; this signal is present from n = 0 to n = ∞. Thus Equation (1) becomes,

$$E = \sum_{n=0}^{\infty} [(0.5)^n]^2 = \sum_{n=0}^{\infty} \left(\frac{1}{2}\right)^{2n}$$

$$E = \sum_{n=0}^{\infty} \left(\frac{1}{4}\right)^n \quad \ldots(3)$$

We have the standard geometric series formula,

$$\sum_{n=0}^{\infty} A^n = 1 + A + A^2 + \ldots = \frac{1}{1-A}$$

$$\therefore \quad E = \frac{1}{1-\frac{1}{4}} = \frac{4}{3}$$

i.e. E is finite.

Note : If 0 < E <∞ then the signal is energy signal. Since the calculated value of energy is finite; the given signal is energy signal.

Ex. 2.2.6 : Determine the energy and power of signal given by,

$$x(n) = \left(\frac{1}{2}\right)^n \quad n \geq 0$$

$$= 3^n \quad n < 0$$

Soln. : The energy of signal is given by,

$$E = \sum_{n=-\infty}^{\infty} |x(n)|^2 \quad \ldots (1)$$

The given signal is, $x(n) = \left(\frac{1}{2}\right)^n$ for the range n≥ 0 and $x(n) = 3^n$ for n < 0. Thus Equation (1) becomes,

$$E = \sum_{n=0}^{\infty} \left[\left(\frac{1}{2}\right)^n\right]^2 + \sum_{n=-1}^{-\infty} [3^n]^2$$

$$\therefore \quad E = \sum_{n=0}^{\infty} \left[\left(\frac{1}{2}\right)^2\right]^n + \sum_{n=-1}^{-\infty} [3^2]^n$$

$$= \sum_{n=0}^{\infty} \left(\frac{1}{4}\right)^n + \sum_{n=-1}^{-\infty} (9)^n \quad \ldots(2)$$

Consider the first summation term. Using geometric series formula we can express it as follows :

$$\sum_{n=0}^{\infty} \left(\frac{1}{4}\right)^n = \frac{1}{1-\frac{1}{4}} = \frac{4}{3} \quad \ldots(3)$$

Now consider the second summation term. To make the limits of the summation positive, put n = – m. The limits change as follows.

When $n = -1 \Rightarrow -m = -1 \therefore m = 1$

When $n = -\infty \Rightarrow -m = -\infty \therefore m = \infty$

$$\therefore \sum_{n=-1}^{-\infty} (9)^n = \sum_{m=+1}^{\infty} (9)^{-m} = \sum_{m=1}^{\infty} \left(\frac{1}{9}\right)^m$$

Now use the standard summation formula,

$$\sum_{n=1}^{\infty} A^n = \frac{A}{1-A}$$

$$\therefore \sum_{m=1}^{\infty} \left(\frac{1}{9}\right)^m = \frac{1/9}{1-\frac{1}{9}} = \frac{\frac{1}{9}}{8/9} = \frac{1}{8} \quad \ldots(4)$$

Putting Equations (3) and (4) in Equation (2) we get,

$$E = \frac{4}{3} + \frac{1}{8}$$

$$\therefore E = \frac{35}{24}$$

Since the energy is finite, the given signal is an Energy signal.

If the energy of signal is finite then its power is zero. Thus the power of given signal is zero.

$$\therefore P = 0$$

Ex. 2.2.7 : Determine the energy and power of the signal given by,

$$x(n) = \left(\frac{1}{3}\right)^n u(n)$$

MU - May 2018, 10 Marks

Soln. :

$$x(n) = \left(\frac{1}{3}\right)^n u(n)$$

We know

$$E = \sum_{n=-\infty}^{+\infty} |x(n)|^2$$

$$= \sum_{n=-\infty}^{+\infty} \left| \left(\frac{1}{3}\right)^n u(n) \right|^2$$

Since there is u(n), we change the limits of summation

$$\therefore E = \sum_{n=0}^{\omega} \left(\frac{1}{3}\right)^{n2}$$

$$\therefore E = \sum_{n=0}^{\omega} \left(\frac{1}{9}\right)^n$$

From the geometric progression formula,

We know

$$\sum_{n=D}^{\omega} a^n = \frac{1}{1-a} \quad ; |a| < 1$$

$$\therefore E = \frac{1}{1-\frac{1}{9}} = 1.125$$

Since $E < \infty$, the given signal is a energy signal.

Since the signal has finite energy, the power of the signal is zero.

2.3 Operation on Signals

It is imperative for students to understand simple manipulation techniques on discrete time signals. Signal processing involves modifying the original signal. This modification is achieved by performing different operations on discrete time signals. The mathematical transformation from one signal to another is represented as,

$$y(n) = T[x(n)]$$

Here x (n) is the input, y (n) is the output and T is the operation that one performs.

The basic operations are :

1. Time shifting
2. Time reversal
3. Time scaling
4. Scalar multiplication
5. Signal addition and multiplication

We shall discuss each one in detail.

2.3.1 Time Shifting

The shift operation takes the input signal x (n) and shifts the signal, resulting in either a delay or an advancement of the signal.

Mathematically it is represented as,

$$y(n) = x(n-k)$$

Here k is the amount of shift required.

Let us see what this equation amounts to.

Solved Examples

Ex. 2.3.1 : $x(n) = \{1, 2, 3, 4, 5\}$ (↑)

Find $y(n) = x(n-2)$

Soln. : We draw x (n)

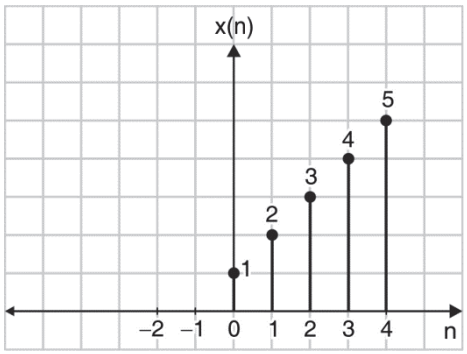


Fig. P.2.3.1

$$y(n) = T[x(n)]$$

$$\therefore \quad y(n) = x(n-2)$$

We take different values of n

$y(0) = x(-2) = 0$ ($\because$ the value of x(n) at n = –2 is 0)

$y(1) = x(-1) = 0$

$y(2) = x(0) = 1$ ($\because$ the value of x(n) at n = 0 is 1)

$y(3) = x(1) = 2$

$y(4) = x(2) = 3$

$y(5) = x(3) = 4$

$y(6) = x(4) = 5$

$y(7) = x(5) = 0$

We plot these values

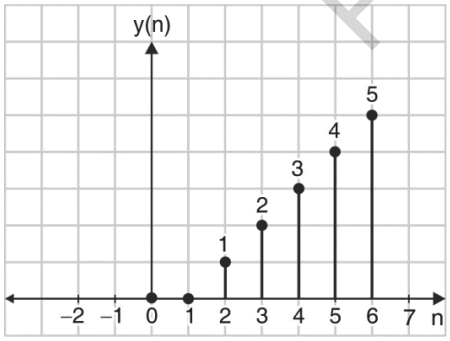


Fig. P. 2.3.1(a)

Hence we state that y (n) = x (n – 2) is a right shifted version of x (n) i.e., y (n) is a delayed version of x (n).

Ex. 2.3.2 : x (n) = {1, 2, 3, 4, 5}
↑

Find y (n) = x (n + 2)

Soln. :

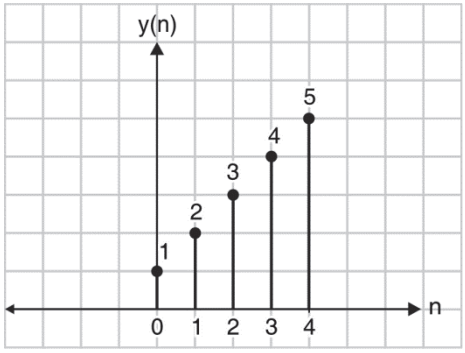


Fig. P.2.3.2

$$y(n) = T[x(n)]$$

$$\therefore \quad y(n) = x(n+2)$$

We take different values of n.

$y(-2) = x(0) = 1$

$y(-1) = x(1) = 2$

$y(0) = x(2) = 3$

$y(1) = x(3) = 4$

$y(2) = x(4) = 5$

$y(3) = x(5) = 0$

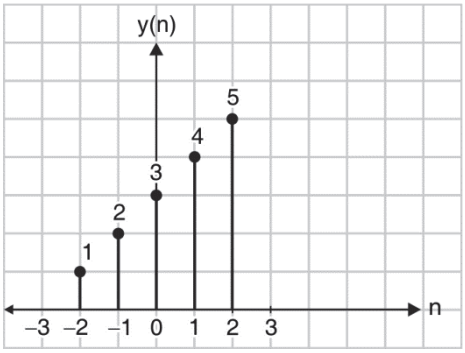


Fig. P. 2.3.2(a)

We hence note that y (n + 2) shifts the output to the left. This is known as time advance.

Hence for y (n – k), if k is > 1 → Right shift → Time delay

if k is < 1 → Left shift → Time advance

2.3.2 Time Reversal

This is also known as the folding operation. It is mathematically given by the equation,

$$y(n) = T[x(n)]$$

$$y(n) = x(-n)$$

This can be very easily understood by taking the same example.

Solved Example

Ex. 2.3.3 : x (n) = {1 , 2, 3, 4, 5} find y(n).
↑

Soln. :

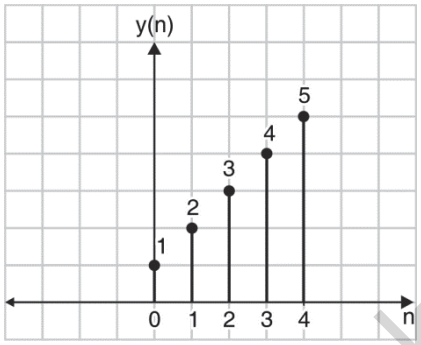


Fig. P. 2.3.3

$$y(n) = T[x(n)]$$

$$y(n) = x(-n)$$

We take different values of n

$$y(-4) = x(-(-4)) = x(4) = 5$$

$$y(-3) = x(3) = 4$$

$$x(-2) = x(2) = 3$$

$$x(-1) = x(1) = 2$$

$$y(0) = x(0) = 1$$

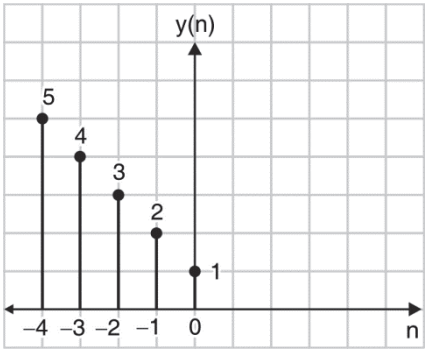


Fig. P.2.3.3(a)

Hence y (n) is a mirror image of x (n). It is a folded version of x (n).

We have used the time reversal operation while computing the odd and even components of a signal.

2.3.3 Time Scaling

This operation is related to change in time scale.

There are two types of time scaling operations :

(a) Down scaling

(b) Up scaling

(a) Down scaling : It is mathematically represented as,

$$y(n) = T[x(n)]$$

$$y(n) = x(2n)$$

We shall explain this with an example.

Solved Example

Ex. 2.3.4 : x (n) = {5, 4, 3, 2, 1 , 2, 3, 4, 5}
↑

Find y (n) = x (2n)

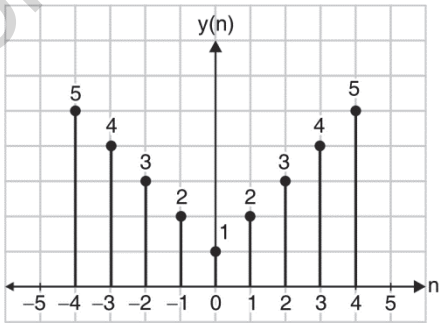


Fig. P. 2.3.4

Soln. :

$$y(n) = T[x(n)]$$

$$y(n) = x(2n)$$

We take different values of n

$$y(0) = x(0) = 1$$

$$y(1) = x(2) = 3$$

$$y(2) = x(4) = 5$$

$$y(3) = x(6) = 0$$

$$y(-1) = x(-2) = 3$$

$$y(-2) = x(-4) = 5$$

$$y(-3) = x(-6) = 0$$

$$\therefore \quad y(n) = \{5, 3, 1, 3, 5\}$$
↑

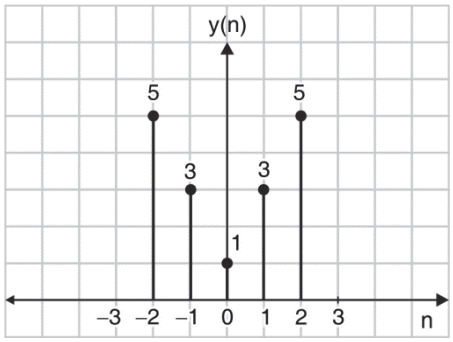


Fig. P.2.3.4(a)

Hence we note that the signal gets compressed. It is also important to note that the shape of x (n) and y (n) remain the same. This is known as down sampling.

(b) Up scaling : It is mathematically represented as,

$$y(n) = x(n/2)$$

We take the same example.

Solved Example

Ex. 2.3.5 : x (n) = {5, 4, 3, 2, 1 , 2, 3, 4, 5} find y(n)
↑

Soln. :

$$y(n) = T[x(n)]$$
$$y(n) = x(n/2)$$

We take different values of n

$$y(0) = x(0) = 1$$
$$y(1) = x(1/2) = \text{No sample}$$
$$y(2) = x(1) = 2$$
$$y(3) = x(3/2) = \text{No sample}$$
$$y(4) = x(2) = 3$$
$$y(5) = x(5/2) = \text{No sample}$$
$$y(6) = x(3) = 4$$
$$y(7) = x(7/2) = \text{No sample}$$
$$y(8) = x(4) = 5$$
$$y(-1) = x(-1/2) = \text{No sample}$$
$$y(-2) = x(-1) = 2$$
$$y(-3) = x(-3/2) = \text{No sample}$$
$$y(-4) = x(-2) = 3$$
$$y(-5) = x(-5/2) = \text{No sample}$$
$$y(-6) = x(-3) = 4$$
$$y(-7) = x(-7/2) = \text{No sample}$$
$$y(-8) = x(-4) = 5$$

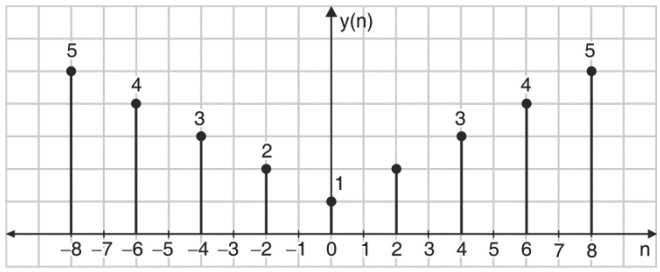


Fig. P. 2.3.5

y (n) is an expanded version of x (n). It is important to note that the shape of x(n) and y(n) remain the same. This is known as up-sampling.

2.3.4 Scalar Multiplication

In this operation, the entire signal is multiplied by a scalar. This is mathematically represented by,

$$y(n) = T[x(n)]$$
$$\text{i.e.,} \quad y(n) = \alpha \cdot x(n)$$

This results in amplification of the signal.

Solved Example

Ex. 2.3.6 : Let x (n) = {1 , 2, 3, 4, 5}, find y (n) = 2 · x (n)
↑

Soln. : $y(n) = T[x(n)]$

$$\therefore \; y(n) = 2 \cdot x(n)$$

We take different values of n

$$y(0) = 2 \cdot x(0) = 2 \times 1 = 2$$
$$y(1) = 2 \cdot x(1) = 2 \times 2 = 4$$
$$y(2) = 2 \cdot x(2) = 2 \times 3 = 6$$
$$y(3) = 2 \cdot x(3) = 2 \times 4 = 8$$
$$y(4) = 2 \cdot x(4) = 2 \times 5 = 10$$

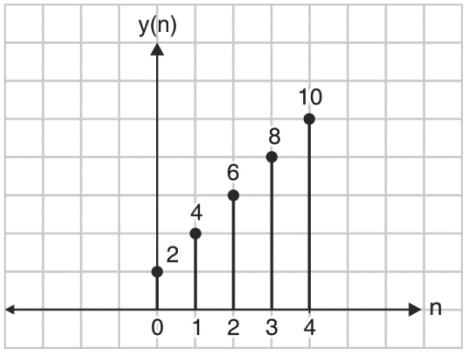


Fig. P. 2.3.6

2.3.5 Signal Addition and Multiplication

In these operations, signals are added or multiplied taking into account of their instances of time (their values of n).

Let us solve an examples to make things clear.

2.3.6 Solved Examples on Signals

Ex. 2.3.7 : Given two signals $x_1(n) = \{1, 1, \underset{\uparrow}{0}, 1, 1\}$ and $x_2(n) = \{2, 2, \underset{\uparrow}{0}, 2, 2\}$.

Perform (i) $x_1(n) + x_2(n)$ (ii) $x_1(n) \times x_2(n)$

Soln. : The first step is to place the two signals exactly one below the other.

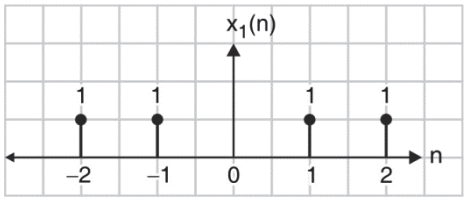


Fig. P.2.3.7

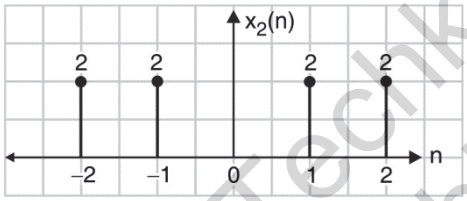


Fig. P.2.3.7(a)

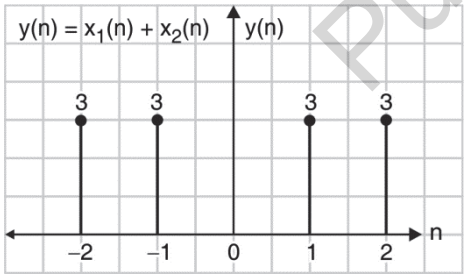


Fig. P.2.3.7(b)

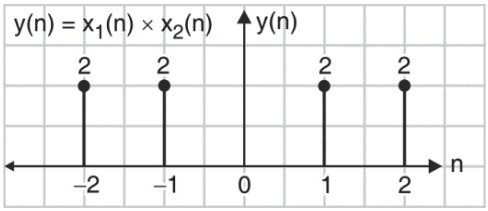


Fig. P. 2.3.7(c)

Bear in mind that we add / multiply only the corresponding terms.

All the operations discussed so far would be used to build up complex operations like the convolution operation.

With classification of signals and operation on signals complete, we now move to discuss various systems.

Ex. 2.3.8 : Draw u(n) – u(n – 5)

Soln. :

u(n) is a unit step while u(n – 5) is a unit step shifted by 5 to the right. We draw them one below the other and perform subtraction.

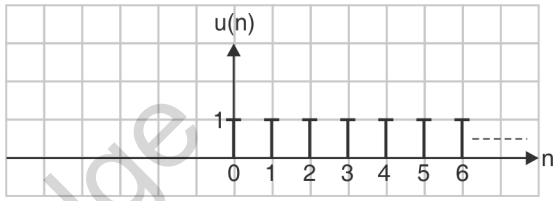


Fig. P.2.3.8

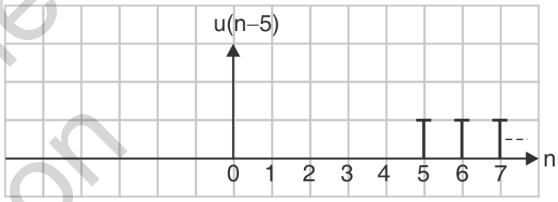


Fig. P.2.3.8(a)

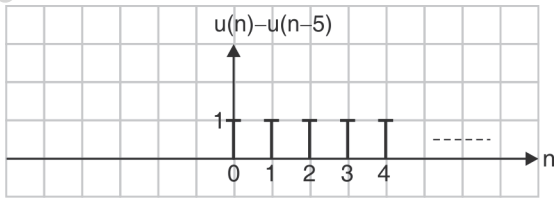


Fig. P. 2.3.8(b)

$\therefore u(n) - u(n-5) = \{1, 1, 1, 1, 1\}$

Ex. 2.3.9 :

For x(n)= {3, 2, 1, 6, 4, 5}, plot the following discrete time signals.

(i) x(n + 1)

(ii) x(–n) u (–n)

(iii) x(n – 1) u(n – 1)

(iv) x(n – 1) u(n)

(v) x(n – 2) **MU - May 2018, 10 Marks**

Soln. :

We begin with drawing x(n).

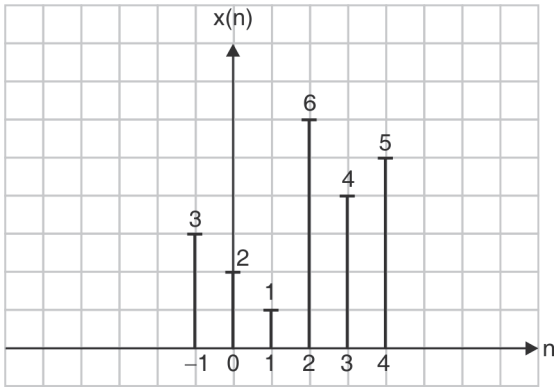


Fig. P. 2.3.9

(i) x(n+1)

This is x(n) shifted to the left by 1.

$$\therefore \quad x(n+1) = \{3, 2, \underset{\uparrow}{1}, 6, 4, 5\}$$

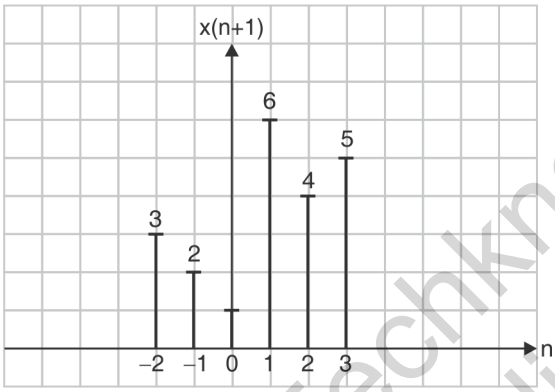


Fig. P.2.3.9(a)

(ii) x(–n) · u(–n)

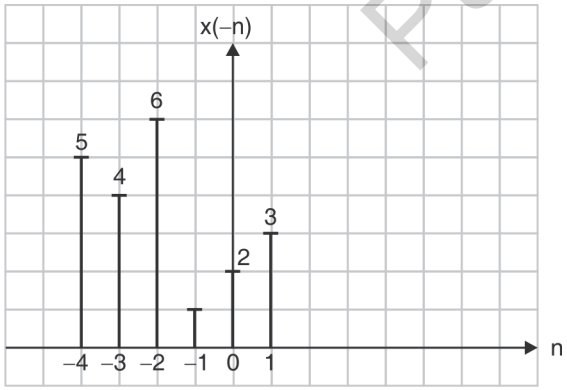


Fig. P.2.3.9(b)

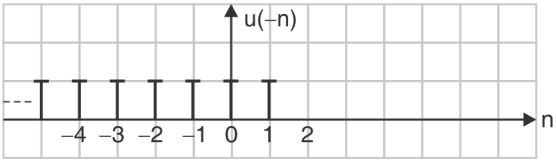


Fig. P.2.3.9(c)

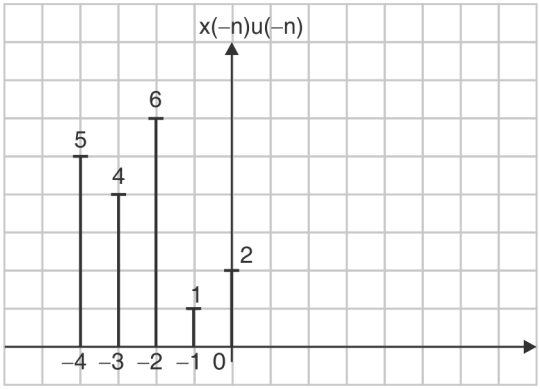


Fig. P.2.3.9(d)

(iii) x(n – 1) · u(–n – 1)

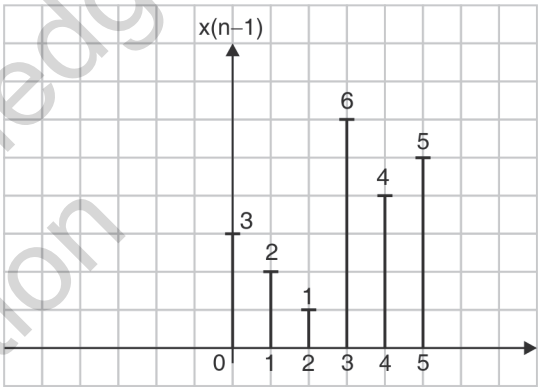


Fig. P.2.3.9(e)

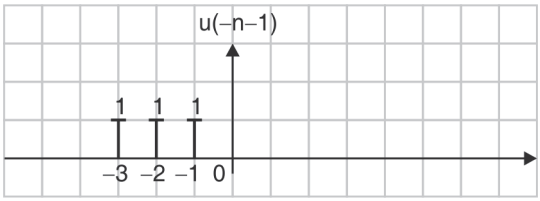


Fig. P.2.3.9(f)

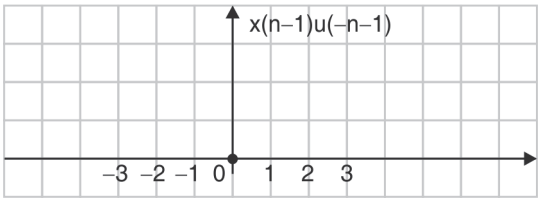


Fig. P.2.3.9(g)

All values are zero.

(iv) x(n – 1) u(n)

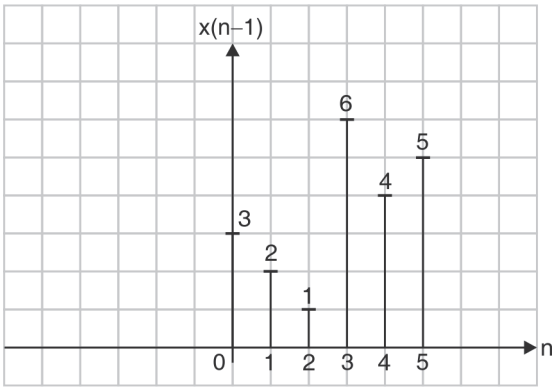


Fig. P.2.3.9(h)

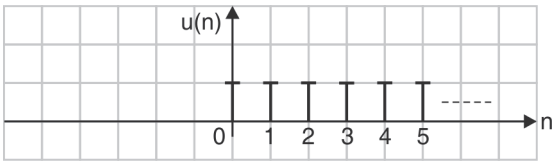


Fig. P.2.3.9(i)

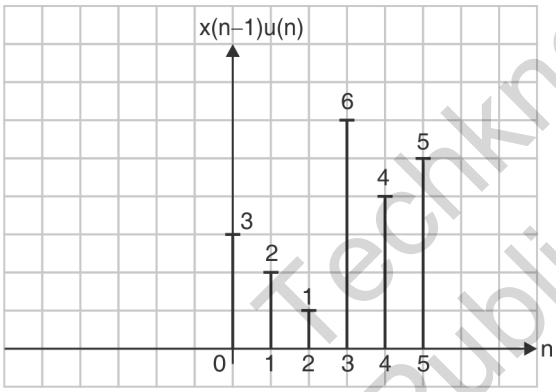


Fig. P.2.3.9(j)

(v) x(n – 2)

This is x(n) shifted to the right by 2.

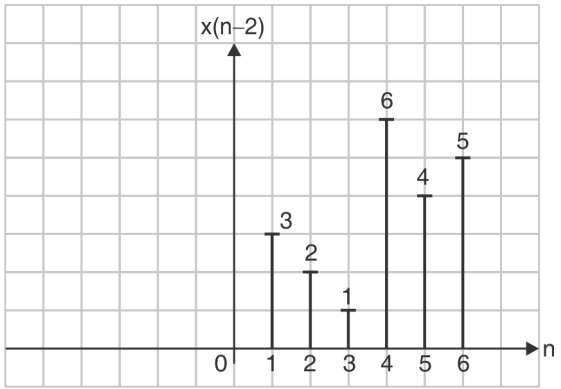


Fig. P.2.3.9(k)

Ex. 2.3.10 : For x(n) = {1, 2, –1, 5, 0, 4}, plot the following discrete time signals.

(i) x(n + 3) (ii) x(–n – 2)

(iii) x(n – 2) δ(n – 2) (iv) x(2n)

MU - Dec. 2018, 10 Marks

Soln. : We begin with drawing x(n)

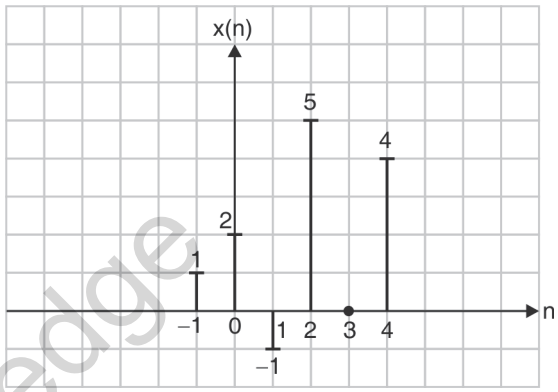


Fig. P.2.3.10

(i) x(n + 3)

This is x(n) shifted to the left by 3.

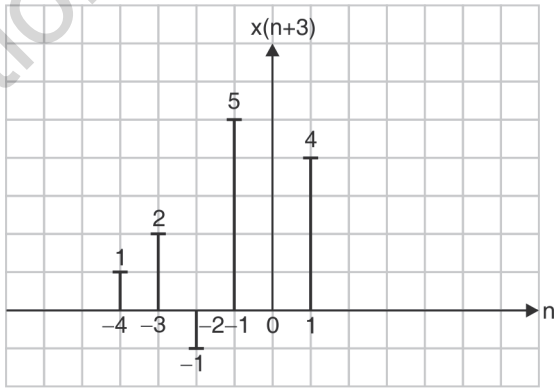


Fig. P.2.3.10(a)

(ii) x(–n – 2)

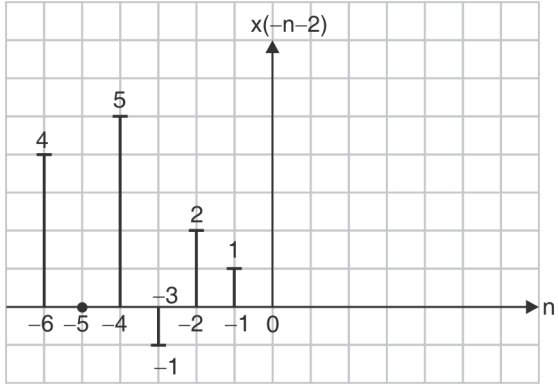


Fig. P.2.3.10(b)

(iii) x(n – 2) δ(n – 2)

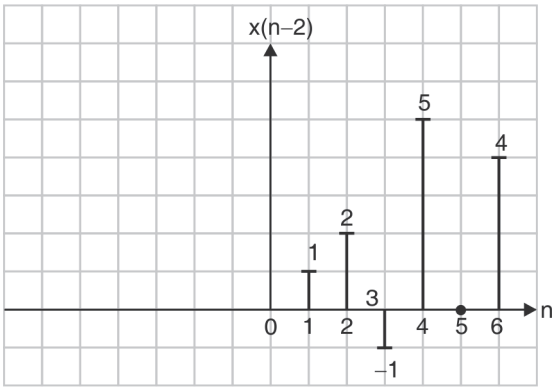


Fig. P.2.3.10(c)

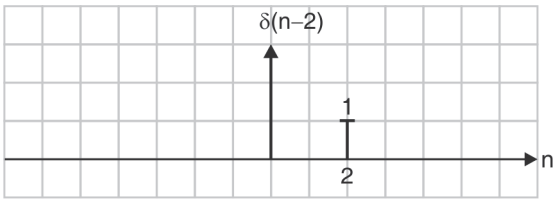


Fig. P.2.3.10(d)

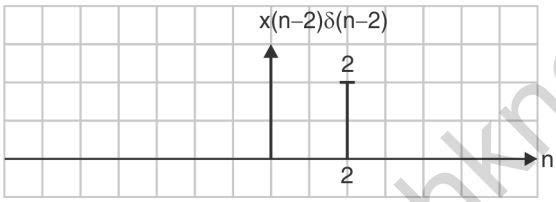


Fig. P.2.3.10(e)

(iv) x(2n)

This is time compression also known as down sampling.

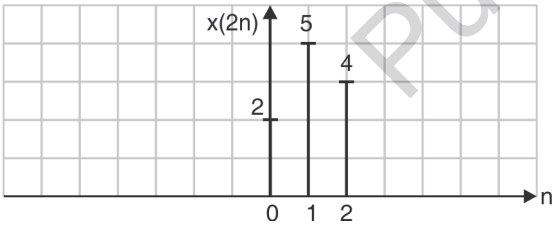


Fig. P.2.3.10(f)

2.4 Discrete Time Systems

As defined earlier, a discrete time system is a device or an algorithm that operates on a discrete time signal. It is mathematically represented as y (n) = T [x (n)]

Discrete time systems are classified according to their general characteristics.

1. Static and Dynamic system (Memory less and Memory systems)
2. Causal and Anti-causal system
3. Linear and Non-linear system
4. Time Variant and Time invariant system
5. Stable and Unstable system

We shall discuss each of them.

1. Static and Dynamic systems

– A system is said to be static if the output of the system depends only on the present input and not on the past or future input.

– Given below are a few examples of static system,

$$y(n) = T[x(n)]$$

$$\text{(i)}\quad y(n) = 4x(n)$$

$$\text{(ii)}\quad y(n) = \log x(n)$$

$$\text{(iii)}\quad y(n) = A\cos x(n)$$

– In each case, y (n) requires only the present value of input x (n)

i.e. from $y(n) = \log x(n)$, we write

$$y(1) = \log x(1)$$

$$y(2) = \log x(2)$$

– Static system are also called memoryless system as we do not need to store pervious input values

– A dynamic system is one in which the output y (n) depends on the present as well as past inputs.

For example, $y(n) = x(n) + x(n-2)$

– In this case, to find out y (4), we need x (4) which is the present input as well as x (4 – 2) = x (2) which is the past input. Hence to find out entire y (n), we need present inputs as well as part inputs.

Some examples of Dynamic systems are :

1. $y(n) = \dfrac{x(n) + x(n-12)}{2}$
2. $y(n) = x(n) - x(n-1)$

– Since past input values need to be stored and retrieved, the system requires a certain amount of memory. Hence Dynamic systems are also called memory systems.

2. Causal and Non-causal system

– A system is said to be causal if the output of the system y (n) at any time n depends only on present and past inputs but does not depend on future inputs.

i.e. $y(n) = x(n) + x(n-2) + x(n-16)$

– If the output of a system also depends on future inputs it known as a Non-Casual systems.

For example, $y(n) = x(n) + x(n+1)$

- In this case, to find out say y (4), we need x (4) which is the present input as well as x (5) which is the future input.
- Hence in an non-causal system, we need to predict the future inputs to get the present input. Given below are a few examples of Causal and Non-Causal systems.

1. $y(n) = \frac{x(n) + x(n-1) + x(n-2)}{3}$ → causal
2. $y(n) = \log(x(n))$ → causal
3. $y(n) = x(2n)$ → Non-causal
4. $y(n) = x(n) - x(n+2)$ → Non-causal

3. Linear and Non-Linear Systems

- A system is said to be linear if it satisfies the superposition principle.
- Superposition principle states that the response of the system to a weighted sum of signals is equal to the corresponding weighted sum of output of the system to individual input signals.

i.e. $T[ax_1(n) + bx_2(n)] = aT[x_1(n)] + bT[x_2(n)]$

A block diagram will make things clearer.

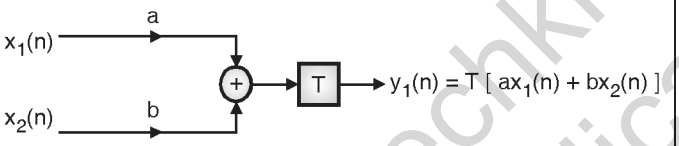


Fig. 2.4.1

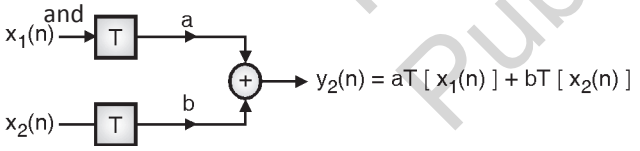


Fig. 2.4.2

Note : The system T is said to be linear if $y_1(n) = y_2(n)$.

Solved Example

Ex. 2.4.1 : Check whether the following systems are linear or non linear.

1. $y(n) = n\,x(n)$
2. $y(n) = e^{x(n)}$
3. $y(n) = x^2(n)$

Soln. : 1. $y(n) = n\,x(n)$

We know

$$y(n) = T[x(n)]$$

$$y(n) = n[x(n)]$$

Here the system multiples the input x(n) with n.

We use two inputs, $x_1(n)$ and $x_2(n)$

We check if $T[x_1(n) + x_2(n)] = T[x_1(n)] + T[x_2(n)]$

(1) $T[x_1(n) + x_2(n)]$

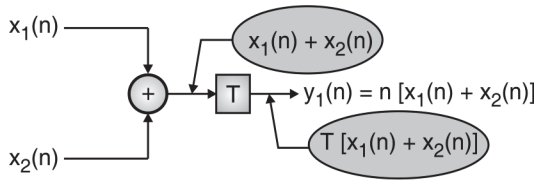


Fig. P. 2.4.1

$$\therefore\ y_1(n) = n[x_1(n) + x_2(n)]$$

Now,

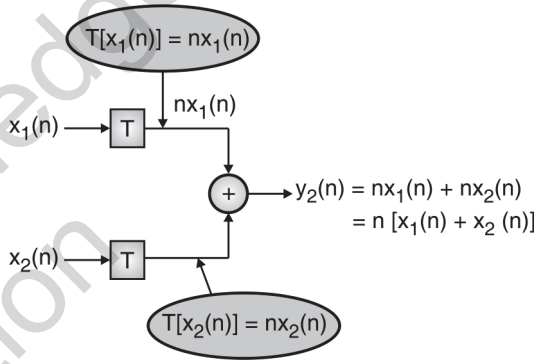


Fig. P. 2.4.1(a)

$$\therefore\ y_2(n) = n[x_1(n) + x_2(n)]$$

$$\therefore\ y_1(n) = y_2(n), \text{ hence system is linear.}$$

(2) $y(n) = e^{x(n)}$

We know,

$$y(n) = T[x(n)]$$

Hence the system in this case computes the exponential of the input.

We use two inputs, $x_1(n)$ and $x_2(n)$

We check for $T[x_1(n) + x_2(n)] = T[x_1(n)] + T[x_2(n)]$

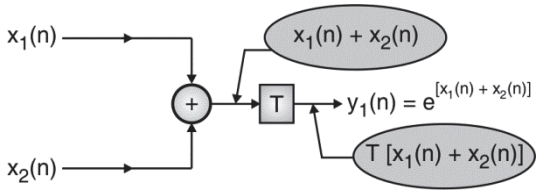


Fig. P.2.4.1(b)

$$y_1(n) = e^{[x_1(n) + x_2(n)]}$$

Now,

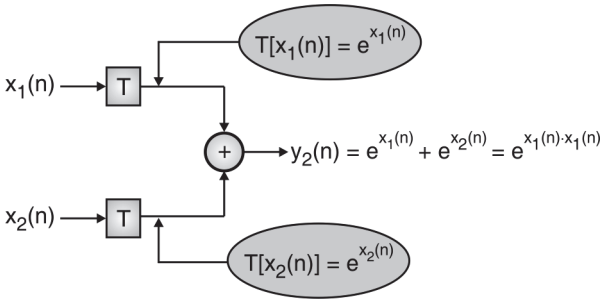


Fig. P.2.4.1(c)

$$y_2(n) = e^{x_1(n) \cdot x_2(n)}$$

Hence $y_1(n) \neq y_2(n)$; Hence system is unstable.

(3) $y(n) = x^2(n)$

We know $y(n) = T[x(n)]$

Hence the system computes the square of the input x(n).

We use to inputs $x_1(n)$ and $x_2(n)$ and check it

$$T[x_1(n) + x_2(n)] = T[x_1(n)] + T[x_2(n)]$$

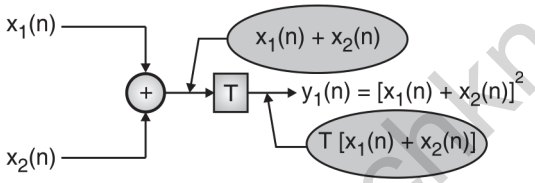


Fig. P. 2.4.1(d)

$$\therefore \quad y_1(n) = [x_1(n) + x_2(n)]^2$$

Now,

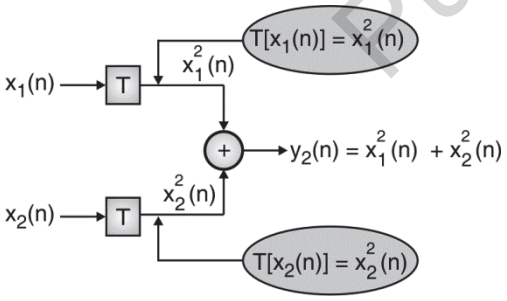


Fig. P. 2.4.1(e)

$$y_2(n) = x_1^2(n) + x_2^2(n)$$

$$\therefore \quad y_1(n) \neq y_2(n)$$

Hence the system is Non-Linear.

4. Time Variant and Time Invariant Systems

A system is Time invariant (Also called shift invariant) if its input-output characteristics do not change with time. If y (n) is the response of the system to input x (n), then y (n – k) will be the response of the system to input x (n – k) i.e. if the input gets shifted by k samples, then the output also shifts by k samples.

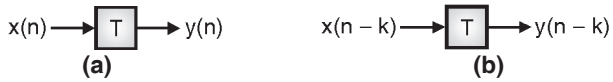


Fig. 2.4.3

To check whether the system in Time invariant or not, we perform the following steps :

Step 1 : Delay input x (n) by k samples and observe the output. We call this output y (n, k).

Step 2 : Delay the output y (n) by k samples i.e., in the given equation y (n), replace n by (n – k). We call this output y (n – k).

If y (n, k) = y (n – k) the system is said to be time invariant or else it is called time variant.

Solved Example

Ex. 2.4.2 : Check whether the following systems are time invariant or time variant.

1. $y(n) = e^{x(n)}$
2. $y(n) = x(n) + x(n-1)$
3. $y(n) = n\,x(n)$

Soln. :

1. $y(n) = e^{x(n)}$

We know $y(n) = T[x(n)]$

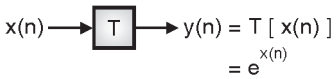


Fig. P.2.4.2

Step 1 : Delay the input

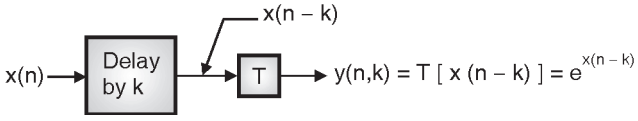


Fig. P.2.4.2(a)

$$\therefore \quad y(n, k) = e^{x(n-k)}$$

Step 2 : Delay the output

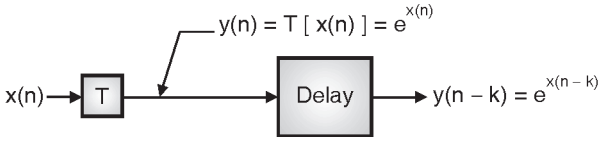


Fig. P.2.4.2(b)

$\therefore$ $y(n-k) = e^{x(n-k)}$

Since y (n, k) = y (n – k), system is Time Invariant.

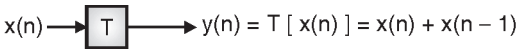


Fig. P.2.4.2(c)

2. y (n) = x (n) + x (n – 1)

$y(n) = T[x(n)]$

Step 1 : Delay the input

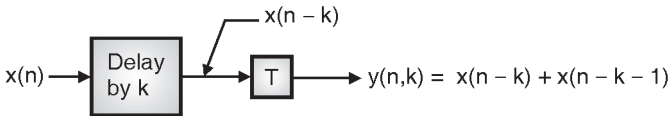


Fig. P.2.4.2(d)

Step 2 : Delay the output

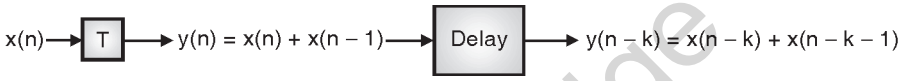


Fig. P.2.4.2(e)

Since, y (n, k) = y (n – k), the system is Time Invariant.

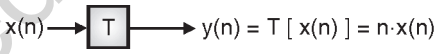


Fig. P.2.4.2 (f)

3. y (n) = n x (n)

$y(n) = T[x(n)]$

Step 1 : Delay the input

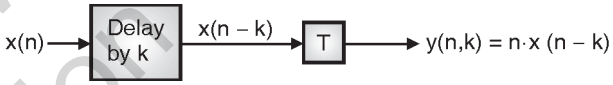


Fig. P.2.4.2 (g)

$y(n, k) = n \cdot x(n-k)$

Step 2 : Delay the output

$y(n-k) = (n-k)\, x\, (n-k)$

Since y (n, k) ≠ y (n – k), the system is Time Variant.

Fig. P.2.4.2 (h)

5. Stable and Unstable systems

A system is said to be stable if for a Bounded input, the system produces a bounded output.

Let M_x be a finite number i.e., $M_x < \infty$. Input is said to be bounded if $|x(n)| \leq M_x < \infty$.

Similarly let M_y is finite number i.e., $M_y < \infty$. Output is said to be bounded if $|y(n)| \leq M_y < \infty$.

Solved Example

Ex. 2.4.3: Check whether the following systems are stable.

i) $y(n) = e^{x(n)}$ ii) $y(n) = x(2n)$

Soln. :

In each of the cases, if x (n) is finite, y (n) will also be finite. Hence both systems are stable.

2.4.1 Representation of a Signal in Terms of Weighted Sum of Shifted Discrete Impulse

Any arbitrary signal can be represented as a summation of shifted and scaled impulses, δ (n). Let us show this with a simple example. Consider an input signal x(n).

x(n) = {0.5, 2, 1, 0.5 , 1, 2, 0.5}

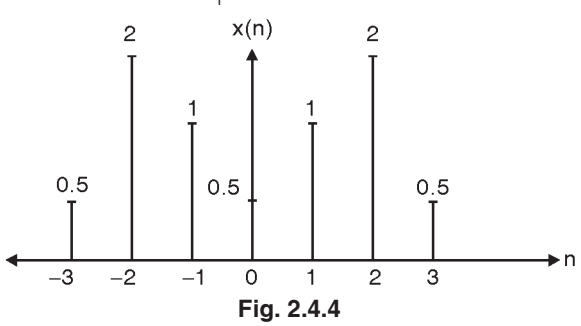


Fig. 2.4.4

The sample x(0) can be obtained by multiplying x(0), the magnitude, with a unit impulse function δ(n).

$$\text{i.e., } x(0) \cdot \delta(n) = \begin{cases} x(0) ; & \text{for } n = 0 \\ 0 ; & \text{for } n \neq 0 \end{cases}$$

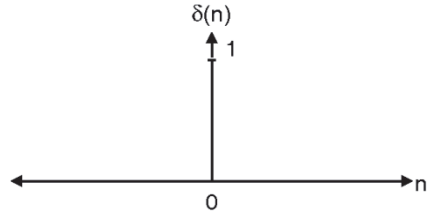


Fig. 2.4.5

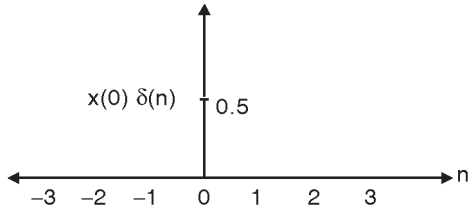


Fig. 2.4.6

We could get the other components of x(n) by shifting the delta function.

$$\text{i.e., } x(-1) \cdot \delta(n+1) = \begin{cases} x(-1) ; & \text{for } n = -1 \\ 0 ; & \text{for } n \neq -1 \end{cases}$$

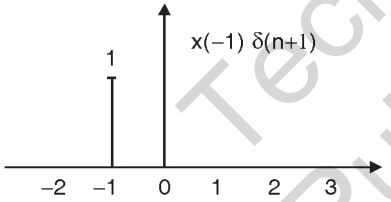


Fig. 2.4.7

In a similar manner

$$x(-2) \cdot \delta(n+2) = \begin{cases} x(-2) ; & \text{for } n = -2 \\ 0 ; & \text{for } n \neq -2 \end{cases}$$

$$x(-3) \cdot \delta(n+3) = \begin{cases} x(-3) ; & \text{for } n = -3 \\ 0 ; & \text{for } n \neq -3 \end{cases}$$

$$x(1) \cdot \delta(n-1) = \begin{cases} x(1) ; & \text{for } n = 1 \\ 0 ; & \text{for } n \neq 1 \end{cases}$$

$$x(2) \cdot \delta(n-2) = \begin{cases} x(2) ; & \text{for } n = 2 \\ 0 ; & \text{for } n \neq 2 \end{cases}$$

$$x(3) \cdot \delta(n-3) = \begin{cases} x(3) ; & \text{for } n = 3 \\ 0 ; & \text{for } n \neq 3 \end{cases}$$

$$\therefore x(n) = x(-3)\,\delta(n+3) + x(-2)\,\delta(n+2) + x(-1)\,\delta(n+1) + x(0)\,\delta(n) + x(1)\,\delta(n-1) + x(2)\,(n-2) + x(3)\,(n-3)$$

∴ In general, we have

$$x(n) = \sum_{k=-\infty}^{\infty} x(n)\,\delta(n-k)$$

Hence a signal can be viewed as a summation of scaled and shifted impulses. This equation will be used in deriving an important equation known as the Convolution sum.

2.5 Impulse Response and Convolution

– As discussed earlier, a simple block diagram of a system is shown in Fig. 2.5.1.

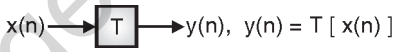


Fig. 2.5.1

– If the input x (n) is a unit impulse δ (n), then the output of the system is known as the impulse response of the system and in denoted as h (n).

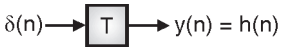


Fig. 2.5.2

$$\therefore h(n) = T[\delta(n)]$$

– The impulse response completely characterizes the system.

– We have shown in the earlier section that any arbitrary input x (n) can be represented as a weighted sum of discrete impulses.

$$\text{i.e., } x(n) = \sum_{k=-\infty}^{+\infty} x(k) \cdot \delta(n-k)$$

We know,

$$y(n) = T[x(n)]$$

$$y(n) = T\left[\sum_{k=-\infty}^{+\infty} x(k) \cdot \delta(n-k)\right] \qquad \ldots(2.5.1)$$

– If we assume the system to be linear, then Equation (2.5.1) can be written as,

$$y(n) = \sum_{k=-\infty}^{+\infty} x(k)\,T[\delta(n-k)]$$

Let, $T[\delta(n-k)] = h(n,k)$

i.e., Let the response to the shifted impulse sequence be denoted by h(n, k).

$$\therefore \quad y(n) = \sum_{k=-\infty}^{+\infty} x(k) \cdot h(n, k) \qquad \ldots(2.5.2)$$

- If we now assume the system to be time invariant,

$$h(n, k) = h(n - k) \qquad \ldots(2.5.3)$$

- Substituting Equation (2.5.3) in Equation (2.5.2), we get ,

$$y(n) = \sum_{k=-\infty}^{+\infty} x(k) \cdot h(n - k)$$

- This equation is called the convolution sum and is one of the most important formula in Digital Signal Processing and Image Processing applications.
- It states that for a Linear Time-Invariant (LTI) system, if the input sequence x(n) and impulse response h(n) are known, y(n) can be found out from the convolution sum.

The convolution sum is represented as,

$y(n) = x(n) * h(n)$ where, * denotes the convolution operation.

2.5.1 Computation of Linear Convolution

There are two methods of computing linear convolution.

1. Graphical method
2. Tabular method

We will begin with discussing the graphical method and then move on to the tabular method which is very easy to perform.

2.5.2 Linear Convolution using Graphical Method

Graphical method gives us a pictorial representation of how complex the convolution operation is Linear convolution is given by the formula.

$$y(n) = \sum_{k=-\infty}^{+\infty} x(k)\, h(n - k) \qquad \ldots(2.5.4)$$

It involves the folding operation, shifting operation, multiplication operation and the summation operation.

In the original formula of convolution, the range of k is $-\infty \le k + \infty$ and the range of n is $-\infty \le n \le +\infty$.

However this ranges depend on the length of x(n) and h(n).

Range of n

Consider the following notations,

y_l = Lowest range of y(n)

y_h = Highest range of y(n)

x_l = Lowest range of x(n)

x_h = Highest range of x(n)

h_l = Lowest range of h(n)

h_h = Highest range of h(n)

The range of n is y(n) will be equal to

$y_l = x_l + h_l$ and $y_h = x_h + h_h$

Let us consider an examples to understand this better.

2.5.2(A) Solved Examples on Graphical Method

Ex. 2.5.1 : What will be the range of y(n) if x(n) = {1,2, 3,1}
↑
h(n) = {1, 2, 2, – 1}
↑

Soln. :

Here $x(n) = \{ \underset{\uparrow}{1}, 2, 3, 1 \}$

$x(0)\ x(1)\ x(2)\ x(3)$

$\therefore \quad x_l = 0, x_h = 3$

Similarly,

$h(n) = \{ 1, \underset{\uparrow}{2}, 2, -1 \}$

$h(-1)\ h(0)\ h(1)\ h(2)$

$\therefore \quad h_l = -1, h_h = 2$

Now $y_l = x_l + h_l$

$= 0 - 1 = -1$

and $y_h = x_h + h_h$

$= 3 + 2 = 5$

Hence the range of n will be $-1 \leq n \leq 5$.

$\therefore$ The output will have values,

$\{y(-1), y(0), y(1), y(2), y(3), y(4), y(5)\}$

Range of k

The range of k will be the same as that of the sequence x(n). In the above example, since $x_l = 0$ and $x_h = 3$, the range of k will be $0 \leq k \leq 3$.

Hence for the above example the formula of convolution reduces to,

$$y(n) = \sum_{k=0}^{3} x(k)\, h(n-k); \qquad -1 \leq n \leq 5$$

Ex 2.5.2 : Consider an audio signal x(n) = {1, 2, 4}. This signal is filtered using a low pass filter having an impulse response h(n) = {1, 1, 1}. Obtain the filtered output signal.

Soln. : Filtering a signal implies the convolution operation. Always remember this. The filtered output is given by the formula.

$$y(n) = x(n) * h(n)$$

i.e.
$$y(n) = \sum_{k=-\infty}^{+\infty} x(k)\, h(n-k)$$

When n = 0, the formula reduces to

$$y(0) = \sum_{k=-\infty}^{+\infty} x(k)\, h(-k)$$

i.e. we need x(k) and h(– k)

The steps involved are as follows :

(1) Draw x(k) and h(– k) one below the other.

(2) Multiply corresponding elements of x(k) and h (– k) and add the resultants.

(3) Shift h(– k) to the right and repeat step (2). Continue shifting to the right till there is no overlap.

(4) Shift h(– k) to the left and repeat step (2).Continue shifting to the left till there is no overlap.

$x(k) = \{1, 2, 4\}$, $h(k) = \{1, 1, 1\}$

Since there are no arrows, the first elements of x(k) and h(k) are taken as the origin.

(1) Draw x(k) and h (– k)

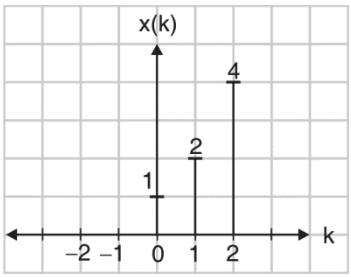


Fig. P. 2.5.2

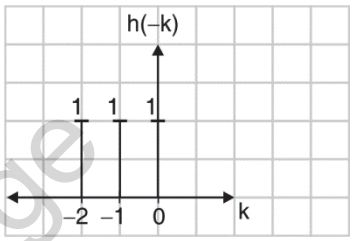


Fig. P. 2.5.2(a)

$$y(0) = \sum x(k)\, h(-k) = 1 \times 1 = 1$$

(2) Shift h(– k) to the right

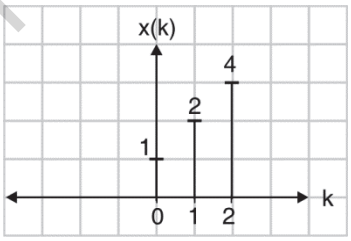


Fig. P. 2.5.2(b)

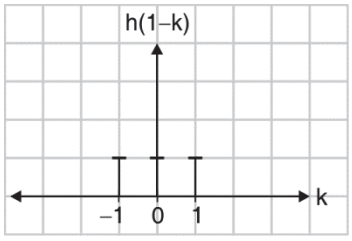


Fig. P. 2.5.2(c)

$$y(1) = \sum x(k)\, h(1-k)$$

$$y(1) = 1(1) + 2(1) = 3$$

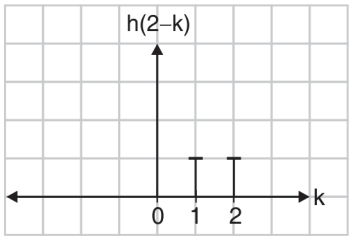


Fig. P. 2.5.2(d)

$y(2) = \sum x(k)\, h(2-k)$

$y(2) = 1(1) + 2(1) + 4(1) = 7$

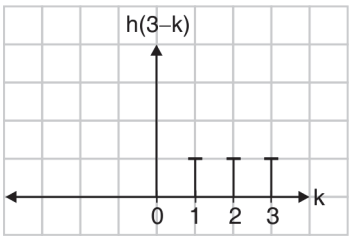


Fig. P. 2.5.2(e)

$y(3) = \sum x(k)\, h(3-k)$

$y(3) = 1(0) + 2(1) + 4(1) = 6$

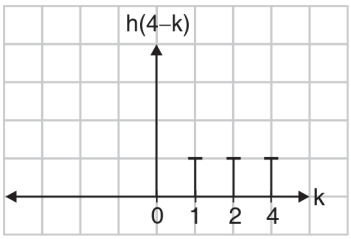


Fig. P. 2.5.2(f)

$y(4) = \sum x(k)\, h(4-k)$

$y(4) = 1(0) + 2(0) + 4(1) = 4$

Now for the 5th shift we have

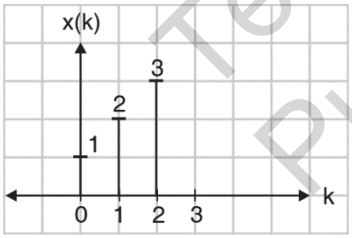


Fig. P. 2.5.2(g)

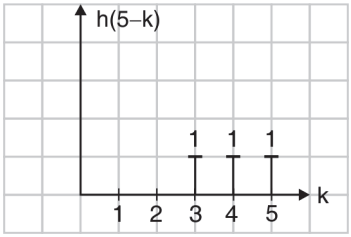


Fig. P. 2.5.2(h)

$y(5) = \sum x(k)\, h(5-k)$

We see there is no overlap between x(k) and h(5 – k) hence the resultant is 0. All subsequent right shifts will give in zero values.

We now shift h(–k) to the left

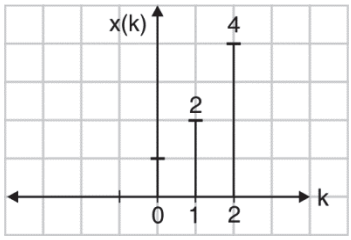


Fig. P. 2.5.2(i)

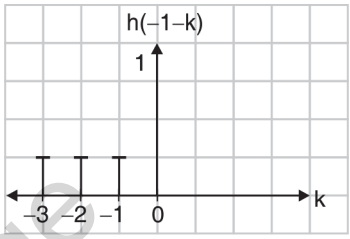


Fig. P. 2.5.2(j)

$y(-1) = 0$

$y(-1) = \sum x(k)\, h(-1-k) = 0$

We see there is no overlap between x(k) and h(–1 – k) hence the resultant is 0. All subsequent left shifts will give us zero values.

$\therefore \quad y(n) = \{y(0), y(1), y(2), y(3), y(4)\}$

$\therefore \quad y(n) = \{1, 3, 7, 6, 4\}$

Hence the filtered audio signal is y(n) = {1, 3, 7, 6,4}. There are two ways to check the result.

(i) Length of y(n) = Length of x(n) + Length of h(n) – 1

$= 3 + 3 - 1 = 5$

We observe that length of y(n) obtained is also 5.

(ii) Sum of elements of x(n) × sum of elements of h(n) = sum of elements of y(n).

i.e. $\sum x(n) \times \sum h(n) = \sum y(n)$

in our case

$\sum x(n) = 1 + 2 + 4 = 7$

$\sum h(n) = 1 + 1 + 1 = 3$

$\therefore \quad \sum x(n) \times \sum h(n) = 7 \times 3 = 21$

Now $\sum y(n) = 1 + 3 + 7 + 6 + 4 = 21$

$\therefore \quad \sum x(n) \times \sum h(n) = \sum y(n)$

Hence the result obtained is correct.

Ex 2.5.3 : Compute linear convolution of the following sequences x(n) = { 1,2, 3, 1}, h(n) = {1, 2, 2, – 1}
↑ ↑

Soln. : $$y(n) = \sum_{k=-\infty}^{+\infty} x(k)\, h(n-k)$$

We draw x(k) and h(– k) and obtain y(0).

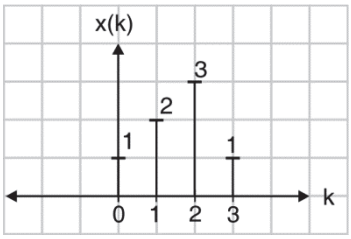


Fig. P. 2.5.3

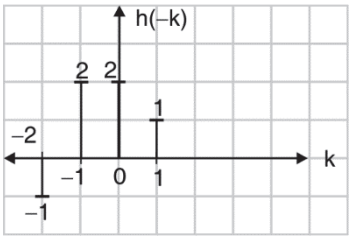


Fig. P. 2.5.3(a)

$y(0) = 1(2) + 2(1) = 4$

We shift h(–k) to the right till there is no overlay

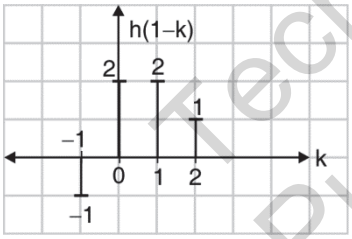


Fig. P. 2.5.3(b)

$y(1) = 1(2) + 2(2) + 3(1) = 9$

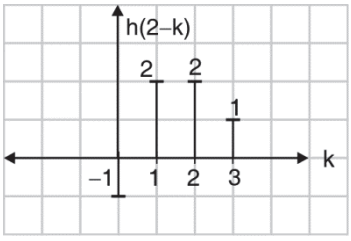


Fig. P. 2.5.3(c)

$y(2) = 1(–1) + 2(2) + 3(2) + 1(1) = 10$

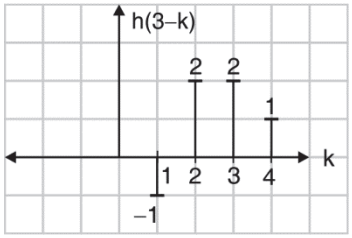


Fig. P. 2.5.3(d)

$y(3) = 2(–1) + 3(2) + 1(2) = 6$

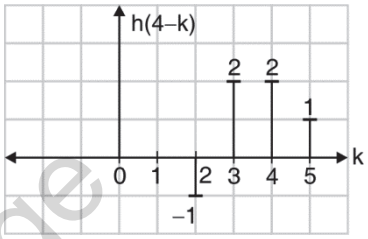


Fig. P. 2.5.3(e)

$y(4) = 3(– 1) + 1(2) = – 1$

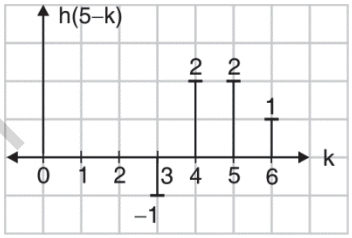


Fig. P. 2.5.3(f)

$y(5) = 1(– 1) = – 1$

beyond this there will be no overlap between x(k) and h(n – k). Hence we stop shifting h(– k) to the right.

We now shift h(–k) to the left.

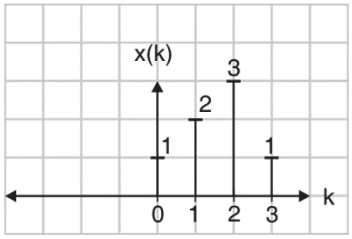


Fig. P. 2.5.3(g)

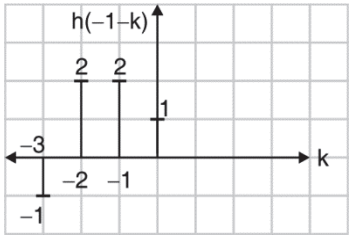


Fig. P. 2.5.3(h)

$y(–1) = 1(1) = 1$

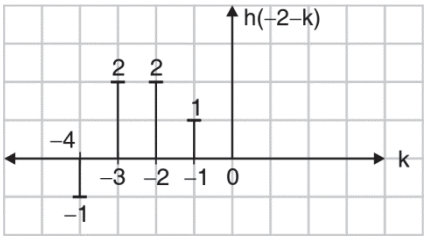


Fig. P. 2.5.3(i)

$$y(-2) = 0$$

Since there is no overlap, we stop shifting h(–k) to the left.

$$\therefore \quad y(n) = \{1, 4, 9, 10, 6, -1, -1\}$$
$$\uparrow$$

We will check if this result is correct

$$\text{length of } y(n) = \text{length of } x(n) + \text{length of } h(n) - 1$$
$$= 4 + 4 - 1 = 7$$

We observe that y(n) obtained has 7 elements

Also

$$\sum x(n) \times \sum h(n) = \sum y(n)$$
$$\sum x(n) = 1 + 2 + 3 + 1 = 7$$
$$\sum h(n) = 1 + 2 + 2 - 1 = 4$$
$$\sum x(n) \times \sum h(n) = 7 \times 4 = 28$$

Now $\sum y(n) = 1 + 4 + 9 + 10 + 6 - 1 - 1 = 28$

Hence the result obtained is correct.

2.5.3 Linear Convolution using Tabular Method

We have learnt how to perform linear convolution using the graphical method. There is a much easier way to performing linear convolution. This is the tabular method.

In this method we follow the given steps.

Step 1 : Form a matrix as shown in Fig. 2.5.3.

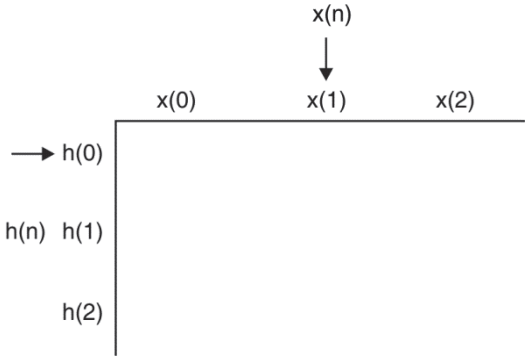


Fig. 2.5.3

We assume arrows are at x(1) and h(0).

We could interchange the positions of x(n) and h(n).

Step 2 : Multiply corresponding elements of x(n) and h(n)

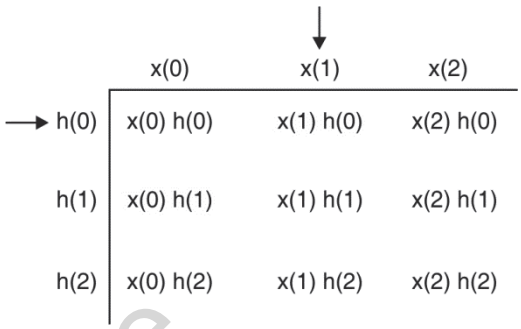


Fig. 2.5.4

Step 3 : We separate the elements diagonally

We shade the section where the arrows intersect. In this case the arrows intersect at x(1)h(0) hence we have shaded the entire diagonal section.

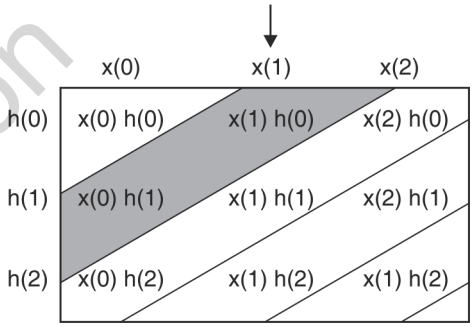


Fig. 2.5.5

Step 4 : We add the elements in each section to obtain y(n). The sum of elements in the shaded section gives us y(0)

$$\therefore \quad y(-1) = x(0) \cdot h(0)$$
$$y(0) = x(0)\,h(1) + x(1)\,h(0)$$
$$y(1) = x(0)\,h(2) + x(1)\,h(1) + x(2)h(0)$$
$$y(2) = x(1)\,h(2) + x(2)\,h(1)$$
$$y(3) = x(2) - h(2)$$
$$\therefore \quad y(n) = \{y(-1), y(0), y(1), y(2), y(3)\}$$
$$\uparrow$$

We will solve a couple of example to understand the tabular method better.

2.5.3(A) Solved Examples on Tabular Method

Ex. 2.5.4 : Perform linear convolution of x(n) = {1,1,0,1, 1}
↑
h (n) = {1, – 2, – 3, 4}
↑

Soln. : We form a matrix of x(n) and h(n)

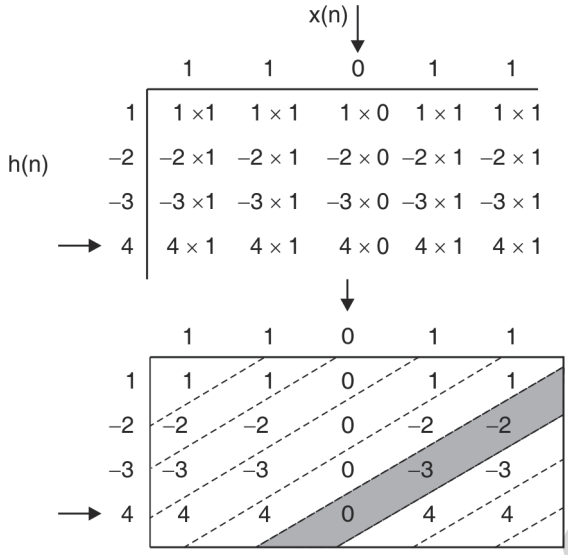


Fig. P. 2.5.4

We add elements from each diagonal section. Since the arrows intersect at 0, we shade that entire segment. The sum of this section will be the origin value.

$\therefore$ y(n) = {1, – 1, – 5, 2, 3, – 5, 1 , 4}
↑

We check whether this result is correct.

Length of y(n) = Length of x(n) + Length of h(n) – 1

= 5 + 4 – 1 = 8

The length of y(n) obtained = 8.

Also, $\sum x(n) \times \sum h(n) = \sum y(n)$

$\sum x(n) = 1 + 1 + 0 + 1 + 1 = 4$

$\sum h(n) = 1 – 2 – 3 + 4 = 0$

$\therefore \sum x(n) \times \sum h(n) = 4 \times 0 = 0$

Now $\sum y(n) = 1 – 1 – 5 + 2 + 3 – 5 + 1 + 4$

$= 0$

$\therefore$ Our result is correct.

Ex. 2.5.5 : The impulse response of a linear time invariant system is h(n) = {1, 2, 1, – 1}. Determine the response of the
↑
system to the input x(n) = {1, 2, 3, 1}

Soln. : Determine the response simply means obtaining y(n).

Given x(n) and h(n), obtaining y(n) means performing linear convolution.

y(n) = x(n) * h(n)

We use the tabular method.

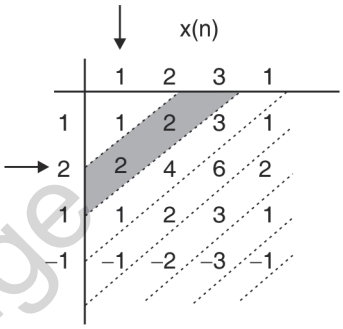


Fig. P. 2.5.5

$\therefore$ y(n) = {1, 4, 8, 8, 3, – 2, – 1}

This is the output response of the system.

Length of y(n) = Length of x(n) + Length of h(n) – 1

= 4 + 4 – 1 = 7

Also, $\sum x(n) \times \sum h(n) = \sum y(n)$

$\sum x(n) = 1 + 2 + 3 + 1 = 7$

$\sum h(n) = 1 + 2 + 1 – 1 = 3$

$\therefore \sum x(n) \times \sum h(n) = 7 \times 3 = 21$

Now $\sum y(n) = 1 + 4 + 8 + 8 + 3 – 2 – 1$

$= 21$

$\therefore$ Our result is correct.

2.6 Correlation

Correlation is basically used when we want to compare two signals. It is a very important operation in signal processing and image processing. Correlation is a measure of the degree to which two signals are similar.

Correlation as a mathematical operation closely resembles convolution.

Correlation of two separate signals is known as cross correlation, while correlation of the signal with itself is known as auto correlation.

1. **Cross correlation :** Correlation between two signals x(n) and y(n), is called cross correlation. It is given by the formula

$$r_{xy}(l) = \sum_{n=-\infty}^{+\infty} x(n) \cdot y(n-l)$$

2. **Auto correlation :** Many times it is necessary to correlate the signal with itself for example determination of time delays between transmitted and received signal.

   Hence the two signals are generated from the same source. It is given by the formula

$$r_{xx}(l) = \sum_{n=-\infty}^{+\infty} x(n)\, x(n-l)$$

Simple method to calculate correlation

We will use one of the important properties of correlation. It is as follows :

$$r_{xy}(l) = x(n) * y(-n) \qquad \ldots(2.6.1)$$

Observe Equation (2.6.1), The L.H.S. indicates cross correlation of x (n) and y (n). R.H.S. term indicates the convolution ('*') operation. It is the convolution of x (n) and y (– n). Here y (– n) is the folded version of y (n).

So this equation implies :

(i) Take signal x (n) as it is.

(ii) Fold sequence y (n) to obtain y (– n).

(iii) Obtain the convolution of x (n) and y (– n).

(iv) The result of convolution will be same as the correlation.

Let us solve the same example using the method.

2.6.1 Solved Examples on Correlation

Ex. 2.6.1 : $x(n) = \{1, 1, \underset{\uparrow}{0}, 1\}$ and $y(n) = \{4, -3, \underset{\uparrow}{-2}, 1\}$

Soln. : In the tabular form simply flip y (n).

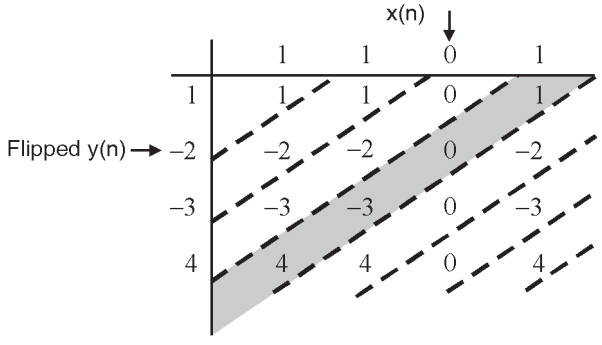


Fig. P. 2.6.1

$$\therefore r_{xy} = \{1, -1, -5, \underset{\uparrow}{2}, 2, -3, 4\}$$

Thus we get

$$r_{xy}(l) = x(n) * y(-n)$$
$$= \{1, -1, -5, \underset{\uparrow}{2}, 2, -3, 4\}$$

Ex. 2.6.2 : Determine the auto-correlation of the following signal :

$$x(n) = \{\underset{\uparrow}{1}, 2, 1, 1\}$$

Soln. :

$$\text{Let } x_1(n) = \{\underset{\uparrow}{1}, 2, 1, 1\},$$
$$x_2(n) = \{\underset{\uparrow}{1}, 2, 1, 1\}$$

We use the tabular method.

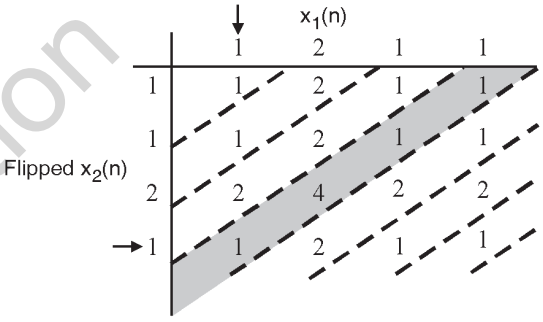


Fig. P.2.6.2

$$\therefore r_{xx} = \{1, 3, 5, \underset{\uparrow}{7}, 5, 3, 1\}$$

Summary

In this chapter we discussed representation of Discrete time signals. We classified discrete time signals as well as discrete time systems. How to operate on discrete time signals was understood. In this chapter we derived one the most important formula which is known as convolution. Two methods of computing the convolution were explained and examples were solved. We also discussed correlation in this chapter

Review Questions

Q. 1 Define discrete time system. Explain any three properties with suitable example.

Q. 2 Define a periodic signal

Q. 3 Define : Even signal and Odd signal

Q. 4 Define power signal.

Q. 5 Sketch and define two standard DT signals $\delta(n)$ and u(n).

Q. 6 With example explain the time scaling and time reversal operations performed on discrete time signals.

Q. 7 With example, explain how linear convolution can be obtained using graphical method.

Q. 8 Explain in brief different properties of convolution.

Q. 9 Explain why the result of linear and circular convolution is not same.

Q.10 How N point circular convolution can be obtained using linear convolution ?

❑❑❑

CHAPTER 3

Discrete Fourier Transform (DFT)

Syllabus :

Introduction to DTFT, DFT, Relation between DFT and DTFT, IDFT, Properties of DFT without mathematical proof (Scaling and Linearity, Periodicity, Time Shift and Frequency Shift, Time, Reversal, Convolution Property and Parsevals' Energy Theorem). DFT computation using DFT properties. Transfer function of DT System in frequency domain using DFT. Linear and Circular Convolution using DFT, Convolution of long sequences, Introduction to 2-D DFT Circular Convolution (without mathematical proof), Linear convolution using Circular Convolution. Spectral Analysis using FFT

3.1 The Fourier Transform

- Fourier transform is named after the French mathematician Jean Baptiste Joseph Fourier who was born in 1768 in Auxerre, France. The Fourier transform is one of the most important transforms in engineering. Most of the branches of engineering owe their growth to the wonderful Fourier transform. The Fourier transform gave a new direction and understanding to the field of Digital Signal Processing.

  So what exactly is the Fourier transform?

- To understand this, Let us look at a simple experiment that we have all conducted in school. Consider Fig. 3.1.1.

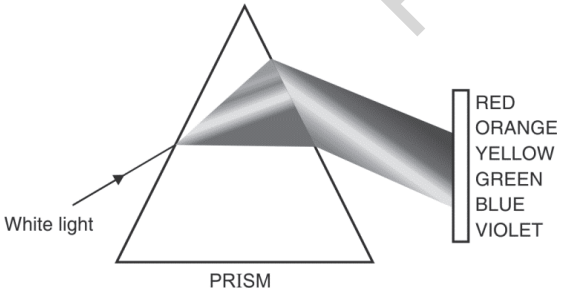


Fig. 3.1.1

- White light when passed through a prism gets split into different wavelengths (colours). Isaac Newton who was born in 1643, in England submitted a paper in 1672 to the Royal Society where he used the word Spectrum to describe the continuous band of colours produced by this apparatus. Newton was the first to note that white light can be split into its constituent colours using a prism.

- Since colours are nothing but wavelengths and wavelengths are related to frequency, splitting of white light into different colours is actually a form of frequency analysis. Newton then placed another prism upside down and showed that constituent colours blended back to give white light again. Fig. 3.1.2.

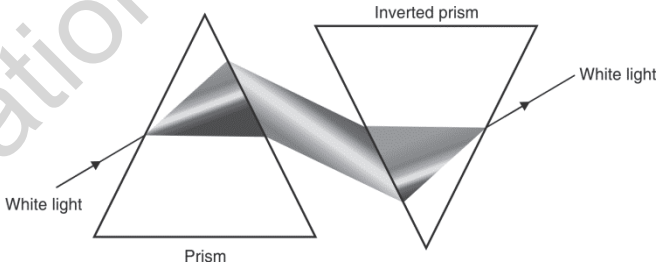


Fig. 3.1.2

Let us summarize what is stated.

- The first prism splits white light into different colours (wavelengths) and the second prism takes these different colours and reproduces white light.
- Joseph Fourier studied the works of Isaac Newton and showed that what the prism does to light, the Fourier transform does to signals.
- White light is replaced by a signal, the first prism by the Fourier transform and the second inverted prism by the Inverse Fourier transform.
- The Fourier transform breaks up a signal of any shape into pure sinusoidal functions. The inverse Fourier transform combines the pure sinusoidal functions to give back the original signal. To put it simply, the Fourier transform tells us the frequencies that are present in a given signal.

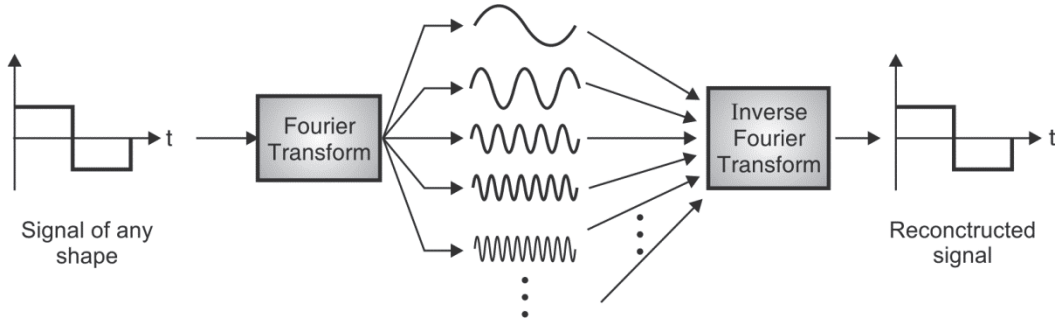


Fig. 3.1.3

3.2 Discrete Time Fourier Transform

- Discrete time Fourier transform, as the name suggest is used when the signal is discrete in time and is a periodic. It is given by the formula.

$$X(\omega) = \sum_{n=\infty}^{+\infty} x(n)\, e^{-j\omega n}$$

- Since x(n) is discrete, we use the summation ($\sum$) operation.
- Hence x(n) is the discrete time signal and X(ω) are the frequency components present in the signal.
- We can obtain x(n) from X (ω) by using the Inverse Discrete Time Fourier Transform (I.D.T.F.T), It is given by the formula,

$$x(n) = \frac{1}{2\pi} \int_{-\pi}^{+\pi} X(\omega)\, e^{+j\omega n}\, d\omega$$

- Here we use the integral operation because X(ω) is continuous.
- The DTFT does not exist for every periodic sequence. A sufficient condition for the existence of the DTFT for a periodic signal x(n) is.

$$\sum_{n=-\infty}^{+\infty} |x(n)| < \infty$$

- In other words, The DTFT exists only if the signal x(n) is absolutely summable.

3.2.1 Solved Examples on DTFT of Standard Signals

Ex. 3.2.1 : Obtain DTFT of unit impulse, δ (n)

Soln. : A unit impulse is shown in Fig. P. 3.2.1.

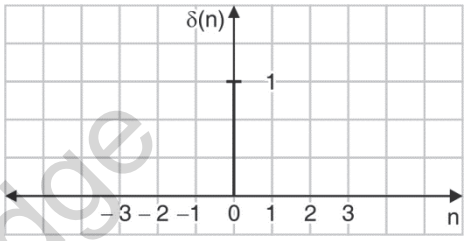


Fig. P. 3.2.1

$$\delta(n) = \begin{cases} 1, & ; n = 0 \\ 0 & ; \text{otherwise} \end{cases}$$

The DTFT is given by the formula,

$$X(\omega) = \sum_{n=-\infty}^{+\infty} x(n)\, e^{-j\omega n}$$

$$\therefore\ X(\omega) = \sum_{n=-\infty}^{+\infty} \delta(n)\, e^{-j\omega n}$$

Since δ (n) exists only at n = 0.

$$\therefore\ X(\omega) = \sum_{n=0} 1.\ e^{-j\omega n} = 1 \times e^{j\omega o}$$

$$\therefore\ X(\omega) = 1$$

$$\therefore\ \delta(n) \xleftrightarrow{FT} 1$$

Ex. 3.2.2 : Obtain DTFT of a unit step signal u(n)

Soln. : A unit step signal is shown in Fig. P. 3.2.2.

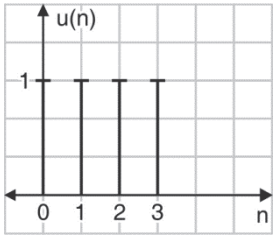


Fig. P. 3.2.2

$$u(n) = \begin{cases} 1 & ; \ n \geq 0 \\ 0 & ; \ \text{otherwise} \end{cases}$$

The DTFT is given by the formula,

$$X(\omega) = \sum_{n=-\infty}^{+\infty} x(n)\, e^{-j\omega n}$$

Since u(n) exists only for $n \geq 0$, we change the limits of the summation.

$$X(\omega) = \sum_{n=0}^{\infty} e^{-j\omega n}$$

$$= \sum_{n=0}^{\infty} (e^{-j\omega})^n$$

We know from the geometric progression formula,

$$\therefore \quad \sum_{n=0}^{\infty} a^n = \frac{1}{1-a} \qquad |a| < 1$$

$$\therefore \quad x(\omega) = \frac{1}{1-e^{-j\omega}}$$

$$\therefore \quad u(n) \xrightarrow{FT} \frac{1}{1-e^{-j\omega}}$$

Ex. 3.2.3 : Obtain DTFT of rectangular pulse

x (n) = A

0 ≤ n ≤ L–1 = 0 ; otherwise

Soln. : The rectangular pulse is an shown in Fig. P. 3.2.3.

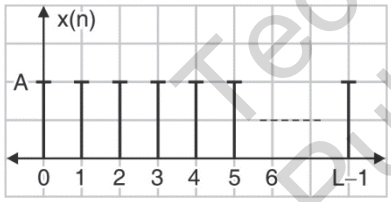


Fig. P. 3.2.3

This is similar to a unit step except that a unit step is infinite in length.

The DTFT is given by the formula,

$$\therefore \quad X(\omega) = \sum_{n=-\infty}^{+\infty} x(n)\, e^{-j\omega n}$$

We change the limits of the summation,

$$\therefore \quad X(\omega) = \sum_{n=0}^{L-1} A\, e^{-j\omega n}$$

$$\therefore \quad A \sum_{n=0}^{\infty} e^{-j\omega n}$$

$$= A \sum_{n=0}^{L-1} (e^{-j\omega})^n$$

We know from the geometric of progression formula.

$$\sum_{K=N_1}^{N_2} a^n = \frac{a^{N1} - a_0^{N_2 - 1}}{1-a}$$

Here $N_1 = 0$, and $N_2 = L - 1$

$$\therefore \quad X(\omega) = A\left[\frac{1-e^{j\omega L}}{1-e^{-j\omega}}\right]$$

$$\therefore \quad \text{Rectangular pulse } A \xleftrightarrow{FT} A\left[\frac{1-e^{-j\omega L}}{1-e^{-j\omega}}\right]$$

Ex. 3.2.4 : Obtain DTFT and sketch the magnitude spectrum for,

$$X(n) = u(n) - u(n-4)$$

Soln. :

Here u(n) is a step input while u(n – 4) is a unit step shifted by 4.

Hence,

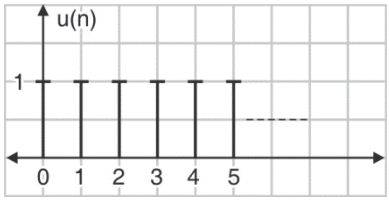


Fig. P. 3.2.4

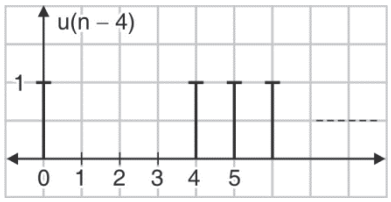


Fig. P. 3.2.4(a)

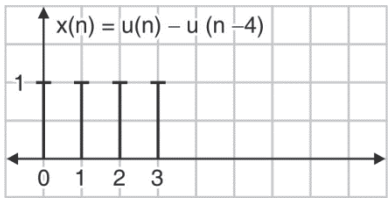


Fig. P. 3.2.4(b)

$$\therefore \quad x(n) = \{1,1,1,1\}$$

$$\uparrow$$

The DTFT is given by the formula,

$$\therefore \quad X(\omega) = \sum_{n=-\infty}^{+\infty} x(n) e^{-j\omega n}$$

In our case

$$\therefore \quad x(\omega) = \sum_{n=0}^{3} 1.\, e^{-j\omega n}$$

Since the limits of the summation are small, we open up the summation.

$$\therefore \quad X(\omega) = e^{-j\omega o} + e^{-j\omega 1} + e^{-j\omega 2} + e^{-j\omega 3}$$

$$\therefore \quad X(\omega) = 1 + e^{-j\omega} + e^{-j2\omega} + e^{-j3\omega}$$

We need to sketch the magnitude spectrum,

$$e^{-j\omega} = \cos\omega - j\sin\omega$$

$$\therefore \quad |e^{-j\omega}| = \sqrt{\cos^2\omega + \sin^2\omega} = 1$$

Similarly,

$$|e^{-j2\omega}| = \sqrt{\cos^2 2\omega + \sin^2 2\omega} = 1$$

and $$|e^{-j3\omega}| = \sqrt{\cos^2 3\omega + \sin^2 3\omega} = 1$$

$$\therefore \quad |X(\omega)| = 1 + 1 + 1 + 1 = 4$$

$$\therefore \quad |X(\omega)| = 4 \qquad \text{...for all } \omega$$

$\therefore$ The Magnitude Spectrum is shown in Fig. P. 3.2.4(c).

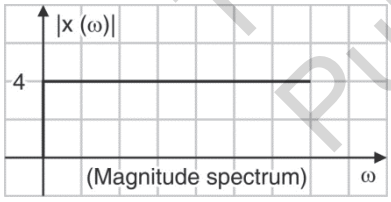


Fig. P. 3.2.4(c)

Note : The spectrum is continuous.

3.3 Relationship between DTFT and DFT

The DTFT is given by the formula,

$$X(k) = \sum_{n=\infty}^{+\infty} x(n)\, e^{-j\omega n} \qquad \ldots (3.3.1)$$

Since the DTFT is continuous, we sample the ω axis to make it discrete.

We sample ω such that

$$\omega = \frac{2\pi k}{N} \qquad \ldots (3.3.2)$$

Where, N = Number of sample.

Substituting Equation (3.3.2) is Equation (3.3.1), we get,

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j\,2\pi kn}{N}} \qquad \ldots (3.3.3)$$

N is normally equal to the length of the input signal x (n)

Equation (3.3.3) is called the DFT and is one of the most important transforms in engineering.

3.4 Discrete Fourier Transform (DFT)

- The DFT is a modification of the Fourier transform so as to use it for discrete time signals and to obtain a discrete frequency spectrum. We shall now proceed to explain the DFT.
- Remember, the objective of the DFT is to obtain the frequencies present in the signal.
- Consider a discrete time input signal x(n) of length N. The DFT of x(n) is denoted by X(k) and is given by,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j\,2\pi kn}{N}}$$

$$k = 0,\ 1,\ 2,\ \ldots N-1 \qquad \ldots(3.4.1)$$

- Since the input signal x(n) is of length N, the summation varies from 0 to N – 1 and Equation (3.4.1) is called as N-point DFT.
- As mentioned in the earlier section, Equation (3.4.1) acts as the prism, breaking down the input signal into various frequency components.
- We can obtain x(n) from X(k) using the Inverse Discrete Fourier Transform IDFT which is given by the formula,

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, e^{\frac{+j\,2\pi kn}{N}}$$

$$n = 0,\ 1,\ 2,\ \ldots N-1 \qquad \ldots(3.4.2)$$

Equation (3.4.2) is called as N–point IDFT.

- This acts as the inverted prism which gives us the original signal x(n) by combining the constituent frequencies.

Note : $\frac{1}{N}$ in Equation (3.4.2) is a scaling function.

- This book uses the scaling function in the IDFT formula while other books choose to use it in the DFT formula. Some other books use $\frac{1}{\sqrt{N}}$ in both DFT and IDFT formulae.
- MATLAB uses $\frac{1}{N}$ in the IDFT formula and hence we too choose to use it there.
- Unless specified the length of x(n) and X(k) are the same.

3.4.1 Solved Examples on DFT of Simple Signals

In most of the examples, we begin by changing the limits of the summation depending on the input signal.

Ex. 3.4.1 : Obtain DFT of a unit impulse signal $\delta(n)$.

Soln.: Here $x(n) = \delta(n)$

The DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; k = 0, 1, \ldots\ldots N-1$$

$$\therefore \quad X(k) = \sum_{n=0}^{N-1} \delta(n)\, e^{\frac{-j2\pi kn}{N}}$$

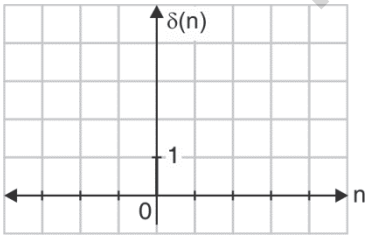


Fig. P.3.4.1

Since $\delta(n)$ exists only at n = 0, for all other values of the summation the result will be equal to zero.

$$\therefore \quad X(k) = \delta(0)\, e^{\frac{-j2\pi k \cdot 0}{N}} = \delta(0) \cdot e^{0}$$

$$\therefore \quad X(k) = 1 \qquad (\because \delta(0) = 1)$$

Hence, $\delta(n) \overset{DFT}{\longleftrightarrow} 1$

Ex. 3.4.2 : Obtain DFT of $\delta(n - n_0)$.

Soln.: Here $x(n) = \delta(n - n_0)$

$\delta(n - n_0)$ is basically $\delta(n)$ shifted to the right by n_0.

$\therefore$ $\delta(n - n_0)$ is 1 at n_0 and 0 at all other values of n.

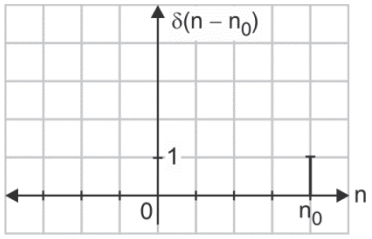


Fig. P.3.4.2

The DFT is given by the equation,

$$\therefore \quad X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; k = 0, 1, \ldots N-1$$

$$\therefore \quad X(k) = \sum_{n=0}^{N-1} \delta(n - n_0)\, e^{\frac{-j2\pi kn}{N}}$$

Since $\delta(n - n_0)$ exist at $n = n_0$, for all other values of the summation, the result will be equal to zero.

$$\therefore \quad X(k) = \delta(n - n_0)\, e^{\frac{-j2\pi k n_0}{N}}$$

$$\therefore \quad X(k) = 1 \cdot e^{\frac{-j2\pi k n_0}{N}} \qquad (\because \delta(n - n_0) = 1)$$

$$\therefore \delta(n - n_0) \overset{DFT}{\longleftrightarrow} e^{\frac{-j2\pi k n_0}{N}}$$

Similarly,

$$\therefore \delta(n + n_0) \overset{DFT}{\longleftrightarrow} e^{\frac{+j2\pi k n_0}{N}}$$

Ex. 3.4.3 : Obtain DFT of $u(n) - u(n-4)$.

Soln.: Here $x(n) = u(n) - u(n-4)$

Let us draw $u(n) - u(n-4)$ to understand what the input is.

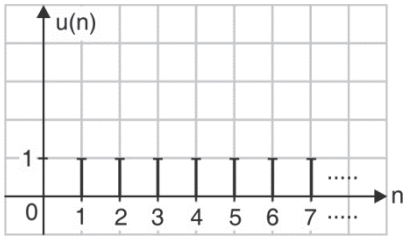


Fig. P.3.4.3

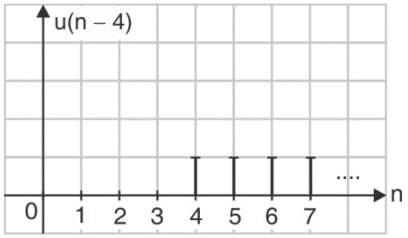


Fig. P.3.4.3(a)

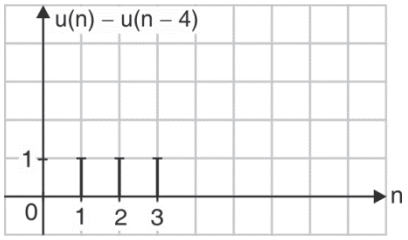


Fig. P.3.4.3(b)

u(n) is a unit step, while u (n – 4) is a unit step shifted to the right by 4.

From the Fig. P. 3.4.3(a) it is clear that,

$$x(n) = \{\underset{\uparrow}{1}, 1, 1, 1\}$$

The DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; k = 0, 1, 2, \ldots N-1$$

We now change the limits of the summation as x(n) exists only from 0 to 3.

$\therefore$ N = 4

$\therefore$ Range of k is also 0 to 3.

$$\therefore \quad X(k) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi kn}{4}} \quad ; k = 0, 1, 2, 3,$$

For each value of k, we vary n from 0 to 3.

(i) k = 0

$$X(0) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 0 \cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 0 \cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 0 \cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 0 \cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 0 \cdot 3}{4}}$$

$$= x(0)\, e^{0} + x(1)\, e^{0} + x(2)\, e^{0} + x(3)\, e^{0}$$

$$= x(0) + x(1) + x(2) + x(3)$$

$$= 1 + 1 + 1 + 1$$

$$\therefore \quad X(0) = 4$$

(ii) k = 1

$$X(1) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 1 \cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 1 \cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 1 \cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 1 \cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 1 \cdot 3}{4}}$$

$$= x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi}{2}} + x(2)\, e^{-j\pi} + x(3)\, e^{\frac{-j3\pi}{2}}$$

$$= 1 + 1 \cdot e^{\frac{-j\pi}{2}} + 1 \cdot e^{-j\pi} + 1 \cdot e^{\frac{-j3\pi}{2}}$$

$$= 1 + 1\left(\cos\frac{\pi}{2} - j\sin\frac{\pi}{2}\right) + 1\,(\cos\pi - j\sin\pi) + 1\left(\cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right)$$

$$= 1 + (0 - j) + (-1 - 0) + (0 + j)$$

$$= 1 - j - 1 + j$$

$$\therefore X(1) = 0$$

(iii) k = 2

$$X(2) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 2 \cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 2 \cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 2 \cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 2 \cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 2 \cdot 3}{4}}$$

$$= x(0)\, e^{0} + x(1)^{-j\pi} + x(2)\, e^{-j2\pi} + x(3)\, e^{-j3\pi}$$

$$= 1 + 1\,(\cos\pi - j\sin\pi) + 1 \cdot (\cos 2\pi - j\sin 2\pi) + 1\,(\cos 3\pi - j\sin 3\pi)$$

$$= 1 + (-1 - 0) + (1 - 0) + (-1 - 0) = 0$$

$$\therefore \quad X(2) = 0$$

(iv) k = 3

$$X(3) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 3 \cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 3 \cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 3 \cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 3 \cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 3 \cdot 3}{4}}$$

$$= x(0)\, e^{0} + x(1)\, e^{\frac{-j3\pi}{2}} + x(2)\, e^{-j3\pi} + x(3)\, e^{\frac{-j9\pi}{2}}$$

$$= 1 + 1\left(\cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right) + 1 \cdot (\cos 3\pi - j\sin 3\pi) + 1\left(\cos\frac{9\pi}{2} - j\sin\frac{9\pi}{2}\right)$$

$= 1 + 1 \cdot (0 + j) + 1 (-1 - 0) + (0 - j)$

$= 1 + j - 1 - j$

$= 0$

$\therefore \quad X(3) = 0$

$\therefore \quad X(k) = \{ X(0),\ X(1),\ X(2),\ X(3)\}$

$\therefore \quad X(k) = \{4, 0, 0, 0\}$

Hence, $\{1, 1, 1, 1\} \overset{DFT}{\longleftrightarrow} \{4, 0, 0, 0\}$

Let us draw the two sequences obtained.

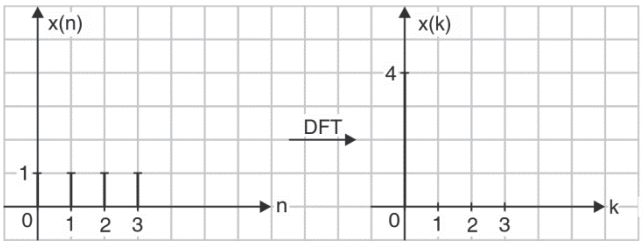


Fig. P.3.4.3(c)

Ex. 3.4.4 : Compute the DFT of the sequence given as $x(n) = (-1)^n$ for

(i) N = 3

(ii) N = 4

Soln.:

(i) N = 3

The range of 'n' is n = 0 to N – 1 that means n = 0 to 2.

Given sequence is , $x(n) = (-1)^n$.

For $n = 0 \Rightarrow x(0) = (-1)^0 = 1$

$n = 1 \Rightarrow x(1) = (-1)^1 = -1$

$n = 2 \Rightarrow x(2) = (-1)^2 = 1$

$\therefore \quad x(n) = \{ 1, -1, 1 \}$
$\uparrow$

According to the definition of DFT

$$X(k) = \sum_{n=0}^{N-1} x(n)\ e^{\frac{-j2\pi kn}{N}} \qquad \ldots k = 0 \text{ to } N-1$$

$$\therefore \quad X(k) = \sum_{n=0}^{2} x(n)\ e^{\frac{-j2\pi kn}{3}} \qquad \ldots k = 0 \text{ to } 2$$

For k = 0

$$X(0) = \sum_{n=0}^{2} x(n)\ e^{\frac{-j2\pi 0 \cdot n}{3}}$$

$X(0) = x(0)\ e^0 + x(1)\ e^0 + x(2)\ e^0$

$\therefore \quad X(0) = x(0) + x(1) + x(2) = 1 - 1 + 1$

$\therefore \quad \mathbf{X(0) = 1}$

For k = 1

$$X(1) = \sum_{n=0}^{2} x(n)\ e^{\frac{-j2\pi n}{3}}$$

$$\therefore \quad X(1) = x(0)\ e^0 + x(1)\ e^{\frac{-j2\pi}{3}} + x(2)\ e^{\frac{-j4\pi}{3}}$$

$$\therefore \quad X(1) = 1 - 1 \cdot e^{\frac{-j2\pi}{3}} + 1\ e^{\frac{-j4\pi}{3}}$$

$\therefore \quad X(1) = 1 - (-0.5 - j\ 0.866) + (-0.5 + j\ 0.866)$

$\therefore \quad \mathbf{X(1) = 1}$

For k = 2

$$X(2) = \sum_{n=0}^{2} x(n)\ e^{\frac{-j2\pi \cdot 2n}{3}}$$

$$\therefore \quad X(2) = x(0)\ e^0 + x(1)\ e^{\frac{-j4\pi}{3}} + x(2)\ e^{\frac{-j8\pi}{3}}$$

$\therefore \quad X(2) = 1 - (-0.5 + j\ 0.866) + (-0.5 - j\ 0.866)$

$\therefore \quad \mathbf{X(2) = 1}$

Thus DFT is, $X(k) = \{1, 1, 1\}$

(ii) N = 4

The range of 'n' is n = 0 to 3.

The sequence x(n) can be written as,

$x(n) = \{ 1, -1, 1, -1\}$
$\uparrow$

According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\ e^{\frac{-j\,2\pi\,k\,n}{N}} \qquad \ldots k = 0 \text{ to } N-1$$

$$\therefore \quad X(k) = \sum_{n=0}^{3} x(n)\ e^{\frac{-j2\pi\,k\,n}{4}} \qquad \ldots k = 0 \text{ to } 3$$

$$\therefore \quad X(k) = \sum_{n=0}^{3} x(n)\ e^{\frac{-j\,\pi\,kn}{2}}$$

For k = 0 $\quad X(0) = \sum_{n=0}^{3} x(n)\ e^0 = \sum_{n=0}^{3} x(n)$

$\therefore \quad X(0) = x(0) + x(1) + x(2) + x(3)$

$= 1 - 1 + 1 - 1$

$\therefore \quad \mathbf{X(0) = 0}$

For k = 1

$$X(1) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j\pi n}{2}}$$

$$\therefore \quad X(1) = x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi}{2}} + x(2)\, e^{-j\pi} + x(3)\, e^{\frac{-j3\pi}{2}}$$

$$\therefore \quad X(1) = 1 - (-0 - j\,1) + (-1 - j\,0) - (0 + j\,1)$$

$$\therefore \quad \mathbf{X(1) = 0}$$

For k = 2

$$X(2) = \sum_{n=0}^{3} x(n)\, e^{-j\pi n}$$

$$\therefore \quad X(2) = x(0)\, e^{0} + x(1)\, e^{-j\pi} + x(2)\, e^{-j2\pi} + x(3)\, e^{-j3\pi}$$

$$\therefore \quad X(2) = 1 - (-1 - j\,0) + (1)\,(1 - j\,0) - 1\,(-1 - j\,0)$$

$$\therefore \quad \mathbf{X(2) = 4}$$

For k = 3

$$X(3) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j3\pi n}{2}}$$

$$\therefore \quad X(3) = x(0)\, e^{0} + x(1)\, e^{\frac{-j3\pi}{2}} + x(2)\, e^{-j3\pi} + x(3)\, e^{\frac{-j9\pi}{2}}$$

$$\therefore \quad X(3) = 1 - (0 + j\,1) + (-1 - j\,0) - (0 - j\,1)$$

$$\therefore \quad \mathbf{X(3) = 0}$$

Thus DFT is,

$$\mathbf{X(k) = \{0, 0, 4, 0\}}$$

Ex. 3.4.5 : Obtain the DFT of the sequence

$x(n) = \delta(n) + 3\,\delta(n-2) + 2\,\delta(n-3)$

Soln. : The given sequence is shown below. The length of the sequence is from 0 to 3.

$$\therefore \quad N = 4$$

We know, $\delta(n) \xleftrightarrow{DFT} 1$

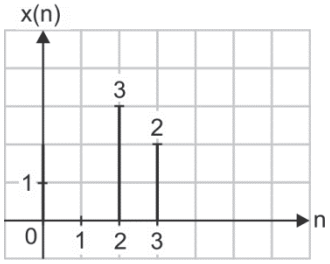


Fig. P.3.4.5

And $\delta(n - n_0) \xleftrightarrow{DFT} e^{\frac{-j2\pi k n_0}{N}}$

$$\therefore \quad 3\,\delta(n-2) \xleftrightarrow{DFT} 3\,e^{\frac{-j2\pi k\cdot 2}{4}} = 3e^{-j\pi k}$$

$$\text{and} \quad 2\,\delta(n-3) \xleftrightarrow{DFT} 2\,e^{\frac{-j2\pi k\cdot 3}{4}} = 2e^{\frac{-j3\pi k}{2}}$$

$\therefore$ DFT of the given sequence is,

$$\mathbf{X(k) = 1 + 3e^{-j\pi k} + 2\,e^{-j3\pi k}}$$

Ex. 3.4.6 : Compute DFT of $x(n) = \cos\left(\frac{2\pi}{N} k_0 n\right)$.

Soln. :

We have, $\cos\theta = \frac{e^{j\theta} + e^{-j\theta}}{2}$

Thus given sequence can be written as,

$$x(n) = \frac{1}{2}\left[e^{\frac{+j2\pi k_0 n}{N}} + e^{\frac{-j2\pi k_0 n}{N}}\right]$$

According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi k n}{N}}$$

$$\therefore \quad X(k) = \frac{1}{2}\sum_{n=0}^{N-1}\left[e^{\frac{+j2\pi k_0 n}{N}} + e^{\frac{-j2\pi k_0 n}{N}}\right]\cdot e^{\frac{-j2\pi k n}{N}}$$

$$\therefore \quad X(k) = \frac{1}{2}\cdot\sum_{n=0}^{N-1} e^{\frac{+j2\pi k_0 n}{N}}\cdot e^{\frac{-j2\pi k n}{N}} + \frac{1}{2}\cdot\sum_{n=0}^{N-1} e^{\frac{-j2\pi k_0 n}{N}}\cdot e^{\frac{-j2\pi k n}{N}}$$

$$\therefore \quad X(k) = \frac{1}{2}\sum_{n=0}^{N-1}\left[e^{\frac{j2\pi(k_0 - k)}{n}}\right]^n + \frac{1}{2}\sum_{n=0}^{N-1}\left[e^{\frac{-j2\pi(k_0 + k)}{N}}\right]^n$$

Now we have standard summation formula,

$$\sum_{n=N_1}^{N_2} a^n = \begin{cases} \dfrac{a^{N_1} - a^{N_2+1}}{1-a}, & a \neq 1 \\ N_2 - N_1 + 1, & a = 1 \end{cases}$$

Here $N_1 = 0$, $N_2 = N - 1$.

$$\therefore\ X(k) = \frac{1}{2}\left[\frac{\left(e^{\frac{j2\pi(k_0-k)}{N}}\right)^0 - \left(e^{\frac{j2\pi(k_0-k)}{N}}\right)^{N-1+1}}{1-e^{\frac{j2\pi(k_0-k)}{N}}}\right]$$

$$+\frac{1}{2}\left[\frac{\left(e^{\frac{-j2\pi(k_0+k)}{N}}\right)^0 - \left(e^{\frac{-j2\pi(k_0+k)}{N}}\right)^{N-1+1}}{1-e^{\frac{-j2\pi(k_0+k)}{N}}}\right]$$

$$= \frac{1}{2}\left[\frac{1-e^{j2\pi(k_0-k)}}{1-e^{\frac{j2\pi}{N}(k_0-k)}}\right] + \frac{1}{2}\left[\frac{1-e^{-j2\pi(k_0+k)}}{1-e^{\frac{-j2\pi}{N}(k_0+k)}}\right]$$

Ex. 3.4.7 : Compute the DFT of x(n) = {1, 2, 3, 4}. Draw the corresponding magnitude and phase spectrum.

Soln.: x(n) = {1, 2, 3, 4}

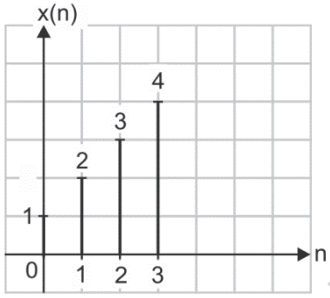


Fig. P.3.4.7

DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi k\cdot n}{4}} \quad ; k = 0, 1, \ldots, N-1$$

Since the length of x(n) is 4, N = 4

$$\therefore\ X(k) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi k n}{4}} \quad ; k = 0, 1, 2, 3,$$

For each value of k, we vary n from 0 to 3.

(i) k = 0

$$\therefore\ X(0) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 0\cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 0\cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 0\cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 0\cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 0\cdot 3}{4}}$$

$$= e^0 + 2\cdot e^0 + 3\cdot e^0 + 4\cdot e^0$$

$$= 1 + 2 + 3 + 4$$

$$\therefore\ \mathbf{X(0) = 10}$$

(ii) k = 1

$$X(1) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 1\cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 1\cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 1\cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 1\cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 1\cdot 3}{4}}$$

$$= x(0)\, e^0 + x(1)\, e^{\frac{-j\pi}{2}} + x(2)\, e^{-j\pi} + x(3)\, e^{\frac{-j3\pi}{2}}$$

$$= 1\cdot 1 + 2\left[\cos\frac{\pi}{2} - j\sin\frac{\pi}{2}\right] + 3\,[\cos\pi - j\sin\pi] + 4\left[\cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right]$$

$$= 1 + 2(0 - j) + 3(-1 - 0) + 4(0 + j)$$

$$= 1 - 2j - 3 + 4j$$

$$\therefore\ \mathbf{X(1) = -2 + 2j}$$

(iii) k = 2

$$X(2) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 2\cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 2\cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 2\cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 2\cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 2\cdot 3}{4}}$$

$$= x(0)\, e^0 + x(1)\, e^{-j\pi} + x(2)\, e^{-j2\pi} + x(3) e^{-j3\pi}$$

$$= 1\cdot 1 + 2(\cos\pi - j\sin\pi) + 3(\cos 2\pi - j\sin 2\pi) + 4(\cos 3\pi - j\sin 3\pi)$$

$$= 1 + 2(-1 - 0) + 3(1 - 0) + 4(-1 - 0)$$

$$= 1 - 2 + 3 - 4$$

$$\therefore \mathbf{X(2) = -2}$$

(iv) k = 3

$$X(3) = \sum_{n=0}^{3} x(n)\, e^{\frac{-j2\pi 3\cdot n}{4}}$$

$$= x(0)\, e^{\frac{-j2\pi 3\cdot 0}{4}} + x(1)\, e^{\frac{-j2\pi 3\cdot 1}{4}} + x(2)\, e^{\frac{-j2\pi 3\cdot 2}{4}} + x(3)\, e^{\frac{-j2\pi 3\cdot 3}{4}}$$

$$= x(0)\, e^0 + x(1)\, e^{-j\frac{3\pi}{2}} + x(2)\, e^{-j3\pi} + x(3)\, e^{-j\frac{9\pi}{2}}$$

$$= 1\cdot 1 + 2\left(\cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right) + 3(\cos 3\pi - j\sin 3\pi) + 4\left(\cos\frac{9\pi}{2} - j\sin\frac{9\pi}{2}\right)$$

$$= 1 + 2(0 + j) + 3(-1 - 0) + 4(0 - j)$$

$$= 1 + 2j - 3 - 4j$$

$$\therefore\ \mathbf{X(3) = -2 - 2j}$$

Hence the final DFT of x(n) is,

$$X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$$

As is evident, the DFT is complex. (As in most practical cases) and hence can be split into magnitude and phase we know,

$$\text{Magnitude} = \sqrt{(\text{Real})^2 + (\text{Imaginary})^2} \text{ and}$$

$$\text{Phase} = \tan^{-1}\left(\frac{\text{Imaginary}}{\text{Real}}\right)$$

$$\therefore \quad |X(k)| = \{10, 2.83, 2, 2.83\}$$

$$\angle X(k) = \{0, 135°, 0°, -135°\}$$

We now draw the magnitude and phase. They are known as magnitude spectrum and phase spectrum.

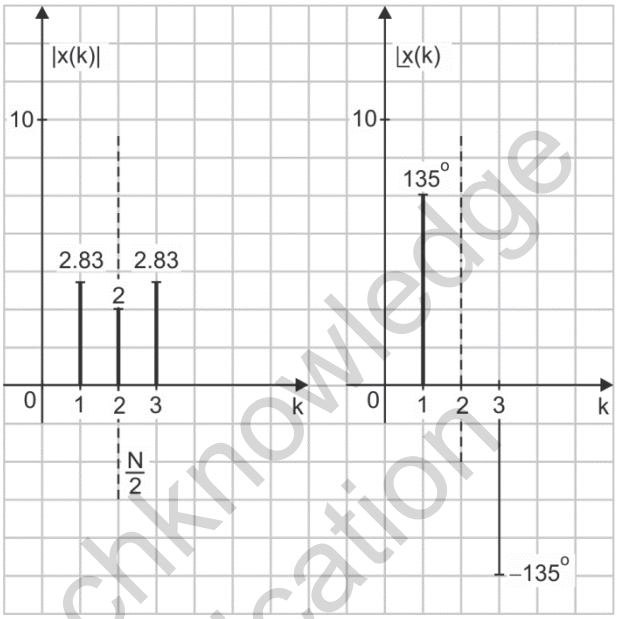


Fig. 3.4.7(a)

Note : We have drawn a dotted line at $\frac{N}{2}$ (since N = 4, $\frac{N}{2}$ = 2) which we will explain later.

Ex. 3.4.8 : Find N = 5 point DFT for x(n) = { 1,0,1,0,1 }.

Soln. :

Given : $x(n) = \{1, 0, 1, 0, 1\}$ and N = 5

This is a five point DFT.

According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}}, \quad k = 0 \text{ to } N-1$$

$$\therefore \quad X(k) = \sum_{n=0}^{4} x(n)\, e^{\frac{-j2\pi kn}{5}}, \quad k = 0 \text{ to } 4$$

For k = 0

$$X(0) = \sum_{n=0}^{4} x(n)\, e^{0} = \sum_{n=0}^{4} x(n)$$

$$\therefore \quad X(0) = x(0) + x(1) + x(2) + x(3) + x(4)$$

$$= 1 + 0 + 1 + 0 + 1$$

$$\therefore \mathbf{X(0) = 3}$$

For k = 1

$$X(1) = \sum_{n=0}^{4} x(n)\, e^{\frac{-j2\pi n}{5}}$$

$$\therefore \quad X(1) = x(0)\, e^{0} + x(1)\, e^{\frac{-j2\pi}{5}} + x(2)\, e^{\frac{-j4\pi}{5}} + x(3)\, e^{\frac{-j6\pi}{5}} + x(4)\, e^{\frac{-j8\pi}{5}}$$

$\therefore$ X (1) = 1 + 0 + [– 0.81 – j0.59] + 0 + [0.31 + j0.95]

$\therefore$ **X (1) = 0.5 + j0.36**

For k = 2

$$X(2) = \sum_{n=0}^{4} x(n)\, e^{\frac{-j4\pi n}{5}}$$

$$\therefore X(2) = x(0)\, e^{0} + x(1)\, e^{\frac{-j4\pi}{5}} + x(2)\, e^{\frac{-j8\pi}{5}} + x(3)\, e^{\frac{-j12\pi}{5}} + x(4)\, e^{\frac{-j16\pi}{5}}$$

$$\therefore X(2) = 1 + 0 + [0.31 + j0.95] + 0 + [-0.81 + j0.59]$$

$\therefore$ **X (2) = 0.5 + j1.54**

For k = 3

$$X(3) = \sum_{n=0}^{4} x(n)\, e^{\frac{-j6\pi n}{5}}$$

$$\therefore X(3) = x(0)\, e^{0} + x(1)\, e^{\frac{-j6\pi}{5}} + x(2)\, e^{\frac{-j12\pi}{5}} + x(3)\, e^{\frac{-j18\pi}{5}} + x(4)\, e^{\frac{-j24\pi}{5}}$$

$$\therefore X(3) = 1 + 0 + [0.31 - j0.95] + 0 + [-0.81 - j0.59]$$

$\therefore$ **X (3) = 0.5 – j1.54**

For k = 4

$$X(4) = \sum_{n=0}^{4} x(n)\, e^{\frac{-j8\pi n}{5}}$$

$$\therefore X(4) = x(0)\, e^{0} + x(1)\, e^{\frac{-j8\pi}{5}} + x(2)\, e^{\frac{-j16\pi}{5}} + x(3)\, e^{\frac{-j24\pi}{5}} + x(4)\, e^{\frac{-j32\pi}{5}}$$

$$\therefore X(4) = 1 + 0 + [-0.81 + j0.59] + 0 + [0.31 - j0.59]$$

$\therefore$ **X (4) = 0.5 – j0.36**

$\therefore$ The 5 point DFT of the given sequence is

X (k) = {3, 0.5 + j0.36, 0.5 + j1.54, 0.5 – j1.54, 0.5 – j 0.36}

Ex. 3.4.9 : Find the DFT of the following finite duration sequence of length L.

$$x(n) = \begin{cases} A & \text{For } 0 \le n \le L-1 \\ 0 & \text{Otherwise} \end{cases}$$

Soln. :

According to the definition of DFT, We have

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}}$$

$$\therefore X(k) = \sum_{n=0}^{L-1} A \cdot e^{\frac{-j2\pi kn}{N}}$$

$$= A \sum_{n=0}^{L-1} \left(e^{\frac{-j2\pi k}{N}} \right)^{n}$$

We have standard summation formula,

$$\sum_{k=N_1}^{N_2} a^{k} = \frac{a^{N_1} - a^{N_2+1}}{1-a}$$

Here $N_1 = 0$, $N_2 = L - 1$ and $a = e^{\frac{-j2\pi k}{N}}$

$$\therefore X(k) = A \left[\frac{\left(e^{\frac{-j2\pi k}{N}} \right)^{0} - \left(e^{\frac{-j2\pi k}{N}} \right)^{L-1+1}}{1 - e^{\frac{-j2\pi k}{N}}} \right]$$

$$\therefore X(k) = A \left[\frac{1 - e^{\frac{-j2\pi kL}{N}}}{1 - e^{\frac{-j2\pi k}{N}}} \right]$$

Ex. 3.4.10 : Derive DFT of the sample data sequence x(n) = {1, 1, 2, 2, 3, 3} and compute magnitude and phase spectrum.

Soln. : According to the DFT equation,

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j\frac{2\pi kn}{N}} \qquad k = 0, 1, 2, \ldots N^{-1}$$

Hence N = 6,

$$X(k) = \sum_{n=0}^{5} x(n) \cdot e^{-j\frac{2\pi kn}{6}} \quad ; k = 0, 1, 2, 3, 4, 5$$

$$X(k) = x(0)\cdot e^{-0} + x(1)\cdot e^{-j\frac{2\pi k}{6}} + x(2)\cdot e^{-j\frac{4\pi k}{6}} + x(3)\cdot e^{-j\frac{6\pi k}{6}} + x(4)\cdot e^{-j\frac{8\pi k}{6}} + x(5)\cdot e^{-j\frac{10\pi k}{6}}$$

For k = 0

$$X(0) = x(0)\cdot e^{0} + x(1)\cdot e^{0} + x(2)\cdot e^{0} + x(3)\cdot e^{0} + x(4)\cdot e^{0} + x(5)\cdot e^{0}$$

$$\therefore \quad X(0) = 1 + 1 + 2 + 2 + 3 + 3 = \mathbf{12}$$

If k = 1

$$X(1) = x(0) + x(1)\cdot e^{-j\frac{2\pi}{6}} + x(2)\cdot e^{-j\frac{4\pi}{6}} + x(3)\cdot e^{-j\frac{6\pi}{6}} + x(4)\cdot e^{-j\frac{8\pi}{6}} + x(5)\cdot e^{-j\frac{10\pi}{6}}$$

$$X(1) = 1 + 1\cdot\left(\cos\frac{\pi}{3} - j\sin\frac{\pi}{3}\right) + 2\cdot\left(\cos\frac{2\pi}{3} - j\sin\frac{2\pi}{3}\right) + 2\cdot(\cos\pi - j\sin\pi) + 3\left(\cos\frac{4\pi}{3} - j\sin\frac{4\pi}{3}\right) + 3\left(\cos\frac{5\pi}{3} - j\sin\frac{5\pi}{3}\right)$$

$$\therefore \quad \mathbf{X(1) = -1.5 + j\,2.6}$$

For k = 2

$$X(2) = x(0) + x(1)\cdot e^{-j\frac{4\pi}{6}} + x(2)\cdot e^{-j\frac{8\pi}{6}} + x(3)\cdot e^{-j\frac{12\pi}{6}} + x(4)\cdot e^{\frac{-16\pi}{6}} + x(5)\cdot e^{\frac{-20\pi}{6}}$$

$$X(2) = 1 + 1\left(\cos\frac{2\pi}{3} - j\sin\frac{2\pi}{3}\right) + 2\left(\cos\frac{4\pi}{3} - j\sin\frac{4\pi}{3}\right) + 2(\cos 2\pi - j\sin 2\pi) + 3\left(\cos\left(\frac{8\pi}{3}\right) - j\sin\frac{8\pi}{3}\right) + 3\left(\cos\frac{10\pi}{3} - j\sin\frac{10\pi}{3}\right)$$

$$\therefore \mathbf{X(2) = -1.5 + j\,0.86}$$

For k = 3

$$X(3) = x(0) + x(1)\cdot e^{-j\frac{6\pi}{6}} + x(2)\cdot e^{-j\frac{12\pi}{6}} + x(3)\cdot e^{-j\frac{18\pi}{6}} + x(4)\cdot e^{-j\frac{24\pi}{6}} + x(5)\cdot e^{-j\frac{30\pi}{6}}$$

$$X(3) = 1 + 1(\cos\pi - j\sin\pi) + 2(\cos 2\pi - j\sin 2\pi) + 2(\cos 3\pi - j\sin 3\pi) + 3(\cos 4\pi - j\sin 4\pi) + 3(\sin 5\pi - j\sin 5\pi)$$

$$\therefore \quad \mathbf{X(3) = 0}$$

For k = 4

$$X(4) = x(0) + x(1)\cdot e^{-j\frac{8\pi}{6}} + x(2)\cdot e^{-j\frac{16\pi}{6}} + x(3)\cdot e^{-j\frac{24\pi}{6}} + x(4)\cdot e^{-j\frac{32\pi}{6}} + x(5)\cdot e^{-j\frac{40\pi}{6}}$$

$$X(4) = 1 + 1\left(\cos\left(\frac{4\pi}{3}\right) - j\sin\frac{4\pi}{3}\right) + 2\left(\cos\frac{8\pi}{3} - j\sin\frac{8\pi}{3}\right) + 2(\cos 4\pi - j\sin 4\pi) + 3\left(\cos\frac{16\pi}{3} - j\sin\frac{16\pi}{3}\right) + 3\left(\cos\frac{20\pi}{3} - j\sin\frac{20\pi}{3}\right)$$

$$\therefore \quad \mathbf{X(4) = -0.5 - j\,0.86}$$

For k = 5

$$X(5) = x(0) + x(1)\cdot e^{-j\frac{10\pi}{6}} + x(2)\cdot e^{-j\frac{20\pi}{6}} + x(3)\cdot e^{-j\frac{30\pi}{6}} + x(4)\cdot e^{-j\frac{40\pi}{6}} + x(5)\cdot e^{-j\frac{50\pi}{6}}$$

$$X(5) = 1 + 1\left(\cos\left(\frac{5\pi}{3}\right) - j\sin\frac{5\pi}{3}\right) + 2\left(\cos\frac{10\pi}{3} - j\sin\frac{10\pi}{3}\right) + 2(\cos 5\pi - j\sin 5\pi) + 3\left(\cos\frac{20\pi}{3} - j\sin\frac{20\pi}{3}\right) + 3\left(\cos\frac{25\pi}{3} - j\sin\frac{25\pi}{3}\right)$$

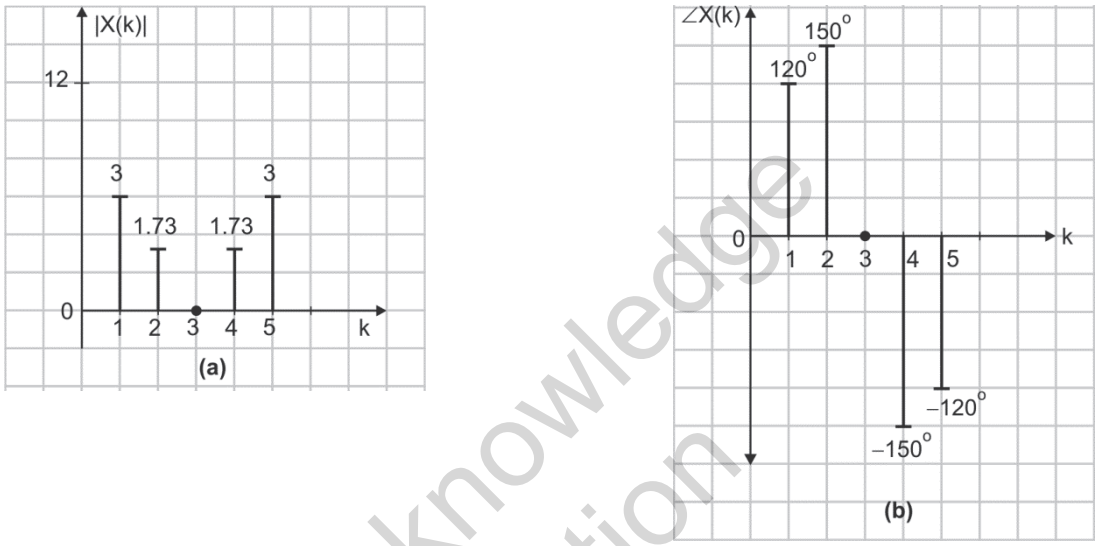
$$\therefore \mathbf{X(5) = -1.5 - j\,2.6}$$

$\therefore \quad X(k) = \{12, -1.5 + j\,2.6, -1.5\ j\,0.86, 0, -1.5 - j\,0.86, -1.5 - j\,2.6\}$

Magnitude $\rightarrow |X(k)| = \{12, 3, 1.73, 0, 1.73, 3\}$

Phase $\rightarrow \angle X(k) = \{0, 120°, 150°, 0, -150°, -120°\}$

We plot the magnitude and phase diagram as shown in Fig. P.3.4.10.



(a) Magnitude spectrum **(b) Phase spectrum**

Fig. P. 3.4.10

Note : Observing the magnitude and phase plot of Ex. 3.4.7 and Ex. 3.4.10 we note that

1. The magnitude plot is symmetric about the $\frac{N}{2}$ point. If N is odd, then it is symmetric about the $\frac{N+1}{2}$ point.
2. The phase plot is anti-symmetric about the $\frac{N}{2}$ point. If N is odd, then it is anti-symmetric about the $\frac{N+1}{2}$ point.

This is always true.

3.5 The Fourier Spectrum

At this stage it is extremely important to understand what the spectrum means. We stated that the Fourier Transform gives us the frequency components present in a signal. How do we get these.

We know how to solve a DFT example.

Let us take a 8 point example :

Solved Examples

Ex. 3.5.1 : Given x(n) = {8, 2, 1, 5, 6, 2, 4, 3}

The DFT of this signal is

X(k) = { 31, 0.58 + 1.58 j, 9 + 4j, 3.41 – 4.4 j, 7, 3.41 + 4.4 j, 9 – 4 j, 0.58 – 1.58j }

Verify this by a program or by using the fft(x) command in MATLAB.

Soln. : Since X(k) is complex, it has magnitude and phase. We plot its Magnitude.

$|X(k)| = \{31, 1.69, 9.84, 5.58, 7, 5.58, 9.84, 1.69\}$

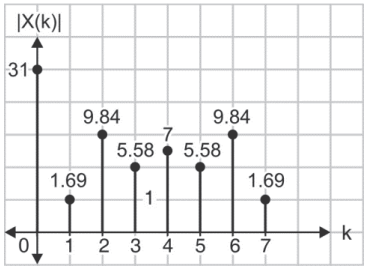


Fig. P.3.5.1

The X-axis is the frequency axis. The following questions come to our mind

(1) What does 0, 1, 2, ..., 7 represent ?

(2) Is the X axis in Hz ?

(3) Does it mean that the original signal x(n) has frequencies 0 Hz, 1 Hz,...., 6 Hz, 7 Hz ?

- These are very important questions that need to be answered.
- Make sure you understand the next few lines.
- As we have seen in earlier chapters, the entire subject of Digital Signal Processing carries no meaning if the Sampling Frequency (F_s) is not known.
- The DFT is basically sampling of the frequency axis.
- Suppose x(n) was obtained by sampling x(t) with a sampling frequency of 8000 Hz.
- A 8 point DFT will split this sampling frequency into 8 equal parts.

$$k \cdot \frac{F_s}{N} \qquad k = 0, 1, 2, \ldots, 7$$

- Since it is a 8 point DFT, N = 8.

  ∴ We get 0 Hz, 1 kHz, 2 kHz, 3 kHz,....,7 kHz

- Hence the X-axis for the given example is,

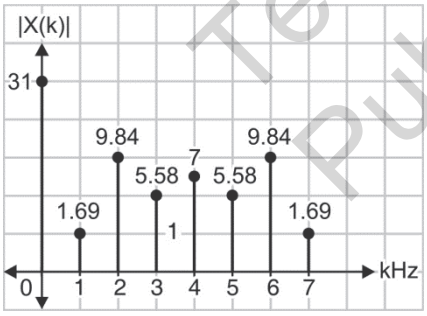


Fig. P.3.5.1(a)

- Hence we state that x(n) is made up of a DC value of 31, 1 Hz signal with magnitude 1.69, 2 Hz signal with magnitude 9.84, 7 Hz signal with magnitude 1.69.
- If we change the sampling frequency, the X-axis changes !!
- Suppose x(n) was obtained by sampling x(t) with a sampling frequency of 10 kHz.
- A 8-point DFT will split up the X-axis by the formula,

$$k \cdot \frac{F_s}{N}; \qquad k = 0,1, 2, \ldots, 7$$

Since we take a 8-point DFT, N = 8

∴ We get 0 Hz, 1250 Hz, 2500 Hz, 3750 Hz, 5000 Hz, 6250 Hz, 7500 Hz and 8750 Hz.

Hence the spectrum would be

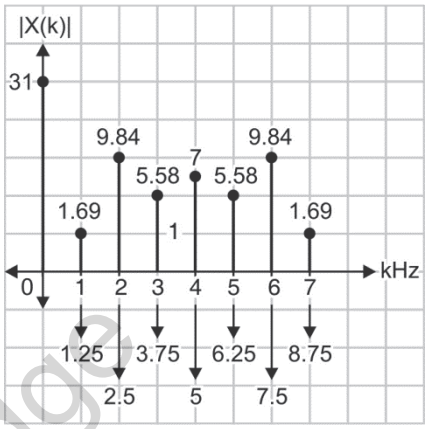


Fig. P. 3.5.1(b)

- Hence we state what has been stated earlier.
- The entire area of Digital Signal Processing makes sense only if we know the sampling frequency. Only if we know the sampling frequency, will we be able to find out the frequency components present in the signal.

3.6 Inverse Discrete Fourier Transform (IDFT)

We shall now learn how to compute the IDFT. It is similar to the DFT except that the negative exponential is replaced by a positive one. The IDFT is given by the equation.

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, e^{\frac{+j\,2\pi kn}{N}} \qquad ; \qquad n = 0, 1, 2, \ldots N-1$$

Solved Example

Ex. 3.6.1 : Compute the IDFT of X(k) = {4, 0, 0, 0}

Soln.: IDFT is given by the equation,

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, e^{\frac{+j\,2\pi kn}{N}} \qquad ; n = 0, 1, \ldots N-1$$

Since length of X(k) is 4, N = 4

$$\therefore \; x(n) = \frac{1}{4} \sum_{k=0}^{3} X(k)\, e^{\frac{+j\,2\pi kn}{4}} \qquad ; n = 0, 1, 2, 3$$

For each value of n, we vary k from 0 to 3

(i) n = 0

$$x(0) = \frac{1}{4}\sum_{k=0}^{3} X(k)\, e^{\frac{+j2\pi k \cdot 0}{4}}$$

$$= \frac{1}{4}\left[X(0)\, e^{\frac{+j2\pi 0 \cdot 0}{4}} + X(1)\, e^{\frac{+j2\pi 1 \cdot 0}{4}} + X(2)\, e^{\frac{+j2\pi 2 \cdot 0}{4}} + X(3)\, e^{\frac{+j2\pi 3 \cdot 0}{4}}\right]$$

$$= \frac{1}{4}\left[X(0)\, e^{0} + X(1)\, e^{0} + X(2)\, e^{0} + X(3)\, e^{0}\right]$$

$$= \frac{1}{4}\left[4 + 0 + 0 + 0\right]$$

$$= 1$$

$\therefore$ **x(0) = 1**

(ii) n = 1

$$x(1) = \frac{1}{4}\sum_{k=0}^{3} X(k)\, e^{\frac{+j2\pi k \cdot 1}{4}}$$

$$= \frac{1}{4}\left[X(0)\, e^{\frac{+j2\pi 0 \cdot 1}{4}} + X(1)\, e^{\frac{+j2\pi 1 \cdot 1}{4}} + X(2)\, e^{\frac{+j2\pi 2 \cdot 1}{4}} + X(3)\, e^{\frac{+j2\pi 3 \cdot 1}{4}}\right]$$

$$= \frac{1}{4}\left[4e^{0} + 0 \cdot e^{\frac{j2\pi}{4}} + 0\, e^{+j\pi} + 0\, e^{\frac{j3\pi}{2}}\right]$$

$\therefore$ **x(1) = 1**

(iii) n = 2

$$x(2) = \frac{1}{4}\sum_{k=0}^{3} X(k)\, e^{\frac{+j2\pi k \cdot 2}{N}}$$

$$= \frac{1}{4}\left[X(0)\, e^{\frac{+j2\pi 0 \cdot 2}{4}} + X(1)\, e^{\frac{+j2\pi 1 \cdot 2}{4}} + X(2)\, e^{\frac{+j2\pi 2 \cdot 2}{4}} + X(3)\, e^{\frac{+j2\pi 3 \cdot 2}{4}}\right]$$

$$= \frac{1}{4}\left[4e^{0} + 0 \cdot e^{j\pi} + 0\, e^{j2\pi} + 0\, e^{j3\pi}\right] = 1$$

$\therefore$ **x(2) = 1**

(iv) n = 3

$$x(3) = \frac{1}{4}\sum_{k=0}^{3} X(k)\, e^{\frac{+j2\pi k \cdot 3}{4}}$$

$$= \frac{1}{4}\left[X(0)\, e^{\frac{+j2\pi 0 \cdot 3}{4}} + X(1)\, e^{\frac{+j2\pi 1 \cdot 3}{4}} + X(2)\, e^{\frac{+j2\pi 2 \cdot 3}{4}} + X(3)\, e^{\frac{+j2\pi 3 \cdot 3}{4}}\right]$$

$$= \frac{1}{4}\left[4e^{0} + 0 \cdot e^{\frac{j3\pi}{2}} + 0\, e^{j3\pi} + 0\, e^{j9\pi/2}\right] = 1$$

$\therefore$ **x(3) = 1**

Hence, we have,

$$x(n) = \{x(0), x(1), x(2), x(3)\}$$

$$\therefore \quad x(n) = \{1, 1, 1, 1\}$$

If we compare this example with Ex. 3.4.3, we can observe that,

$$X(k) = \{4, 0, 0, 0\} \xleftrightarrow{\text{IDFT}} x(n) = \{1, 1, 1, 1\}$$

3.7 Computing DFT by Matrix Method

We will now learn how to compute the DFT of a sequence by using the matrix method. This is a much easier way as compared to the earlier technique which involved opening up the summation sign for each value of X(k).

The DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \qquad ; k = 0, 1, 2, \ldots, N-1$$

"We define a new term W_N which is called the twiddle factor."

$$W_N = e^{\frac{-j2\pi}{N}}$$

Substituting this is the DFT equation we get,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn} \quad ; k = 0, 1, 2, \ldots N-1 \quad \ldots(3.7.1)$$

Similarly the IDFT which is given by the equation,

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, e^{\frac{+j2\pi kn}{N}} \qquad ; n = 0, 1, \ldots N-1$$

can be written in terms of W_N as,

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, W_N^{-kn} \qquad \ldots(3.7.2)$$

Equation (3.7.1) can be represented in the matrix form as

$$\text{DFT} \rightarrow X(k) = [W_N]\, x(n) \qquad \ldots(3.7.3)$$

$$\text{where, } x(n) = \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ : \\ x(N-1) \end{bmatrix}_{N \times 1} ; X(k) = \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ : \\ X(N-1) \end{bmatrix}_{N \times 1}$$

and

$$W_N^{nk} = \begin{array}{c} \\ k \downarrow \\ 0 \\ 1 \\ 2 \\ \vdots \\ N-1 \end{array} \begin{array}{c} n \rightarrow \\ \begin{bmatrix} W_N^0 & W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & W_N^2 & \cdots & W_N^{N-1} \\ W_N^0 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \\ W_N^0 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \\ 0 \quad 1 \quad 2 \quad \cdots \quad N-1 \end{array} N \times N$$

$$\ldots(3.7.4)$$

Each value of W_N^{nk} matrix is obtained by multiplying n and k. W_N is called a $N \times N$ DFT matrix.

Note : Changing n and k co-ordinates will not make any difference.

Similarly, Equation (3.7.2) can be written as,

$$\text{IDFT} \rightarrow x(n) = \frac{1}{N} [W_N^*]\, X(k) \qquad \ldots(3.7.5)$$

Before we start solving examples of DFT using the matrix method, let us try to generate W_N matrix.

3.7.1 Solved Examples of DFT

Ex. 3.7.1 : Generate a 4×4 DFT matrix.

Soln.: Since we require a 4×4 matrix N = 4.

Therefore both, n and k will vary from 0 to 3.

$$W_4^{nk} = \begin{array}{c} \\ k \downarrow \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} n \rightarrow \\ \begin{bmatrix} W_4^{0.0} & W_4^{0.1} & W_4^{0.2} & W_4^{0.3} \\ W_4^{1.0} & W_4^{1.1} & W_4^{1.2} & W_4^{1.3} \\ W_4^{2.0} & W_4^{2.2} & W_4^{2.2} & W_4^{2.3} \\ W_4^{3.0} & W_4^{3.1} & W_4^{3.2} & W_4^{3.3} \end{bmatrix} \\ 0 \quad 1 \quad 2 \quad 3 \end{array} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix}$$

We know $W_N = e^{\frac{-j\,2\pi}{N}}$

We calculate each value of the matrix,

$$W_4^0 = e^{-j\frac{2\pi 0}{4}} = 1$$

$$W_4^1 = e^{-j\frac{2\pi}{4} \cdot 1} = -j$$

$$W_4^2 = e^{-j\frac{2\pi}{4} \cdot 2} = -1$$

$$W_4^3 = e^{-j\frac{2\pi}{4} \cdot 3} = +j$$

$$W_4^4 = e^{-j\frac{2\pi}{4} \cdot 4} = +1$$

$$W_4^6 = e^{-j\frac{2\pi}{4} \cdot 6} = -1$$

$$W_4^9 = e^{-j\frac{2\pi}{4} \cdot 9} = -j$$

$$\therefore \quad W_4^{nk} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & +1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \qquad \ldots(1)$$

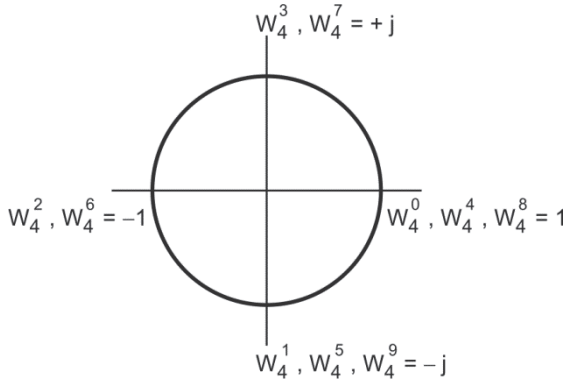
Hence a 4×4 DFT matrix is shown in Equation (1).

As mentioned earlier, W_N is called the Twiddle factor.

The word twiddle means "Turn in a twisting or spinning motion". In other words, to move in a turning (circular) fashion. A lot of us have a habit of twiddling our tumbs while waiting for an oral exam.

Now W_N is called a twiddle factor because it is cyclic in nature.

Shown below are W_4 values mapped on a circle.



$$\therefore \quad W_4^0 = W_4^4 = W_4^8 = 1$$
$$W_4^1 = W_4^5 = W_4^9 = -j$$
$$W_4^2 = W_4^6 = -1$$
$$W_4^3 = W_4^7 = +j$$

Fig. P. 3.7.1(a)

Similarly for W_8, we can draw the values of the 8×8 matrix using a circle.

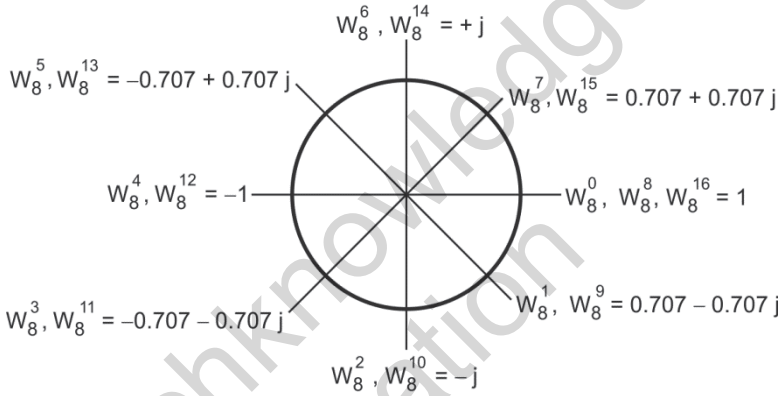


Fig. P. 3.7.1(b)

Let us calculate each of these values and verify

$$\therefore \quad W_8^0 = e^{-j\frac{2\pi}{8}\cdot 0} = \cos 0 - j \sin 0 = 1$$

$$\therefore \quad W_8^1 = e^{-j\frac{2\pi}{8}\cdot 1} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - 0.707\,j$$

$$\therefore \quad W_8^2 = e^{-j\frac{2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$\therefore \quad W_8^3 = e^{-j\frac{2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - 0.707\,j$$

$$\therefore \quad W_8^4 = e^{-j\frac{2\pi}{8}\cdot 4} = \cos\pi - j\sin\pi = -1$$

$$\therefore \quad W_8^5 = e^{-j\frac{2\pi}{8}\cdot 5} = \cos\frac{5\pi}{4} - j\sin\frac{5\pi}{4} = -0.707 + 0.707\,j$$

$$\therefore \quad W_8^6 = e^{-j\frac{2\pi}{8}\cdot 6} = \cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2} = +j$$

$$\therefore \quad W_8^7 = e^{-j\frac{2\pi}{8}\cdot 7} = \cos\frac{7\pi}{4} - j\sin\frac{7\pi}{4} = 0.707 + 0.707\,j$$

$$\therefore \quad W_8^8 = e^{-j\frac{2\pi}{8}\cdot 8} = \cos 2\pi - j\sin 2\pi = +1$$

$$\therefore \quad W_8^9 = e^{-j\frac{2\pi}{8}\cdot 9} = \cos\frac{9\pi}{4} - j\sin\frac{9\pi}{4}$$
$$= 0.707 - 0.707\,j$$

$$\therefore \quad W_8^{10} = e^{-j\frac{2\pi}{8}\cdot 10} = \cos\frac{5\pi}{2} - j\sin\frac{5\pi}{2} = -j$$

$$\therefore \quad W_8^{11} = e^{-j\frac{2\pi}{8}\cdot 11} = \cos\frac{11\pi}{4} - j\sin\frac{11\pi}{4}$$
$$= -0.707 + 0.707\,j$$

$$\therefore \quad W_8^{12} = e^{j\frac{2\pi}{8}\cdot 12} = \cos 3\pi - j\sin 3\pi = -1$$

$$\therefore \quad W_8^{13} = e^{j\frac{2\pi}{8}\cdot 13} = \cos\frac{13\pi}{4} - j\sin\frac{13\pi}{4}$$
$$= -0.707 + 0.707\,j$$

$$\therefore \quad W_8^{14} = e^{j\frac{2\pi}{8}\cdot 14} = \cos\frac{7\pi}{2} - j\sin\frac{7\pi}{2} = +j$$

$$\therefore \quad W_8^{15} = e^{j\frac{2\pi}{8}\cdot 15} = \cos\frac{15\pi}{4} - j\sin\frac{15\pi}{4}$$
$$= 0.707 + 0.707\,j$$

$$\therefore \quad W_8^{16} = e^{j\frac{2\pi}{8}\cdot 16} = \cos 4\pi - j\sin 4\pi = 1$$

Using these values we can easily generalize a 8×8 DFT matrix.

$$W_8^{nk} = \begin{array}{c} \\ k\downarrow \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} n \rightarrow \\ \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} W_8^0 & W_8^0 & W_8^0 & W_8^0 & W_8^0 & W_8^0 & W_8^0 & W_8^0 \\ W_8^0 & W_8^1 & W_8^2 & W_8^3 & W_8^4 & W_8^5 & W_8^6 & W_8^7 \\ W_8^0 & W_8^2 & W_8^4 & W_8^6 & W_8^8 & W_8^{10} & W_8^{12} & W_8^{14} \\ W_8^0 & W_8^3 & W_8^6 & W_8^9 & W_8^{12} & W_8^{15} & W_8^{18} & W_8^{21} \\ W_8^0 & W_8^4 & W_8^8 & W_8^{12} & W_8^{16} & W_8^{20} & W_8^{24} & W_8^{28} \\ W_8^0 & W_8^5 & W_8^{10} & W_8^{15} & W_8^{20} & W_8^{25} & W_8^{30} & W_8^{35} \\ W_8^0 & W_8^6 & W_8^{12} & W_8^{18} & W_8^{24} & W_8^{30} & W_8^{36} & W_8^{42} \\ W_8^0 & W_8^7 & W_8^{14} & W_8^{21} & W_8^{28} & W_8^{35} & W_8^{42} & W_8^{49} \end{bmatrix} \end{array}$$

Since W_8 is cyclic we have,

$$\therefore \quad W_8^0 = W_8^8 = W_8^{16} = W_8^{24} = W_8^{32} = W_8^{40} = 1$$

$$\therefore \quad W_8^1 = W_8^9 = W_8^{17} = W_8^{25} = W_8^{33} = W_8^{41} = W_8^{49}$$

$$= 0.707 - 0.707\,j$$

$$\therefore \quad W_8^2 = W_8^{10} = W_8^{18} = W_8^{26} = W_8^{34} = W_8^{42} = j$$

$$\therefore \quad W_8^3 = W_8^{11} = W_8^{19} = W_8^{27} = W_8^{35} = W_8^{43}$$

$$= -0.707\ \ 0.707\,j$$

$$\therefore \quad W_8^4 = W_8^{12} = W_8^{20} = W_8^{28} = W_8^{36} = W_8^{44} = 1$$

$$\therefore \quad W_8^5 = W_8^{13} = W_8^{21} = W_8^{29} = W_8^{37} = W_8^{45}$$

$$= -0.707 + 0.707\,j$$

$$\therefore \quad W_8^6 = W_8^{14} = W_8^{22} = W_8^{30} = W_8^{38} = W_8^{46} = j$$

$$\text{and} \quad W_8^7 = W_8^{15} = W_8^{23} = W_8^{31} = W_8^{39} = W_8^{47}$$

$$= 0.707 + 0.707\,j$$

Using these values we generate a 8×8 DFT matrix.

$$W_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707 - j\,0.707 & -j & -0.707 - j\,0.707 & -1 & -0.707 + j\,0.707 & j & 0.707 + j\,0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707 - j\,0.707 & j & 0.707 - j\,0.707 & -1 & 0.707 + j\,0.707 & -j & -0.707 + j\,0.707 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707 + j\,0.707 & -j & 0.707 + j\,0.707 & -1 & 0.707 - j\,0.707 & j & -0.707 - j\,0.707 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & 0.707 + j\,0.707 & j & -0.707 + j\,0.707 & -1 & -0.707 - j\,0.707 & -j & 0.707 - j\,0.707 \end{bmatrix}$$

Let us now solve a few examples by using the DFT matrix method.

Ex. 3.7.2 : Compute the DFT of the sequence x(n) = { 1, 1, 1, 1}

Soln.: This is the same sequence which was solved in Ex. 3.4.3.

The DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn},$$

Where, $W_N^{nk} = e^{\frac{-j\,2\pi kn}{N}}$

Since the length of x(n) = 4, the DFT equation reduces to,

$$X(k) = \sum_{n=0}^{3} x(n)\ W_4^{kn}\ ;\ k = 0, 1, 2, 3$$

In matrix form it is written as,

$$X(k) = [W_4]\, x(n)$$

Where, $x(n) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

$$\therefore \quad X(k) = \underset{W_4}{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}} \underset{x(n)}{\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}}$$

$$\therefore \quad X(k) = \begin{bmatrix} 4 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Hence, $X(k) = [\,4, \quad 0, \quad 0, \quad 0\,]$

We note the answer is the same as that obtained in Example 3.4.3.

Ex. 3.7.3 : Compute the DFT of x(n) = {1, 2, 3, 4}

Soln. : This is the same as Example 3.5.7.

Since x(n) is of length 4, N = 4 and we generate a DFT matrix of size 4×4.

$$\therefore \quad X(k) = [W_4]_{4\times 4}\, x(n)$$

$$\therefore \quad X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$X(k) = \begin{bmatrix} 10 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

$$\therefore \quad X(k) = \{\, 10, \quad -2+2j, \quad -2, \quad -2-2j \,\}$$

This is the same as obtained in Example 3.5.7.

You must have realized that the matrix method is much easier that the conventional method of opening the summation.

Note : The DFT matrix depends only on the length of x(n) and not the values of x(n). Because of this for every 4 point input, we use the same 4×4 DFT matrix.

A 4×4 DFT matrix can be easily memorized using the 5 steps.

Step 1 : $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & & & \\ 1 & & & \\ 1 & & & \end{bmatrix}$

Step 2 : $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & & \\ 1 & & & \\ 1 & & & -j \end{bmatrix}$

Step 3 : $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & & +j \\ 1 & & & \\ 1 & +j & & -j \end{bmatrix}$

Step 4 : $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & & j \\ 1 & & +1 & \\ 1 & j & & -j \end{bmatrix}$

Step 5 : $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & +j \\ 1 & -1 & +1 & -1 \\ 1 & +j & -1 & -j \end{bmatrix}$

Ex. 3.7.4 : Compute the IDFT of X(k) = {10, –2 + 2j, –2, –2 – 2j}.

Soln. : The IDFT is given by the equation

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, W_N^{-nk} \qquad ; n = 0, 1, 2, \ldots N-1$$

In matrix form it is written as,

$$x(n) = \frac{1}{N} [W_N^*]\, X(k)$$

Since X(k) is of length 4, N = 4 and we generate a IDFT matrix of size 4×4.

$$\therefore \quad x(n) = \frac{1}{4} \underset{W_4^*}{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & +1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}} \underset{X(k)}{\begin{bmatrix} 10 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}}$$

$$x(n) = \frac{1}{4} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 16 \end{bmatrix}$$

$$x(n) = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$\therefore$ x(n) = { 1 2 3 4 }

Ex. 3.7.5 : Compute the DFT of x(n) = {1, 1, 0, 0}

MU - Dec. 2018, 10 Marks

Soln. :

Since x(n) is of length 4, N = 4 and we generate a DFT matrix of size 4×4.

$$\therefore \quad X(k) = [W_4]_{4\times 4}\, x(n)$$

$$\therefore \quad X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$X(k) = \begin{bmatrix} 2 \\ 1-j \\ 0 \\ 1+j \end{bmatrix}$$

$\therefore$ X(k) = { 2, 1– j, 0, 1 + j}

Ex. 3.7.6 : Compute the IDFT of X(k) = {2, 1 – j, 0, 1 + j}.

MU - Dec. 2016, 5 Marks

Soln. : The IDFT is given by the equation

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, W_N^{-nk} \qquad ; n = 0, 1, 2, \ldots N-1$$

In matrix form it is written as,

$$x(n) = \frac{1}{N} [W_N^*]\, X(k)$$

Since X(k) is of length 4, N = 4 and we generate a IDFT matrix of size 4×4.

$$\therefore \quad x(n) = \frac{1}{4} \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & +1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}}_{W_4^*} \underbrace{\begin{bmatrix} 2 \\ 1-j \\ 0 \\ 1+j \end{bmatrix}}_{X(k)}$$

$$x(n) = \frac{1}{4} \begin{bmatrix} 4 \\ 4 \\ 0 \\ 0 \end{bmatrix}$$

$$x(n) = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$\therefore$ x(n) = { 1 1 0 0 }

Ex. 3.7.7 : For a continuous time signal equation, s(t) = sin [2 π 1000 t] + 0.5 sin [2 π 2000 t] Sample the given signal at 8000 samples/sec and find out 8 point DFT. Plot magnitude and phase response.

Soln. :

Given : Sampling rate F_s = 8000 sample / sec. Discrete signal is obtained by replacing t by n/F_s

$$\therefore \quad x(n) = \sin\left[2\pi\left(\frac{1000}{8000}\right)\right] n + 0.5 \sin\left[2\pi\left(\frac{2000}{8000}\right) n\right] = \sin\left(\frac{\pi}{4}\right) n + 0.5 \sin\left(\frac{\pi}{2}\right) n$$

For n = 0 to 7

$x(0) = \sin 0 + 0.5 \sin 0 = 0$

$x(1) = \sin\frac{\pi}{4} + 0.5 \sin\frac{\pi}{2} = 1.207$

$x(2) = \sin\frac{\pi}{2} + 0.5 \sin \pi = 1$

$x(3) = \sin\frac{3\pi}{4} + 0.5 \sin\frac{3\pi}{2} = 0.207$

$x(4) = \sin \pi + 0.5 \sin 2\pi = 0$

$x(5) = \sin\frac{5\pi}{4} + 0.5 \sin\frac{5\pi}{2} = -0.207$

$x(6) = \sin\frac{3\pi}{2} + 0.5 \sin 3\pi = -1$

$x(7) = \sin\frac{7\pi}{4} + 0.5 \sin\frac{7\pi}{2} = -1.207$

$\therefore$ x(n) = {0, 1.207, 1, 0.207, 0, – 0.207 – 1, – 1.207}

Since the length of x(n) is equal to 8, we use a 8×8 DFT matrix,

$\therefore$ X (k) = $[w_8]$ x (n)

$$X(k) = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & (0.707 - j\,0.707) & -j & (-0.707 - j\,0.707) & -1 & (-0.707 + j\,0.707) & j & (0.707 + j\,0.707) \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & (-0.707 - j\,0.707) & j & (0.707 - j\,0.707) & -1 & (0.707 + j\,0.707) & -j & (-0.707 - j\,0.707) \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & (0.707 + j\,0.707) & -j & (0.707 + 0.707) & -1 & (0.707 - j\,0.707) & j & (-0.707 - j\,0.707) \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & (0.707 + j\,0.707) & j & (-0.707 + j\,0.707) & -1 & (-0.707 - j\,0.707) & -j & (0.707 - j\,0.707) \end{vmatrix} \begin{vmatrix} 0 \\ 1.207 \\ 1 \\ 0.207 \\ 0 \\ -0.207 \\ -1 \\ -1.207 \end{vmatrix}$$

$$= \begin{bmatrix} 0 \\ -0.3\,j \\ -2\,j \\ 0.006\,j \\ 0 \\ 0.3\,j \\ 2\,j \\ -0.006\,j \end{bmatrix} = \begin{bmatrix} 0 \\ -j3.99 \\ j2 \\ j0.0003 \\ 0 \\ -j0.0003 \\ j2 \\ j3.99 \end{bmatrix}$$

$X(k) = \{0, -j3.99, -j2, j0.0003, 0, -j0.0003, j2, j3.99\}$

$|X(k)| = \{0, 3.99, 2, 0.0003, 0, 0.0003, 2, 3.99\}$

$\angle X(k) = \{0, -90°, -90°, 0, -90°, 90°, 90°,\}$

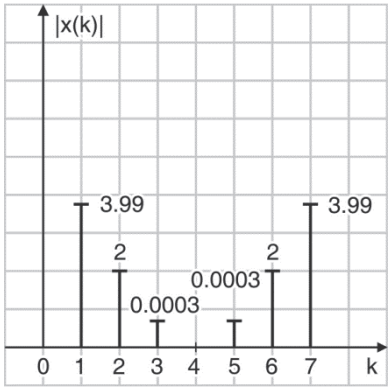


Fig. P. 3.7.7

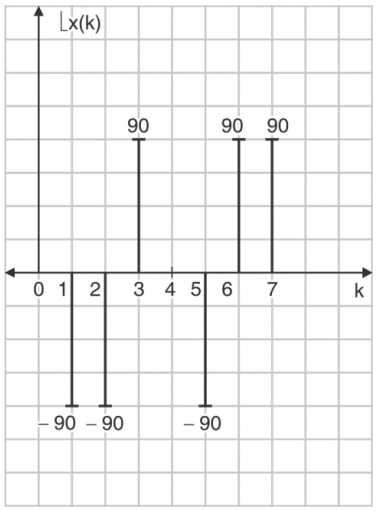


Fig. P. 3.7.7(a)

Ex. 3.7.8 : Compute the DFT of the following frequency

$$x(n) = \cos\frac{n\pi}{4} \qquad \therefore \quad 0, 1, 2, 3$$

Soln. : $x(n) = \cos\left(\frac{n\pi}{4}\right)$ n = 0, 1, 2, 3

We substitute values of n

$$\therefore \quad x(10) = \cos\left(\frac{0\,\pi}{4}\right) = 1$$

$$\therefore \quad x(1) = \cos\left(\frac{\pi}{4}\right) = 0.707$$

$$\therefore \quad x(2) = \cos\left(\frac{2\pi}{4}\right) = 0$$

$$\therefore \quad x(3) = \cos\left(\frac{3\pi}{4}\right) = -0.707$$

$$\therefore \quad x(n) = \{1, 0.707, 0, -0, 707\}$$

A 4 – point DFT is matrix form is given by the equation

$$X(k) = [W_4]\, x(n)$$

$$y(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 0.707 \\ 0 \\ -0.707 \end{bmatrix}$$

$$\therefore \quad y(k) = \{1, 1 - j\,1.414, 1, 1 + j\,1.414\}$$

Ex . 3.7.9 : Compute circular convolution of $x_1(n) = \{1,1,2,2\}$ $x_2(n) = \{1, 2, 3, 4\}$

Soln. :

$y(n) = x_1(n) \circledast x_2(n)$

We generate a circular matrix of $x_2(n)$

$$\therefore \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & 4 & 3 & 2 \\ 2 & 1 & 4 & 3 \\ 3 & 2 & 1 & 4 \\ 4 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 15 \\ 17 \\ 15 \\ 13 \end{bmatrix}$$

$$\therefore \quad y(n) = \{15, 17, 15, 13\}$$

Ex. 3.7.10 : Compute the 8-point DFT of the sequence $x(n) = \{0, 1, 2, 3\}$. Also draw the magnitude and phase plot.

Soln. : Since we require a 8-point DFT, we append four zeros to the original input sequence.

$$\therefore \quad x(n) = \{0, 1, 2, 3, 0, 0, 0, 0\}$$

As N = 8, we generate a 8×8 DFT matrix.

$$X(k) = [W_8]\, x(n)$$

$$X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707 - j\,0.707 & -j & -0.707 - j\,0.707 & -1 & -0.707 + j\,0.707 & j & 0.707 + j\,0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707 - j\,0.707 & j & 0.707 - j\,0.707 & -1 & 0.707 + j\,0.707 & -j & -0.707 + j\,0.707 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707 + j\,0.707 & -j & 0.707 + j\,0.707 & -1 & 0.707 - j\,0.707 & j & -0.707 - j\,0.707 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & 0.707 + j\,0.707 & j & -0.707 + j\,0.707 & -1 & -0.707 - j\,0.707 & -j & 0.707 - j\,0.707 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving the above matrix operation, we obtain

$$\therefore \quad X(k) = \begin{bmatrix} 6 \\ -1.414 - j\,4.83 \\ -2 + j2 \\ 1.414 - j\,0.83 \\ -2 \\ 1.414 + j\,0.83 \\ -2 - j2 \\ -1.414 + j\,4.83 \end{bmatrix}$$

$\therefore$ X(k) ={6, –1.414 – j 4.83, –2 + j2, 1.414 – j 0.83,
–2, 1.414 + j 0.83, –2 – j2, – 1.414 + j 4.83}

Since X(k) is complex, we plot the magnitude and phase.

$$|X(k)| = \sqrt{(\text{Real})^2 + (\text{Imaginary})^2}$$

$\therefore$ |X(k)| = {6, 5.03, 2.83, 1.64, 2, 1.64, 2.83, 5.03}

$$\angle X(k) = \tan^{-1}\left(\frac{\text{Imaginary}}{\text{Real}}\right)$$

$\therefore$ $\angle$X(k) = {0°, – 106.31°, 135°, –30.4°, 0°, 30.4°,
– 135°, 106.31°}

We now draw the magnitude and phase spectrum.

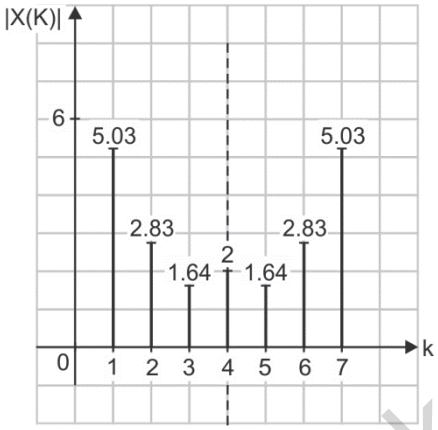


Fig. P.3.7.10

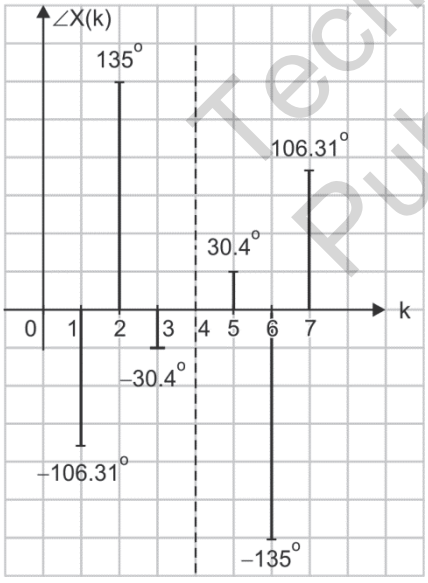


Fig. P.3.7.10(a)

A dotted line is drawn at $\frac{N}{2} = 4$.

Notice that the magnitude spectrum is symmetric about the $\frac{N}{2}$ point while the phase spectrum is anti-symmetric about the $\frac{N}{2}$ point. This is always true.

Ex. 3.7.11 : Determine 2-point and 4-point DFT of a sequence,

x (n) = u (n) – u (n – 2)

Sketch the magnitude of DFT in both the cases.

Soln. :

We begin with Identify the signal

x (n) = u(n) – u (n –2) is shown in Fig. P.3.7.11.

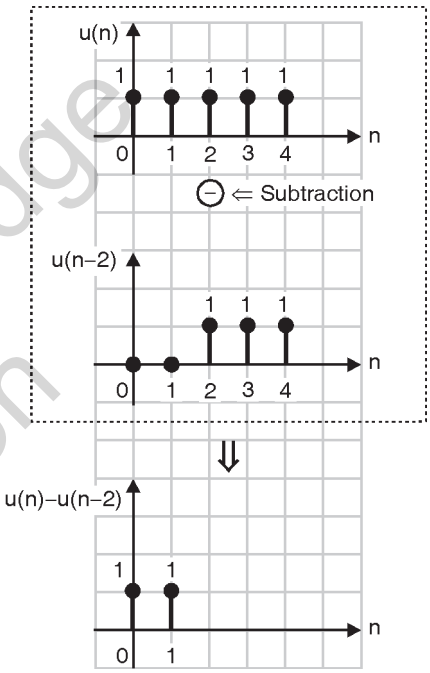


Fig. P. 3.7.11

$\therefore$ x (n) = {1,1}

(i) Two point DFT

Since N = 2, we use a 2 × 2

DFT matrix

$$\therefore \quad X(k) = [W_2]_{2\times 2}\, x(n) \qquad \ldots (1)$$

Generation of W_2 is shown below n = 0 n = 1

$$W_2^{kn} = \begin{matrix} k=0 \\ k=1 \end{matrix} \begin{bmatrix} W_2^0 & W_2^0 \\ W_2^0 & W_2^1 \end{bmatrix}$$

Now, $W_2^0 = e^{\frac{-j2\pi}{2} \cdot 0} = 1$

$$W_2^1 = e^{\frac{-j2\pi}{2} \cdot 1} = e^{-j\pi} = -1$$

$$\therefore \quad W_2^{nk} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

From Equation (1) we write $W_2\, X(n)$

$$X(k) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\therefore \quad X(k) = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

$$\therefore \quad X(k) = \{2,0\}$$

$$|X(k)| = \sqrt{(\text{Real})^2 + (\text{Img})^2}$$

$$\therefore \quad |X(k)| = \sqrt{2^2} = 2$$

$\therefore$ Magnitude plot is shown in Fig. P. 3.7.11(a).

Fig. P.3.7.11(a)

(ii) Four point DFT

Since N = 4, we use 4 × 4 DFT matrix

We append two zeros to x (n) to get x (n) = {1,1,0,0}

We have already learnt how to generate a 4 × 4 DFT matrix

$$\therefore \quad X(k) = [W_4]_{4\times 4}\, x(n)$$

$$\therefore \quad X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore \quad X(k) = \begin{bmatrix} 2 \\ 1-j \\ 0 \\ 1+j \end{bmatrix}$$

$$\therefore \quad X(k) = \{2, 1-j, 0, 1+j\}$$

$$|X(k)| = \sqrt{(\text{Real})^2 + (\text{Imaginary})^2}$$

$$\therefore \quad |X(k)| = \{2, 1.414, 0, 1.414\}$$

The magnitude plot is shown in Fig. P. 3.7.11(b).

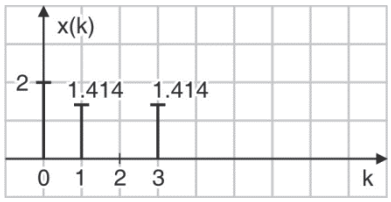


Fig. P. 3.7.11(b)

Ex. 3.7.12 : Compute the DFT of four point sequence x(n) = {0, 1, 2, 3}

Soln. :

The four point DFT in the matrix form is given by,

$$X(k) = [W_4] \cdot x(n)$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

$$\therefore \quad X(k) = \begin{bmatrix} 0+1+2+3 \\ 0-j-2+3j \\ 0-1+2-3 \\ 0+j-2-3j \end{bmatrix} = \begin{bmatrix} 6 \\ -2j+2j \\ -2 \\ -2-2j \end{bmatrix}$$

$$\therefore \quad X(k) = \{6, -2+2j, -2, -2-2j\}$$

Ex. 3.7.13 : Calculate 8 point DFT of

x (n) = { 1, 2, 1, 2 }

Soln. :

First we will make length of given sequence '8' by appending zeros to x(n).

$$\therefore \quad x(n) = \{1, 2, 1, 2, 0, 0, 0, 0\}$$

Now

$$X(k) = [W_8]\, x(n)$$

We have already learnt – how to compute W_8, which is a 8×8 DFT matrix.

$$X(k)= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707 - j\,0.707 & -j & -0.707 - j\,0.707 & -1 & -0.707 + j\,0.707 & j & 0.707 + j\,0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707 - j\,0.707 & j & 0.707 - j\,0.707 & -1 & 0.707 + j\,0.707 & -j & -0.707 + j\,0.707 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707 + j\,0.707 & -j & 0.707 + j\,0.707 & -1 & 0.707 - j\,0.707 & j & -0.707 - j\,0.707 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & 0.707 + j\,0.707 & j & -0.707 + j\,0.707 & -1 & -0.707 - j\,0.707 & -j & 0.707 - j\,0.707 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 1 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore X(k) = \begin{bmatrix} 1+2+1+2+0+0+0+0 \\ 1+1.414-j\,1.414-j-1.414-j\,1.414+0+0+0+0 \\ 1-j\,2-1+j\,2+0+0+0+0 \\ 1-1.414-j\,1.414+j+1.414-j\,1.414+0+0+0+0 \\ 1-2+1-2+0+0+0+0 \\ 1-1.414+j\,1.414-j+1.414+j\,1.414+0+0+0+0 \\ 1+j\,2-1-j\,2+0+0+0+0 \\ 1+1.414+j\,1.414+j-1.414+j\,1.414+0+0+0+0 \end{bmatrix}$$

$$\therefore \quad X(k) = \begin{bmatrix} 6 \\ 1-j3.828 \\ 0 \\ 1-j\,1.828 \\ -2 \\ 1+j\,1.828 \\ 0 \\ 1+j3.828 \end{bmatrix}$$ This is the required DFT.

$\therefore$ The 8 point DFT of the sequence is

$\therefore$ X (k) ={6,1 – j3.828, 0, 1 – j 1.828, – 2, 1 + j 8. 28, 0, 1 + j 3.828}

Ex. 3.7.14 : Compute 8 point DFT of the sequence x (n) = {0, 1, 2, 3}. Sketch the magnitude and phase plot also

Soln. : Given sequence is, x (n) = {0, 1, 2, 3}

It is asked to calculate 8 point DFT.

We append four zero to x (n) to make its length equal to 8,

$\therefore$ x (n) = {0, 1, 2, 3, 0, 0, 0, 0}

Now $X(k) = [W_8]\, x(n)$

$$X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707 - j\,0.707 & -j & -0.707 - j\,0.707 & -1 & -0.707 + j\,0.707 & j & 0.707 + j\,0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707 - j\,0.707 & j & 0.707 - j\,0.707 & -1 & 0.707 + j\,0.707 & -j & -0.707 + j\,0.707 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707 + j\,0.707 & -j & 0.707 + j\,0.707 & -1 & 0.707 - j\,0.707 & j & -0.707 - j\,0.707 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & 0.707 + j\,0.707 & j & -0.707 + j\,0.707 & -1 & -0.707 - j\,0.707 & -j & 0.707 - j\,0.707 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore X(k) = \begin{bmatrix} 6 \\ -1.414 - j\,4.828 \\ -2 + j\,2 \\ 1.414 - j\,0.828 \\ -2 \\ 1.414 + j\,0.828 \\ -2 - j\,2 \\ -1.414 + j\,4.828 \end{bmatrix}$$

$$\text{Magnitude response} = |X(k)| = \sqrt{(\text{Real})^2 + (\text{Imaginary})^2}$$

$$\therefore |X(k)| = \{6, 5.03, 2.83, 1.64, 2, 1.64, 2.83, 5.03\}$$

$$\text{Phase response} = \tan^{-1}\left\{\frac{\text{Imaginary}}{\text{Real}}\right\}$$

$\angle X(k) = \{0°, -106.3°, 135°, -30.36°, 0°, 30.36°, -135°, 106.32°\}$

The magnitude plot and phase plot are shown in Fig. P. 3.7.14.

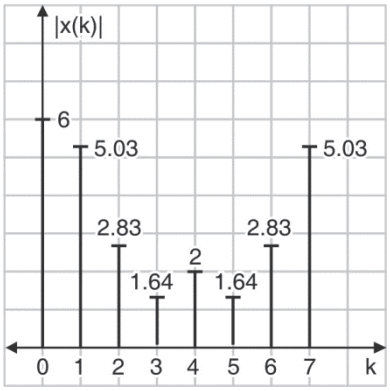


Fig. P. 3.7.14

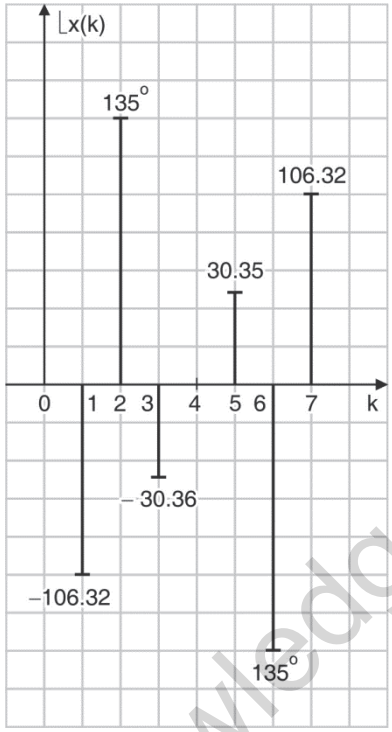


Fig. P. 3.7.14(a)

Ex. 3.7.15 : Plot the magnitude and phase spectrum of the sampled data sequence = {2, 0, 0, 1} which was obtained using a sampling frequency of 20 Hz.

Soln. : To obtain magnitude and phase response we will calculate DFT as follows,

$$X(k) = [W_4]\, x(n)$$

$$\therefore\ X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2+j \\ 1 \\ 2-j \end{bmatrix}$$

$$\therefore\ X(k) = \{3,\ 2+j,\ 1,\ 2-j\}$$

Magnitude plot

$$|X(k)| = \sqrt{R_e^2 + I_m^2}$$

$$\therefore\ |X(k)| = \{3,\ 2.24,\ 1,\ 2.24\}$$

$$\angle X(k) = \tan^{-1}\left(\frac{\text{Img}}{\text{Real}}\right)$$

$$\angle X(k) = \{0°,\ 26.56°,\ 0°,\ -26.56°\}$$

We draw the, magnitude and phase plot

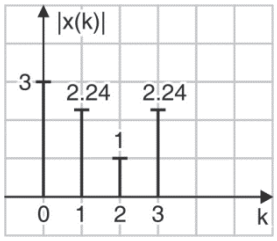


Fig. P. 3.8.15

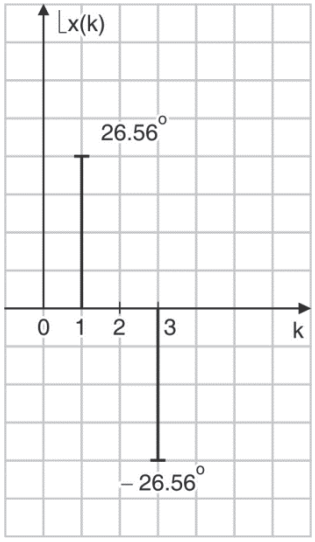


Fig. P. 3.7.15(a)

Ex. 3.7.16 : Compute the sequence x(n) for which the DFT is {20, – 4 + 4j, – 4, – 4 – 4j}

Soln. : Since we need to find the sequence x(n), we compute the IDFT.

The IDFT is given by the equation

$$\therefore \quad x(n) = \frac{1}{N} [W_4^*] X(k)$$

Since N = 4, we use a 4 × 4 IDFT matrix.

$$x(n) = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & +1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}\begin{bmatrix} 20 \\ -4+4j \\ -4 \\ -4-4j \end{bmatrix}$$

$$\therefore x(n) = \frac{1}{4}\begin{bmatrix} 8 \\ 16 \\ 24 \\ 32 \end{bmatrix}$$

$$\therefore x(n) = \{2, 4, 6, 8\}$$

Given below is a MATLAB code to compute the DFT of a sequence of any length.

Program to Compute the DFT using Matrix Method

```
clc
clear all
x={0 1 2 3 0 0 0 0];
N=length (x);

for k = 0 : 1 : N–1
    for n = 0 : 1 : N–1
        W (n+1, k+1) =exp (–j*2*pi*k*n/N);
    end
end
X=W*x'
X_Mag=abs (X);
X_Ph=rad2deg (angle (X));
subplot (2, 2, 1)
stem (X_Mag)
title ('Magnitude plot')
subplot (2, 2, 2)
stem (X_Ph)
title ('Phase plot')

X =
    6.0000 + 0.0000i
  – 1.4142 – 4.8284i
  – 2.0000 + 2.0000i
  1.4142 – 0.8284i
  – 2.0000 – 0.0000i
  1.4142 + 0.8284i
  – 2.0000 – 2.0000i
 –1.4142 + 4.8284i
```

Ex. 3.7.17 : Find the DFT of following sequence and check your result by using IDFT x (n) = {0, 1, 2, 3}

Soln. :

We have already obtained DFT of x (n) of Ex. 2.7.2

$$\text{It is } X(k) = \{6, 2j-2, -2, -2j-2\}$$

Calculation of IDFT :

$$x(n) = \frac{1}{N}\left[W_N^*\right] X_N$$

$$\therefore \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}\begin{bmatrix} 6 \\ 2j-2 \\ -2 \\ -2j-2 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 6+2j-2-2-2j-2 \\ 6-2-2j+2-2+2j \\ 6-2j+2-2+2j+2 \\ 6+2+2j+2+2-2j \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 0 \\ 4 \\ 8 \\ 12 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

$$\therefore \quad x(n) = \{0, 1, 2, 3\}$$

Ex. 3.7.18 : Show that the basis matrix for DFT is unitary.

Consider N = 4.

Soln. : A matrix is said to be unitary is

$$W_N = W_N^{*T} = I$$

We need to check is

$$W_4 \cdot W_N^{*T} = I$$

We have,

$$W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$\text{and } W_4^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

Now

$$W_4 \cdot W_4^{*T} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

$$4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \therefore 4\,[I]$$

Hence DFT matrix is unitary

3.8 Discrete Fourier Transform (DFT) Properties

MU - May 2016, Dec. 2016, May 2017

Q. State any five DFT properties.
(May 2016, Dec. 2016, May 2017, 10 Marks)

Like any other transform, the DFT has several important properties. These properties help us in computing the DFT easily.

The properties that we study here are listed below.

1. Linearity
2. Periodicity
3. Circular time shift
4. Circular frequency shift
5. Time reversal
6. Symmetry property
7. Complex Conjugate property
8. Parseval's theorem
9. Multiplication
10. Circular Convolution property

Though the Proof of each of the properties is not a part the syllabus, we shall prove them for the sake of continuity. Students can choose to ignore them.

We will also solve examples using them.

3.8.1 Linearity

MU - Dec. 2015, Dec. 2017, May 2018

Q. State the DFT Properties : Linearity.
(Dec. 2015, Dec. 2017, May 2018, 2 Marks)

$$\text{If } x_1(n) \xleftrightarrow{\text{DFT}} X_1(k) \text{ and } x_2(n) \xleftrightarrow{\text{DFT}} X_2(k) \text{ then } a\,x_1(n) + b x_2(n) \xleftrightarrow{\text{DFT}} a\,X_1(k) + bX_2(k)$$

Proof : According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$\text{Let } x(n) = ax_1(n) + bx_2(n)$$

$$\therefore X(k) = \sum_{n=0}^{N-1} [ax_1(n) + bx_2(n)]\, e^{\frac{-j2\pi nk}{N}}$$

$$= \sum_{n=0}^{N-1} a\,x_1(n)\, e^{\frac{-j2\pi nk}{N}} + \sum_{n=0}^{N-1} bx_2(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$= a \sum_{n=0}^{N-1} x_1(n)\, e^{\frac{-j2\pi nk}{N}} + b \sum_{n=0}^{N-1} x_2(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$= a\,X_1(k) + b\,X_2(k)$$

Hence

$$\text{If } x_1(n) \xleftrightarrow{\text{DFT}} X_1(k) \text{ and } x_2(n) \xleftrightarrow{\text{DFT}} X_2(k), \text{ then } ax_1(n) + bx_2(n) \xleftrightarrow{\text{DFT}} aX_1(k) + bX_2(k)$$

Solved Example

Ex. 3.8.1 : $x_1(n) = \{1, 2, 3, 4\}$ and $x_2(n) = \{5, 6, 7, 8\}$

Compute the DFT of the sequence $x_3(n) = 2x_1(n) + 3x_2(n)$

Soln. : From the Linearity property we know

$$\text{If } x_1(n) \xleftrightarrow{\text{DFT}} X_1(k)$$

$$x_2(n) \xleftrightarrow{\text{DFT}} X_2(k)$$

$$\text{then, } a\,x_1(n) + bx_2(n) \xleftrightarrow{\text{DFT}} aX_1(k) + bX_2(k)$$

$$\text{Now, } x_3(n) = 2x_1(n) + 3x_2(n)$$

$$\therefore \text{DFT}\{x_3(n)\} = \text{DFT}\{a\,x_1(n)\} + \text{DFT}\{b\,x_2(n)\}$$

is $x_3(n) \xleftrightarrow{DFT} aX_1(k) + bX_2(k)$

We calculate DFT of $x_1(n)$ and $x_2(n)$

$$X_1(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\therefore \quad X_1(k) = \begin{bmatrix} 10 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

Similarly $$X_2(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

$$\therefore \quad X_2(k) = \begin{bmatrix} 26 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

$$\therefore \quad DFT\{x_3(n)\} = X_3(k) = 2X_1(k) + 3X_2(k)$$

$$= 2\begin{bmatrix} 10 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix} + 3\begin{bmatrix} 26 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

$$= \begin{bmatrix} 20 \\ -4+4j \\ -4 \\ -4-4j \end{bmatrix} + \begin{bmatrix} 78 \\ -6+6j \\ -6 \\ -6-6j \end{bmatrix}$$

$$\therefore \quad X_3(k) = \begin{bmatrix} 98 \\ -10+10j \\ -10 \\ -10-10j \end{bmatrix}$$

$$\therefore \quad X_3(k) = \{98, -10+10j, -10, -10-10j\}$$

3.8.2 Periodicity

MU - Dec. 2015, Dec. 2017, May 2018

Q. State the DFT Properties : Periodicity. **(Dec. 2015, Dec. 2017, May 2018, 2 Marks)**

If $x(n) \xleftrightarrow{DFT} X(k)$

then $X(k+N) = X(k)$

i.e., The DFT is periodic with a period N.

Proof : According to the definition of the DFT

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

We replace k by (k + N)

$$\therefore \quad X(k+N) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi n(k+N)}{N}}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} \cdot e^{\frac{-j2\pi nN}{N}}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} \cdot e^{-j2\pi n}$$

Now, $e^{-j2\pi n} = \cos 2\pi n - j\sin 2\pi n = 1$

$$\therefore \quad X(k+N) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$= X(k)$$

$$\therefore \quad X(k+N) = X(k)$$

Hence the DFT is periodic with period N.

Circular shift of a sequence :

Before we move to the next properties, let us understand how to visualize a periodic signal.

Consider a sequence x(n),

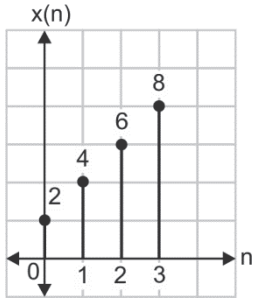


Fig. 3.8.1(a)

A periodic extension of a sequence x(n) can be written as,

$$x_p(n) = \sum_{l=-\infty}^{+\infty} x(n - lN); \qquad \text{where N is the period.}$$

This is shown in Fig. 3.8.1 (b).

A simple way of visualizing a periodic signal is to imagine x(n) on a circle in the counter clockwise direction.

The Fig. 3.8.1(b) will make this clear.

$x_p(n) = X((n))_N$ is the periodic extension of x(n).

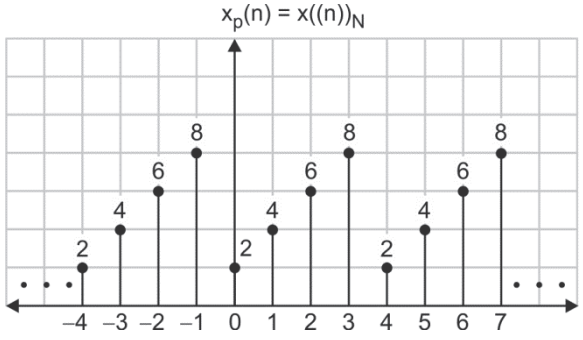


Fig. 3.8.1(b)

As stated earlier, the periodic signal x_p (n) can be viewed as lying on a circle in the counter-clockwise direction.

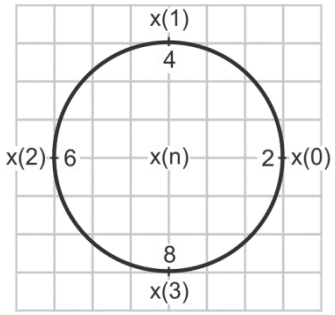


Fig. 3.8.1(c)

As we move around the circle in the counter-clockwise direction we get,

x(0), x(1), x(2), x(3), x(0), x(1), x(2), x(3), …

$\therefore \quad x_p(n) = x[(n \text{ modulo } N)] = x_p((n))_N$

If we now shift the periodic signal by say 2.

We get,

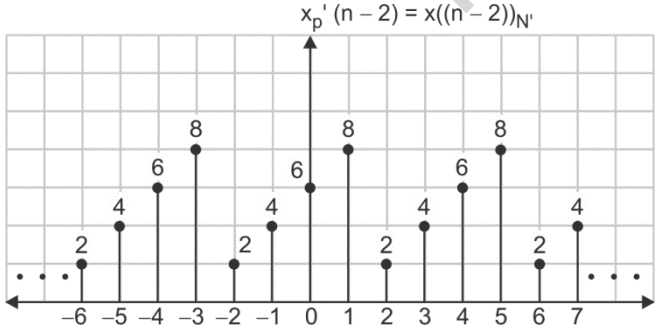


Fig. 3.8.1(d)

This is nothing but a circular shift of the circle as shown in Fig. 3.8.1(e)

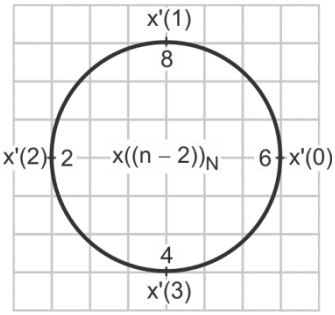


Fig. 3.8.1(e)

Here $((n))_N$ denotes n modulo N.

$\therefore$ $x((n-2))_N$ denotes x(2 modulo N)

A general way of representing a periodic signal is,

$\therefore \quad x_p(n) = x[(n \text{ modulo } N)]$ or $x((n))_N$.

If n is between 0 and N – 1, then leave it as it is. If not, then add or subtract multiplies of N from n until the result is between 0 and N – 1.

Example : $x(-3 \text{ Modulo } 4) = x((-3))_4 = x(1)$

$x(11 \text{ Modulo } 8) = x((11))_8 = x(2)$

The shifted version of $x_p(n)$ shown in Fig. 3.8.1(d) can be written as,

$$x'_p(n) = x_p(n-k) = \sum_{l=-\infty}^{+\infty} x(n - lN - k)$$

In general if

$x(n) = \{x(0), x(1), x(2), \ldots\ldots, x(N-1)\}$

then $x((n-1)_N = \{x(N-1), x(0), x(1), \ldots, x(N-2)\}$

$x((n-2))_N = \{x(N-2), x(N-1), x(0), x(1), \ldots, x(N-3)\}$

: :
: :
: :

$x((n-N))_N = \{x(0), x(1), \ldots\ldots, x(N-1)\}$

$\therefore \quad x((n-N))_N = x(n)$

In a similar way,

$x((n-m))_N = x(N-m+n)$

We now prove the circular shift property of the DFT.

3.8.3 Circular Time Shift

MU - Dec. 2015, Dec. 2017, May 2018

Q. State the DFT Properties : Time Shift.
(Dec. 2015, Dec. 2017, May 2018, 2 Marks)

If $x(n) \xleftrightarrow{DFT} X(k)$

then $x((n-m))_N \xleftrightarrow{DFT} e^{\frac{-j2\pi mk}{N}} \cdot X(k)$

$$DFT\{x(n)\} = X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$\therefore DFT\{x((n-m))_N\} = \sum_{n=0}^{m-1} x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} + \sum_{n=m}^{N-1} x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} \quad \ldots(3.8.1)$$

We see that the above equation is made up of two parts. We work on the first part.

Since, $x((n-m))_N = x(N-m+n)$, we have

$$\sum_{n=0}^{m-1} x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} = \sum_{n=0}^{m-1} x(N-m+n)\, e^{\frac{-j2\pi nk}{N}}$$

Let $N-m+n = l$

$$n = l-N+m$$

$$\therefore \sum x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} = \sum_{l=N-m}^{N-1} x(l)\, e^{\frac{-j2\pi k(-N+m+l)}{N}}$$

$$= \sum_{l=N-m}^{N-1} x(l)\, e^{\frac{-j2\pi k(l+m)}{N}} \cdot e^{+j2\pi k}$$

Since $e^{+j2\pi k} = 1$ for k = 0, 1, 2, ... we have,

$$\sum x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} = \sum_{l=N-m}^{N-1} x(l)\, e^{\frac{-j2\pi k(l+m)}{N}} \quad \ldots(3.8.2)$$

Similarly for the second part of Equation (3.8.1) we have,

$$\sum_{n=m}^{N-1} x((n-m))_N\, e^{\frac{-j2\pi nk}{N}} = \sum_{l=0}^{N-1-m} x(l)\, e^{\frac{-j2\pi k(m+l)}{N}} \quad \ldots(3.8.3)$$

Substituting Equations (3.8.2) and (3.8.3) in Equation (3.8.1) we get,

$$DFT\{x((n-m))_N\} = \sum_{l=N-m}^{N-1} x(l)\, e^{\frac{-j2\pi k(l+m)}{N}} + \sum_{l=0}^{N-m-1} x(l)\, e^{\frac{-j2\pi k(m+l)}{N}}$$

$$= e^{\frac{-j2\pi mk}{N}} \sum_{l=0}^{N-1} x(l)\, e^{\frac{-j2\pi k}{N}}$$

$$DFT\{x((n-m))_N\} = e^{-j2\pi km/N} \cdot X(k)$$

$$\therefore x((n-m))_N \xleftrightarrow{DFT} e^{\frac{-j2\pi mk}{N}} \cdot X(k)$$

Solved Examples

Ex. 3.8.2 : (i) Given x(n) = {1, 2, 3, 4}

Find the DFT of x(n).

(ii) Using results obtained in (i) and not otherwise, obtain the DFT of the following sequences.

$x_1(n) = \{4, 1, 2, 3\}$

$x_2(n) = \{3, 4, 1, 2\}$

Soln. :

(i) We first find the DFT of x(n) using the matrix notation

$$X = W_N^{nk} \cdot x$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\therefore X(k) = \{10, -2+2j, -2, -2-2j\}$$

(ii) We now assume that x(n) is periodic, we plot x(n)

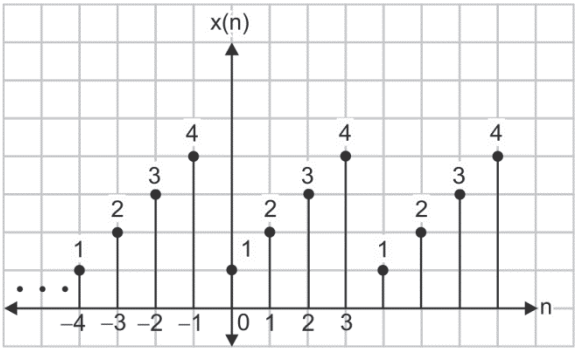


Fig. P. 3.8.2

Now $x_1(n)$ is x(n) shifted to the right by one.

Note, $x_1(n)$ can also be x(n) shifted to the left by 3.

i.e. $x_1(n) = x(n-1)$ OR $x_1(n) = x(n+3)$

Let's take $x_1(n) = x(n-1)$

(Both will give same answer)

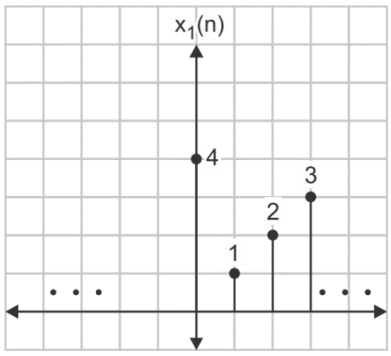


Fig. P. 3.8.2(a)

We know

If $x(n) \xleftrightarrow{DFT} X(k)$

$x(n-p) \xleftrightarrow{DFT} e^{\frac{-j2\pi pk}{N}} \cdot X(k)$

i.e., $x(n-p) \xleftrightarrow{DFT} W_N^{pk} X(k)$

$\therefore x(n-1) \xleftrightarrow{DFT} W_N^{k} \cdot X(k)$

$\therefore x_1(n) = x(n-1) \xleftrightarrow{DFT} W_N^{k} \cdot X(k)$

We already know $X(k) = \{10, -2+2j, -2, -2-2j\}$

i.e., $X_1(k) = W_4^k X(k)$

$X_1(0) = W_4^0 X(0) = (1)(10)$

$= 10$

$X_1(1) = W_4^1 X(1) = (-j)(-2+2j) = 2+j2$

$X_1(2) = W_4^2 X(2) = (-1)(-2) = 2$

$X_1(3) = W_4^3 X(3) = (j)(-2-2j) = 2-j2$

$\therefore$ DFT $\{4, 1, 2, 3\} = \{10, 2+j2, 2, 2-j2\}$

$\therefore X_1(k) = \{10, 2+j2, 2, 2-j2\}$

Similarly for $x_2(n)$, we observe that $x_2(n)$ is simply $x(n)$ shifted to the left by 2 or to the right by 2.

$\therefore x_2(n) = x(n+2) = x(n-2)$

Let us take $x_2(n) = x(n+2)$

(Both will give the same answer)

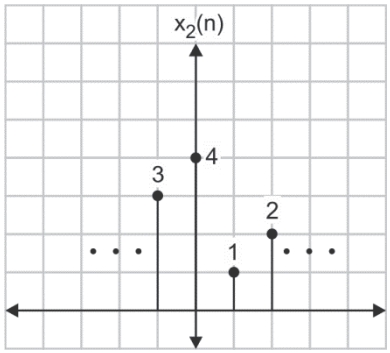


Fig. P. 3.8.2(b)

Since $x(n-p) \xleftrightarrow{DFT} W_N^{pk} \cdot X(k)$

$\therefore x(n+2) \xleftrightarrow{DFT} W_4^{2k} X(k)$

$X_2(k) = W_4^{2k} X(k)$

$X_2(0) = W_4^0 X(k) = (1) \cdot (10) = 10$

$X_2(1) = W_4^2 X(k) = (-1)(-2+2j) = 2-2j$

$X_2(2) = W_4^4 X(k) = (1)(-2) = -2$

$X_2(3) = W_4^6 X(k) = (-1)(-2-2j) = 2+2j$

$\therefore$ DFT $\{3, 4, 1, 2\} = \{10, 2-j2, -2, 2+j2\}$

$\therefore X_2(k) = \{10, 2-2j, -2, 2+2j\}$

Ex. 3.8.3 :

(i) x (n) = {1, 2, 3, 4} find DFT X (k)

(ii) Using results obtained in part (i) and not otherwise find the DFT of following sequences

$x_1(n) = \{4, 1, 2, 3\}$ $x_2(n) = \{2, 3, 4, 1\}$

$x_3(n) = \{3, 4, 1, 2\}$ $x_4(n) = \{4, 6, 4, 6\}$

Soln. :

(i)
$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1+2+3+4 \\ 1-2j-3+4j \\ 1-2+3-4 \\ 1+2j-3-4j \end{bmatrix} = \begin{bmatrix} 10 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

$\therefore X(k) = \{10, -2+2j, -2, -2, -2j\}$

(ii) (a) Given $x_1(n) = \{4, 1, 2, 3\}$ and $x(n) = \{1, 2, 3, 4\}$. Thus $x_1(n)$ is obtained by delaying $x(n)$ by one position

$\therefore \quad x_1(n) = x(n-1)$

According to circular time shift property

$$x(n-l) \xleftrightarrow{DFT} X(k)\, e^{\frac{-j2\pi kl}{N}}$$

Here $l = 1$

$$\therefore \quad X_1(k) = X(k)\, e^{\frac{-j2\pi k}{N}}$$

For $k = 0 \Rightarrow \quad X_1(0) = X(0)\, e^0 = 10$

$k = 1 \Rightarrow \quad X_1(1) = X(1)\, e^{\frac{-j2\pi}{4}}$

$$= (-2 + 2j)\left[\cos\frac{2\pi}{4} - j\sin\frac{2\pi}{4}\right]$$

$\therefore \quad X_1(1) = (-2 + 2j)(-j)$

$\therefore \quad X_1(1) = 2 + j2$

For $k = 2 \Rightarrow \quad X_1(2) = X(2)\, e^{\frac{-j4\pi}{4}} = X(2)\, e^{-j\pi}$

$\therefore \quad X_1(2) = (-2)[\cos\pi - j\sin\pi]$

$= (-2)(-1)$

$\therefore \quad X_1(2) = 2$

For $k = 3 \Rightarrow \quad X_1(3) = X(3)\, e^{\frac{-j6\pi}{4}}$

$$= (-2 - 2j)\left[\cos\frac{6\pi}{4} - j\sin\frac{6\pi}{4}\right]$$

$\therefore \quad X_1(3) = (-2 - 2j)(+j)$

$\therefore \quad X_1(3) = 2 - j2$

$\therefore \quad X_1(k) = \{10, 2 + j2, 2, 2 - j2\}$

(b) Given $x_2(n) = \{2, 3, 4, 1\}$ and $x(n) = \{1, 2, 3, 4\}$. Thus $x_2(n)$ is obtained by advancing $x(n)$ by $\pm$ position

$\therefore \quad x_2(n) = x(n+1)$

According to circular time shifting property,

$$x(n-l) \xleftrightarrow{DFT} X(k)\, e^{\frac{-j2\pi kl}{N}}$$

$$x(n+1) \xleftrightarrow{DFT} X(k)\, e^{\frac{-j2\pi k}{N}}$$

$$X_2(k) = X(k)\, e^{\frac{-j2\pi k}{N}}$$

For $k = 0 \Rightarrow \quad X_2(0) = X(0)\, e^{\circ} = 10$

For $k = 1 \Rightarrow \quad X_2(1) = X(1)\, e^{\frac{-j2\pi}{4}}$

$$= (-2 + 2j)\left[\cos\frac{2\pi}{4} + j\sin\frac{2\pi}{4}\right]$$

$\therefore \quad X_2(1) = (-2 + 2j)(0 + j)$

$\therefore \quad X_2(1) = -2 - j2$

For $k = 2 \Rightarrow \quad X_2(2) = X(2)\, e^{\frac{-j4\pi}{4}}$

$= (-2)[\cos\pi + j\sin\pi]$

$\therefore \quad X_2(2) = (-2)[-1]$

$\therefore \quad X_2(2) = 2$

For $k = 3 \Rightarrow \quad X_2(3) = X(3)\, e^{\frac{-j6\pi}{4}}$

$$= (-2 - 2j)\left[\cos\frac{6\pi}{4} + j\sin\frac{6\pi}{4}\right]$$

$\therefore \quad X_2(3) = (-2 - 2j)[-j]$

$\therefore \quad X_2(3) = -2 + 2j$

$\therefore \quad X_2(k) = \{10, -2, -j2, 2, -2 + j2\}$

(c) Given $x_3(n) = \{3, 4, 1, 2\}$ and $x(n) = \{1, 2, 3, 4\}$. That means $x_3(n)$ is obtained by delaying $x(n)$ by 2 positions.

$\therefore \quad x_3(n) = x(n-2)$

According to circular time shifting property,

$$x(n-l) \xleftrightarrow{DFT} X(k)\, e^{\frac{-j2\pi kl}{N}}$$

$$x(n-2) \xleftrightarrow{DFT} X(k)\, e^{\frac{-j4\pi k}{4}}$$

$\therefore \quad X_3(k) = X(k)\, e^{-j\pi k}$

For $k = 0 \Rightarrow X_3(0) = X(0)\, e^{\circ} = 10$

For $k = 1 \Rightarrow X_3(1) = X(1)\, e^{-j\pi} = (-2 + 2j)[\cos - j\sin\pi]$

$\therefore \quad X_3(1) = (-2 + 2j)(-1)$

$\therefore \quad X_3(1) = 2 - 2j$

For $k = 2 \Rightarrow X_3(2) = X(2) e^{-j2\pi}$

$= (-2)[\cos 2\pi - j\sin 2\pi]$

$\therefore \quad X_3(2) = (-2)(1)$

$\therefore \quad X_3(2) = -2$

For $k = 3 \Rightarrow X_3(3) = X(3)\, e^{-j3\pi}$

$= (-2 - j2)[\cos 3\pi - j\sin 3\pi]$

$\therefore \quad X_3(3) = (-2 - j2)[-1]$

$\therefore \quad X_3(3) = 2 + j2$

$\therefore \quad X_3(k) = \{10, 2 - j2, -2, 2 + j2\}$

(d) Given $x_4(n) = \{4, 6, 4, 6\}$ and $x(n) = \{1, 2, 3, 4\}$

Thus $x_4(n)$ and $x(n)$ are related as,

$$x_4(n) = x(n) + x(n \mp 2)$$

Using half period shift property,

$$X_4(k) = X(k) + (-1)^k X(k) \ldots [\because \; x(n \mp 2) \leftrightarrow (-1)^k X(k)]$$

For $k = 0 \Rightarrow X_4(0) = X(0) + (-1)^0 X(0) = 10 + 10$

$$X_4(0) = 20$$

For $k = 1 \Rightarrow X_4(1) = X(1) + (-1)^1 X(1) = -2 + 2j + 2 - 2j$

$$\therefore \quad X_4(1) = 0$$

For $k = 2 \Rightarrow X_4(2) = X(2) + (-1)^2 X(2) = -2 + (-2)$

$$\therefore \quad X_4(2) = -4$$

For $k = 3 \Rightarrow X_4(3) = X(3) + (-1)^3 X(3) = -2 - 2j + 2 + 2j$

$$\therefore \quad X_4(3) = 0$$

$$\therefore \quad X_4(k) = \{20, 0, -4, 0\}$$

Ex. 3.8.4 : For a given sequence x (n) = {2, 0, 0, 1}, perform following operations :

(i) Find out the 4 point DFT of x (n)

(ii) Plot x (n), its periodic extension $x_p(n)$ and $x_p(n-3)$

(iii) Find out 4 point DFT of $x_p(n-3)$

(iv) Add phase angle in (i) with factor $-\left[\frac{2\pi rk}{N}\right]$

where N = 4, r = 3 k = 0, 1, 2, 3

(v) Comment to the result you had in point (i) and (ii)

Soln. :

Given : x (n) = {2, 0, 0, 1}

(i) $X(k) = [W_4]\, x_4$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2+0+0+1 \\ 2+0+0+j \\ 2+0+0-1 \\ 2+0+0-j \end{bmatrix} = \begin{bmatrix} 3 \\ 2+j \\ 1 \\ 2-j \end{bmatrix}$$

$$\therefore \; X(k) = \{3, 2+j, 1, 2-j\}$$

(ii) The plot of x (n), $x_p(n)$ and $x_p(n-3)$ are shown in Figs. P. 3.8.4(a), P. 3.8.4(b) and P. 3.8.4(c) respectively.

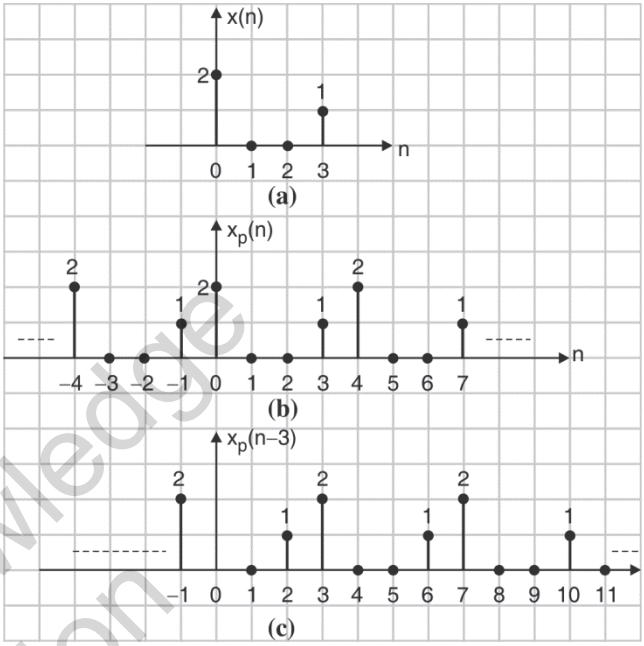


Fig. P. 3.8.4

(iii) In Fig. P. 3.8.4, consider the sequence in the basic range; that means n = 0 to 3.

$$\therefore \; x'(n) = \{0, 0, 1, 2\}$$

Comparing this to the original sequence,

x (n) = {2, 0, 0, 1} we can write,

$$x'(n) = x((n-3))$$

According to circular time shifting property,

$$x(n-l)_N \longleftrightarrow X(k)\, e^{\frac{-j2\pi kl}{N}}$$

Here $l = 3, \; N = 4$

and $X(k) = \{3, 2+j, 1, 2-j\}$

So, DFT of x′ (n) is,

$k = 0 \Rightarrow X(0)\, e^{-j0} = 3$

$k = 1 \Rightarrow X(1)\, e^{\frac{-j2\pi\, 1\, 3}{4}} = (2+j) \cdot e^{\frac{-j3\pi}{2}}$

$$= (2 + j)(0 + j) = -1 + j2$$

$$k = 2 \Rightarrow X(2) \cdot e^{\frac{-j2\pi\, 2\, 3}{4}} = 1 \cdot e^{-j3\pi} = -1$$

$$k = 3 \Rightarrow X(3)\, e^{\frac{-j2\pi\, 3\, 3}{4}}$$

$$= (2 - j) \cdot e^{\frac{-j9\pi}{2}} = (2 - j)(0 - j) = -1 - j2$$

$$\therefore \quad X'(k) = \{3, -1 + j2, -1, -1 - j2\}$$

(iv) We have, $X(k) = \{3, 2 + j, 1, 2 - j\}$

The added phase angle is $e^{\frac{-j2\pi rk}{N}}$; $N = 4$, $r = 3$ and $k = 0, 1, 2, 3$

That means phase angle is $e^{\frac{-j2\pi 3k}{4}} = e^{\frac{-j3\pi k}{2}}$

Due to this added phase angle DFT will be same as DFTOF (ii).

(v) The DFTOF $x_p(n)$ is same as that of $x(n)$; because DFT is periodic in nature.

Ex. 3.8.5 : Consider the finite length sequence x(n) shown in Fig. P. 3.8.5. The five point DFT of x (n) is denoted by X (k). Plot the sequence whole DFT is

$Y(k) = e^{-\frac{4\pi k}{5}} X(k)$

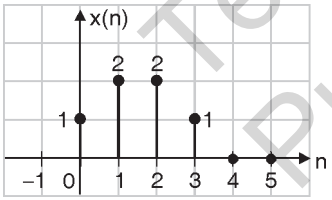


Fig. P. 3.8.5

Soln. : The given sequence contains 5 points. It can be written as,

$$x(n) = \{\underset{\uparrow}{1}, 2, 2, 1, 0\}$$

According to circular time shifting property,

$$x((n - l))_N \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)\, W_N^{kl}$$

We have $W_N = e^{\frac{-j2\pi}{N}}$ Here $N = 5$

$$\therefore \quad x((n - l))_5 \overset{\text{DFT}}{\underset{5}{\longleftrightarrow}} X(k)\, e^{\frac{-j2\pi\, kl}{5}} \qquad \ldots(1)$$

Given : $Y(k) = e^{\frac{-j4\pi 4}{5}} X(k)$ …(2)

Comparing Equations (1) and (2) we note $l = 2$

$$\therefore \quad y(n) = x(n - 2)_5$$

This simply means y(n) is obtained by circularly delaying x(n) by 2 position. We arrange the values of x(n) on a circle in the counter otherwise direction.

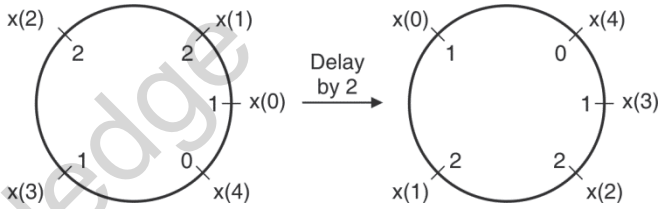


Fig. P. 3.8.5(a)

$$\therefore \quad y(n) = \{1, 0, 1, 2, 2\}$$

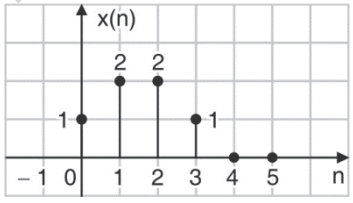


Fig. P. 3.8.5(b)

Ex. 3.8.6 :

$$x[n] = \begin{cases} 1 & 0 \le n \le 3 \\ 0 & 4 \le n \le 7 \end{cases}$$

(i) Find DFT X [k]

(ii) Using the result obtained in (i) find the DFT of the following sequences.

$$x_1[n] = \begin{cases} 1 & n = 0 \\ 0 & 1 \le n \le 4 \\ 1 & 5 \le n \le 7 \end{cases}$$

and $$x_2[n] = \begin{cases} 0 & 0 \le n \le 1 \\ 1 & 2 \le n \le 5 \\ 0 & 6 \le n \le 7 \end{cases}$$

Soln. :

(i) **Given,** $x(n) = \begin{cases} 1, & 0 \le n \le 3 \\ 0, & 4 \le n \le 7 \end{cases}$

$$\therefore \quad x(n) = \{1, 1, 1, 1, 0, 0, 0, 0\}$$

The DFT of x(n) is calculated as follows.

$$\therefore \quad X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0.707 - j\,0.707 & -j & -0.707 - j\,0.707 & -1 & -0.707 + j\,0.707 & j & 0.707 + j\,0.707 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -0.707 - j\,0.707 & j & 0.707 - j\,0.707 & -1 & 0.707 + j\,0.707 & -j & -0.707 + j\,0.707 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -0.707 + j\,0.707 & -j & 0.707 + j\,0.707 & -1 & 0.707 - j\,0.707 & j & -0.707 - j\,0.707 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ 1 & 0.707 + j\,0.707 & j & -0.707 + j\,0.707 & -1 & -0.707 - j\,0.707 & -j & 0.707 - j\,0.707 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\therefore \quad X(k) = \begin{bmatrix} 1+1+1+1+0+0+0+0 \\ 1+0.707-j0.707-j-0.707-j0.707+0+0+0+0 \\ 1-j-1+j+0+0+0+0 \\ 1-0.707-j0.707+j+0.707-j0.707+0+0+0+0 \\ 1-1+1-1+0+0+0+0 \\ 1-0.707+j0.707-j+0.707+j0.707 \\ 1+j-1-j+0+0+0+0 \\ 1+0.707+j0.707+j-0.707+j0.707+0+0+0+0 \end{bmatrix}$$

$$\therefore \quad X(k) = \{4,\ 1 - j2.414,\ 0,\ 1 - j0.414,\ 0,\ 1 + j0.414,\ 0,\ 1 + j2.414\}$$

(ii) Given $x_1(n) = \begin{cases} 1, & n = 0 \\ 0, & 1 \le n \le 4 \\ 1, & 5 \le n \le 7 \end{cases}$

$$x_1(n) = \{1, 0, 0, 0, 0, 1, 1, 1\}$$

Comparing with x(n), we can write

$$x_1(n) = x((n-5))$$

According to circular time shifting property,

$$x_1((n-l)) \overset{DFT}{\underset{N}{\leftrightarrow}} X(k) \cdot W_N^{kl}$$

$$\therefore \quad x_1(n) = x((n-5)) \overset{DFT}{\underset{N}{\leftrightarrow}} X(k) \cdot W_N^{k.5}$$

Now $W_N^k = e^{\frac{-j2\pi k}{N}}$

$$\therefore \ W_N^{5k} = e^{\frac{-j2\pi}{8} \times 5k}$$

$$= e^{-j\frac{10\,\pi\,k}{8}} = -0.707 + j0.707$$

$X_1(0) = e^0 \cdot X(0) = 4$

$X_1(1) = e^{-j\frac{10\pi}{8}} X(1)$

$= (-0.707 + j0.707)(1 - j2.414) = 1 + j\,2.41$

$X_1(2) = e^{-j10\frac{\pi}{8} \times 2}\ X(2) = e^{-j\frac{20\pi}{8}} X(2) = 0$

$X_1(3) = e^{-j\frac{10\pi}{8} \times 3}\ X(3) = e^{-j\frac{30\pi}{8}} X(3)$

$= (0.707 + j0.707)(1 - j0.414) = 1 + j0.414$

$X_1(4) = e^{-j\frac{10\pi}{8} \times 4}\ X(4) = 0$

$X_1(5) = e^{-j\frac{10\pi}{8} \times 5} X(5)$

$= (0.707 - j0.707)(1 + j0.414) = 1 - j0.414$

$X_1(6) = e^{-j\frac{10\pi}{8} \times 6}\ X(6) = 0$

$X_1(7) = e^{-j\frac{10\pi}{8} \times 7} X(7)$

$= e^{\frac{-j70\pi}{8}} (1 + j2.414) = 1 - j\,2.41$

$\therefore \quad X_1(k) = \{4,\ 2.41 - j,\ 0,\ 1 + j0.414,\ 0,$

$1 - j0.414,\ 0,\ 2.41 + j\}$

Given $x_2(n) = \begin{cases} 0, & 0 \le n \le 1 \\ 1, & 2 \le n \le 5 \\ 0, & 6 \le n \le 7 \end{cases}$

$$\therefore \quad x_2(n) = \{0, 0, 1, 1, 1, 1, 0, 0\}$$

$$\text{Thus} \quad x_2(n) = x((n+2))$$

Using circular time shifting property

$$\therefore \quad x_2(n) = x((n+2)) \overset{\text{DFT}}{\underset{N}{\leftrightarrow}} X(k) \cdot W_N^{-kl}$$

$$\therefore \quad X_2(k) = X(k) \cdot W_8^{-2k}$$

$$\text{Now} \quad W_8^{2k} = e^{\frac{-j2\pi.2k}{8}} = e^{\frac{-j\pi k}{z}}$$

$$\therefore \quad X_2(k) = X(k) \cdot e^{-\frac{j\pi k}{2}}$$

$$\therefore \quad X_2(0) = x(0) \cdot e^0 = 4$$

$$X_2(1) = x(1) \cdot e^{-\frac{j\pi}{2}} = (1 - j2.414)\,(0 + j1)$$

$$= -2.414 - j$$

$$X_2(2) = X(2) \cdot e^{j\pi} = 0$$

$$X_2(3) = X(3)\, e^{\frac{j3\pi}{2}} = (1 - j0.414)\,(0 - j) = 0.414 + j$$

$$X_2(4) = X(4)\, e^{j2\pi} = 0$$

$$X_2(5) = X(5)\, e^{\frac{j5\pi}{2}} = (1 + j0.414)\,(0 + j) = 0.414 - j$$

$$X_2(6) = X(6)\, e^{j3\pi} = 0$$

$$X_2(7) = X(7)\, e^{\frac{j7\pi}{2}} = (1 + j2.414)\,(0 - j) = -2.414 - j$$

$$X_2(k) = \{4, -2.414 - j, 0, 0.414 + j, 0, 0.414 - j, 0 - 2.414 + j\}$$

3.8.4 Circular Frequency Shift

If $x(n) \overset{\text{DFT}}{\longleftrightarrow} X(k)$

then $x(n)\, e^{\frac{+j2\pi mn}{N}} \overset{\text{DFT}}{\longleftrightarrow} X((k-m))_N$

Proof :

$$\text{DFT}\{x(n)\} = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$\therefore \text{DFT}\{x(n)\, e^{\frac{+j2\pi mn}{N}}\}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{\frac{+j2\pi mn}{N}} \cdot e^{\frac{-j2\pi nk}{N}}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi n(k-m)}{N}}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi n(N+k-m)}{N}}$$

$$= X(N + k - m)$$

$$= X((k-m))_N$$

$$\text{DFT}\{x(n)\, e^{+j2\pi mn/N}\} = X((k-m))_N$$

This is the circular frequency shift property.

Solved Example

Ex. 3.8.7 : Given signal x(n) = {2, 2, 1, 4} has a 4-point DFT X(k). Without performing DFT or IDFT, find out the sequence $x_1(n)$ which would have a 4 point DFT X(k – 1).

Soln. : This is what the problem states

$$x(n) \overset{\text{DFT}}{\longleftrightarrow} X(k)$$

then

$$?? \overset{\text{DFT}}{\longleftrightarrow} X(k-1)$$

The output in the frequency domain is shifted. Hence we use the circular frequency shift property.

i.e., if $x(n) \overset{\text{DFT}}{\longleftrightarrow} X(k)$

then $x(n)\, e^{\frac{j2\pi m \cdot n}{N}} \overset{\text{DFT}}{\longleftrightarrow} X(k-m)$

In this case m = 1

$$\therefore \quad x(n)\, e^{\frac{j2\pi n}{N}} \overset{\text{DFT}}{\longleftrightarrow} X(k-1)$$

$$\therefore \quad x_1(n) = x(n)\, e^{\frac{j2\pi n}{4}}$$

$$x_1(0) = x(0)\, e^{\frac{j2\pi 0}{4}} = x(0) = 2$$

$$x_1(1) = x(1) e^{\frac{j2\pi.1}{4}} = x(1) \cdot (+j) = 2j$$

$$x_1(2) = x(2)\, e^{\frac{j2\pi.2}{4}} = x(2) \cdot (-1) = -1$$

$$x_1(3) = x(3)\, e^{\frac{j2\pi 3}{4}} = x(3) \cdot (-j) = -4j$$

$$\therefore \quad x_1(n) = \{2, 2j, -1, -4j\}$$

Ex. 3.8.8 : For a discrete time sequence x(n) = (1, 2, 3, 4), DFT is given by X(k) = {10, – 2 + 2j, – 2, – 2 – 2j}. Compute the DFT of $\hat{x}$ (n) = {3, 4, 1, 2} using circular time shift property of DFT.

Soln. :

Given : $x(n) = \{1, 2, 3, 4\}$ and

$\hat{x}(n) = \{3, 4, 1, 2\}$.

That means $\hat{x}(n)$ is obtained by circularly delaying x (n) by 2 positions. It is as shown in Fig. P. 3.8.8.

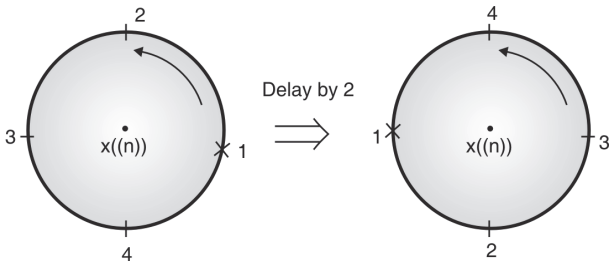


Fig. P. 3.8.8

$$\therefore \hat{x}(n) = x((n-2))$$

According to circular time shifting property,

$$x((n-l))_N \xleftrightarrow[N]{DFT} X(k)\, W_N^{kl}$$

Here, $N = 4$ and $l = 2$

$$\therefore \hat{x}(n) = x((n-2))_4 \xleftrightarrow[4]{DFT} X(k)\, W_4^{2k}$$

but, $W_N = e^{\frac{-j2\pi}{N}}$

$$\therefore W_4 = e^{\frac{-j2\pi}{4}} = e^{\frac{-j\pi}{2}}$$

Let, DFT of $\hat{x}(n) = \hat{X}(k)$

$$\therefore \hat{X}(k) = X(k) \cdot W_4^{2k} = X(k) \cdot e^{\frac{-j\pi}{2} \cdot 2k}$$

$$\therefore \hat{X}(k) = X(k)\, e^{-j\pi k}$$

Given, $X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$

for $k = 0 \Rightarrow \hat{X}(0) = X(0)\, e^0 = X(0) = 10$

for $k = 1 \Rightarrow \hat{X}(1) = X(1)\, e^{-j\pi}$

$= (-2 + 2j)(\cos\pi - j\sin\pi)$

$= (-2 + 2j)(-1) = 2 - 2j$

for $k = 2 \Rightarrow \hat{X}(2) = X(2)\, e^{-j2\pi}$

$= -2(\cos 2\pi - j\sin 2\pi) = -2$

and for $k = 3 \Rightarrow \hat{X}(3) = X(3)\, e^{-j3\pi}$

$= (-2 - 2j)(\cos 3\pi - j\sin 3\pi)$

$= (-2 - 2j)(-1) = 2 + 2j$

$$\therefore \hat{X}(k) = \{10, 2 - 2j, -2, 2 + 2j\}$$

3.8.5 Time Reversal

MU - Dec. 2015, Dec. 2017, May 2018

Q. State the DFT Properties : Time Reversal **(Dec. 2015, Dec. 2017, May 2018, 2 Marks)**

If x(n) is periodic,

$$x((-n))_N = x(N - m) \qquad 0 \le n \le N - 1$$

If $x(n) \xleftrightarrow{DFT} X(k)$

$$x(N - n) \xleftrightarrow{DFT} X(N - k)$$

OR

$$x((-n))_N \xleftrightarrow{DFT} X(N - k)$$

Proof : $DFT\{x(m)\} = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$

$$\therefore DFT\{x(N - n)\} = \sum_{n=0}^{N-1} x(N - n)\, e^{\frac{-j2\pi nk}{N}}$$

Let, $N - n = m \qquad \therefore n = N - m$

$$\therefore DFT\{x(N - n)\} = \sum_{m=0}^{N-1} x(m)\, e^{\frac{j\pi k(N-m)}{N}}$$

$$= \sum_{m=0}^{N-1} x(m)\, e^{\frac{-j2\pi k N}{N}} \cdot e^{\frac{j2\pi km}{N}}$$

but,

$$= \sum_{m=0}^{N-1} x(m)\, e^{\frac{j\pi km}{N}}$$

We now insert $e^{\frac{-j2\pi mN}{N}}$ since $e^{\frac{-j2\pi mN}{N}}$

$\therefore$ We get,

$$DFT\{x(N - m)\} = \sum_{m=0}^{N-1} x(m)\, e^{\frac{-j2\pi km}{N}} \cdot e^{\frac{-j2\pi mN}{N}}$$

$$= \sum x(m)\, e^{\frac{-j2\pi m(N-k)}{N}}$$

Comparing this with the standard DFT equation, we have

$$\{x(N - m)\} \xleftrightarrow{DFT} X(N - k)$$

OR

$$x((-m))_N \xleftrightarrow{DFT} X(N - k)$$

$$x((-n))_N \xleftrightarrow{DFT} X((-k))_N$$

Solved Example

Ex. 3.8.9 : Find the DFT of the given sequence x(n)= {1, 2, 3, 4}. Using the results obtained in (i) and not otherwise find the DFT of the signal $x_1(n) = \{1, 4, 3, 2\}$

Soln. :

(i) We find the DFT using the matrix method

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$X(k) = [W_N] \cdot x \quad ; \quad W_N \text{ is the DFT matrix.}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\therefore \quad X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$$

(ii) Here $x_1(n) = \{1, 4, 3, 2\}$

On careful observation, we note that

$x_1(n) = x(-n)$ (We assume x (n) to be periodic)

i.e., if $x(n) \xleftrightarrow{DFT} X(k)$

then $x(-n) \xleftrightarrow{DFT} X(-k)$

We have already shown that

$$X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$$

$$\therefore \quad X(-k) = \{10, -2 - 2j, -2, -2 + 2j\}$$

$$\therefore \quad DFT\{x_1(n)\} = \{10, -2 - 2j, -2, -2 + 2j\}$$

3.8.6 Symmetry Property

- The symmetry properties of DFT are derived in the similar way of DTFT symmetry properties. We know that DFT of sequence x (n) is denoted by X (k).
- Now if x (n) and X (k) are complex valued sequence then it can be represented as follows :

$$x(n) = x_R(n) + j\, x_I(n), \; 0 \le n \le N-1 \qquad \ldots(3.8.4)$$

and $$X(k) = X_R(k) + j\, X_I(k), \; 0 \le k \le N-1 \qquad \ldots(3.8.5)$$

Here 'R' stands for real part and 'I' stands for imaginary part.

According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \qquad \ldots(3.8.6)$$

Putting Equation (3.8.5) in Equation (3.8.6)

$$X(k) = \sum_{n=0}^{N-1} [\, x_R(n) + j\, x_I(n)\,]\, e^{\frac{-j2\pi kn}{N}} \qquad \ldots(3.8.7)$$

But according to Euler's identify,

$$e^{\frac{-j2\pi kn}{N}} = \cos\left(\frac{2\pi kn}{N}\right) - j \sin\left(\frac{2\pi kn}{N}\right)$$

Putting this value in Equation (3.8.7) we get,

$$X(k) = \sum_{n=0}^{N-1} [\, x_R(n) + j\, x_I(n)\,] \left[\cos\left(\frac{2\pi kn}{N}\right) - j \sin\left(\frac{2\pi kn}{N}\right)\right]$$

$$\therefore \; X(k) = \sum_{n=0}^{N-1} \left[x_R(n) \cdot \cos\left(\frac{2\pi kn}{N}\right) - j\, x_R(n) \sin\left(\frac{2\pi kn}{N}\right) + j\, x_I(n) \cdot \cos\left(\frac{2\pi kn}{N}\right) - j^2\, x_I(n) \sin\left(\frac{2\pi kn}{N}\right)\right]$$

Here $j^2 = -1$; and writing summation for real and imaginary parts separately we get,

$$X(k) = \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi kn}{N}\right) + x_I(n) \sin\left(\frac{2\pi kn}{N}\right)\right] - j \sum_{n=0}^{N-1} \left[x_R(n) \sin\left(\frac{2\pi kn}{N}\right) - x_I(n) \cos\left(\frac{2\pi kn}{N}\right)\right] \qquad \ldots(3.8.8)$$

Comparing Equations (3.8.8) and (3.8.5) we can write,

$$X_R(k) = \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi kn}{N}\right) + x_I(n) \sin\left(\frac{2\pi kn}{n}\right)\right] \qquad \ldots(3.8.9)$$

and

$$X_I(k) = -\sum_{n=0}^{N-1} \left[x_R(n) \sin\left(\frac{2\pi kn}{N}\right) - x_I(n) \cos\left(\frac{2\pi kn}{N}\right)\right] \qquad \ldots(3.8.10)$$

Equations (3.8.9) and (3.8.10) are obtained by using definition of DFT. Similarly we can obtain real and imaginary parts of x (n) using definition of IDFT.

$$x_R(n) = \frac{1}{n} \sum_{k=0}^{N-1} \left[X_R(k) \cos\left(\frac{2\pi kn}{N}\right) - X_I(k) \sin\left(\frac{2\pi kn}{N}\right)\right] \qquad \ldots(3.8.11)$$

and

$$x_I(n) = \frac{1}{N}\sum_{k=0}^{N-1}\left[X_R(k)\sin\left(\frac{2\pi kn}{N}\right) + X_I(k)\cos\left(\frac{2\pi kn}{N}\right)\right] \quad \ldots(3.8.12)$$

Now we will consider different cases as follows :

Case (i) : When x (n) is real valued

Statement : If x (n) is real valued then

$$X(N-k) = X(-k) = X^*(k)$$

Proof : According to the definition of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn} \quad \ldots(3.8.13)$$

Replacing k by N – k,

$$X(N-k) = \sum_{n=0}^{N-1} x(n)\, W_N^{(N-k)n}$$

$$\therefore \quad X(N-k) = \sum_{n=0}^{N-1} x(n)\, W_N^{Nn}\, W_N^{-kn} \quad \ldots(3.8.14)$$

Now we have, twiddle factor $W_N = e^{-\frac{j2\pi}{N}}$

$$\therefore \quad W_N^{Nn} = \left(e^{-\frac{j2\pi}{N}}\right)^{Nn}$$

$$= e^{-j2\pi n} = \cos 2\pi n - j \sin 2\pi n$$

Since n is an integer, cos 2 πn = 1 and sin 2 πn = 0

$$\therefore \quad W_N^{Nn} = 1$$

Thus Equation (3.8.14) becomes,

$$X(N-k) = \sum_{n=0}^{N-1} x(n)\, W_N^{-kn} \quad \ldots(3.8.15)$$

Comparing Equation (3.8.15) with definition of DFT (Equation (3.8.13)) we get,

$$X(N-k) = X(-k) \quad \ldots(3.8.16)$$

Now using Equation (3.8.13) we can write,

$$X^*(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{-kn} \quad \ldots(3.8.17)$$

Thus, from Equations (3.8.16) and (3.8.17) we get,

$$X(N-k) = X(-k) = X^*(k)$$

Case (ii) : When x (n) is real and even

Statement : When x (n) is real and even which means,

$$x(n) = x(N-n) \text{ then DFT becomes,}$$

$$X(k) = X_R(k)$$

Proof : Since imaginary part is zero; putting $x_I(n) = 0$ in Equation (3.8.9) we get,

$$X_R(k) = \sum_{n=0}^{N-1} X_R(n)\cos\left(\frac{2\pi kn}{N}\right)$$

Similarly IDFT can be written by putting $X_I(k) = 0$ in Equation (3.8.11).

$$\therefore \quad X_R(n) = \frac{1}{N}\sum_{n=0}^{N-1} X_R(k)\cos\left(\frac{2\pi kn}{N}\right)$$

Case (iii) : When x (n) is real and odd

Statement : When x (n) is real and odd which means,

$$x(n) = -x(N-n) \text{ then the DFT becomes,}$$

$$X(k) = -j\sum_{n=0}^{N-1} x_R(n)\sin\left(\frac{2\pi kn}{N}\right)$$

Proof : Since x (n) is real, we will put $x_I(n) = 0$ in Equation (3.8.8). Similarly x (n) is odd and 'cos' is even function so we can write, $\cos\left(\frac{2\pi kn}{N}\right) = 0$. Thus first summation in Equation (3.8.8) becomes zero. In the second summation of Equation (3.8.8), putting $x_I(n) = 0$ we get,

$$X(k) = -j\sum_{n=0}^{N-1} x_R(n)\sin\left(\frac{2\pi kn}{N}\right)$$

Similarly IDFT can be written as,

$$x(n) = j\frac{1}{N}\sum_{k=0}^{N-1} X_R(k)\sin\left(\frac{2\pi kn}{N}\right)$$

Case (iv) : When x (n) is purely imaginary sequence

When x (n) is purely imaginary which means $x_R(n) = 0$ and $x(n) = j\, x_I(n)$ then putting $x_R(n) = 0$ in Equation (3.8.9) we get,

$$X_R(k) = \sum_{n=0}^{N-1} x_I(n)\sin\left(\frac{2\pi kn}{N}\right)$$

And putting $x_R(n) = 0$ in Equation (3.8.10) we get,

$$X_I(k) = \sum_{n=0}^{N-1} x_I(n)\cos\left(\frac{2\pi kn}{N}\right)$$

Symmetry properties can be summarized as shown in Fig. 3.8.2.

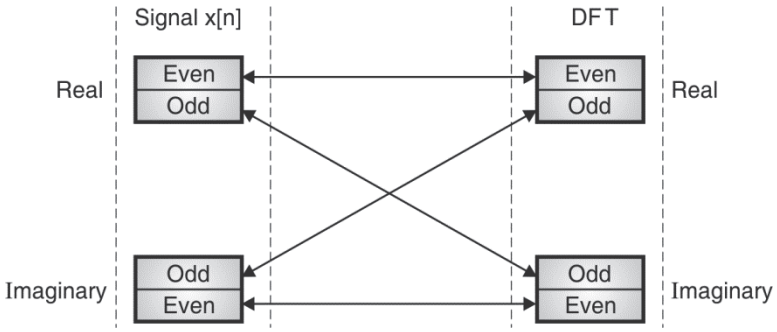


Fig. 3.8.2 : Summary of symmetry property

This summary is shown in Table 3.8.1.

Table 3.8.1

N point sequence x [n] $0 \le n \le N-1$	N point DFT
$x^*[n]$	$X^*[N-k]$
$x^*[N-n]$	$X^*[k]$
$x_R[n]$	$X_{ce}[k] = \frac{1}{2}[X[k] + X^*[N-k]]$
$j\,x_I[n]$	$X_{ce}[k] = \frac{1}{2}[X[k] - X^*[N-k]]$
$x_{ce}[n] = \frac{1}{2}[x[n] + x^*[N-n]]$	$X_R[k]$
$x_{co}[n] = \frac{1}{2}[x[n] + x^*[N-n]]$	$j\,X_I[k]$

Solved Example

Ex. 3.8.10 : The first five points of the 8 point DFT of a real valued sequence are,
{0.25, 0.125 – j0.3018, 0, 0.125 – j 0.0518, 0} Determine the remaining three points.

Soln. : Given DFT points are :

$X(0) = 0.25$

$X(1) = 0.125 - j\,0.3018$

$X(2) = 0$

$X(3) = 0.125 - j\,0.0518$

$X(4) = 0$

Given sequence is a real valued sequence. According to the symmetry property we have,

$$X^*(k) = X(N-k)$$

$$\text{or } X(k) = X^*(N-k) \quad \ldots(1)$$

This is 8 point DFT. Thus N = 8

$$\therefore X(k) = X^*(8-k) \quad \ldots(2)$$

Now we want remaining three samples namely X (5) X (6) and X (7). Putting k = 5 in Equation (2),

$$X(5) = X^*(8-5) = X^*(3)$$

We have $X(3) = 0.125 - j\,0.0518$

$\therefore X^*(3) = 0.125 + j\,0.0518$

$\therefore X(5) = 0.125 + j\,0.0518$

Putting k = 6 in Equation (2),

$$X(6) = X^*(8-6) = X^*(2)$$

We have $X(2) = 0$, Thus $X^*(2) = 0$

$\therefore X(6) = 0$

Similarly putting k = 7 in Equation (2) we get,

$$X(7) = X^*(8-7) = X^*(1)$$

We have $X(1) = 0.125 - j\,0.3018$

$\therefore X(7) = 0.125 + j\,0.3018$

Ex. 3.8.11 : The first five DFT points of real and even sequence x(n) of length eight are given below. Find remaining three points.

$$X(k) = \{5, 1, 0, 2, 3, \ldots\}$$

Soln. : Given DFT points are :

$$X(0) = 5, X(1) = 1, X(2) = 0,$$

$$X(3) = 2 \text{ and } X(4) = 3.$$

According to symmetry property we have,

$$X^*(k) = X(N-k)$$

$$\therefore X(k) = X^*(N-k)$$

This is 8 point DFT. Thus N = 8

$$\therefore X(k) = X^*(8-k)$$

$$\therefore X(5) = X^*(8-5) = X^*(3)$$

$$\therefore X(5) = 2$$

$$X(6) = X^*(8-6) = X^*(2)$$

$$\therefore X(6) = 0$$

$$X(7) = X^*(8-7) = X^*(1)$$

$$\therefore X(7) = 1$$

Ex. 3.8.12 : Compute DFT of the sequence $x_1(n) = \{1, 2, 4, 2\}$ using property and not otherwise compute DFT of $x_2(n) = \{1 + j, 2 + 2j, 4 + 4j, 2 + 2j\}$

Soln. : $x_1(n) = \{1, 2, 4, 2\}$

$$\therefore X_1(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 4 \\ 2 \end{bmatrix}$$

$$\therefore X_1(k) = \begin{bmatrix} 1+2+4+2 \\ 1-2j-4+2j \\ 1-2+4-2 \\ 1+2j-4-2j \end{bmatrix} = \begin{bmatrix} 9 \\ -3 \\ 1 \\ -3 \end{bmatrix}$$

Now $x_2(n) = \{1 + j, 2 + 2j, 4 + 4j, 2 + 2j\}$

$$\therefore x_2(n) = x_1(n) + j\, x_1(n)$$

$$\therefore X_2(k) = X_1(k) + j\, X_1(k)$$

$$\therefore X_2(k) = \{9 + 9j, -3 - 3j, 1 + j, -3 - 3j\}$$

Ex. 3.8.13 : For a 8-point DFT, the first five DFT coefficients are $X(k) = X\{10, 2 + 3j, 1 + 2j, j, 4\}$ Find the remaining coefficients.

Soln. :

Given :

$$X(0) = 10, \quad X(1) = 2 + 3j, \quad X(2) = 1 + 2j,$$

$$X(3) = j, \quad X(4) = 4$$

We know, $X(k) = X^*(N - k)$

In this case, N = 8

$$\therefore X(5) = X^*(8-5) = X^*(3) = +j$$

$$X(6) = X^*(8-6) = X^*(2) = 1 - 2j$$

$$X(7) = X^*(8-7) = X^*(1) = 2 - 3j$$

Therefore the entire DFT sequence is,

$$X(k) = \{10, 2 + 3j, 1 + 2j, j, 4, j, 1 - 2j, 2 - 3j\}$$

Note : The magnitude of the DFT is symmetric about the N/2 point. While the phase of the DFT is antisymmetric about the N/2 point.

Ex. 3.8.14 : The first five points of the 8-point DFT of a real valued sequence are,

$X(k) = \{10, 2.2 - j\,0.3, 0, 1.2 - j0.13, 0\}$

Determine the remaining three points.

Soln. : (We use the symmetry property here)

i.e. $X(k) = X^*(N - k)$

We have,

$$X(0) = 10, \quad X(1) = 2.2 - j0.3, \quad X(2) = 0,$$

$$X(3) = 1.2 - j\,0.13, \quad X(4) = 0$$

Since this is a 8-point DFT, N = 8

$$\therefore X(k) = X^*(8-k)$$

$$\therefore X(5) = X^*(8-5) = X^*(3)$$

$$X(6) = X^*(8-6) = X^*(2)$$

$$X(7) = X^*(8-7) = X^*(1)$$

$$\therefore X(5) = 1.2 + j0.13$$

$$X(6) = 0$$

$$X(7) = 2.2 + j0.3$$

$\therefore$ The final 8-point DFT sequence is,

$$X(k) = \{10, 2.2 - j\,0.3, 0, 1.2 - j0.13, 0, 2.2 + j0.3\}$$

Ex. 3.8.15 : A sequence is given as x (n) = {1 + 2j, 1 + 3j, 2 + 4j, 2 + 2j}. From basic definition. Find X (k).

If $x_1(n) = \{1, 1, 2, 2\}$

$x_2(n) = \{2, 3, 4, 2\}$

Find $X_1(k)$ and $X_2(k)$ by using DFT only.

Soln. :

Given : x (n) = {1 + 2j, 1 + 3j, 2 + 4j, 2 + 2j}

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1+2j \\ 1+3j \\ 2+4j \\ 2+2j \end{bmatrix}$$

$$= \begin{bmatrix} 1+2j+1+3j+2+4j+2+2j \\ 1+2j-j+3-2-4j+2j-2 \\ 1+2j-1-3j+2+4j-2-2j \\ 1+2j+j-3-2-4j-2j+2 \end{bmatrix}$$

$$= \begin{bmatrix} 6+11j \\ -j \\ +j \\ -2-3j \end{bmatrix}$$

$\therefore$ X (k) = {6 + +11j, – j, j, – 2 – 3j}

Now, $x = x_1(n) + j\, x_2(n)$

$x^*(n) = x_1(n) - j\, x_2(n)$

$x(n) + x^*(n) = 2\, x_1(n)$

$$x_1(n) = \frac{x(n) + x^*(n)}{2}$$

$$\therefore \quad X_1(k) = \frac{X(k) + X^*(-k)}{2}$$

And $$X_2(k) = \frac{X(k) - X^*(-k)}{2j}$$

(a) $$X_1(k) = \frac{X(k) + X^*(-k)}{2}$$

X (k) = {6 + 11j, – j, + j, – 2 – 3j}

$X^*(k)$ = {6 – 11j, + j, – j, – 2 + j3}

$X^*(-k)$ = {6 – 11j, – 2, + j3, – j + j}

Hence, $$X_1(0) = \frac{X(0) + X^*(0)}{2}$$

$$= \frac{6 + 11j + 6 - 11j}{2} = 6$$

$$X_1(1) = \frac{X(1) + X^*(-1)}{2}$$

$$= \frac{-j - 2 + j3}{2} = -1 + j$$

$$X_1(2) = \frac{X(2) + X^*(-2)}{2} = \frac{j - j}{2} = 0$$

$$X_1(3) = \frac{X(3) + X^*(-3)}{2}$$

$$= \frac{-2 - j3 + j}{2} = -1 - j$$

$X_1(k) = \{X_1(0), X_1(1), X_1(2), X_1(3)\}$

= {6, – 1 + j, 0, – 1 – j}

(b) $$X_2(k) = \frac{X(k) - X^*(-k)}{2j}$$

$$X_2(0) = \frac{X(0) - X^*(0)}{2j}$$

$$= \frac{6 + 11j - 6 + 11j}{2j} = 11$$

$$X_2(1) = \frac{X(1) - X^*(-1)}{2j}$$

$$= \frac{-j + 2 - j3}{2j} = \frac{2 - j4}{2j}$$

$$= \frac{j}{j} \times \frac{2 - j4}{2j} = \frac{2j + 4}{-2} = -2 - j$$

$$X_2(2) = \frac{X(2) - X^*(-2)}{2j} = \frac{j + j}{2j} = 1$$

$$X_2(3) = \frac{X(3) - X^*(-3)}{2j}$$

$$= \frac{-2 - j3 - j}{2j} = \frac{j}{j}\left(\frac{-2 - j4}{2j}\right) = -2 + j$$

$X_2(k) = \{X_2(0), X_2(1), X_2(2), X_3(3)\}$

= {11, – 2 – j, 1, – 2 + j}

3.8.7 Complex Conjugate Property

If $x(n) \xleftrightarrow{DFT} X(k)$

then $x^*(n) \xleftrightarrow{DFT} X^*(N-k) = X^*((-k))_N$

Proof : From the DFT equation we have,

$$\text{DFT}\{x(n)\} = \sum_{N=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$\therefore \quad \text{DFT}\{x^*(n)\} = \sum_{n=0}^{N-1} x^*(n)\, e^{\frac{-j2\pi nk}{N}}$$

$$= \sum_{n=0}^{N-1} [x(n)\, e^{j2\pi nk/N}]^*$$

$$= \sum_{n=0}^{N-1} [x(n)\, e^{-j2\pi n(N-k)/N}]^*$$

$$= X^*(N-k)$$

$$\therefore \quad \text{DFT}[x^*(n)] = X^*(N-k) = X^*((-k))_N$$

3.8.8 Parseval's Theorem

If $x(n) \overset{DFT}{\longleftrightarrow} X(k)$

Then Energy of the signal is,

$$E = \sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

Proof : Energy of a signal is given by the equation,

$$E = \sum_{n=0}^{N-1} |x(n)|^2 \quad \ldots(3.8.18)$$

$$E = \sum_{n=0}^{N-1} x(n)\, X^*(n) \quad \ldots(3.8.19)$$

From the IDFT equation we have,

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)\, W_N^{+nK}$$

$$\therefore \quad x^*(n) = \frac{1}{N} \sum_{k=0}^{N-1} X^*(k)\, W_N^{+nK} \quad \ldots(3.8.20)$$

We substitute Equation (3.8.20) in Equation (3.8.19)

$$\therefore \quad E = \sum_{k=0}^{N-1} x(n) \left[\frac{1}{N} \sum_{n=0}^{W-1} X^*(k)\, W_N^{+nk} \right]$$

Rearranging the terms, we have,

$$E = \frac{1}{N} \sum_{k=0}^{N-1} X^*(k) \cdot \left[\sum_{k=0}^{N-1} x(n)\, W_N^{nk} \right]$$

$$\therefore \quad E = \frac{1}{N} \sum_{k=0}^{N-1} X^*(k) \cdot X(k)$$

$$E = \frac{1}{N} \Sigma |X(k)|^2 \quad \ldots(3.8.21)$$

$\therefore$ We have,

$$E = \sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

Solved Example

Ex. 3.8.16 : X(k) = {10, –2, 0, 2}.

Compute the energy of the signal x(n).

Soln. : From Parseval's Energy theorem, we have,

$$E = \sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

$$\therefore \quad E = \frac{1}{4} \{ |X(0)|^2 + |X(1)|^2 + |X(2)|^2 + |X(3)|^2 \}$$

$$= \frac{1}{4} \{ 100 + 4 + 0 + 4 \}$$

$$E = 27$$

Ex. 3.8.17 : Verify Parseval's Theorem for sequence,

$x(n) = \left(\frac{1}{2}\right)^n u(n)$

Soln. : According to Parseval's Theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2 \quad \ldots(1)$$

Step I : Need to prove this equality.

Consider value of N = 4. That means n = 0 to N – 1

Let, $\therefore$ n = 0 to 3

Thus sequence x (n) is generated as follows,

For $n = 0 \Rightarrow \quad x(0) = \left(\frac{1}{2}\right)^0 = 1$

$n = 1 \Rightarrow \quad x(1) = \left(\frac{1}{2}\right)^1 = \frac{1}{2}$

$n = 2 \Rightarrow \quad x(2) = \left(\frac{1}{2}\right)^2 = \frac{1}{4}$

$n = 3 \Rightarrow \quad x(3) = \left(\frac{1}{2}\right)^3 = \frac{1}{8}$

$$\therefore \quad x(n) = \left\{ \underset{\uparrow}{1}, \frac{1}{2}, \frac{1}{4}, \frac{1}{8} \right\}$$

Step II : Consider L.H.S. of Equation (1).

$$L.H.S = \sum_{n=0}^{N-1} |x(n)|^2 = \sum_{n=0}^{3} |x(n)|^2$$

$$L.H.S = |x(0)|^2 + |x(1)|^2 + |x(2)|^2 + |x(3)|^2$$

$$\therefore \quad L.H.S = 1 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{4}\right)^2 + \left(\frac{1}{8}\right)^2$$

$$\therefore \quad L.H.S = 1.328125 \quad \ldots(2)$$

Step III : Consider R.H.S term,

$$R.H.S = \frac{1}{N} \sum_{n=0}^{N-1} |X(k)|^2$$

First we will calculate DFT X (k) using matrix method.

$$X(k) = [W_4]\, X_N$$

$$\therefore \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{4} \\ \frac{1}{8} \end{bmatrix}$$

$$\therefore X(k) = \{1.875, \quad 0.75 - j0.375, \quad 0.625, \quad 0.75 + j0.375\}$$

Now are will calculate $|X(k)|$.

$$\therefore |X(k)| = \{1.875, 0.838525, 0.625, 0.838525\}$$
$$\uparrow$$

Step IV :

$$\text{R.H.S} = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

$$\therefore \text{R.H.S} = \frac{1}{4} \sum_{k=0}^{3} |X(k)|^2$$

$$\therefore \text{R.H.S} = \frac{1}{4} \{ |X(0)|^2 + |X(1)|^2 + |X(2)|^2 + |X(3)|^2 \}$$

$$\therefore \text{R.H.S} = 1.328125$$

Since L.H.S= R.H.S ; Parseval's theorem is verified.

Ex. 3.8.18 : State and prove Parseval's theorem for the following sequence x(n) = {1, 2, 3, 4}.

Soln. :

Given :

$$x(n) = \{1, 2, 3, 4\}$$

According to parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

Step I :

Here $N = 4$

$$\text{L.H.S.} = \sum_{n=0}^{3} |x(n)|^2$$

$$= |x(0)|^2 + |x(1)|^2 + |x(2)|^2 + |x(3)|^2$$

$$= 1 + (2)^2 + (3)^2 + (4)^2$$

$$= 1 + 4 + 9 + 16 = 30$$

Step II : First we will calculate DFT of x (n) that is X (k)

$$X(k) = [W_4]\, x_N$$

$$\therefore X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 10 \\ -2 + 2j \\ -2 \\ -2 - 2j \end{bmatrix}$$

$$\therefore X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$$

Step III :

$$\text{R.H.S.} = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

Now $|X(k)| = \{10, 2.828, 2, 2.828\}$
$\uparrow$

$$\therefore \text{R.H.S.} = \frac{1}{4} \sum_{k=0}^{3} |X(k)|^2$$

$$= \frac{1}{4} [|X(0)|^2 + |X(1)|^2 + |X(2)|^2 + |X(3)|^2]$$

$$= \frac{1}{4} [100 + 7.997584 + 4 + 7.997584]$$

R.H.S. = 30

Step IV : Since L.H.S. = R.H.S, Parseval's theorem is verified.

3.8.9 Multiplication of Two Sequences

If $x_1(n) \xleftrightarrow{\text{DFT}} X_1(k)$

and $x_2(n) \xleftrightarrow{\text{DFT}} X_2(k)$

then $x_1(n) \cdot x_2(n) \xleftrightarrow{\text{DFT}} \frac{1}{N} [X_1(k) \circledast X_2(k)]$

3.8.10 Circular Convolution Property

MU - Dec. 2015, Dec. 2017, May 2018

Q. State the DFT Properties : Convolution .
(Dec. 2015, Dec. 2017, May 2018, 2 Marks)

If $x_1(n) \xleftrightarrow{DFT} X_1(k)$

and $x_2(n) \xleftrightarrow{DFT} X_2(k)$

then, $x_1(n) \circledast x_2(n) \xleftrightarrow{DFT} X_1(k) \cdot X_2(k)$.

$$X_3(k) = X_1(k)\, X_2(k)$$

Hence, ($\circledast$) is the symbol used for circular convolution.

We find $x_3(n)$ for which DFT is $X_3(k)$. Some books also use the symbol Ⓝ to denote circular convolution.

Proof : From the Periodic convolution we know,

$$x_{3p}(n) = \sum_{m=0}^{N-1} x_{1p}(m)\, x_{2p}(n-m)$$

The subscript p stands for periodic.

$$\text{i.e., } x_3((n))_N = \sum_{m=0}^{N-1} x_1((m))_N \; x_2((n-m))_N$$

for $0 \le n \le N-1$ (one period)

$$x_3((n))_N = x_3(n),$$

$$x_1((m))_N = x_1(m)$$

$$\therefore \quad x_3(n) = \sum_{m=0}^{N-1} x_1(m)\; x_2((n-m))_N \qquad \ldots(3.8.22)$$

Now equation RHS of Equation (3.8.22), represents circular convolution of $x_1(n)$ and $x_2(n)$ which is represented as ($\circledast$).

$$\therefore \quad x_3(n) = x_1(n) \circledast x_2(n)$$

$$\therefore \quad DFT\{x_3(n)\} = DFT\{x_1(n) \circledast x_2(n)\}$$

$$= X_3(k) = X_1(k) \cdot X_2(k)$$

$$\therefore \quad x_1(n) \circledast x_2(n) \xleftrightarrow{DFT} X_1(k) \cdot X_2(k)$$

Hence, circular convolution reduces to multiplication while working with the DFT.

We will be solving a few examples after discussing one more property of the DFT.

3.9 Circular Convolution

One of the most important formula of DFT is that of circular convolution. We shall now see how to perform circular convolution.

Given two periodic sequences $x_1(n)$ and $x_2(n)$, circular convolution can be found out using the following two techniques.

1. Concentric circle method
2. Matrix multiplication method

We shall explain each one in detail.

3.9.1 Concentric Circle Method

Given two periodic sequences $x_1(n)$ and $x_2(n)$

$$y(n) = x_1(n) \circledast x_2(n)$$

Step 1 : Plot the samples of $x_1(n)$ evenly around the outer circle in the *counter clockwise* direction.

Step 2 : Plot the samples of $x_2(n)$ evenly around the inner circle in the *clockwise*. Make sure the positions of $x_1(0)$ and $x_2(0)$ coincide.

Step 3 : Multiply all the corresponding samples of $x_1(n)$ and $x_2(n)$ and add. This will give us $y_1(0)$.

Step 4 : Rotate the inner circle in the direction of the outer circle (*counter clockwise*) by one step and repeat step 3.

Step 5 : Repeat step 4 till $x_1(0)$ and $x_2(0)$ again coincide.

We shall take an example to understand these steps.

3.9.1(A) Solved Example on Concentric Circle Method

Ex. 3.9.1 : Perform circular convolution on the given two periodic sequences.

$x_1(n) = \{1, 2, 3, 4\}$, $x_2(n) = \{4, 1, 1, 2\}$

Soln. : $y(n) = x_1(n) \circledast x_2(n)$

We take two concentric circles. Plot $x_1(n)$ on the outer circle in the counter clockwise direction and $x_2(n)$ on the inner circle in the clockwise direction i.e.,

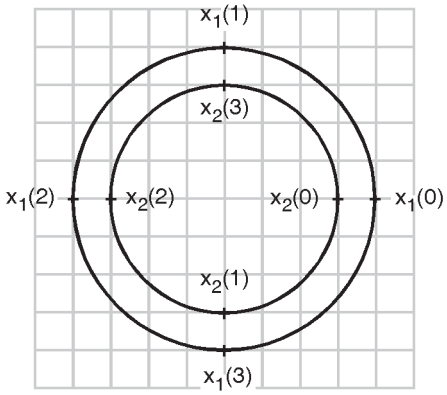


Fig. P. 3.9.1

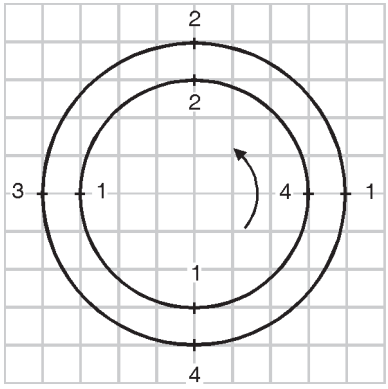


Fig. P. 3.9.1(a)

We multiply the corresponding samples and add. This gives us y(0).

$$\therefore \quad y(0) = 4 \times 1 + 2 \times 2 + 1 \times 3 + 1 \times 4 = 15$$

We now rotate the inner circle in the anticlock-wise direction by one step. This gives us,

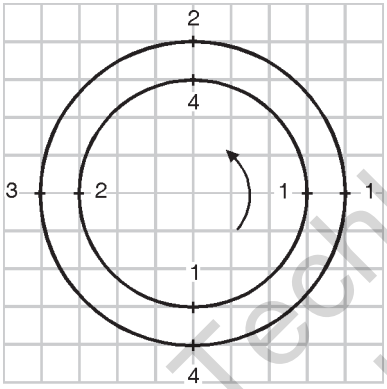


Fig. P.3.9.1(b)

$$\therefore \quad y(1) = 1 \times 1 + 4 \times 2 + 2 \times 3 + 1 \times 4 = 19$$

We continue rotating the inner circle in steps of one.

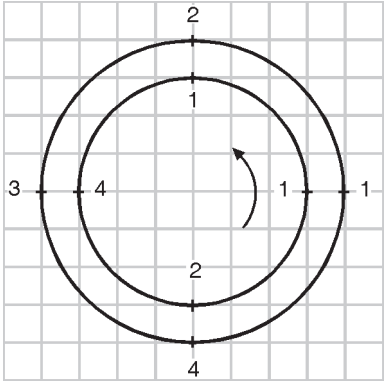


Fig. P.3.9.1(c)

$$y(2) = 1 \times 1 + 1 \times 2 + 4 \times 3 + 2 \times 4 = 23$$

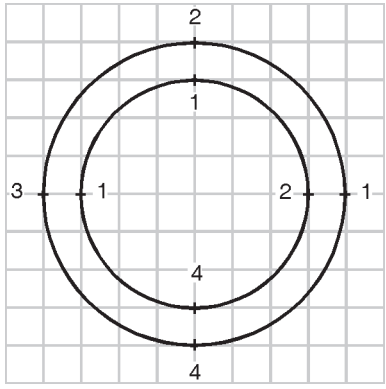


Fig. P. 3.9.1(d)

$$y(3) = 2 \times 1 + 1 \times 2 + 1 \times 3 + 4 \times 4 = 23$$

If we now rotate the inner circle, we come back to the starting point. Hence we stop here.

$$\therefore \quad y(n) = \{15, 19, 23, 23\}$$

The second method of performing circular convolution is using the Matrix method.

As is evident circular convolution can be performed only when $x_1(n)$ and $x_2(n)$ are of the same length.

If they are not of the same length, then we need to zero pad the shorter sequence and make it equal to the longer sequence.

Ex. 3.9.2 : Find the circular convolution of two sequences, $x_1(n) = \{1, -1, 2, -4\}$ and $x_2(n) = \{1, 2\}$

Soln. : Since $x_2(n)$ is shorter than $x_1(n)$, we zero pad it so that their lengths become equal

$$\therefore \quad x_2(n) = \{1, 2, 0, 0\}$$

We now perform circular convolution.

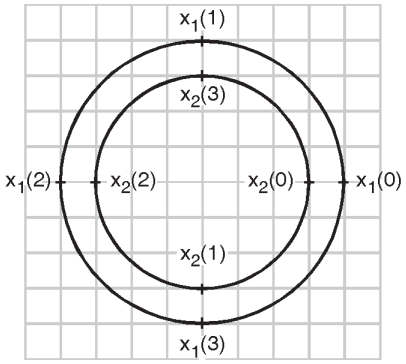


Fig. P. 3.9.2

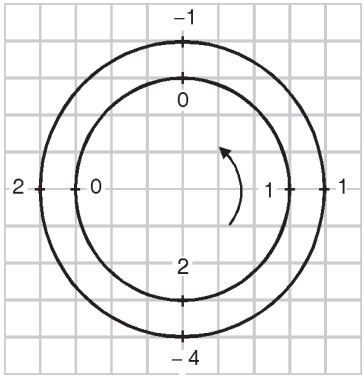


Fig. P. 3.9.2(a)

$$y(0) = 1 \times 1 + 0 \times -1 + 0 \times -2 + 2 \times -4 = -7$$

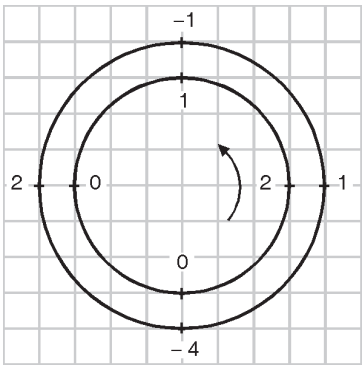


Fig. P. 3.9.2(b)

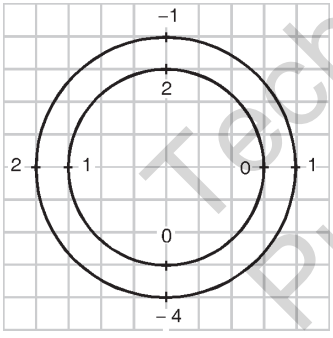


Fig. P. 3.9.3(c)

$$y(1) = 2 \times 1 + 1 \times -1 + 0 \times 2 + 0 \times 4 = 1$$

$$y(2) = 0 \times 1 + 2 \times -1 + 1 \times 2 + 0 \times -4 = 0$$

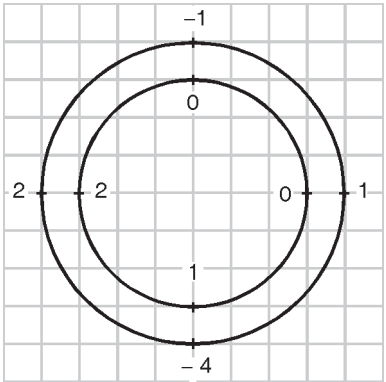


Fig. P. 3.9.2(d)

$$y(3) = 0 \times 1 + 0 \times -1 + 2 \times 2 + 1 \times -4$$

$$= 0$$

$$\therefore \quad y(n) = \{-7, 1, 0, 0\}$$

Ex. 3.9.3 : Determine

$$y(n) = x(n) \textcircled{N} h(n)$$

Where

$$x(n) = \{1, 2, 3, 1\}; h(n) = \{4, 3, 2, 2\}$$

Soln. :

We generate a circular matrix of h(n)

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 4 & 2 & 2 & 3 \\ 3 & 4 & 2 & 2 \\ 2 & 3 & 4 & 2 \\ 2 & 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$$\therefore \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 17 \\ 19 \\ 22 \\ 19 \end{bmatrix}$$

$$\therefore \quad y(n) = \{17, 19, 22, 19\}$$

Ex. 3.9.4 : For the following sequences

$$x_1(n) = \begin{cases} 1 & 0 \le n \le 2 \\ 0 & \text{otherwise} \end{cases}$$

$$x_2(n) = \begin{cases} 1 & 0 \le n \le 2 \\ 0 & \text{otherwise} \end{cases}$$

Compute linear convolution using circular convolution.

Soln. :

Given : $x_1(n) = \begin{cases} 1 & 0 \le n \le 2 \\ 0 & \text{otherwise} \end{cases}$

and $x_2(n) = \begin{cases} 1 & 0 \le n \le 2 \\ 0 & \text{otherwise} \end{cases}$

$$\therefore \quad x_1(n) = \{1, 1, 1\} \text{ and } x_2(n) = \{1, 1, 1\}$$

Here, $L_1 = L_2 = 3$

To result of linear convolution would have a length $L_1 + L_2 - 1$

$$\therefore \quad 3 + 3 - 1 = 5$$

We now append zeros to $x_1(n)$ and $x_2(n)$ to make their lengths equal to 5.

$$\therefore \quad x_1(n) = \{1,1,1,0,0\} \text{ and}$$

$$x_2(n) = \{1,1,1,0,0\}$$

The circular convolution is performed as follows.

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 1 \end{bmatrix}$$

$$\therefore \quad y(n) = \{1, 2, 3, 2, 1\}$$

This is the result that we would have got had we used circular convolution.

Let us verify the result by performing linear convolution. We use the tabular method.

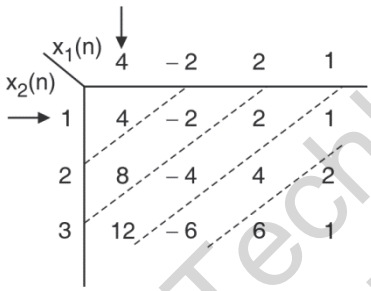


Fig. P. 3.9.4

$$\therefore \quad y(n) = \{1, 2, 3, 2, 1\}$$

Hence we get the same result.

Ex. 3.9.5 : Using the DFT method, obtain the circular convolution of the following :

$x_1(n) = [1 \;\; 2 \;\; 1 \;\; -2]$

$x_2(n) = [3 \;\; -2 \;\; 1 \;\; -3]$

Verify your result using the graphical method.

Soln. : From the DFT property we know,

$$x_1(n) \circledast x_2(n) \xleftrightarrow{\text{DFT}} X_1(k) \cdot X_2(k)$$

We begin with computing the DFT of $x_1(n)$ and $x_2(n)$

Step I :

Given :

$$x_1(n) = \{1, 2, 1, -2\}$$

Now, $X_1(k) = [W_4]\, X_{1N}$

$$\therefore \; X_1(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 1 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} 1+2+1-2 \\ 1-2j-1-2j \\ 1-2+1+2 \\ 1+2j-1+2j \end{bmatrix} = \begin{bmatrix} 2 \\ -4j \\ 2 \\ 4j \end{bmatrix}$$

Similarly,

$$X_2(k) = [W_4]\, X_{2N}$$

$$\therefore \; X_2(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 3 \\ -2 \\ 1 \\ -3 \end{bmatrix}$$

$$= \begin{bmatrix} 3-2+1-3 \\ 3+2j-1-3j \\ 3+2+1+3 \\ 3-2j-1+3j \end{bmatrix} = \begin{bmatrix} -1 \\ 2-j \\ 9 \\ 2+j \end{bmatrix}$$

Step II :

Let $Y(k) = X_1(k) \cdot X_2(k)$

$$= \{2, -4j, 2, 4j\} \cdot \{-1, 2-j, 9, 2+j\}$$

$$\therefore \; Y(k) = \{-2, -4-8j, 18, -4+8j\}$$

Step III :

The result of circular convolution is obtained by performing IDFT of Y (k) as follows,

$$y(n) = \frac{1}{N}\left[W_4^* \right] Y(k)$$

$$= \frac{1}{4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} -2 \\ -4-8j \\ 18 \\ -4+8j \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} -2-4-8j+18-4+8j \\ -2-4j+8-18+4j+8 \\ -2+4+8j+18+4-8j \\ -2+4j-8-18-8-4j \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 8 \\ -4 \\ 24 \\ -36 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 6 \\ -9 \end{bmatrix}$$

$$\therefore \quad y(n) = \{2, -1, 6, -9\}$$

This is the final Result we verify the result by performing direct circular convolution. We generate a circular matrix of $x_1(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & -2 & 1 & 2 \\ 2 & 1 & -2 & 1 \\ 1 & 2 & 1 & -2 \\ -2 & 1 & 2 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ -2 \\ 1 \\ -3 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 6 \\ -9 \end{bmatrix}$$

$$\therefore \quad y(n) = 2, -1, 6, -9\}$$

Which is the same as obtained using the DFT.

3.9.2 Matrix Method

In this method, we generate a circular matrix of $x_2(n)$ and multiply it by $x_1(n)$.

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ \vdots \\ y(N-1) \end{bmatrix} = \begin{bmatrix} x_2(0) & x_2(N-1) & x_2(N-2) & \cdots & x_2(1) \\ x_2(1) & x_2(0) & x_2(N-1) & \cdots & x_2(2) \\ x_2(2) & x_2(1) & x_2(0) & \cdots & x_2(3) \\ \vdots & \vdots & x_2(1) & \cdots & \vdots \\ x_2(N-2) & & \vdots & & \\ x_2(N-1) & x_2(N-2) & & & x_2(0) \end{bmatrix}\begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ \vdots \\ x_1(N-1) \end{bmatrix}$$

Note : Since the next column of the circular matrix will be the same as the first column we stop. The x_2 matrix is called a circular matrix.

3.9.2(A) Solved Example on Matrix Method

Ex. 3.9.6 : Perform circular convolution on the given two sequence $x_1(n) = \{1, 2, 3, 4\}$, $x_2(n) = \{4, 1, 1, 2\}$.

Soln. :

Soln. : We generate a circular matrix of $x_2(n)$ and multiply by $x_1(n)$.

$$y(n) = x_1(n) \circledast x_2(n)$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 4 & 2 & 1 & 1 \\ 1 & 4 & 2 & 1 \\ 1 & 1 & 4 & 2 \\ 2 & 1 & 1 & 4 \end{bmatrix}\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$\therefore$ We get $\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 15 \\ 19 \\ 19 \\ 23 \end{bmatrix}$

i.e., $y(n) = \{15, 19, 19, 23\}$

This is the same result as obtained using the concentric circle method.

Ex. 3.9.7 : Given the two sequences of length 4 are,

$x(n) = \{0,1,2,3\}$, $h(n) = \{2,1,1,2\}$

Find circular convolution

Soln. : The lengths of x (n) and h (n) are equal. We plot x (n) on the outer circle in counter clock wise direction, and h(n) on the inner circle in the clock wise direction. We multiply and add corresponding elements.

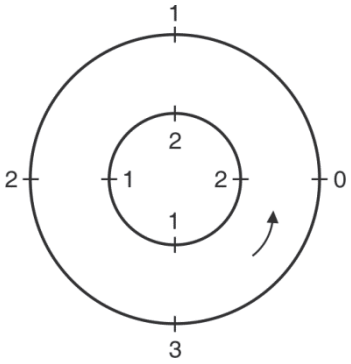


Fig. P. 3.9.7(a)

$\therefore \quad y(0) = (0 \times 2) + (1 \times 2) + (2 \times 1) + (3 \times 1) = 7$

We now rotate the inner circle and continue to multiply and add the corresponding elements. Hence we have

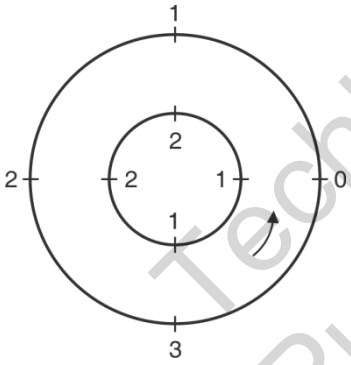


Fig. P. 3.9.7(b)

$\therefore \quad y(1) = (0 \times 1) + (1 \times 2) + (2 \times 2) + (3 \times 1) = 9$

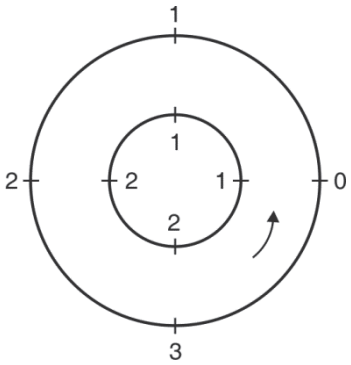


Fig. P. 3.9.7(c)

$\therefore \quad y(2) = (0 \times 1) + (1 \times 1) + (2 \times 2) + (3 \times 2) = 11$

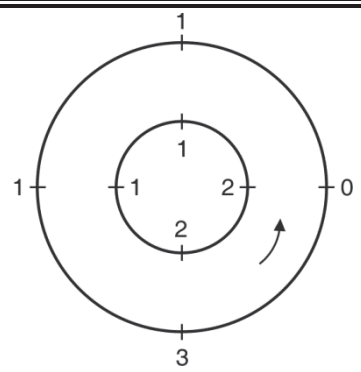


Fig. P. 3.9.7(d)

$\therefore \quad y(3) = (0 \times 2) + (1 \times 1) + (2 \times 1) + (3 \times 2) = 9$

If we rotate one more time, we come back to the starting position. Hence we stop the process.

$\therefore \quad y(n) = \{7, 9, 11, 9\}$

Ex. 3.9.8 : Find circular convolution for the following sequence using graphical method.

(i) $x_1(n) = \delta(n) + \delta(n-1) + \delta(n-2)$

$x_2(n) = 2\delta(n) - \delta(n-1) + 2\delta(n-2)$

(ii) $x_1(n) = \delta(n) + \delta(n-2) - \delta(n-2) + \delta(n-3)$

$x_2(n) = \delta(n) - \delta(n-2) + \delta(n-4)$

Soln. :

(i) $x_1(n)$ and $x_2(n)$ can be written is the form

$x_1(n) = \{1, 1, 1\}$ and $x_2(n) = \{2, -1, 2\}$

$\uparrow$ $\uparrow$

Since both $x_1(n)$ and $x_2(n)$ are of equal lengths, we place $x_1(n)$ on the outer circle in the counter clockwise direction and $x_2(n)$ on the inner circular in the clockwise direction.

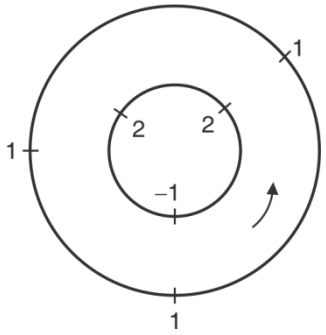


Fig. P. 3.9.8(a)

$y(0) = (1 \times 2) + (1 \times 2) + (1 \times (-1)) = 3$

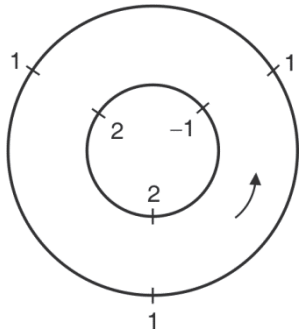


Fig. P. 3.9.8(b)

$y(1) = (1 \times (-1)) + (1 \times 2) + (1 \times 2) = 3$

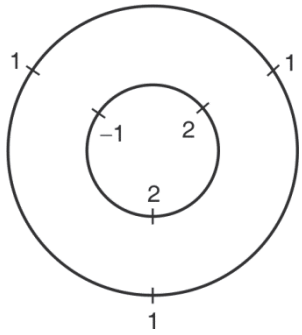


Fig. P. 3.9.8(c)

$$y(2) = (1 \times 2) + (1 \times (-1)) + 1\,(2) = 3$$

If we now rotate the inner circle, we come back to the starting point. Hence we stop the process.

$$\therefore \quad y(n) = \{3, 3, 3\}$$

(ii) $x_1(n)$ and $x_2(n)$ can be written is the form

$$x_1(n) = \{1, 1, -1, -1\}$$

and $$x_2(n) = \{1, 0, 1, 0, 1\}$$

We append a zero to $x_1(n)$ to make the lengths of $x_1(n)$ and $x_2(n)$ equal.

$$\therefore \quad x_1(n) = \{1, 1, -1, -1, 0\} \text{ and}$$

$$x_2(n) = \{1, 0, 1, 0, 1\}$$

We place $x_1(n)$ on the outer circle, and $x_2(n)$ on the inner circle

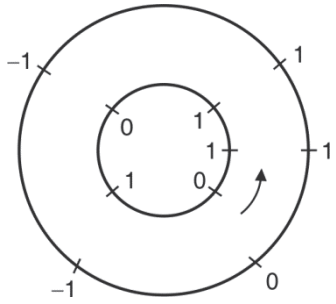


Fig. P. 3.9.8(d)

$y(0) = (1 \times 1) + (1 \times 1) + (-1 \times 0) + (-1 \times 1) + (0 \times 0) = 1$

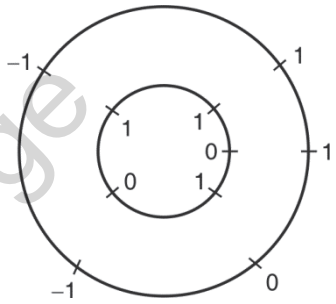


Fig. P. 3.9.8(e)

$y(1) = (1 \times 0) + (1 \times 1) + (-1 \times 1) + (-1 \times 0) + (0 \times 1) = 0$

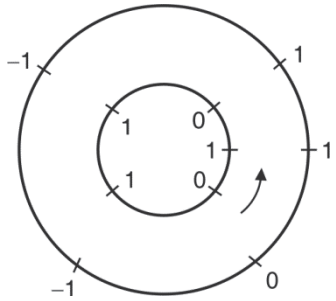


Fig. P. 3.9.8(f)

$y(2) = (1 \times 1) + (1 \times 0) + (-1 \times 1) + (-1 \times 1) + (0 \times 0) = -1$

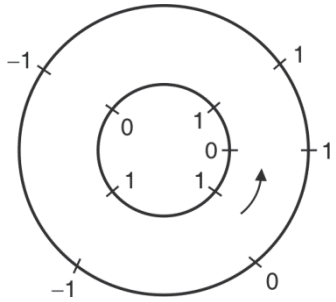


Fig. P. 3.9.8(g)

$y(3) = (1 \times 0) + (1 \times 1) + (-1 \times 0) + (-1 \times 1) + (0 \times 1) = 0$

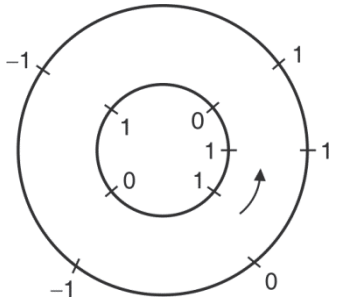


Fig. P. 3.9.8(h)

$\therefore y(4) = (1 \times 1) + (1 \times 0) + (-1 \times 1) + (-1 \times 0) + (0 \times 1) = 0$

$\therefore \quad y(n) = \{1, 0, -1, 0, 0\}$

Note : Unlike Linear convolution, where the length of the output signal is $L_1 + L_2 - 1$, the length of the output signal in circular convolution is max (L_1, L_2)

3.9.3 Getting Linear Convolution from Circular Convolution

- We can get the linear convolution result using circular convolution by making a small change to the method described in the earlier example.
- The main difference between linear and circular convolution is the length.
- Given two sequences $x_1(n)$ and $x_2(n)$ of lengths L_1 and L_2 respectively, the length of y(n) using linear convolution would be $L_1 + L_2 - 1$ while the length of the result of circular convolution will be max(L_1, L_2).
- What we do here is append zeros to both $x_1(n)$ and $x_2(n)$ so that the lengths of both the sequences are $L_1 + L_2 - 1$. After this if we perform circular convolution, we get the result of linear convolution.
- Let us take an example to prove the above statement.

3.9.4 Solved Examples

Ex. 3.9.9 : Perform circular convolution on the following sequences so that the result is the same as that of Linear convolution

$x_1(n) = \{1, 2, 3, 1\}$, $x_2(n) = \{1, 1, 1\}$

Soln. : Here the length of $x_1(n) = 4$ and length of $x_2(n) = 3$. Linear convolution would give us a result of length $L_1 + L_2 - 1$ i.e., $4 + 3 - 1 = 6$

We now append zeros to both $x_1(n)$ and $x_2(n)$ so that they are both of length 6.

$$\therefore \quad x_1(n) = \{1, 2, 3, 1, 0, 0\}$$

$$x_2(n) = \{1, 1, 1, 0, 0, 0\}$$

Note : In this case we need to append two zeros to $x_1(n)$ and three zeros to $x_2(n)$.

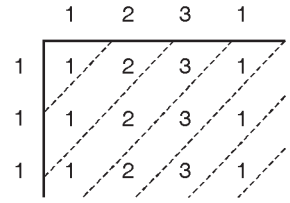
We now perform circular convolution. We shall use the matrix method for its simplicity.

$$y(n) = x_1(n) \circledast x_2(n)$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore \quad y(n) = \{1, 3, 6, 6, 4, 1\}$$

We now check if this is same as what we would have got had we performed linear convolution. We shall again use the matrix method $y(n) = x_1(n) * x_2(n)$



$$\therefore \quad y(n) = \{1, 3, 6, 6, 4, 1\}$$

Hence we see that the two results are the same.

Go ahead try out your own sums. This technique will always work. Given below is a simple program to implement circular convolution.

CIRCULAR CONVOLUTION

```
Clear all
clc

x= [1 2 6 4 7 3 4];
h= [4 3 2 2];
N2= length (x);
N3= length (h);
N = max (N2,N3);
%% Append Zeros to the shorter signal and make the two
equal
if  N2< N
        x=[x zeros(1,N–N2) ];
        else
        h=[h zeros (1,N–N3) ];
end
b=h′ ; c=b ;
for n=1:1:N – 1
```

```
    b=[b(N);b(1:N–1,:) ];
    c=[c b];
end
CIR_CONV=c*x′
```

Ex . 3.9.10 : Compute circular convolution of

$x_1(n) = \{1,1,2,2\}$

$x_2(n) = \{1, 2, 3, 4\}$

Soln. :

$$y(n) = x_1(n) \circledast x_2(n)$$

We generate a circular matrix of $x_2(n)$

$$\therefore \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & 4 & 3 & 2 \\ 2 & 1 & 4 & 3 \\ 3 & 2 & 1 & 4 \\ 4 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 15 \\ 17 \\ 15 \\ 13 \end{bmatrix}$$

$$\therefore \quad y(n) = \{15, 17, 15, 13\}$$

Ex. 3.9.11: Compute DFT of $X(n) = 1, \ 0 \le n \le 2 = 0$ Otherwise N = 4, Find $|X(k)|$ and $\angle X(k)$

Soln. : Since N = 4, we have

$$x(n) = \{1,1,1,0\}$$
$$\uparrow$$

A 4 – point DFT in matrix form is given by the equation

$$X(k) = [W_4]\, x(n)$$

$$X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$\therefore \ X(k) = \{3, -j, 1, j\}$$

Now,

Magnitude $|X(k)| = \{3,1,1,1\}$

$$\text{Phase } \angle X(k) = \tan^{-1}\left\{\frac{\text{Imaginary}}{\text{Real}}\right\}$$

$$\therefore \quad \angle X(k) = \{0, -90°, 0, 90°\}$$

Ex. 3.9.12 : Use the four point DFT and IDFT to determine the circular convolution of the given two sequences.

$x_1(n) = \{1, 2, 3, 1\}$

$x_2(n) = \{4, 3, 2, 2\}$

Soln. : We use the **circular convolution** property

$$\text{i.e. } x_1(n) \circledast x_2(n) \xleftrightarrow{\text{DFT}} X_1(k) \cdot X_2(k)$$

We find the DFT's of $x_1(n)$ and $x_2(n)$

$$X_1 = [W_N] \cdot x_1$$

$$\begin{bmatrix} X_1(0) \\ X_1(1) \\ X_1(2) \\ X_1(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$$\therefore X_1(k) = \{7, -2-j, 1, -2+j\}$$

Similarly

$$X_2 = [W_N] \cdot x_2$$

$$\begin{bmatrix} X_2(0) \\ X_2(1) \\ X_2(2) \\ X_2(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 2 \end{bmatrix}$$

$$\therefore \quad X_2(k) = \{11, 2-j, 1, 2+j\}$$

$$X_3 = X_1(k) \cdot X_2(k) = \{77, -5, 1, -5\}$$

Since this is in the Fourier domain, we finally calculate the IDFT.

$$X_3(k) = \frac{1}{N} [W_N]^* \cdot X_3$$

$$\begin{bmatrix} x_3(0) \\ x_3(1) \\ x_3(2) \\ x_3(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 77 \\ -5 \\ 1 \\ -5 \end{bmatrix}$$

$$\therefore \quad x_3 = \{17, 19, 22, 19\}$$

This is the result that we would obtain if we circularly convolved $x_1(n)$ and $x_2(n)$

$$\therefore \quad x_1(n) \circledast x_2(n) = \{17, 19, 22, 19\}$$

Let us check our result by performing circular convolution of $x_1(n)$ and $x_2(n)$

$$\therefore \quad x_3(n) = x_1(n) \circledast x_2(n)$$

$$\begin{bmatrix} x_3(0) \\ x_3(1) \\ x_3(2) \\ x_3(3) \end{bmatrix} = \begin{bmatrix} 4 & 2 & 2 & 3 \\ 3 & 4 & 2 & 2 \\ 2 & 3 & 4 & 2 \\ 2 & 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$\therefore \quad x_3(n) = \{17, 19, 22, 19\}$

Hence we get the same result as what we got earlier.

Ex. 3.9.13 : Compute circular convolution of $x_1(n) = \{1,2,3,4\}$ and $x_2(n) = \{2,1,2,1\}$

Soln. :

$$x_1(n) = \{1,2,3,4\}$$

$$x_2(n) = \{2,1,2,1\}$$

$$y(n) = x_1(n) \circledast x_2(n)$$

OR $$y(n) = x_1(n) \textcircled{N} x_2(n)$$

We generate a circular matrix of $x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 14 \\ 16 \\ 14 \\ 16 \end{bmatrix}$$

$\therefore \quad y(n) = \{14, 16, 14, 16\}$

Ex. 3.9.14 : Perform circular convolution of

$x_1(n) = \{2, 1, 2, 1\}$

$x_2(n) = \{4, 3, 2, 1\}$

Soln. : We generate a circular matrix or $x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 4 & 1 & 2 & 3 \\ 3 & 4 & 1 & 2 \\ 2 & 3 & 4 & 1 \\ 1 & 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 16 \\ 14 \\ 16 \\ 14 \end{bmatrix}$$

$\therefore \quad y(n) = \{16,14,16,14\}$

Ex. 3.9.15 : Find linear and circular convolution of the following sequences

$x_1[n] = [4, -2, 2, 1]$

$x_2[n] = [1, 2, 3]$

Soln. :

Given :

(i) Linear convolution

$$x_1(n) = [4, -2, 2, 1]$$

$$x_2(n) = [1, 2, 3]$$

Number of sample in $x_1(n)$ is $L_1 = 4$

Number of sample in $x_2(n)$ is $L_2 = 3$

So from linear convolution number of samples will be,

$$L_2 + L_2 - 1 = 6$$

$$x(n) = x_1(n) * x_2(n)$$

We use the tabular method,

We assume the first value as the origin,

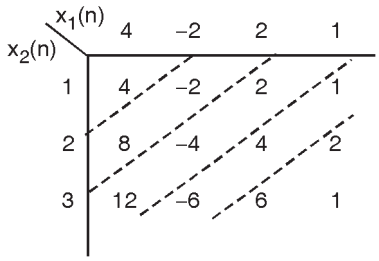


Fig. P. 3.9.15

$\therefore \quad y(n) = \{4, 6, 10, -1, 8, 1\}$

(ii) Circular Convolution

For circular convolution, the length of both the signals should be equal.

We append one zero to $x_2(n)$

$$\therefore \quad x_1(n) = \{4, -2, 2, 1\}$$

$$x_2(n) = \{1, 2, 3, 0\}$$

We generate a circular matrix of $x_2(n)$

$$y_1(n) = x_1(n) \circledast x_2(n)$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ -2 \\ 2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 12 \\ 9 \\ 10 \\ -1 \end{bmatrix}$$

$$\therefore \quad y(n) = \{12, 9, 10, 1\}$$

Ex. 3.9.16 : Find circular convolution of two finite duration signals

$$x_1(n) = \{1, -1, -2, 3, -1\}$$

$$x_2(n) = \{1, 2, 3\}$$

Soln. : For circular convolution, length of both signals should be equal. We append two zeros to $x_2(n)$.

$$\therefore \quad x_1(n) = \{1, -1, -2, 3, -1\} \text{ and}$$

$$x_2(n) = \{1, 2, 3, 0, 0\}$$

Now,

$$y(n) = x_1(n) \circledast x_2(n)$$

We generate a circular matrix of $x_2(n)$.

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 3 & 2 \\ 2 & 1 & 0 & 0 & 3 \\ 3 & 2 & 1 & 0 & 0 \\ 0 & 3 & 2 & 1 & 0 \\ 0 & 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \end{bmatrix} = \begin{bmatrix} 8 \\ -2 \\ -1 \\ -4 \\ -1 \end{bmatrix}$$

$$\therefore \quad y(n) = \{8, -2, -1, -4, -1\}$$

Ex. 3.9.17 : Use the four point DFT and IDFT to determine the circular convolution of sequences

$x_1(n) = (1, 2, 3, 1)$ $\quad$ $x_2(n) = (4, 3, 2, 2)$

Soln. :

Step I : The four point DFT of $x_1(n)$ is $X_1(k)$ and it is given by,

$$X_1(k) = [W_4]\, x_1$$

We have,

$$[W_4] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$\therefore \quad X_1(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 7 \\ -2-j \\ 1 \\ -2+j \end{bmatrix}$$

$$\therefore \quad X_1(k) = \{7, -2-j, 1, -2+j\} \qquad \ldots(1)$$

Similarly,

$$X_2(k) = [W_4]\, x_{2N}$$

$$\therefore \quad X_2(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 2 \end{bmatrix}$$

$$\therefore \quad X_2(k) = \begin{bmatrix} 4+3+2+2 \\ 4-3j-2+2j \\ 4-3+2-2 \\ 4+3j-2-2j \end{bmatrix} \begin{bmatrix} 11 \\ 2-j \\ 1 \\ 2+j \end{bmatrix}$$

$$\therefore \quad X_2(k) = \{11, 2-j, 1, 2+j\}$$

Step II : Now according to property of circular convolution,

$$x_1(n) \circledast x_2(n) = X_1(k) \cdot X_2(k) = Y(k)$$

$$\therefore \quad Y(k) = \{7, -2-j, 1, -2+j\} \cdot \{11, +2-j, 1, 2+j\}$$

$$\therefore \quad Y(k) = \{77, -5, 1, -5\}$$

Step III : Let the result of $x_1(n) \circledast x_2(n)$ be sequence y(n). It is obtained by computing IDFT of Y(k).

According to the definition of IDFT we have,

$$y(n) = \frac{1}{N} [W_N^*] \cdot X_N$$

$$= \frac{1}{4}\,[\,W_4^{*}\,]\cdot Y_N$$

$$\therefore\ y(n) = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & +j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & +j \end{bmatrix}\begin{bmatrix} 77 \\ -5 \\ 1 \\ -5 \end{bmatrix}$$

$$y(n) = \frac{1}{4}\begin{bmatrix} 68 \\ 76 \\ 88 \\ 76 \end{bmatrix} = \begin{bmatrix} 17 \\ 19 \\ 22 \\ 19 \end{bmatrix}$$

$$y(n) = \{17, 19, 22, 19\}$$

Ex. 3.9.18 : Find the response of FIR filter with impulse response h(n) = {1, 2, 4} to the input sequence x (n) = {1, 2} using circular convolution.

Soln. :

Step I :

$$x(n) = \{1, 2\},\ h(n) = \{1, 2, 4\}$$

Here $L_1 = 2$ and $L_2 = 3$

Hence the length of linear convolution result will be

$$L_1 + L_2 = -1 = 2 + 3 - 1 = 4$$

We append zero to make the length of x(n) and h (n) = 4

$$\therefore\ x(n) = \{1,2,0,0\} \text{ and } h(n) = \{1,2,4,0\}$$

We compute the DFT of x(n) and h (n) and use the property,

$$x(n) \circledast h(n) \xleftrightarrow{\text{DFT}} X(k)\cdot H(k)$$

$$\therefore\ X(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}\begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1+2 \\ 1-2j \\ 1-2 \\ 1+2j \end{bmatrix}$$

$$\therefore\ X(k) = \{3, 1-2j, 3-3+2j\}$$

$$\text{And}\quad H(k) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}\begin{bmatrix} 1 \\ 2 \\ 4 \\ 0 \end{bmatrix}$$

$$\therefore\ H(k) = \{7, -3-2j, 3, -3+2j\}$$

$$\text{Now } Y(k) = X(k)\,.\,H(k)$$

We perform element by element multiplication

$$\therefore\ Y(k) = \{21, -7+4j, -3, -7-4j\}$$

We finally compute the IDFT of y (k),

$$\text{Now}\quad y(n) = \frac{1}{N}[W_4^{*}]\,Y_N$$

$$\therefore\ y(n) = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & -j \end{bmatrix}\begin{bmatrix} 21 \\ -7+4j \\ -3 \\ -7-4j \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 21 -7+4j-3-7-4j \\ 21-7j+4j^2+3+7j+4j^2 \\ 21+7-4j-3+7+4j \\ 21+7j-4j^2+3-7j-4j^2 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 4 \\ 16 \\ 32 \\ 32 \end{bmatrix}$$

$$\therefore\ y(n) = \{1, 4, 8, 8\}$$

$$\uparrow$$

This is the final response of the FIR filter.

Verify.

Let us, Verify the result by performing linear convolution using tabular method

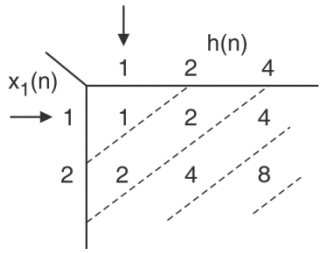


Fig. P. 3.9.18

$$y(n) = \{1, 4, 8, 8\}$$

Hence we get the same result.

Ex. 3.9.19 : If x(n) = {1, 2, 3, 2}, h (n) = {1, 0, 2, 0}

(i) Find circular convolution using time domain method.

(ii) Find linear convolution using circular convolution.

Soln. :

(i) We will solve it using the graphical method and then test our result using the matrix method.

$$y(n) = x_1(n) \circledast x_2(n)$$

Since the length's of both x_1 (n) and x_2 (n) are the same, we do not need to append zeros to either of the signals.

We take two concentric circles. Steps involved are as follows :

(a) Plot x_1 (n) on outer circle in counter clockwise direction.

(b) Plot x_2 (n) on inner circle in clockwise direction.

(c) Ensure the origins of the two signals are aligned. Multiply and add corresponding elements.

(d) Move inner circle in the clockwise direction and repeat step (c).

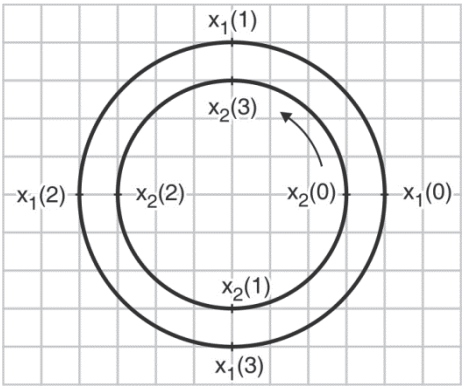


Fig. P. 3.9.19

Hence we have,

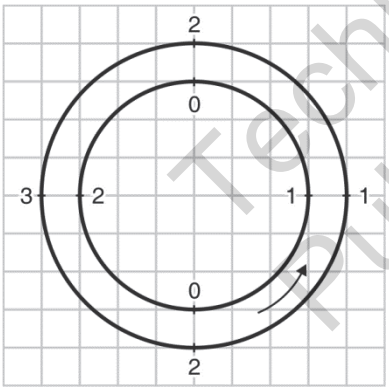


Fig. P. 3.9.19(a)

$\therefore$ y (0) $=(1 \times 1) + (2 \times 0) + (3 \times 2) + (2 \times 0) = 7$

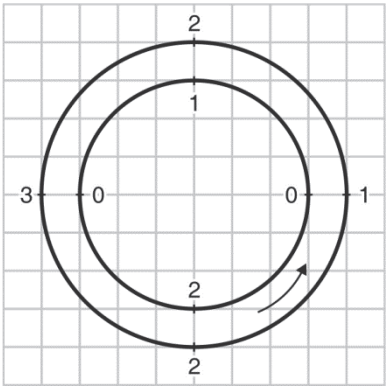


Fig. P. 3.9.19(b)

$\therefore$ y (0) $=(1 \times 0) + (2 \times 1) + (3 \times 0) + (2 \times 2) = 6$

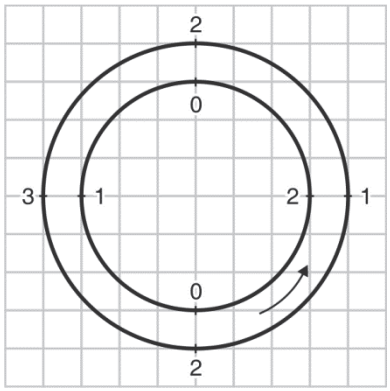


Fig. P. 3.9.19(c)

$\therefore$ y (0) $=(1 \times 2) + (2 \times 0) + (3 \times 1) + (2 \times 0) = 5$

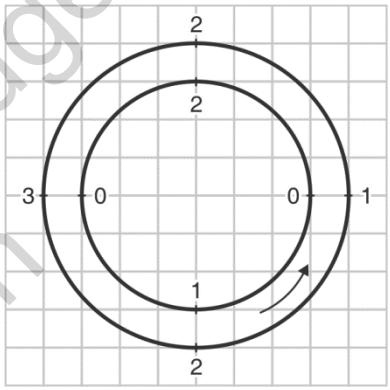


Fig. P. 3.9.19(d)

$\therefore$ y (0) $=(1 \times 0) + (2 \times 2) + (3 \times 0) + (2 \times 1) = 6$

If we now rotate the inner circle, we come back to the starting position. Hence we stop the process.

$\therefore$ y(n) = { 7, 6, 5, 6 }

$\therefore$ x_1 (n) $\circledast$ x_2 (n) = { 7, 6, 5, 6 }

We verify this result using the matrix method. We form a circular matrix of x_2 (n).

$$\therefore \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} x_2(0) & x_2(3) & x_2(2) & x_2(1) \\ x_2(1) & x_2(0) & x_2(3) & x_2(2) \\ x_2(2) & x_2(1) & x_2(0) & x_2(3) \\ x_2(3) & x_2(2) & x_2(1) & x_2(0) \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ x_1(3) \end{bmatrix}$$

$$\therefore \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 2 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 7 \\ 6 \\ 5 \\ 6 \end{bmatrix}$$

$\therefore$ y (n) = { 7, 6, 5, 6 }

which is the same as obtained using the graphical method.

(ii) Find linear convolution using circular convolution. Since length of x_1 (n) = 4 and length of x_2 (n) = 4, linear convolution would give us a result of length

$L_1 + L_2 - 1 = 4 + 4 - 1 = 7$

We now append zeroes to both x_1 (n) and x_2 (n). So that they are both of length 7.

$\therefore \quad x_1(n) = \{1, 2, 3, 2, 0, 0, 0\}$

$x_2(n) = \{1, 0, 2, 0, 0, 0, 0\}$

$y(n) = x_1(n) \circledast x_2(n)$

We will now perform circular convolution using the matrix method.

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \\ y(6) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 2 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\therefore \quad y(n) = \{1, 2, 5, 6, 6, 4, 0\}$

We now verify this result by performing regular convolution.

$y(n) = x_1(n) * x_2(n)$

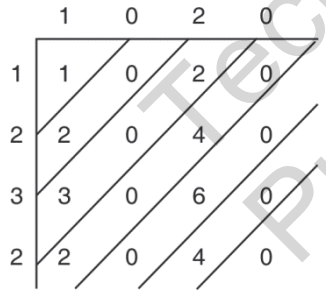


Fig. P. 3.9.19(e)

$\therefore \quad y(n) = \{1, 2, 5, 6, 6, 4, 0\}$

Hence, we observe that we get the same result using linear convolution.

3.10 Filtering of Long Duration Signals

- Filtering of a signal is required for various reasons like removal of noise, boosting certain frequencies while attenuating the rest, to name a few.
- Linear filtering is basically linear convolution that we have studied earlier.
- Hence if x(n) is our signal (song, ecg, etc) which has been corrupted by noise, and h (n) is the impulse response (filter), then the filtered output signal y (n) is given by the linear convolution operation.

$$y(n) = x(n) * h(n)$$

i.e. $$y(n) = \sum_{k=-\infty}^{+\infty} x(k)\ h(n-k)$$

- In the real world, due to high sampling rates, x(n) is very long and usually runs into thousands of samples. Because of its length, it is difficult to perform regular linear convolution at one time.
- The way devised to deal with this issue is to break x(n) into smaller blocks of size N. Computation on each of these blocks is then done separately. Each of these processed blocks are then, appended (connected) to each other to obtain the final filtered output y (n).
- There are two methods of performing this kind of filtering

1. Overlap - save method.
2. Overlap - add method.

We shall discuss both these methods in detail.

3.10.1 Overlap - Save Method

As mentioned earlier, we break x(n) into smaller blocks of length N. (x_1 (n), x_2 (n), …). It is important to note that while the length N can be our choice, it should be longer than the length of h (n).

i.e. $N >$ length h (n).

We use the following notations.

Where, N = Length of each block

M = Length of h (n)

L = Values from current x(n)

$\therefore \quad N = (M - 1) + L$

- Each block x_i (n) of length N is formed such that it contains (M – 1) values from the previous block and the remaining L values from the current x(n).
- Since there is an overlap of (M – 1) values from the previous block, it known as overlap- save method.
- As the 1st block does not have a previous block, we append (M – 1) zeros to it at the beginning. A pictorial representation would help us understand this better.

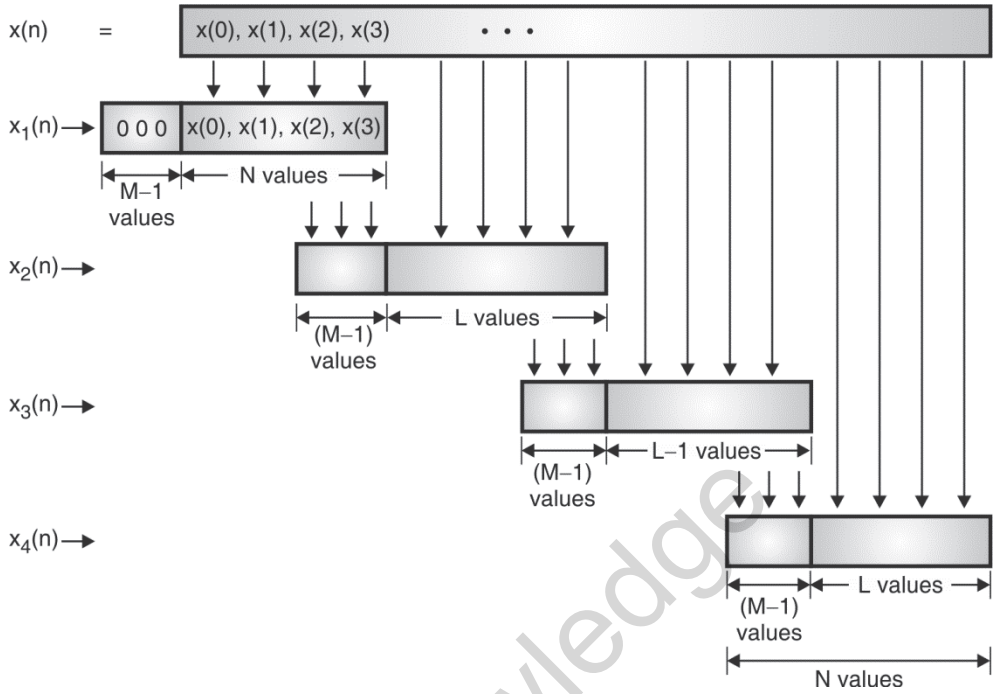


Fig. 3.10.1

Once these block are created, each of these are "circularly convolved" with h (n).

$$\text{i.e.} \quad y_1(n) = x_1(n) \circledast h(n)$$

$$y_2(n) = x_2(n) \circledast h(n)$$

$$\vdots \qquad \vdots \text{ etc.}$$

The final filtered output y (n) is obtained by discarding the first (M – 1) values of each result. $y_1(n)$, $y_2(n)$, … and appending the remaining values. A pictorial representation is shown in Fig. 3.10.2.

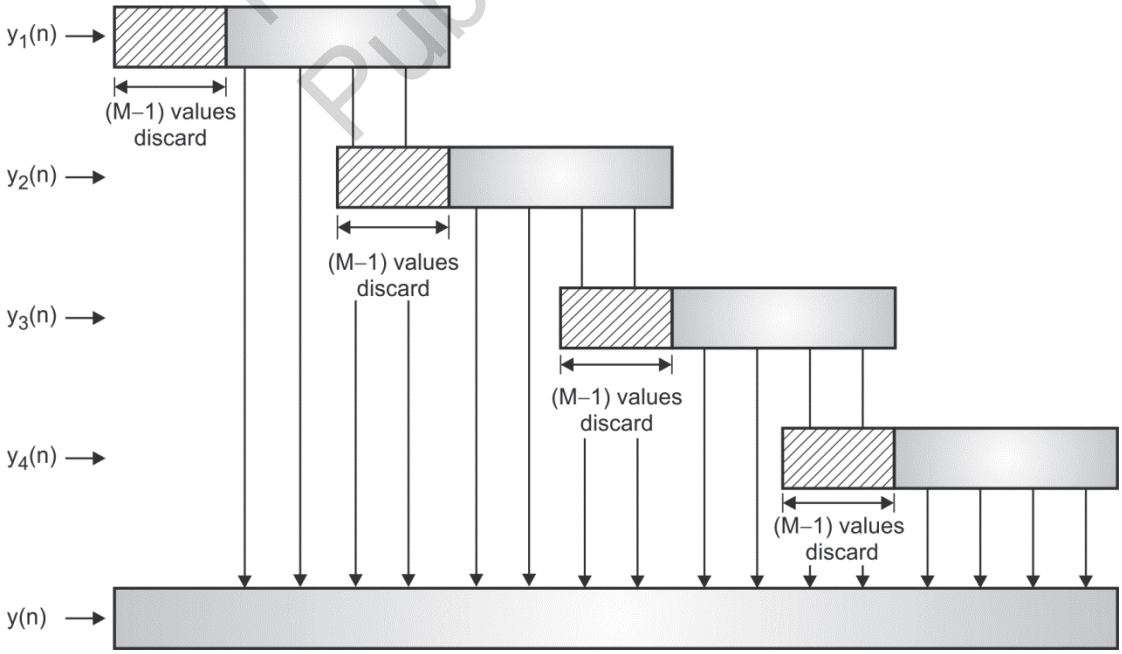


Fig. 3.10.2

This y (n) is the same as what we would obtain by performing linear convolution

i.e. $y(n) = x(n) \circledast h(n)$

It is important to note that we obtained y (n) by performing circular convolution of x(n) with individual blocks.

Let us solve an example to understand the method explained.

Solved Example

Ex. 3.10.1 : Let $x(n) = \{1, 2, 3, 4, 5, 6, 7\}$ and $h(n) = \{1, 0, 2\}$

Perform convolution using overlap save method.

Soln. : We choose a block size of N. In this method, each block contains (M – 1) data points of the previous block followed by L new data points. For the first block, the first (M – 1) values are set to zero. In our case M – 1 = 2. We choose N = 5 and L = 3.

$$x(n) = \{1, 2, 3, 4, 5, 6, 7\}$$

$\therefore x_1(n) = \{0, 0, 1, 2, 3\}$

$x_2(n) = \{2, 3, 4, 5, 6\}$

$x_3(n) = \{5, 6, 7, 0, 0\}$

Now $h(n) = \{1, 0, 2\}$

Since the block size = 5, we append 2 zeros to h(n)

$$\therefore \quad h(n) = \{1, 0, 2, 0, 0\}$$

We now perform circular convolution.

$$\therefore \quad y_1(n) = x_1(n) \circledast h(n) = \{4, 6, 1, 2, 5\}$$

$$\therefore \quad y_2(n) = x_2(n) \circledast h(n) = \{12, 15, 8, 11, 14\}$$

$$\therefore \quad y_3(n) = x_3(n) \circledast h(n) = \{5, 6, 17, 12, 14\}$$

We arrange the results with an overlap of (M – 1) values as shown and discard the first (M – 1) values of each result. In short, we discard the first (M – 1) = 2 values of each result.

4 6 1 2 5

Discard

12 15 8 11 14

Discard

5 6 17 12 14

Discard

$\therefore y(n) = \{1, 2, 5, 8, 11, 14, 17, 12, 14\}$

We check the result by performing linear convolution

$$y(n) = x(n) \circledast h(n)$$

	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
2	2	4	6	8	10	12	14

$y(n) = \{1, 2, 5, 8, 11, 14, 17, 12, 14\}$

3.10.2 Overlap - Add Method

We use the same notations used in overlap–save method. The overlap - add method is similar to the overlap–save method except the way the block are created.

In this method, each block is created by taking L values from x(n) and appending (M – 1) zeros at the end of each block.

The size of each block is N = L + (M – 1).

A pictorial representation would help us understand this better.

Once these blocks are created, each of these are "Circularly convolved" with h (n), just like overlap–save method.

i.e. $y_1(n) = x_1(n) \circledast h(n)$

$y_2(n) = x_2(n) \circledast h(n)$

The final filtered output y (n), is obtained by overlapping and adding the last (M–1) values of result $y_1(n)$, $y_2(n)$, ...etc.

A pictorial representation is shown in Fig. 3.10.3.

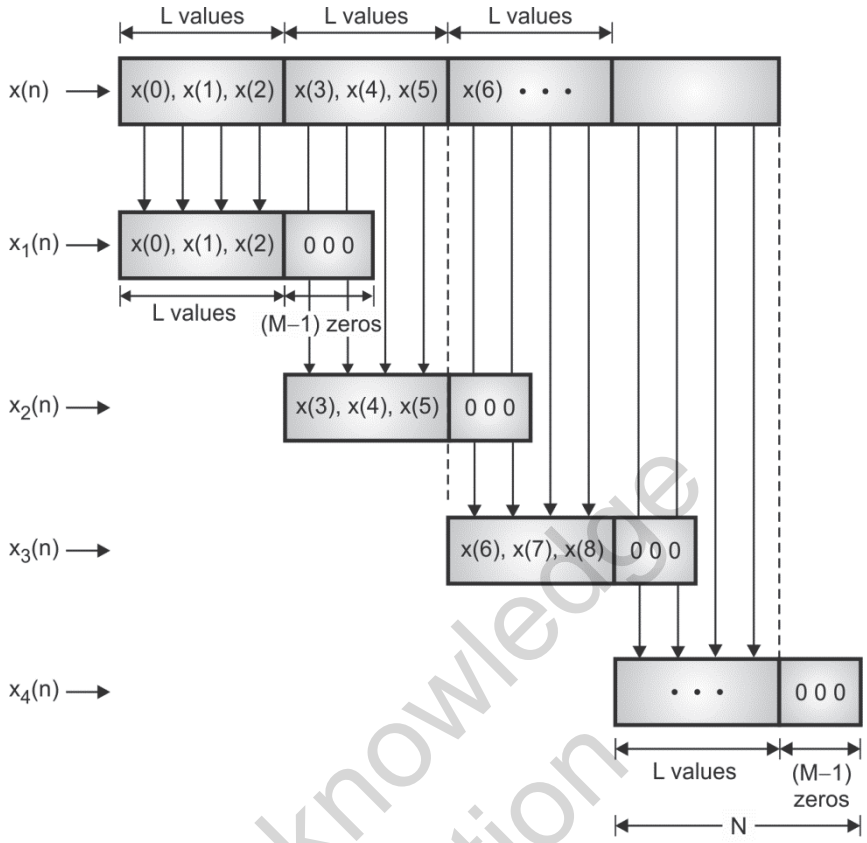


Fig. 3.10.3

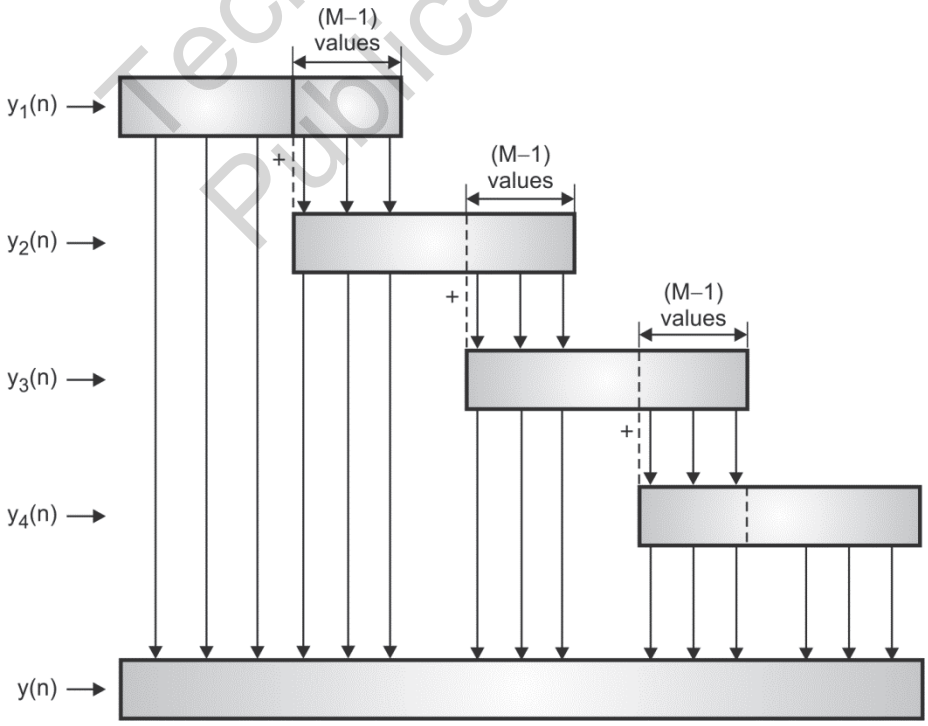


Fig. 3.10.4

3.10.3 Solved Examples on Method

Let us solve an example to understand the method just discussed.

Ex. 3.10.2 : Let x(n) = {1, 2, 3, 4, 5, 6, 7}

and h (n) = {1, 0, 2}

Perform convolution using overlap add method.

Soln. : Overlap–add and overlap save methods are used to perform convolution on long sequences. In both these methods, the input is divided into blocks. Circular convolution is then performed on each of the blocks.

Here, we assume that the length of input, x(n) is L and the length of the impulse response, h (n) = M.

Overlap - add method

In this, the input is divided into blocks of data of size L and (M – 1) zeros are appended to it. This makes the size of data blocks N = L + (M – 1).

In our example we choose a block size of N = 5.

Since h (n) has a length of 3, we append M – 1 = (3 –1) = 2 zeros to L. The length of L = 3.

$\therefore$ $x(n) = \{1, 2, 3, 4, 5, 6, 7\}$

$x_1(n) = \{1, 2, 3, 0, 0\}$

$x_2(n) = \{4, 5, 6, 0, 0\}$

$x_3(n) = \{7, 0, 0, 0, 0\}$

We increase the size of h (n) by appending zeros so that it is equal to N = 5.

$\therefore$ $h(n) = \{1, 0, 2, 0, 0\}$

We now perform circular convolution

$\therefore$ $y_1(n) = x_1(n) \circledast h(n) = \{1, 2, 5, 4, 6\}$

$\therefore$ $y_2(n) = x_1(n) \circledast h(n) = \{4, 5, 14, 10, 12\}$

$\therefore$ $y_3(n) = x_3(n) \circledast h(n) = \{7, 0, 14, 0, 0\}$

These results are placed as shown below and added. The last two terms are added. Since we had appended 2 zeros to h (n).

y = 1 2 5 4 6

↓ ↓ Add

4 5 14 10 12

↓ ↓ Add

7 0 14 0 0

$\therefore$ $y(n) = \{1, 2, 5, 8, 11, 14, 17, 12, 14\}$

We would have got the same answer had we performed linear convolution as shown earlier.

Ex. 3.10.3 : Find linear convolution using overlap-add and overlap-save method of the following sequences :

x[n] = {1, 2, – 1, 2, 3, – 2, – 3, – 1, 1, 1, 2, – 1};

h[n] = {1, 2, 3}

Compare the results and state the usage of each method.

Soln. :

Overlap save method

We will form blocks of length 4.

Here N = 4

Length of h = M = 3

$\therefore$ M – 1 = 2

Now to make each block we need L = 2 data from x (n). The different blocks are formed as follows :

For the 1st block, we append M – 1 = 2 zeros

$x_1(n) = \{0, 0, 1, 2\}$

$\therefore x_1(n) = \{0, 0, 1, 2\}$

M–1 zeros; 2 data from x(n)

$x_2(n) = \{1, 2, -1, 2\}$

M–1 from x(n); New data

Similarly other blocks can be formed,

$x_3(n) = \{-1, 2, 3, -2\}$

$x_4(n) = \{3, -2, -3, -1\}$

$x_5(n) = \{-3, -1, 1, 1\}$

$x_6(n) = \{1, 1, 2, -1\}$

$x_7(n) = \{2, -1, 0, 0\}$

Given $h(n) = \{1, 2, 3\}$

We append zeros to h(n) so that it is equal to N = 4

$\therefore$ $h(n) = \{1, 2, 3, 0\}$

The different output blocks are calculated as follows :

$y_1(n) = x_1(n) \circledast h(n)$

We create a circular of h(n)

$$\therefore \quad y_1(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} 0+0+3+4 \\ 0+0+0+6 \\ 0+0+1+0 \\ 0+0+2+2 \end{bmatrix} = \begin{bmatrix} 7 \\ 6 \\ 1 \\ 4 \end{bmatrix}$$

$$\therefore \quad y_1(n) = \{7, 6, 1, 4\} \quad ....(1)$$

$$y_2(n) = x_2(n) \circledast h(n)$$

$$\therefore \quad y_2(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ -1 \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} 1+0-3+4 \\ 2+2+0+6 \\ 3+4-1+0 \\ 0+6-2+2 \end{bmatrix} = \begin{bmatrix} 2 \\ 10 \\ 6 \\ 6 \end{bmatrix}$$

$$\therefore \quad y_2(n) = \{2, 10, 6, 6\} \quad ...(2)$$

$$y_3(n) = x_3(n) \circledast h(n)$$

$$\therefore \quad y_3(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ 3 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} -1+0+9-4 \\ -2+2+0-6 \\ -3+4+3+0 \\ 0+6+6-2 \end{bmatrix} = \begin{bmatrix} 4 \\ -6 \\ 4 \\ 10 \end{bmatrix}$$

$$\therefore \quad y_3(n) = \{4, -6, 4, 10\} \quad ...(3)$$

$$y_4(n) = x_4(n) \circledast h(n)$$

$$\therefore \quad y_4(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ -2 \\ -3 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 3+0-9-2 \\ 6-2+0-3 \\ 9-4-3+0 \\ 0-6-6-1 \end{bmatrix} = \begin{bmatrix} -8 \\ 1 \\ 2 \\ -13 \end{bmatrix}$$

$$\therefore \quad y_4(n) = \{-8, 1, 2, -13\} \quad ...(4)$$

$$y_5(n) = x_5(n) \circledast h(n)$$

$$\therefore \quad y_5(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} -3 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -3+0+3+2 \\ -6-1+0+3 \\ -9-2+1+0 \\ 0-3+2+1 \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \\ -10 \\ 0 \end{bmatrix}$$

$$\therefore \quad y_5(n) = \{2, -4, -10, 0\} \quad ...(5)$$

$$y_6(n) = x_6(n) \circledast h(n)$$

$$\therefore \quad y_6(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+0+6-2 \\ 2+1+0-3 \\ 3+2+2+0 \\ 0+3+4-1 \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \\ 7 \\ 6 \end{bmatrix}$$

$$\therefore \quad y_6(n) = \{5, 0, 7, 6\} \quad ...(6)$$

$$y_7(n) = x_7(n) \circledast h(n)$$

$$\therefore \quad y_7(n) = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 0 \\ 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ -1 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 2+0+0+0 \\ 4-2+0+0 \\ 6-2+0+0 \\ 0-3+0+0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 4 \\ -3 \end{bmatrix}$$

$$\therefore \quad y_7(n) = \{2, 2, 4, -3\} \quad ...(7)$$

The final output y (n) is obtained by discarding initial 2 samples from each output block and then by connecting all output blocks one after other.

$$\therefore \quad y(n) = \{1, 4, 6, 6, 4, 10, 2, -13, -1, 0, 0, 7, 6, 4, -3\}$$

Let us verify the result obtained by using regular linear convolution we use the tabular method.

	1	2	–1	2	3	–2	–3	–1	1	1	2	–1
1	1	2	–1	2	3	–2	–3	–1	1	1	2	–1
2	2	4	–2	4	6	–4	–6	–2	2	2	4	–2
3	3	6	–3	6	9	–6	–9	–3	3	3	6	–3

$\therefore$ y(n) = {1, 4, 6, 6, 4, 10, 2, –13, –1, 0, 0, 7, 6, 4, –3}

3.11 Summary of DFT properties

1. Linearity	$ax_1(n) + b\,x_2(n) \xleftrightarrow{DFT} a\,X_1(k) + b\,X_2(k)$
2. Periodicity	X (k) = X (k + N)
3. Circular time shift	$x(n-m)_N \xleftrightarrow{DFT} e^{-j2\pi km/N} X(k)$
4. Circular frequency shift	$x(n)\, e^{-j2\pi km/N} \xleftrightarrow{DFT} X(k-m)_N$
5. Time reversal	$x((-n))_N \xleftrightarrow{DFT} x((-k))_N$
6. Complex conjugate	$x^*((n))_N \xleftrightarrow{DFT} x((-k))_N$ $x(n) \xleftrightarrow{DFT} X(N-K)$
7. Parsevals theorem	$\sum_{n=0}^{N-1} \lvert x(n)\rvert^2 = \frac{1}{N}\sum_{n=0}^{N-1} \lvert X(k)\rvert^2$
8. Circular convolution	$x_1(n) \circledast x_2(n) \xleftrightarrow{DFT} X_1(k) \cdot X_2(k)$

3.12 2 D- Discrete Fourier Transform (2-DFT)

1-D DFT is useful while dealing with 1-Dimensiinal signals. Images are 2 Dimensional signals and we need to use 2-D DFT to compute their frequency components. The 2-D DFT is given by the formula,

$$F(u, v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x, y)\, e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \quad \ldots. (3.12.1)$$

We use the separability property while computing the 2-D DFT.

3.12.1 The Separability Property

This property states that a 2-D DFT can be separated into two 1-D DFTs.

We know (assume a square image)

$$F(u, v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x, y)\, e^{-j2\pi\left(\frac{ux}{N}+\frac{vy}{N}\right)} \quad \ldots(3.12.2)$$

This can be split up as,

$$F(u, v) = \sum_{x=0}^{N-1} e^{-j2\pi\frac{ux}{N}} \sum_{y=0}^{N-1} f(x, y)\, e^{\frac{-j2\pi vy}{N}}$$

$$\text{Let } F(x, v) = \sum_{y=0}^{N-1} f(x, y)\, e^{-j2\pi\frac{vy}{N}}$$

$$\therefore F(u, v) = \sum_{y=0}^{N-1} f(x, v)\, e^{-j2\pi\frac{ux}{N}}$$

As stated, the principle advantage of the separability property is that the 2-D DFT can be obtained in two steps by successive application of 1-D DFT.

F(u, v) can be obtained by applying 1-D DFT along the rows and then along the columns. One could also take the 1-D DFT first along the columns and then along the rows.

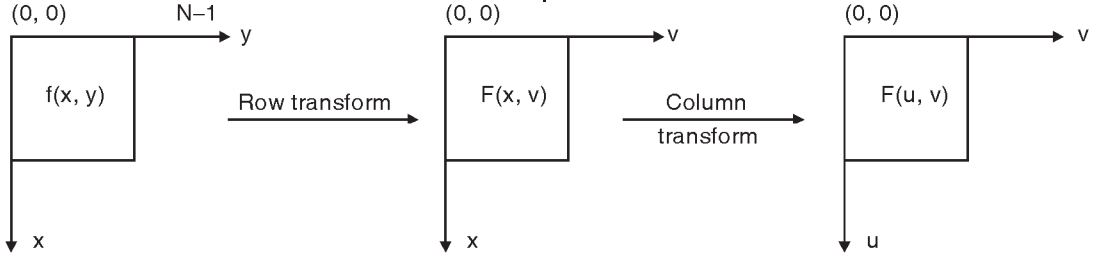


Fig. 3.12.1

3.12.2 Solved Examples on 2- DFT

Ex. 3.12.1 : Find the DFT of the image :

0	1	2	1
1	2	3	2
2	3	4	3
1	2	3	2

Soln. : We have studied the 1-D DFT matrix. We shall use the DFT along the rows and then along the columns.

Length of each row is N = 4 ∴ we need a 4×4 DFT matrix.

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ 0 \\ -2 \end{bmatrix} \rightarrow \text{DFT of } 1^{st} \text{ row}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ -2 \\ 0 \\ -2 \end{bmatrix} \rightarrow \text{DFT of } 2^{nd} \text{ row}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 12 \\ -2 \\ 0 \\ -2 \end{bmatrix} \rightarrow \text{DFT of } 3^{rd} \text{ row}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ -2 \\ 0 \\ -2 \end{bmatrix} \rightarrow \text{DFT of } 4^{th} \text{ row}$$

Hence we have an intermediate stage $\begin{bmatrix} 4 & -2 & 0 & -2 \\ 8 & -2 & 0 & -2 \\ 12 & -2 & 0 & -2 \\ 8 & -2 & 0 & 2 \end{bmatrix}$

Now using the 1-D DFT along the columns of this intermediate image we get

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 8 \end{bmatrix} = \begin{bmatrix} 32 \\ -8 \\ 0 \\ -8 \end{bmatrix} \rightarrow \text{DFT of } 1^{st} \text{ column}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} -2 \\ -2 \\ -2 \\ -2 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \text{DFT of } 2^{nd} \text{ column}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \text{DFT of } 3^{rd} \text{ column}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -i & 1 & -i \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} -2 \\ -2 \\ -2 \\ -2 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \text{DFT of } 4^{th} \text{ column}$$

∴ The final DFT of the entire image is

32	– 8	0	– 8
– 8	0	0	0
0	0	0	0
– 8	0	0	0

Another way of computing the 2D-DFT is by using the formula

$$\mathbf{F = T f T'} \qquad ...(1)$$

where T is the N x N DFT matrix.

The DFT matrix is both symmetric as well as separable and hence can be obtained using the relationship

$$\mathbf{F = T f T} \qquad ...(2)$$

We shall solve the same example by using this formula

Ex. 3.12.2 : Find the DFT of the given image.

$$\begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 2 \end{bmatrix}$$

Soln. : Since f is a 4×4 matrix, T which is the DFT matrix is given below

$$T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$F = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 8 & 12 & 8 \\ -2 & -2 & -2 & -2 \\ 0 & 0 & 0 & 0 \\ -2 & -2 & -2 & -2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$F = \begin{bmatrix} 32 & -8 & 0 & -8 \\ -8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -8 & 0 & 0 & 0 \end{bmatrix}$$

This is the same as what we had got using the conventional method.

Summary

This chapter deals with understanding the DFT. The importance of the DFT was discussed. Examples were solved using the conventional method as well as the matrix method. Computation of Inverse DFT was explained. Drawing the Magnitude as well as the Phase spectrum was carried out. We also explained how to identify the frequency components from the x- axis of the DFT spectrum. Properties of the DFT were listed and examples were solved using each of the properties. The chapter also introduced the 2-D DFT.

Review Questions

Q. 1 State and prove differentiation property of Fourier transform.

Q. 2 State and prove circular convolution property of Fourier transform.

Q. 3 State and prove properties of Discrete Time Fourier Transform (DTFT).

Q. 4 Explain the convolution property of Fourier transform of a DT signal.

Q. 5 State and prove following properties of Fourier transform :

(i) Periodicity (ii) Linearity

Q. 6 Explain Overlap-save and Overlap-add algorithms.

Q. 7 Explain Circular convolution.

Q. 8 Prove that linear convolution is obtained using circular convolution property of DFT.

Q. 9 With the help of neat flow diagram explain DIF FFT algorithm consider N = 8.

Q. 10 Explain the relationship between DTFT and DFT.

❑❑❑

CHAPTER 4

Fast Fourier Transform

Syllabus :

Need of FFT, Radix-2 DIT-FFT algorithm, DIT-FFT Flow graph for N = 4 and 8, Inverse FFT algorithm.

4.1 Introduction

We studied the Discrete Fourier Transform (DFT) in the Chapter 3.

The DFT is used to identify the frequency components present in a discrete time signal. The DFT converts a time domain signal into a frequency domain signal.

A N–point DFT is given by the equation

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk} \quad ; \quad k = 0, 1, \ldots, N-1 \qquad \ldots(4.1.1)$$

In the matrix form it is given by the equation

$$X = [W]x \qquad \ldots(4.1.2)$$

4.2 Computational Complexity of DFT

Now, the question that we need to ask ourselves is How many multiplications and additions are required to compute a N-point DFT ?

We will solve an example to answer this question.

Solved Example

Ex. 4.2.1 : Compute the DFT of the sequence $x(n) = \{0, 1, 2, 1\}$.

Soln. : Since the length of the signal is 4,

$$N = 4 \rightarrow 4\text{-Point DFT}$$

We generate a 4×4 DFT matrix and use Equation (4.1.2).

$$W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & +1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

	Number of multiplication	Number of addition
$x(0) \rightarrow$	$4\times$	$3+$
$x(1) \rightarrow$	$4\times$	$3+$
$x(2) \rightarrow$	$4\times$	$3+$
$x(3) \rightarrow$	$4\times$	$3+$
	$16\times$	$12+$

We have also written the number of multiplication and addition required to calculate each X(k)

$\therefore X(0) = 1\cdot x(0) + 1\cdot x(1) + 1\cdot x(2) + 1\cdot x(3)$ $\rightarrow$4 multi. & 3 add

$X(1) = 1\cdot x(0) + (-j)\cdot x(1) + (-1)\cdot x(2) + j\cdot x(3)$ $\rightarrow$4 multi. & 3 add

$X(2) = 1\cdot x(0) + (-1)x(1) + (1)\cdot x(2) + -(1)x(3)$ $\rightarrow$4 multi. & 3 add

$X(3) = 1\cdot x(0) + j\,x\cdot(1) + (-1)\cdot x(2) + -(j)\cdot x(3)$ $\rightarrow$4 multi. & 3 add

16multi. & 12add

$\therefore X(k) = \{4, -2, 0, -2\}$

As is shown each X(k) requires 4 multiplications and 3 additions.

Hence a 4-point DFT requires a total of 16 multiplications and 12 additions. i.e.$(4)^2$ multiplications and 4×3additions.

We can generalize this by stating the A N-point DFT requires N^2 multiplications and N (N – 1) additions.

Therefore computing the DFT is computationally very expensive.

It has been mentioned in earlier chapters that because of high sampling rates in the real world, the length of x(n) is very large and runs into thousands of samples.

Computing the DFT of such signals would require a machine which has a lot of computation power. It would also take a lot of time.

To cut a long story short, the DFT is computationally very demanding and we require something that would reduce the number of computations.

This is where the Fast Fourier Transform comes in.

Fast Fourier Transform (FFT)

- In 1965, Cooley and Tukey proposed a method of computing the DFT which required only N $\log_2$ N calculations.
- This method came to be known as the Fast Fourier Transform (F.F.T)
- The FFT refers to a class of algorithms for efficiently computing the DFT.
- Hence FFT is not a new transform but a set of algorithms used for computing the DFT efficiently.
- The results obtained from the DFT and the FFT are the same.
- The computational efficiency of the FFT becomes clear when the values of N^2 and N $\log_2$ N are compared for several values of N.

Number of Samples	Number of computations (DFT) N^2	Number of computations (FFT) N $\log_2$ N
32	1024	160
128	16,384	896
2048	4,194,304	22,528

- We notice that as N increases, the FFT becomes more efficient.
- As mentioned earlier, FFT is an algorithm. The commonly used technique is called the RADIX-2 FFT algorithm.
- We have studied the Twiddle factor in the previous chapter. There are two important properties of the Twiddle factor.

$$W_N^{k+\frac{N}{2}} = -W_N^k \quad \text{Symmetric Property}$$

$$W_N^{k+N} = W_N^k \quad \text{Periodic property}$$

- The FFT exploits these two properties to reduce the number of complex multiplications.
- The basic principle of the FFT algorithm is to break (decompose) a N-point DFT into smaller DFTS. As mentioned earlier, FFT is an algorithm and is known as Radix-2 which simply means that the number of output -points is expressed an a power of 2 i.e. $N=2^m$, where m is an integer.
- Hence when the FFT algorithm breaks a DFT into smaller DFTS' the output of the smaller DFTS' are always powers of 2.
- The two commonly used algorithms are
  1. Decimation in Time FFT (DIT-FFT)
  2. Decimation in Frequency FFT (DIF-FFT)
- We shall derive and discuss each one in detail but before doing that let us try to visualize a N-point DFT formula.

  We know a N-point DFT is given by the equation

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk};\ k = 0, 1, \ldots, N-1$$

- Here x(n) is the input and X(k) the output. This equation can be pictorially represented as shown in Fig. 4.2.1

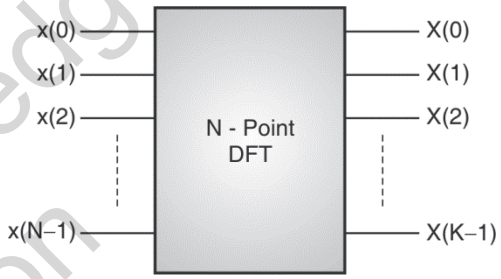


Fig. 4.2.1

- The DFT block computes the DFT of the input x(n) and gives us the output X(k).
- Both, DIF-FFT and DIT-FFT, are based on the basic principle of breaking this N-Point DFT into smaller DFT Blocks.

4.3 Decimation In Time FFT (DIT-FFT)

- To decimate means to kill. Decimation in time means killing (re-arranging) the time signal.
- Here we exploit the symmetric and periodic property of the twiddle factor W_N

$$W_N^{k+N/2} = -W_N^k \rightarrow \text{Symmetric}$$

$$W_N^{k+N} = W_N^k \rightarrow \text{Periodic}$$

- These two important properties reduce the number of computations.

A N-point-DFT is given by the equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk} \qquad k = 0, 1, 2, \ldots N-1$$

– In decimation-in-time technique, we split the input x(n) into odd and even parts.

$$X(k) = \sum_{n \to \text{even}} x(n)\, W_N^{nk} + \sum_{n \to \text{odd}} x(n)\, W_N^{nk}$$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_N^{2nk} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{(2n+1)k}$$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_N^{2nk} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{2nk}$$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_N^{2nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{2nk}$$

Now, $W_N^2 = W_{N/2}$

$$X(k) = \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk}}_{\text{N/2 point DFT}} + \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x(2n+1)\, W_{N/2}^{nk}}_{\text{N/2 point DFT}} \quad \ldots(4.3.1)$$

Let $\sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk} = F_1(k)$

and $\sum_{n=0}^{\frac{N}{2}-1} x(2n+1)\, W_{N/2}^{nk} = F_2(k)$; $k = 0, 1, \ldots \frac{N}{2}$

$$\therefore X(k) = F_1(k) + W_N^k F_2(k) \quad \ldots (4.3.2)$$

This equation gives us the first $\frac{N}{2}$ values

– Since $F_1(k)$ and $F_2(k)$ are periodic with period N/2 we can write

$F_1(k + N/2) = F_1(k)$ and

$F_2(k + N/2) = F_2(k)$

also $W_N^{k+N/2} = -W_N^k$

$\therefore$ Equation (4.3.2) can be written as,

$$\therefore X(k) = F_1(k) + W_N^k F_2(k) ; \quad k = 0,1, \ldots \frac{N}{2} - 1$$

$$\therefore X\left(k + \frac{N}{2}\right) = F_1(k) - W_N^k F_2(k); k = 0, 1, \ldots \frac{N}{2} - 1 \quad \ldots (4.3.3)$$

– We observe that $F_1(k)$ requires $(N/2)^2$ complex multiplications and the same applies to $F_2(k)$. Furthermore, there are N/2 additional complex multiplications required to compute $W_N^k F_2(k)$. Hence the computation of X(k) requires complex multiplications which is almost half the computations required by the conventional method.

$$2 \times \left(\frac{N}{2}\right)^2 + \frac{N}{2} = \frac{N^2}{2} + \frac{N}{2}$$

Let us take an example where N = 8,

– Equation (4.3.3) can be shown using a signal flow graph.

Since in DIT - FFT, we have split the input x(n) into even and odd parts, we have

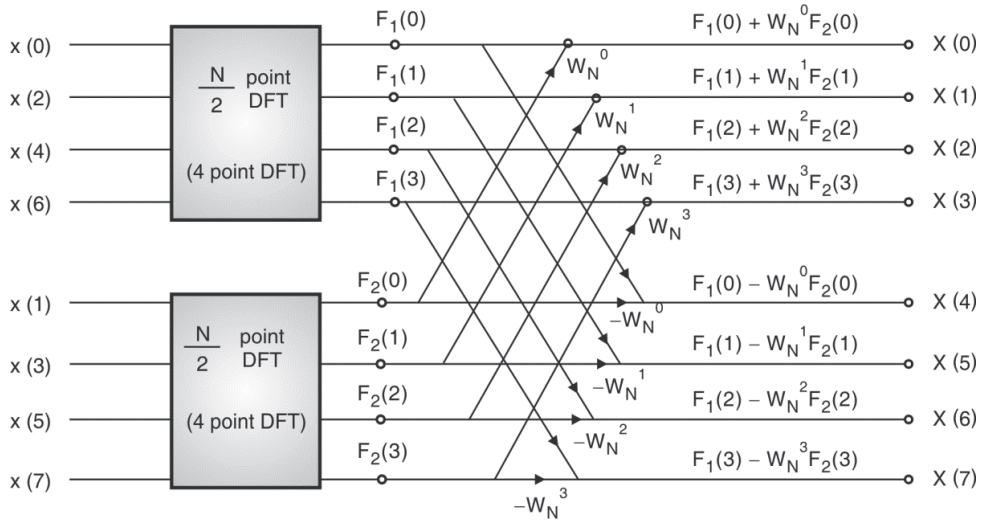


Fig. 4.3.1

– As stated earlier, $F_1(k)$ and $F_2(k)$ are N/2 point DFT's (4 point DFT's if N = 8). These can be further split up into odd and even parts. We rewrite Equation 4.3.1.

$$\text{i.e.} \quad X(k) = \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk}}_{F_1(k)} + W_N^k \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x(2n+1)\, W_{N/2}^{nk}}_{F_2(k)}$$

– Splitting up $F_1(k)$ and $F_2(k)$ into odd and even parts, we get,

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1)\, W_{N/2}^{nk}$$

2 parts of N/4 each (A) 2 parts of N/4 each (B)

(A)
$$F_1(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk} = \sum_{n=0}^{\frac{N}{4}-1} x(4n)\, W_{N/2}^{2nk} + \sum_{n=0}^{\frac{N}{4}-1} x(4n+2)\, W_{N/2}^{(2n+1)k}$$

$$= \sum_{n=0}^{\frac{N}{u}-1} x(un)\, W_{N/L}^{2nk} + \sum_{n=0}^{\frac{N}{u}-1} x(4n+2)\, W_{N/2}^{2nk} . W_{N/2}^{k}$$

$$= \sum x(un)\, W_{N/2}^{2nk} + W_{N/2}^{k} \sum x(un+2)\, W_{N/2}^{2nk}$$

Now $W_{N/2}^{2} = W_{N/4}^{1}$

$$\therefore \quad F_1(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)\, W_{N/2}^{nk} = \underbrace{\sum_{n=0}^{\frac{N}{4}-1} x(4n)\, W_{N/4}^{nk}}_{\text{N/4 point DFT}} + W_{N/2}^{k} \underbrace{\sum_{n=0}^{\frac{N}{4}-1} x(4n+2)\, W_{N/4}^{nk}}_{\text{N/4 point DFT}}$$

$$\therefore \quad F_1(k) = \sum_{n=0}^{\frac{N}{4}-1} x(4n)\, W_{N/4}^{nk} + W_{N/2}^{k} \sum_{n=0}^{\frac{N}{4}-1} x(4n+2)\, W_{N/4}^{nk}$$

Let $\sum_{n=0}^{\frac{N}{4}-1} x(4n)\, W_{N/4}^{nk} = G_1(k)$

and $\sum_{n=0}^{\frac{N}{4}-1} x(4n+2)\, W_{N/4}^{nk} = G_2(k)$

$$\therefore F_1(k) = G_1(k) + W_{N/2}^{k} \; G_2(k)$$

This equation will give us the first $\frac{N}{4}$ values

In a similar manner we can write for (B).

(B) $F_2(k) = H_1(k) + W_{N/2}^{k} H_2(k)$

Now, $G_1(k)$, $G_2(k)$, $H_1(k)$ and $H_2(k)$ are all N/4 point DFT's and are periodic with period N/4. Using the symmetric and periodic properties of the twiddle factor we get

$$\left.\begin{aligned} F_1(k) &= G_1(k) + W_{N/2}^{k} \; G_2(k) \\ \therefore \quad F_1(k + N/4) &= G_1(k) - W_{N/2}^{k} \; G_2(k) \\ F_2(k) &= H_1(k) + W_{N/2}^{k} \; H_2(k) \\ \therefore \quad F_2(k + N/4) &= H_1(k) - W_{N/2}^{k} \; H_2(k) \end{aligned}\right\} k = 0, 1, 2, \ldots, \frac{N}{4} - 1 \qquad \ldots (4.3.4)$$

For N = 8, k will have values 0, 1.

We have split $F_1(k)$ and $F_2(k)$ which were N/2 point DFT's into two N/4 point DFT's each.

We draw the signal flow graph using Equation (4.3.4) and the earlier Fig. 4.3.1. Let N = 8

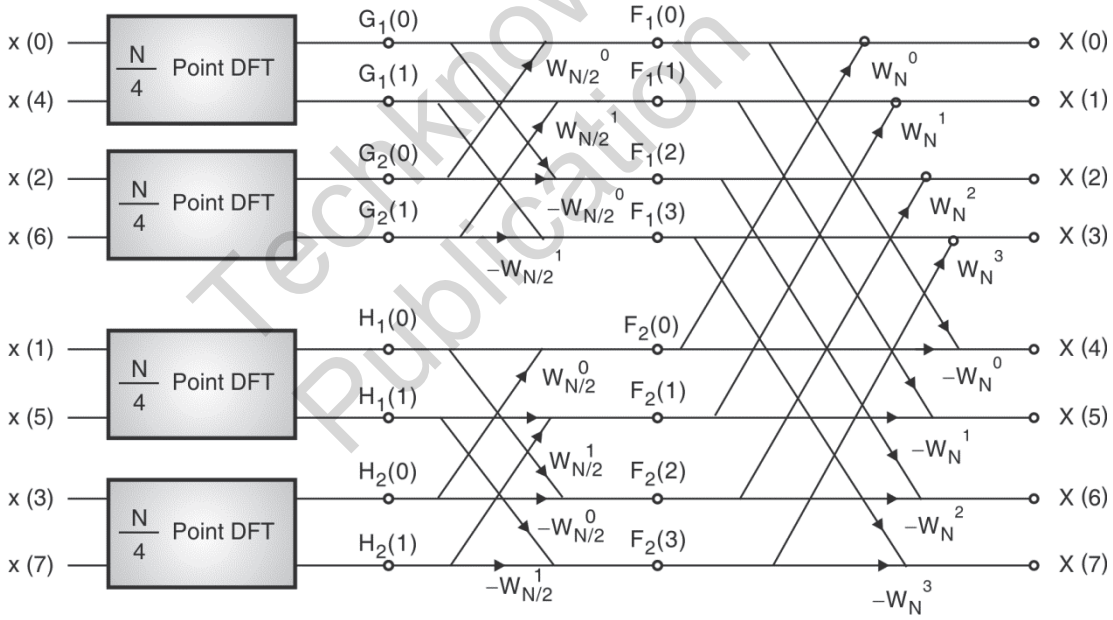


Fig. 4.3.2

A 2-Point DFT is drawn as,

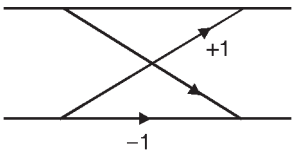


Fig. 4.3.3

Hence the final signal flow graph is shown in Fig. 4.3.4.

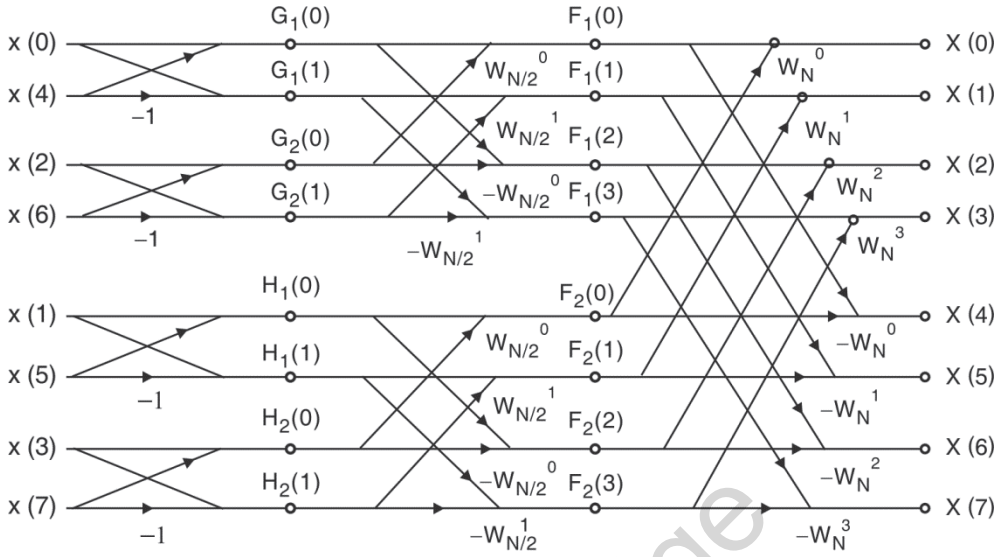


Fig. 4.3.4 : 8-point DIT-FFT Butterfly diagram

This diagram is also called an 8-point butterfly diagram.

The DIT-FFT refers to the method in which the input samples are broken into odd and even parts. This time decimation leads to the scrambled order of the input.

The first decimation gives x(0), x(2), x(4), x(6), x(1)....

The second decimation gives x(0), x(4), x(2), x(6)....

4.4 Bit Reversal Format

The shuffling of the input data is known as bit-reversal. This is because the order of the input data can be obtained by reversing the bits of the binary representation of the normal input data index order

$$x(4) = 100 \rightarrow BR \rightarrow 001 \rightarrow x(1)$$

	Original Samples	Binary Representation	Bit reversed	Decimal	New order of input
x(0) ⟶	0	000	000	0 ⟶	x(0)
x(1) ⟶	1	001	100	4 ⟶	x(4)
x(2) ⟶	2	010	010	2 ⟶	x(2)
x(3) ⟶	3	011	110	6 ⟶	x(6)
x(4) ⟶	4	100	001	1 ⟶	x(1)
x(5) ⟶	5	101	101	5 ⟶	x(5)
x(6) ⟶	6	110	011	3 ⟶	x(3)
x(7) ⟶	7	111	111	7 ⟶	x(7)

Hence the input that is fed to the algorithm is in the bit reversed format.

4.4.1 Solved Examples on Bit Reversal Format

Ex. 4.4.1 : An 8-point sequence x(n) is given by , {0, 1, 2, 3, 2, 1, 5, 2, 1}

Find the DFT of the sequence using DIT-FFT.

Soln. : Since N = 8, we draw a 8-point DIT-FFT Butterfly diagram remember, the input to the network is in the bit reversed order.

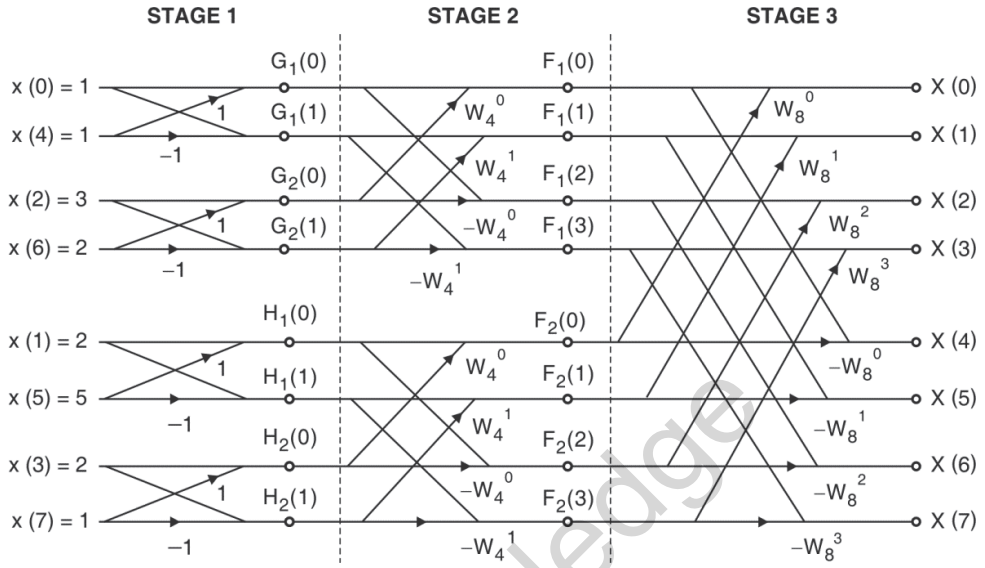


Fig. P.4.4.1

$x(n) = \{1, 2, 3, 2, 1, 5, 2, 1\}$

$x(0)\ x(1)\ x(2)\ x(3)\ x(4)\ x(5)\ x(6)\ x(7)$

Here $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi 1}{4}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi\cdot 1}{8}} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j0.707$$

$$W_8^2 = e^{\frac{-j2\pi\cdot 2}{8}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi\cdot 3}{8}} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - j0.707$$

We write down the equations at each stage and calculate the outputs.

Stage 1

$G_1(0) = x(0) + x(4) = 1 + 1 = 2$

$G_1(1) = x(0) - x(4) = 1 - 1 = 0$

$G_2(0) = x(2) + x(6) = 3 + 2 = 5$

$G_2(1) = x(2) - x(6) = 3 - 2 = 1$

$H_1(0) = x(1) + x(5) = 2 + 5 = 7$

$H_1(1) = x(1) - x(5) = 2 - 5 = -3$

$H_2(0) = x(3) + x(7) = 2 + 1 = 3$

$H_2(1) = x(3) - x(7) = 2 - 1 = 1$

Stage 2

$F_1(0) = G_1(0) + W_4^0 G_2(0) = 2 + 1 \cdot (5) = 7$

$F_1(1) = G_1(1) + W_4^1 G_2(1) = 0 + (-j)(1) = -j$

$F_1(2) = G_1(0) - W_4^0 G_2(0) = 2 - 1 \cdot (5) = -3$

$F_1(3) = G_1(1) - W_4^1 G_2(1) = 0 - (-j)(1) = +j$

$F_2(0) = H_1(0) + W_4^0 H_2(0) = 7 + 1 \cdot (3) = 10$

$F_2(1) = H_1(1) + W_4^1 H_2(1) = -3 + (-j)(1) = -3 - j$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 7 - 1 \cdot (3) = 4$

$F_2(3) = H_1(1) - W_4^1 H_2(1)$

$= -3 - (-j)(1) = -3 + j$

Stage 3

$X(0) = F_1(0) + W_8^0 F_2(0) = 7 + 1 \cdot (10) = 17$

$X(1) = F_1(1) + W_8^1 F_2(1)$

$= -j + (0.707 - j\,0.707)(-3 - j)$

$= -2.83 + j0.41$

$X(2) = F_1(2) + W_8^2 F_2(2)$

$= -3 + (-j)(4) = -3 - j4$

$X(3) = F_1(3) + W_8^3 F_2(3)$

$= j + (-0.707 - j0.707)(-3 + j)$

$= 2.83 + j2.41$

$$X(4) = F_1(0) - W_8^0 F_2(0) = 7 - 1 \cdot (10) = -3$$

$$X(5) = F_1(1) - W_8^1 F_2(1) = -j - (0.707 - j0.707)(-3 - j) = 2.83 - j2.41$$

$$X(6) = F_1(2) - W_8^2 F_2(2) = -3 - (-j)(4) = -3 + j4$$

$$X(7) = F_1(3) - W_8^3 F_2(3) = j - (-0.707 - j0.707)(-3 + j) = -2.83 - j0.41$$

$$\therefore \quad X(k) = \{17, -2.83 + j0.41, -3 - j4, 2.83 + j2.41, -3, 2.83 - j2.41, -3 + j4, -2.83 - j0.41\}$$

Ex. 4.4.2 Find s-point DIT of the sequence x(n) = {1, 2, 3, 4, 4, 3, 2, 1} using DIT Radix 2 FFT

Soln. : Since N = 8, we draw a 8-point DIT-FFT diagram

x(n) = { 1 , 2 , 3 , 4 , 4 , 3 , 2 , 1 }

x(0) x(1) x(2) x(3) x(4) x(5) x(6) x(7)

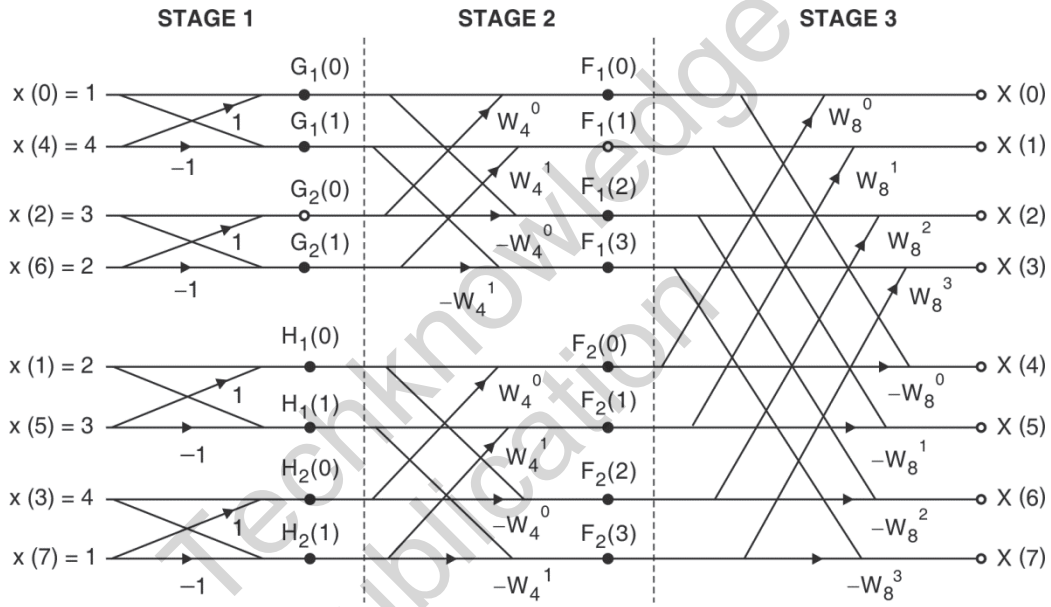


Fig. P. 4.4.2

Here $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{4} \cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{4} \cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - j0.707$$

We write down the equation at each stage and calculate the outputs.

Stage 1

$$G_1(0) = x(0) + x(4) = 1 + 4 = 5$$

$$G_1(1) = x(0) - x(4) = 1 - 4 = -3$$

$$G_2(0) = x(2) + x(6) = 3 + 2 = 5$$

$$G_2(1) = x(2) - x(6) = 3 - 2 = 1$$

$$H_1(0) = x(1) + x(5) = 2 + 3 = 5$$

$$H_1(1) = x(1) - x(5) = 2 - 3 = -1$$

$$H_2(0) = x(3) + x(7) = 4 + 1 = 5$$

$$H_2(1) = x(3) - x(7) = 4 - 1 = 3$$

Stage 2

$$F_1(0) = G_1(0) + W_4^0 G_2(0) = 5 + 1(5) = 10$$

$$F_1(1) = G_1(1) + W_4^1 G_2(1) = -3 + (-j)(1) = -3 - j$$

$$F_1(2) = G_1(0) - W_4^0 G_2(0) = 5 - 1(5) = 0$$

$F_1(3) = G_1(1) - W_4^1 G_2(1) = -3-(-j)(1) = -3+j$

$F_2(0) = H_1(0) + W_4^0 H_2(0) = 5+1(5) = 10$

$F_2(1) = H_1(1) + W_4^1 H_2(1) = -1+(-j)(3) = -3-j3$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 5-1(5) = 0$

$F_2(3) = H_1(1) - W_4^1 H_2(1) = -1-(-j)(3) = -1+j3$

Stage 3

$X(0) = F_1(0) + W_8^0 F_2(0) = 10+(1)(10) = 20$

$X(1) = F_1(1) + W_8^1 F_2(1)$

$= (-3-j) + (0.707 - j\,0.707)(-1-j3)$

$= -5.825 - j\,2.414$

$X(2) = F_1(2) + W_8^2 F_2(2)$

$= 0 + (-j)(0) = 0$

$X(3) = F_1(3) + W_8^3 F_2 + (3)$

$= (-3+j) + (-0.707 - j0.707)(-1+j3)$

$= -0.172 - j\,0.414$

$X(4) = F_1(0) - W_8^0 F_2(0) = 10-(1)(10) = 0$

$X(5) = F_1(1) - W_8^1 F_2(1)$

$= (-3-j) - (0.707 - j0.707)(-1-j3)$

$= -0.172 + j0.414$

$X(6) = F_1(2) - W_8^2 F_2(2) = 0-(-j)(0) = 0$

$X(7) = F_1(3) - W_8^3 F_2(3)$

$= (-3-j) - (-0.707 - j0.707)(-1+j3)$

$= -5.828 + j2.414$

$\therefore \quad X(k) = \{20, -5.828 - j2.414, 0, -0.172 - j0.414, 0, -0.172 + j0.414, 0, -5.828 + j\,2.414$

Ex. 4.4.3 : Given $x(n)=2^n$. Find X(k) using DIT-FFT algorithms. Assume x(n) is of length 8.

Soln. : Here N=8we vary n from 0to 7.

$\therefore \quad x(n) = 2^n, \quad 0 \le n < 7$

$\therefore \quad x(0) = 2^0 = 1$

$\therefore \quad x(1) = 2^1 = 2$

$\therefore \quad x(2) = 2^2 = 4$

$\therefore \quad x(3) = 2^3 = 8$

$\therefore \quad x(4) = 2^4 = 16$

$\therefore \quad x(5) = 2^5 = 32$

$\therefore \quad x(6) = 2^6 = 64$

$\therefore \quad x(7) = 2^7 = 128$

$\therefore \quad x(n) = \{1, 2, 4, 8, 16, 32, 64, 128\}$

Since N= 8, we draw a 8 point DIT-FFT. Remember the input is the bit-reversed format.

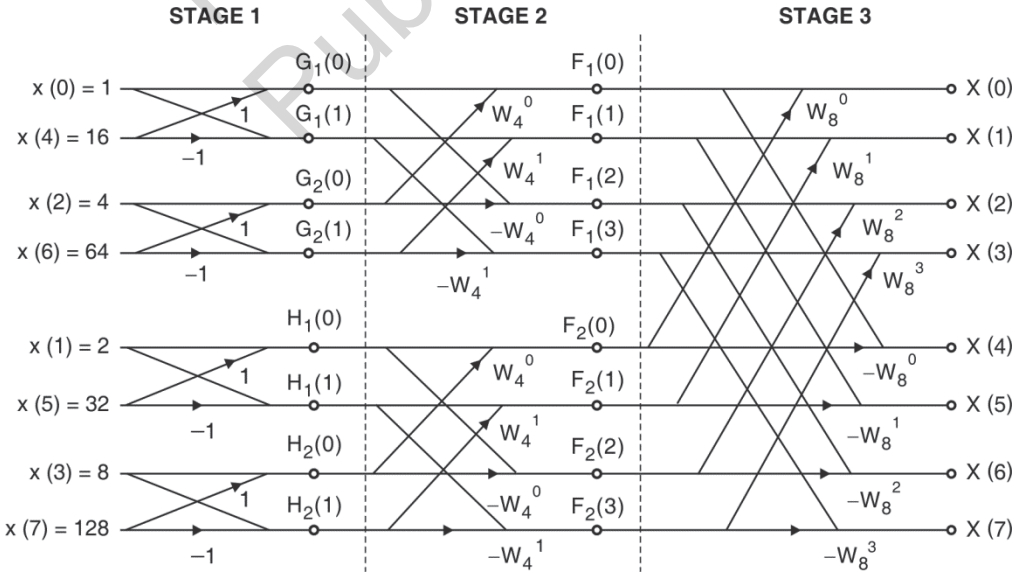


Fig. P.4.4.3

Here $W_4^0 = 1$

$W_4^1 = e^{\frac{-j2\pi}{4}\cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$

Similarly, $W_8^0 = 1$

$$W_8^1 = e^{\frac{-j2\pi}{8}\cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2}$$

$$= 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4}$$

$$= -0.707 - j\,0.707$$

We write down the equations at each stage

Stage 1

$G_1(0) = x(0) + x(4) = 1 + 16 = 17$

$G_1(1) = x(0) - x(4) = 1 - 16 = -15$

$G_2(0) = x(2) + x(6) = 4 + 64 = 68$

$G_2(1) = x(2) - x(6) = 4 - 64 = -60$

$H_1(0) = x(1) + x(5) = 2 + 32 = 34$

$H_1(1) = x(1) - x(5) = 2 - 32 = -30$

$H_2(0) = x(3) + x(7) = 8 + 128 = 136$

$H_2(1) = x(3) - x(7) = 8 - 128 = -120$

Stage 2

$F_1(0) = G_1(0) + W_4^0 G_2(0) = 17 + 1 \cdot (68) = 85$

$F_1(1) = G_1(1) + W_4^1 G_2(1)$

$= -15 + (-j)(-60) = -15 + j60$

$F_1(2) = G_1(0) - W_4^0 G_2(0)$

$= 17 - 1(68) = -51$

$F_1(3) = G_1(1) - W_4^1 G_2(1)$

$= -15 - (-j)(-60) = -15 - j\,60$

$F_2(0) = H_1(0) + W_4^0 H_2(0)$

$= 34 + 1 \cdot (136) = 170$

$F_2(1) = H_1(1) + W_4^1 H_2(1)$

$= -30 + (-j)(-120) = -30 + j120$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 34 - 1 \cdot (136) = -102$

$F_2(3) = H_1(1) - W_4^1 H_2(1)$

$= -30 - (-j)(-120) = -30 - j120$

Stage 3

$X(0) = F_1(0) + W_8^0 F_2(0) = 85 + 1(170) = 255$

$X(1) = F_1(1) + W_8^1 F_2(1)$

$= (-15 + j60) + (0.707 - j0.707)(-30 + j120)$

$= 48.64 + j166.07$

$X(2) = F_1(2) + W_8^2 F_2(2)$

$= -51 + (-j)(-102) = -51 + j102$

$X(3) = F_1(3) + W_8^3 F_2(3)$

$= (-15 - j60) + (-0.707 - j0.707)(-30 - j120)$

$= -78.64 + j46.07$

$X(4) = F_1(0) - W_8^0 F_2(0) = 85 - 1 \cdot (170) = -85$

$X(5) = F_1(1) - W_8^1 F_2(1)$

$= (-15 + j60) - (0.707 - j0.707)(-30 + j120)$

$= -78.64 - j46.07$

$X(6) = F_1(2) - W_8^2 F_2(2)$

$= -51 - (-j)(-120) = -51 - j102$

$X(7) = F_1(3) - W_8^3 F_2(3)$

$= (-15 - j60) - (-0.707 - j0.707)(-30 - j120)$

$= 48.6 - j166.07$

$X(k) = \{255, 48.64 + j166.047, -51 + j102,$

$-78.64 + j46.07, -85, -78.64 - j46.07,$

$-51 - j102, 48.6 - j166.07\}$

Ex.4.4.4 : Compute the DFT of x(n)= {1,0,2,0,3,0,4,0} using DIT-FFT.

Soln. : Since N = 8, we drawn a 8-point DIT-FFT Butterfly diagram. The input is in the bit reversed order.

$x(n) = \{1, 0, 2, 0, 3, 0, 4, 0\}$

Here $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi\cdot 1}{u}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

Similarly,

$W_8^0 = 1$

$$W_8^1 = e^{\frac{-j2\pi\cdot 1}{8}} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j0.707$$

$$W_8^2 = e^{\frac{-j2\pi\cdot 2}{8}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi\cdot 3}{8}} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - j0.707$$

We write down the equation at each stage and calculate the outputs

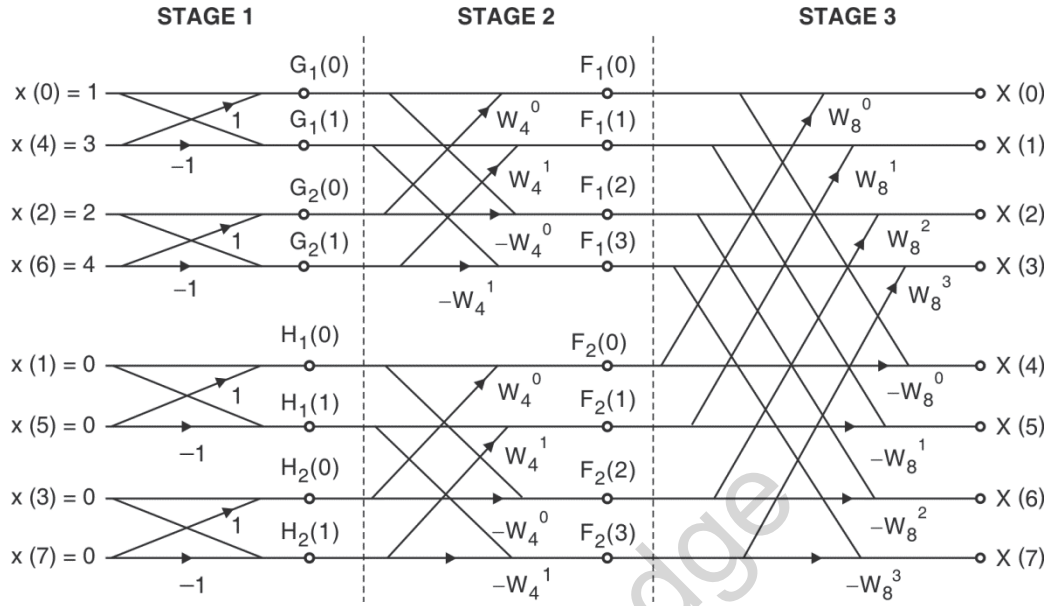


Fig. P. 4.4.4

Stage 1

$G_1(0) = x(0) + x(4) = 1 + 3 = 4$

$G_1(1) = x(0) - x(4) = 1 - 3 = -2$

$G_2(0) = x(2) + x(6) = 2 + 4 = 6$

$G_2(1) = x(2) - x(6) = 2 - 4 = -2$

$H_1(0) = x(1) + x(5) = 0 + 0 = 0$

$H_1(1) = x(1) - x(5) = 0 - 0 = 0$

$H_2(0) = x(3) + x(7) = 0 + 0 = 0$

$H_2(1) = x(3) - x(7) = 0 - 0 = 0$

Stage 2

$F_1(0) = G_1(0) + W_4^0 G_2(0) = 4 + 1 \cdot (6) = 10$

$F_1(1) = G_1(1) + W_4^1 G_2(1) = -2 + (-j)(-2)$

$= -2 + j2$

$F_1(2) = G_1(0) - W_4^0 G_2(0) = 4 - 1 \cdot (6) = -2$

$F_1(3) = G_1(1) - W_4^1 G_2(1) = -2 - (-j)(-2)$

$= -2 - j2$

$F_2(0) = H_1(0) + W_4^0 H_2(0) = 0 + 1 \cdot (0) = 0$

$F_2(1) = H_1(1) + W_4^1 H_2(1) = 0 + (-j)(0) = 0$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 0 - 1 \cdot (0) = 0$

$F_2(3) = H_1(1) - W_4^1 H_2(1) = 0 - (-j)(0) = 0$

Stage 3

$X(0) = F_1(0) + W_8^0 F_2(0) = 10 + 1 \cdot (0) = 10$

$X(1) = F_1(1) + W_8^1 F_2(1)$

$= (-2 + 2j) + (0.707 - j0.707) \cdot 0$

$= -2 + j2$

$X(2) = F_1(2) + W_8^2 F_2(2) = -2 + (-j)(0) = -2$

$X(3) = F_1(3) + W_8^3 F_2(3)$

$= (-2 - 2j) + (-0.707 - j0.707) \cdot 0$

$= -2 - j2$

$X(4) = F_1(0) - W_8^0 F_2(0) = 10 - 1 \cdot (0) = 10$

$X(5) = F_1(1) - W_8^1 F_2(1)$

$= (-2 + 2j) - (0.707 - j0.707) \cdot 0$

$= -2 + j2$

$X(6) = F_1(2) - W_8^2 F_2(2) = -2 - (-j)(0)$

$= -2$

$$X(7) = F_1(3) - W_8^3 F_2(3)$$

$$= (-2 - 2j) - (-0.707 - j0.707)\cdot 0$$

$$= -2 - j2$$

$$\therefore \quad X(k) = \{10, -2 + j2, -2, -2 - j2, 10, -2 + j2, -2, -2 - j2\}$$

Ex. 4.4.5 : Sample the given continuous time signal x(t) = sin (2 π 1000t)+ 0.5 sin (2π 1000 t) at 8000 sample sec. Find out the Eight point DFT using the DIT-FFT algorithm.

Soln. : $x(t) = \sin(2\pi 1000 t) + 0.5 \sin(2\pi 2000 t)$

To obtain a sampled version of x(t), we replace t by $\frac{n}{F_s}$.

Here F_s = 8000 samples /sec

$$\therefore x(n) = \sin\left(\frac{2\pi 1000 n}{F_s}\right) + 0.5 \sin\left(\frac{2\pi 2000 n}{F_s}\right)$$

$$\therefore x(n) = \sin\left(\frac{2\pi 1000 n}{8000}\right) + 0.5 \sin\left(\frac{2\pi 2000 n}{8000}\right)$$

$$\therefore x(n) = \sin\left(\frac{\pi n}{4}\right) + 0.5 \sin\left(\frac{\pi n}{2}\right) \quad \ldots(1)$$

Since we require a 8 point DFT, we vary n from 0 to 7 in Equation (1),

$$x(0) = \sin\left(\frac{\pi}{4}\cdot 0\right) + 0.5 \sin\left(\frac{\pi}{2} - 0\right) = 0$$

Similarly, x(1) = 1.2

x(2) = 1

x(3) = 0.2

x(4) = 0

x(5) = – 0.2

x(6) = – 1

x(7) = – 1.2

$$x(n) = \{0, 1.2, 1, 0.2, 0, -0.2, -1, -1.2\}$$

x(0) x(1) x(2) x(3) x(4) x(5) x(6) x(7)

Since N = 8, we draw a 8 – point DIT-FFT Butterfly diagram. The input is in bit reversed order.

Here, $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi\cdot 1}{4}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi\cdot 1}{8}} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi\cdot 2}{8}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi\cdot 3}{8}} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4}$$

$$= -0.707 - j\,0.707$$

We write down the equations at each stage and calculate the outputs.

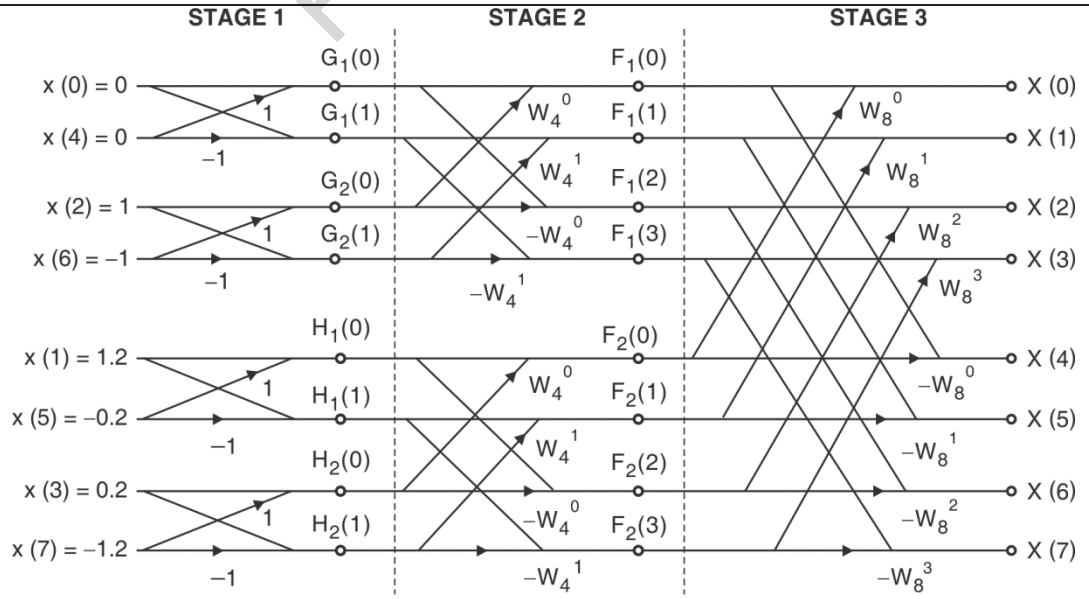


Fig. P.4.4.5

Stage 1

$$G_1(0) = x(0) + x(4) = 0 + 0 = 0$$

$$G_1(1) = x(0) - x(4) = 0 - 0 = 0$$

$$G_2(0) = x(2) + x(6) = 1 + (-1) = 0$$

$$G_2(1) = x(2) - x(6) = 1 - (-1) = 2$$

$$H_1(0) = x(1) + x(5) = 1.2 + (-0.2) = 1$$

$$H_1(1) = x(1) - x(5) = 1.2 - (-0.2) = 1.4$$

$$H_2(0) = x(3) + x(7) = 0.2 + (-1.2) = -1$$

$$H_2(1) = x(3) - x(7) = 0.2 - (-1.2) = 1.4$$

Stage 2

$$F_1(0) = G_1(0) + W_4^0 G_2(0) = 0 + 1 \cdot (0) = 0$$

$$F_1(1) = G_1(1) + W_4^1 G_2(1) = 0 + (-j)(2) = -j2$$

$$F_1(2) = G_1(0) - W_4^0 G_2(0) = 0 - 1 \cdot (0) = 0$$

$$F_1(3) = G_1(1) - W_4^1 G_2(1) = 0 - (-j)(2) = +j2$$

$$F_2(0) = H_1(0) + W_4^0 H_2(0) = 1 + 1(-1) = 0$$

$$F_2(1) = H_1(1) + W_4^1 H_2(1)$$

$$= 1.4 + (-j)(1.4) = 1.4 - j1.4$$

$$F_2(2) = H_1(0) - W_4^0 H_2(0)$$

$$= 1 - 1(-1) = 2$$

$$F_2(3) = H_1(1) - W_4^1 H_2(1)$$

$$= 1.4 - (-j)(1.4) = 1.4 + j1.4$$

Stage 3

$$X(0) = F_1(0) + W_8^0 F_2(0) = 0 + 1 \cdot (0) = 0$$

$$X(1) = F_1(1) + W_8^1 F_2(1)$$

$$= -j2 + (0.707 - j0.707)(1.4 - j1.4) = -j3.98$$

$$X(2) = F_1(2) + W_8^2 F_2(2) = 0 + (-j)(2) = -j2$$

$$X(3) = F_1(3) + W_8^3 F_2(3)$$

$$= j2 + (-0.707 - j0.707)(1.4 + j1.4)$$

$$= j0.02$$

$$X(4) = F_1(0) - W_8^0 F_2(0) = 0 - 1(0) = 0$$

$$X(5) = F_1(1) - W_8^1 F_2(1)$$

$$= -j2 - (0.707 - j0.707)(1.4 - j1.4) = -j0.02$$

$$X(6) = F_1(2) - W_8^2 F_2(2) = 0 - (-j)(2) = j2$$

$$X(7) = F_1(3) - W_8^3 F_2(3)$$

$$= j2 - (-0.707 - j0.707)(1.4 + j1.4)$$

$$= j0.398$$

$$\therefore \quad X(k) = \{0, -j3.98, -2j, j0.02, 0 - j0.02, j2, j0.398\}$$

Ex. 4.4.6 : An eight point sequence x(n)is given by $x_1(n) = \{1, 2, 3, 4, 5, 6, 7, 8\}$

(i) Find DFT of $x_1(n)$ using any of the FFT technique

(ii) Let $x_2(n) = \{5, 6, 7, 8, 1, 2, 3, 4\}$

Using appropriate DFT property and answer of earlier part determine $X_2(k)$

(iii) Again we DFT property and Find $x_3(n) = x_1(n) + x_2(n)$

Soln. :

(i) Computing the DFT of $X_1(n)$:

We use the DIT-FFT. Since N = 8, we draw the 8-point DIT-FFT Butterfly diagram. The input is in the bit reversed order.

$$x_1(n) = \{ 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \}$$
$$x(0)x(1)x(2)x(3)x(4)x(5)x(6)x(7)$$

Here, $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi 1}{4}} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi \cdot 1}{8}} = \cos\frac{\pi}{4} - j \sin\frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi \cdot 2}{8}} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi \cdot 3}{8}} = \cos\frac{3\pi}{4} - j \sin\frac{3\pi}{4}$$

$$= -0.707 - j\,0.707$$

We write down the equations at each stage and calculate the outputs.

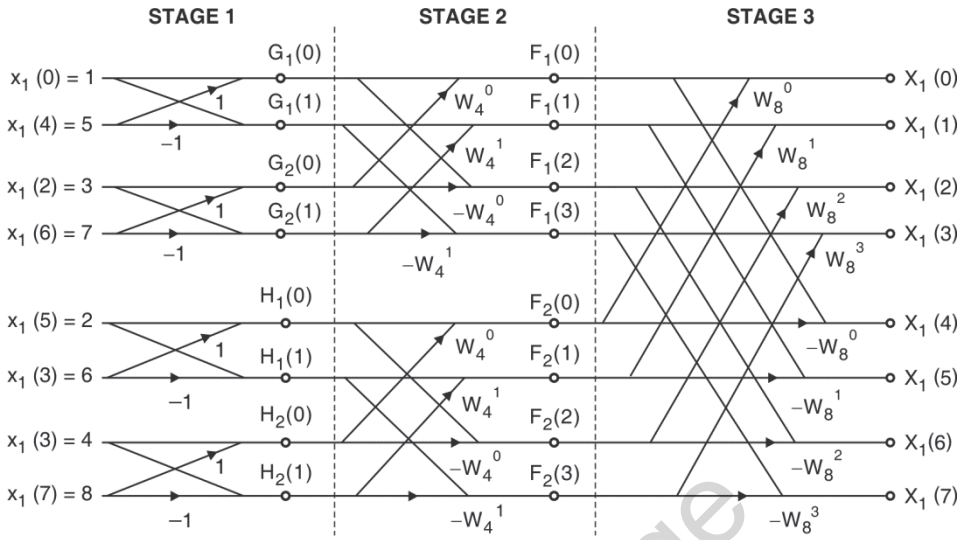


Fig. P. 4.4.6

Stage 1

$G_1(0) = x_1(0) + x_1(4) = 1 + 5 = 6$

$G_1(1) = x_1(0) - x_1(4) = 1 - 5 = -4$

$G_2(0) = x_1(2) + x_1(6) = 3 + 7 = 10$

$G_2(1) = x_1(2) - x_1(6) = 3 - 7 = -4$

$H_1(0) = x_1(1) + x_1(5) = 2 + 6 = 8$

$H_1(1) = x_1(1) - x_1(5) = 2 - 6 = -4$

$H_2(0) = x_1(3) + x_1(7) = 4 + 8 = 12$

$H_2(1) = x_1(3) - x_1(7) = 4 - 8 = -12$

Stage 2

$F_1(0) = G_1(0) + W_4^0 G_2(0) = 6 + 1 \cdot (10) = 16$

$F_1(1) = G_1(1) + W_4^1 G_2(1) = -4 + (-j)(-4) = -4 + j4$

$F_1(2) = G_1(0) - W_4^0 G_2(0) = 6 - 1 \cdot (10) = -4$

$F_1(3) = G_1(1) - W_4^1 G_2(1) = -4 - (-j)(-4) = -4 - j4$

$F_2(0) = H_1(0) + W_4^0 H_2(0) = 8 + 1 \cdot (12) = 20$

$F_2(1) = H_1(1) + W_4^1 H_2(1) = -4 + (-j)(-12) = -4 + j12$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 8 - 1 \cdot (12) = -4$

$F_2(3) = H_1(1) - W_4^1 H_2(1) = -4 - (-j)(-12) = -4 - j12$

Stage 3

$X_1(0) = F_1(0) + W_8^0 F_2(0) = 16 + 1 \cdot (20) = 36$

$X_1(1) = F_1(1) + W_8^1 F_2(1)$

$= (-4 + j4) + (0.707 - j0.707)(-4 + j12)$

$X_1(2) = F_1(2) + W_8^2 F_2(2) = -4 + (-j)(-4) = -4 + j4$

$X_1(3) = F_1(3) + W_8^3 F_2(3)$

$= (-4 + j4) + (-0.707 - j0.707)(-4 + j12)$

$X_1(4) = F_1(0) - W_8^0 F_2(0) = 16 - 1 \cdot (20) = -4$

$X_1(5) = F_1(1) - W_8^1 F_2(1)$

$= (-4 + j4) - (0.707 - j0.707)(-4 + j12)$

$X_1(6) = F_1(2) - W_8^2 F_2(2) = -4 - (-j)(-4) = -4 - j4$

$X_1(7) = F_1(3) - W_8^3 F_2(3)$

$= (-4 - j12) - (-0.707 - j0.707)(-4 - j12)$

$X(k) = \{36, -4 + j9.65, -4+j4, -4 + j1.65, -4,$

$-4 - j1.65, -4 - j4, -4 - j9.656\}$

(ii) Compute the DFT of $x_2(n)$ using appropriate DFT property.

Here, $x_1(n) = \{1, 2, 3, 4, 5, 6, 7, 8\}$

and $x_2(n) = \{5, 6, 7, 8, 1, 2, 3, 4\}$

Imagine $x_1(n)$ and $x_2(n)$ to lie on 2 circles.

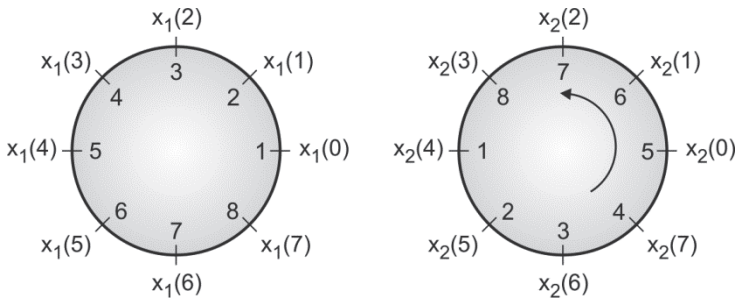


Fig. P. 4.4.6(a)

If we rotate the $x_2(n)$ circle in the clockwise direction by 4 steps, we would get the $x_1(n)$.circle.

Hence, $x_2(n)$ is a delayed version of $x_1(n)$.

$$\therefore \quad x_2(n) = x_1((n-4))$$

From the time shifting property of the DFT we know,

If $x(n) \xleftrightarrow[N]{DFT} X(k)$

Then $x((n-m))_N \xleftrightarrow[N]{DFT} e^{\frac{-j2\pi km}{N}} X(k)$

i.e. $X_2(k) = DFT\{x_2(n)\} = DFT\{x_1((n-4))\}$

$$= e^{\frac{-j2\pi km}{N}} X_1(k)$$

Since the delay is of 4 units, m = 4, N = 8.

$\therefore$ We can write

$$X_2(k) = e^{\frac{-j2\pi 4 \cdot k}{8}} X_1(k)$$

$$\therefore \quad X_2(k) = X_1(k)\, e^{-j\pi k}$$

We have calculated $X_1(k)$ in the previous part of the example.

We vary k from 0 to 7

$X_2(0) = e^{-j\pi \cdot 0} X_1(0) = (1) \cdot X_1(0) = (1).36 = 36$

$X_2(1) = e^{-j\pi \cdot 1} X_1(1) = (-1) \cdot X_1(1)$

$= (-1)(-4 + j9.65) = 4 - j9.65$

$X_2(2) = e^{-j\pi \cdot 2} X_1(2) = (1) X_1(2) = (1)(-4 + j4)$

$= -4 + j4$

$X_2(3) = e^{-j\pi \cdot 3} X_1(3) = (-1) X_1(3)$

$= (-1)(-4 + j1.65) = 4 - j1.65$

$X_2(4) = e^{-j\pi \cdot 4} X_1(4) = (1) X_1(4) = (1)(-4) = -4$

$X_2(5) = e^{-j\pi \cdot 5} X_1(5) = (1) X_1(5) = (-1)(-4 - j\,1.65)$

$= 4 + j1.65$

$X_2(6) = e^{-j\pi \cdot 6} X_1(6) = (1) X_1(6) = (1)(-4 - j4)$

$= -4 - j\,4$

$X_2(7) = e^{-j\pi \cdot 7} X_1(7) = (-1) X_1(7) = (-1)(-4 - j9.65)$

$= 4 + j9.65$

$\therefore \quad X_2(k) = \{36, 4 - j9.65, -4 + j4, 4 - j1.65, -4, 4 + j1.65, -4 - j4, 4 + j\,9.65\}$

(iii) Compute $X_3(k)$ if $x_3(n) = x_1(n) + x_2(n)$.

We use the linearity property which is as follows then

If $x(n) \xleftrightarrow{DFT} X(k)$

$$a x_1(n) + b\, x_2(n) \xleftrightarrow{DFT} a\, X_1(k) + b\, X_2(k)$$

Since, $x_3(n) = x_1(n) + x_2(n)$

$X_3(k) = DFT\{x_3(n)\} = DFT\{x_1(n) + x_2(n)\}$

$\therefore \quad X_3(k) = X_1(k) + X_2(k)$

$X_3(0) = X_1(0) + X_2(0) = 36 + 36 = 72$

$X_3(1) = X_1(1) + X_2(1) = (-4 + j9.65) + (-4 - j\,9.65)$

$= 0$

$X_3(2) = X_1(2) + X_2(2) = (-4 + j4) + (-4 - j\,4)$

$= -8 + j8$

$X_3(3) = X_1(3) + X_2(3) = (-4 + j\,1.65) + (4 - j1.65) = 0$

$X_3(4) = X_1(4) + X_2(4) = (-4) + (-4) = -8$

$X_3(5) = X_1(5) + X_2(5) = (-4 - j\,1.65) + (4 + j\,1.65) = 0$

$X_3(6) = X_1(6) + X_2(6) = (-4 - j4) + (-4 - j4) = -8 - j8$

$X_3(7) = X_1(7) + X_2(7) = (-4 - j9.65) + (4 + j9.65) = 0$

$\therefore \quad X_3(k) = \{72, 0, -8 + j8, 0, -8, 0, -8 - j8, 0\}$

Ex. 4.4.7 : Compute the DFT of the sequence X(0) ={1, 2, 3, 4}.Use DIT-FFT algorithm.

Soln. : Since the length of the sequence is 4, N = 4.We use the 4 - point DIT-FFT Butterfly diagram. The input is in the bit reversed order,

$$W_4^0 = e^{\frac{j2\pi}{4} \cdot 0} = 1$$

$$W_4^1 = e^{\frac{-j\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

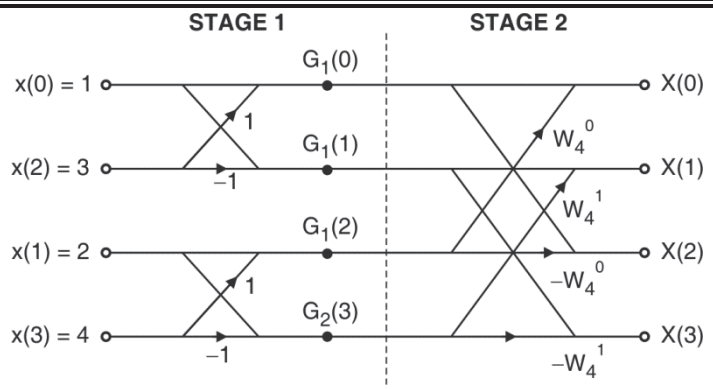


Fig. P. 4.4.7

Stage 1

$G_1(0) = x(0) + x(2) = 1 + 3 = 4$

$G_1(1) = x(0) - x(2) = 1 - 3 = -2$

$G_1(2) = x(1) + x(3) = 2 + 4 = 6$

$G_1(3) = x(1) - x(3) = 2 - 4 = -2$

Stage 2

$X(0) = G_1(0) + W_4^0 G_1(2) = 4 + 1(6) = 10$

$X(1) = G_1(1) + W_4^1 G_1(3) = -2 + (-j)(-2)$

$= -2 + j2$

$X(2) = G_1(0) - W_4^0 G_1(2) = 4 - 1(6) = -2$

$X(3) = G_1(1) - W_4^1 G_1(3) = -2 - (-j)(-2)$

$= -2 - j2$

$X(k) = \{10, -2 + j2, -2, -2 - j2\}$

All along we have observed that in DIT-FFT, the input gets reordered and the output is in order.

Ex. 4.4.8 : Compute 4-point DFT of the sequence given by $x(n) = (-1)^n$ using DIT-FFT algorithms.

Soln. Since N = 4, we vary n from 0 to 3

$\therefore x(n) = (-1)^n$

$\therefore x(0) = (-1)^0 = 1$

$\therefore x(1) = (-1)^1 = -1$

$\therefore x(2) = (-1)^2 = 1$

$\therefore x(3) = (-1)^3 = 1$

$\therefore x(n) = \{1, 2, 4, 8, 16, 32, 64, 128\}$

$x(n) = \{ 1 , -1 , 1 , -1 \}$

$x(0)\ \ x(1)\ \ x(2)\ \ x(3)$

The twiddle factors required are

$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$

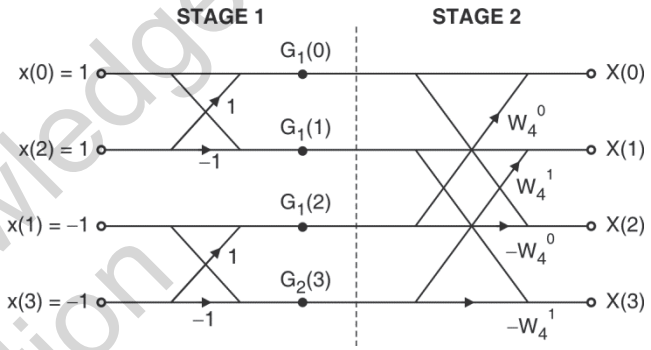


Fig. P. 4.4.8

Stage 1

$G_1(0) = x(0) + x(2) = 1 + 1 = 2$

$G_1(1) = x(0) - x(2) = 1 - 1 = 0$

$G_1(2) = x(1) + x(3) = -1 - 1 = -2$

$G_1(3) = x(1) - x(3) = -1 - (-1) = 0$

Stage 2

$X(0) = G_1(0) + W_4^0 G_1(2) = 2 + (1)(-2) = 0$

$X(1) = G_1(1) + W_4^1 G_1(3) = 0 + (-j)(0) = 0$

$X(2) = G_1(0) + W_4^0 G_1(2) = 2 - (1)(-2) = 4$

$X(3) = G_1(0) + W_4^1 G_1(3) = 2 - (-j)(0) = 0$

$\therefore X(k) = \{0, 0, 4, 0\}$

Ex. 4.4.9 : Compute 4-point DFT of the sequence x(n) = {1, 2, 3, 1} using DIT – FFt Radix–2 algorithms.

Soln. :

Here, N = 4

$x(n) = \{ 1 , 2 , 3 , 1 \}$

$x(0)\ \ x(1)\ \ x(2)\ \ x(3)$

The twiddle factors required are

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

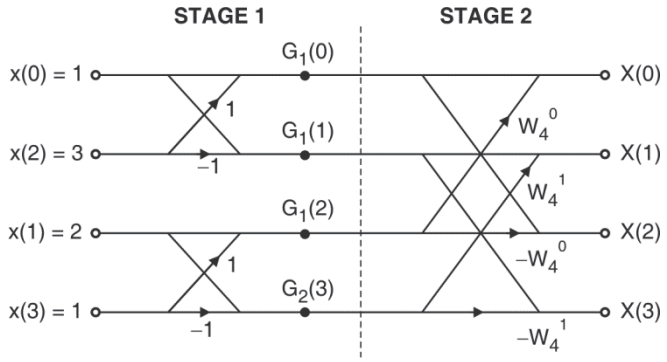


Fig. P. 4.4.9

Stage 1

$$G_1(0) = x(0) + x(2) = 1 + 3 = 4$$

$$G_1(1) = x(0) - x(2) = 1 - 3 = -2$$

$$G_1(2) = x(1) + x(3) = 2 + 1 = 3$$

$$G_1(3) = x(1) - x(3) = 2 - 1 = 1$$

Stage 2

$$X(0) = G_1(0) + W_4^0 G_1(2) = 4 + (1)(3) = 7$$

$$X(1) = G_1(1) + W_4^1 G_1(3) = -2 + (-j)(1) = -2 - j$$

$$X(2) = G_1(0) - W_4^0 G_1(2) = 4 - (1)(3) = 1$$

$$X(3) = G_1(1) - W_4^1 G_1(3) = -2 - (-j)(1) = -2 + j$$

$$\therefore X(k) = \{7, -2-j, 1, -2+j\}$$

Ex. 4.4.10 : Given x(n) = [0, 1, 2, 3].

Compute X(k) using DIT – FFT.

Soln. :

Here, N = 4

$$x(n) = \{\underset{x(0)}{0}, \underset{x(1)}{1}, \underset{x(2)}{2}, \underset{x(3)}{3}\}$$

The twiddle factors required are

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

We draw a 4-points DIT – FFT Butterfly

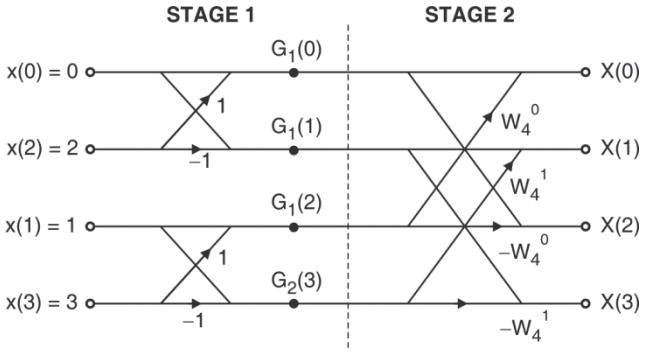


Fig. P. 4.4.10

Stage 1

$$G_1(0) = x(0) + x(2) = 0 + 2 = 2$$

$$G_1(1) = x(0) - x(2) = 0 - 2 = -2$$

$$G_1(2) = x(2) + x(3) = 1 + 3 = 4$$

$$G_1(3) = x(1) - x(3) = 1 - 3 = -2$$

Stage 2

$$X(0) = G_1(0) + W_4^0 G_1(2) = 2 + 1(4) = 6$$

$$X(1) = G_1(1) + W_4^1 G_1(3)$$

$$= -2 + (-j)(-2) = -2 + j2$$

$$X(2) = G_1(0) + W_4^0 G_1(2) = 2 - 1(4) = -2$$

$$X(3) = G_1(1) + W_4^1 G_1(3) = -2 - (-j)(-2) = -2 - j2$$

$$\therefore X(k) = \{6, -2+j2, -2, -2-j2\}$$

4.5 Decimation in Frequency FFT (DIT-FFT)

- DIF-FFT is another important FFT-algorithm. Decimation in frequency implies retaining the order of the input sequence but getting an output which is shuffled up.
- In DIF-FFT, we split up the input sequence into the first N/2 data points and the second N/2 data points (instead of odd and even as was the case in DIT-FFT).

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{nk} + \sum_{n'=\frac{N}{2}}^{N-1} x(n') W_N^{n'k}$$

In the second sum, put $n' = n + \frac{N}{2}$

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)\, W_N^{nk} + \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{(n+N/2)\,k}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)\, W_N^{nk} + \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{nk}\, W_N^{nk}\, W_N^{kN/2}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)\, W_N^{nk} + W_N^{kN/2} \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{nk}$$

Now

$$W_N^{kN/2} = (-1)^k$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)\, W_N^{nk} + (-1)^k \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{nk}$$

$$\therefore X(k) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + (-1)^k\, x\left(n+\frac{N}{2}\right)\right] W_N^{nk}$$

We now split X(k) into odd and even parts,

$$(2k) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + x\left(n+\frac{N}{2}\right)\right] W_N^{2nk} \quad (\because (-1)^{2k} = +1)$$

$$X(2k+1) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^{(2k+1)n}$$

$$(\because (-1)^{2k+1} = -1)$$

Now $W_N^2 = W_{N/2}^1$

$$\therefore X(2k) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + x\left(n+\frac{N}{2}\right)\right] W_{N/2}^{nk};$$

$$k = 0, 1, \ldots \frac{N}{2} - 1 \ldots (4.5.1)$$

$$\therefore X(2k+1) = \sum_{n=0}^{\frac{N}{2}-1} \left\{\left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n\right\} W_{N/2}^{nK}$$

$$; k = 0,1,2, \ldots \frac{N}{2} - 1 \ldots (4.5.2)$$

Let $g_1(n) = \left[x(n) + x\left(n+\frac{N}{2}\right)\right]$

and $g_2(n) = \left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n$

$$\therefore X(2k) = \sum_{n=0}^{\frac{N}{2}-1} g_1(n)\, W_{N/2}^{nk} \quad \ldots (4.5.3)$$

and $$X(2k+1) = \sum_{n=0}^{\frac{N}{2}-1} g_2(n)\, W_{N/2}^{nk} \quad \ldots (4.5.4)$$

We now draw the signal flow graph using Equations (4.5.1), (4.5.2), (4.5.3) and (4.5.4) using N = 8.

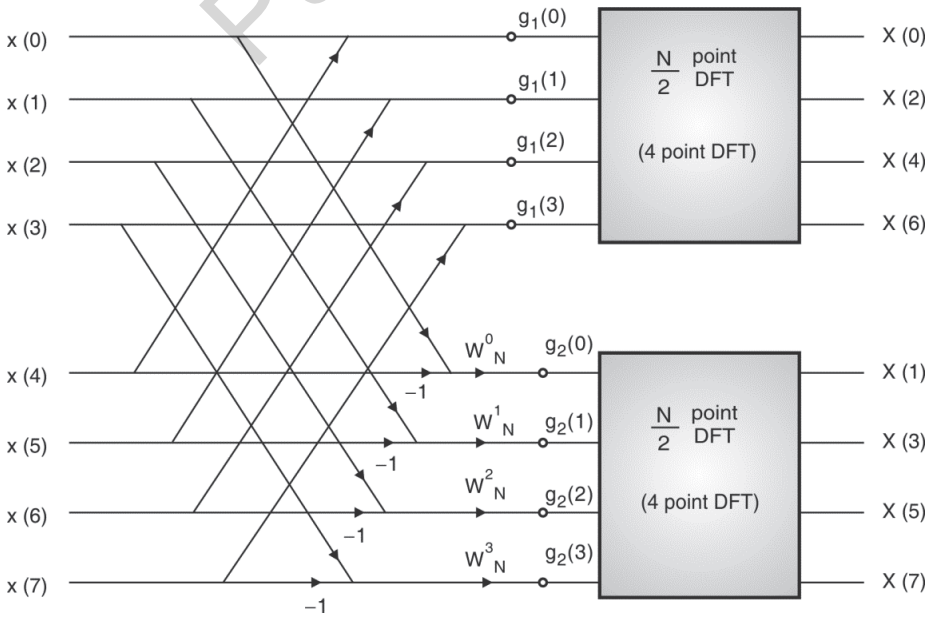


Fig. 4.5.1

As is evident, the order of the input is retained but the output is shuffled.

Equations (4.5.3) and (4.5.4) represent two N/2 point DFT's. Since N = 8, these two equations represent two 4-point DFT's.

Each of these can be further split up into two - 2 point DFT's. We will do just that,

$$X(2k) = \sum_{n=0}^{\frac{N}{2}-1} g_1(n)\, W_{N/2}^{nk}$$

Splitting X(2k) we get,

$$X(2k) = \sum_{n=0}^{\frac{N}{4}-1} g_1(n)\, W_{N/2}^{nk} + \sum_{n=0}^{\frac{N}{4}-1} g_1\left(n+\frac{N}{4}\right) W_{N/2}^{k\,(n+N/4)}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} g_1(n)\, W_{N/2}^{nk} + W_{N/2}^{kN/4} \sum_{n=0}^{\frac{N}{4}-1} g_1\left(n+\frac{N}{4}\right) W_{N/2}^{nk}$$

Now, $W_{N/2}^{kN/4} = (-1)^k$

$$X(2k) = \sum_{n=0}^{\frac{N}{4}-1} g_1(n)\, W_{N/2}^{nk} + (-1)^k \sum_{n=0}^{\frac{N}{4}-1} g_1\left(n+\frac{N}{4}\right) W_{N/2}^{nk}$$

$$X(2k) = \sum_{n=0}^{\frac{N}{4}-1} \left[g_1(n) + (-1)^k\, g_1\left(n+\frac{N}{2}\right) \right] W_{N/2}^{nk}$$

Splitting X(2k) into odd and even parts we get,

$$X(4k) = \sum_{n=0}^{\frac{N}{4}-1} \left[g_1(n) + g_1\left(n+\frac{N}{4}\right) \right] W_{N/4}^{nk};$$

$$k = 0, 1, \ldots \frac{N}{4} - 1 \qquad \ldots(4.5.5)$$

$$X(4k+2) = \sum_{n=0}^{\frac{N}{4}-1} \left[g_1(n) - g_1\left(n+\frac{N}{4}\right) \right] W_{N/2}^{n}\, W_{N/4}^{nk};$$

$$k = 0, 1, \ldots \frac{N}{4} - 1 \qquad \ldots(4.5.6)$$

We also split X(2K + 1) into two N/4 point DFTs. Using Equations (4.5.5) and (4.5.6), we draw the signal flow graph.

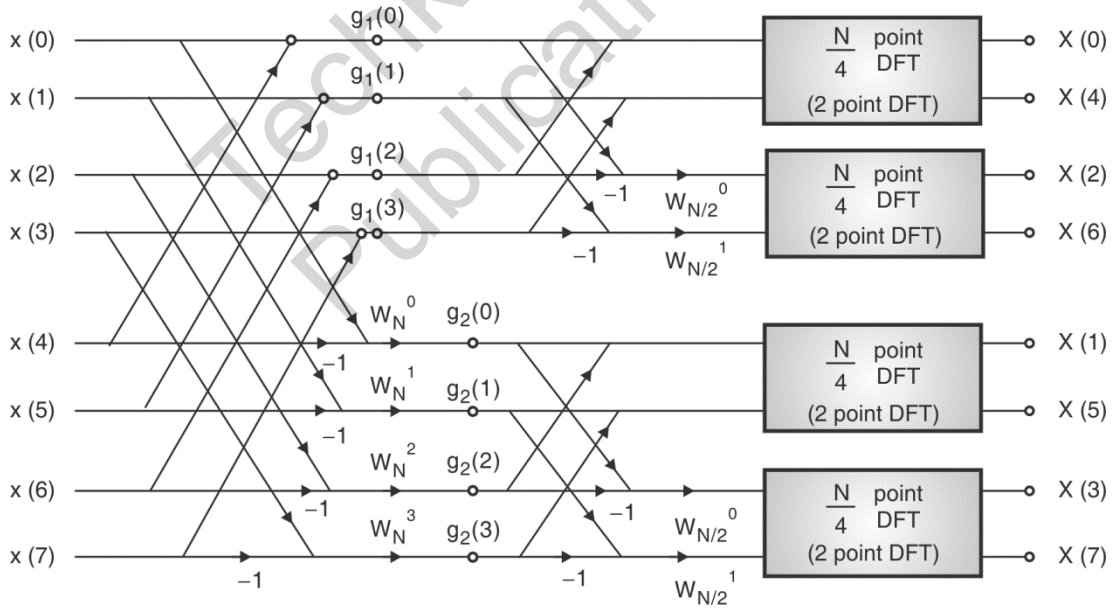


Fig. 4.5.2

A 2-point DFT can be represented as shown in Fig. 4.5.3.

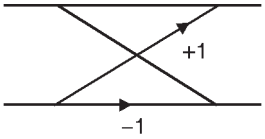


Fig. 4.5.3

The final 8-point DIF-FFT signal flow graph is shown in Fig. 4.5.4.

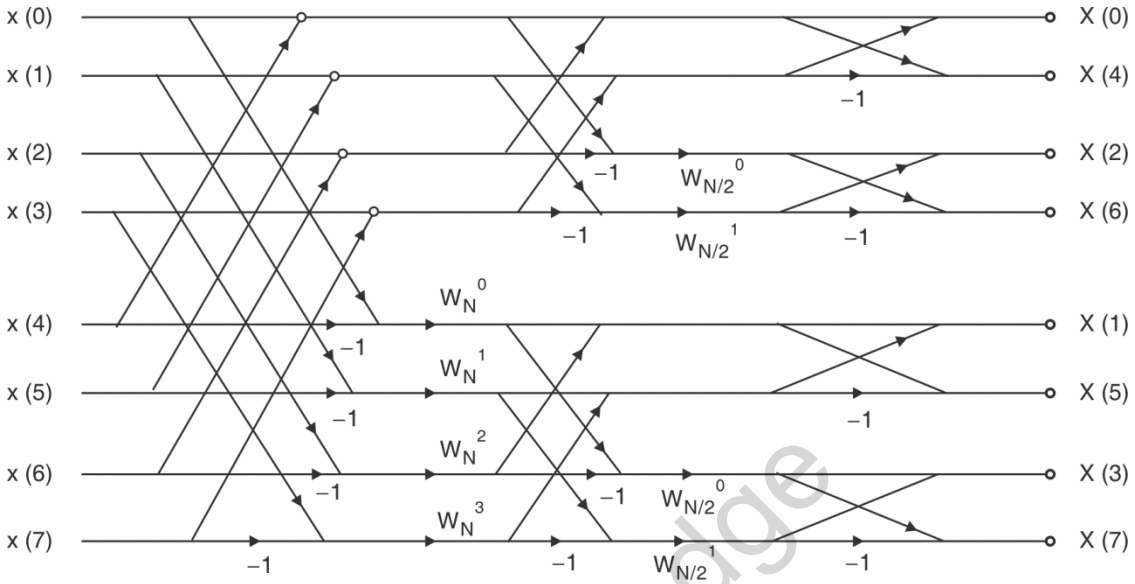


Fig. 4.5.4 : 8-point DIF-FFT Butterfly diagram

In the DIF-FFT, the output is in the bit reversed format while the input is unaltered.

The number of computations required by the DIT-FFT and the DIF-FFT are the same.

As stated earlier, FFT refers to a class of algorithms for efficiently computing the DFT. Hence FFT is not an approximation of the DFT. It is the DFT with a reduced number of computations.

4.5.1 Solved Examples on DIT-FFT

Ex. 4.5.1 : A 8-Point sequence x(n) is given by {1, 2, 3, 2, 1, 5, 2, 1} find the DFT of the sequence using DIF-FFT.

Soln. : x(n)={ 1, 2,3, 2, 1, 5, 2, 1}

Since N = 8 we draw a 8-point DIF-FFT Butterfly diagram. Remember the input is ordered but the output obtained is in bit reversed order.

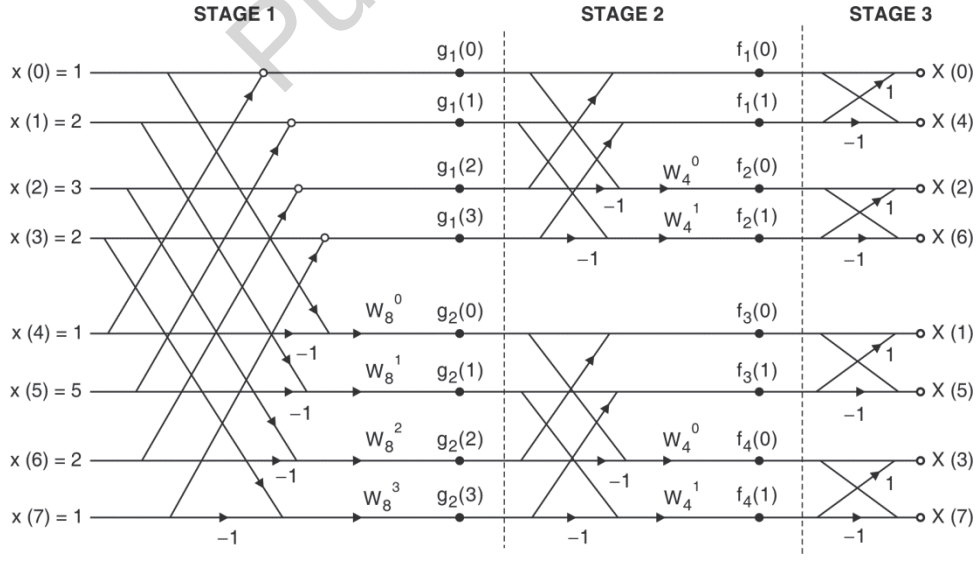


Fig. P. 4.5.1

Here $W_4^0 = e^{\frac{-j2\pi 0}{4}} = \cos 0 - j \sin 0 = 1$

$$W_4^1 = e^{\frac{-j2\pi 1}{4}} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = e^{\frac{-j2\pi}{8}\cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{8}\cdot 1} = \cos\frac{\pi}{4} - j \sin\frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j \sin\frac{3\pi}{4} = -0707 - j0.707$$

Stage 1

$g_1(0) = x(0) + x(4) = 1 + 1 = 2$

$g_1(1) = x(1) + x(5) = 2 + 5 = 7$

$g_1(2) = x(2) + x(6) = 3 + 2 = 5$

$g_1(3) = x(3) + x(7) = 2 + 1 = 3$

$g_2(0) = [x(0) - x(4)]\, W_8^0 = [1 - 1]\,(1) = 0$

$g_2(1) = [x(1) - x(5)]\, W_8^1$

$= [2 - 5]\,(0.707 - j0.707) = -2.12 + j2.12$

$g_2(2) = [x(2) - x(6)]\, W_8^2 = [3 - 2]\,(-j) = -j$

$g_2(3) = [x(3) - x(7)]\, W_8^3$

$= [2 - 1]\,(-0.707 - j0.707) = -0.707 - j0.707$

Stage 2

$f_1(0) = g_1(0) + g_1(2) = 2 + 5 = 7$

$f_1(1) = g_1(1) + g_1(3) = 7 + 3 = 10$

$f_2(0) = [g_1(0) - g_1(2)]\, W_4^0 = [2 - 5]\,(1) = -3$

$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1 = [7 - 3]\,(-j) = -4j$

$f_3(0) = g_2(0) + g_2(2) = 0 + (-j) = -j$

$f_3(1) = g_2(1) + g_2(3)$

$= (-2.12 + j2.12) + (-0.707 - j0.707)$

$= -2.827 + j1.413$

$f_4(0) = [g_2(0) - g_2(2)]W_4^0 = 0 - (-j)$

$= 0 - (-j)\,(1) = +j$

$f_4(1) = [g_2(1) - g_2(3)]W_4^1$

$= [(-2.12 + j2.12) - (-0.707 - j0.707)]\,(-j)$

$= 2.827 + j1.413$

Stage 3

$X(0) = f_1(0) + f_1(1) = 7 + 10 = 17$

$X(4) = f_1(0) - f_1(1) = 7 - 10 = -3$

$X(2) = f_2(0) + f_2(1) = -3 + (-4j) = -3 - 4j$

$X(6) = f_2(0) - f_2(1) = -3 - (-4j) = -3 + 4j$

$X(1) = f_3(0) + f_3(1) = (-j) + (-2.827 + j1.413) = -2.827 + j0.413$

$X(5) = f_3(0) - f_3(1) = (-j) - (-2.827 + j\,1.413) = 2.827 - j\,2.413$

$X(3) = f_4(0) + f_4(1) = j + (2.827 + j1.413)$

$= 2.827 + 2.413$

$X(7) = f_4(0) - f_4(1) = j - (2827 + j1413)$

$= 2.827 - j0.413$

We now rearrange the output in the proper order

$\therefore$ $X(k) = \{17, -2.827 + j\,0.413, -3 - 4j, 2.827 + j2.413, -3, 2.827 - j2.413, -3 + 4j, -2.827 - j0.413\}$

Ex. 4.5.2 : Find 8-point DFT using Radix – 2 – DIF FFT algorithms for a given sequences

x(n) = {–1, 0, 2, 0, – 4, 0, 2, 0}

Soln. :

$x(n) = \{-1, 0, 2, 0, -4, 0, 2, 0\}$

x(0) x(1) x(2) x(3) x(4) x(5) x(6) x(7)

We draw a 8-point DIF-FFT Butterfly diagram,

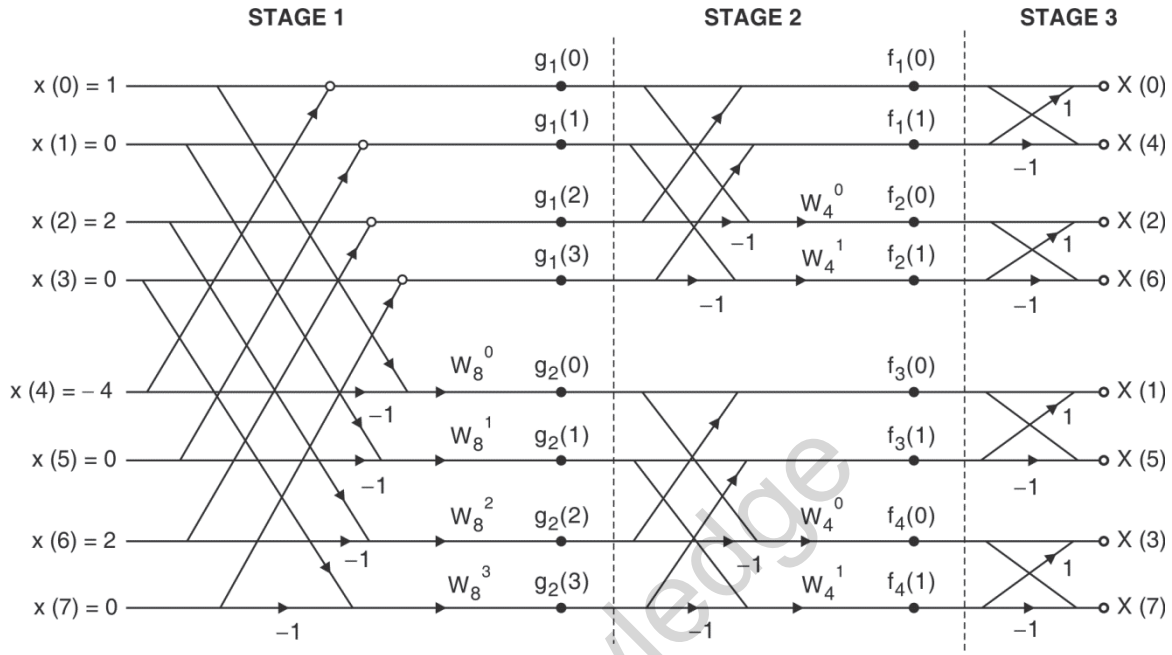


Fig. P. 4.5.2

Here,

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos \frac{\pi}{4} - j \sin \frac{\pi}{4} = 0.707 - j0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8} \cdot 2} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8} \cdot 3} = \cos \frac{3\pi}{4} - j \sin \frac{3\pi}{4} = -0.707 - j\,0.707$$

Stage 1

$g_1(0) = x(0) + x(4) = -1 + (-4) = -5$

$g_1(1) = x(1) + x(5) = 0 + 0 = 0$

$g_1(2) = x(2) + x(6) = 2 + 2 = 4$

$g_1(3) = x(3) + x(7) = 0 + 0 = 0$

$g_2(0) = [x(0) - x(4)]\, W_8^0 = [-1 - (-4)]\,(1) = 3$

$g_2(1) = [x(1) - x(5)]\, W_8^1$

$= [0 - 0]\,(0.707 - j0.707) = 0$

$g_2(2) = [x(2) - x(6)]\, W_8^2 = [2 - 2]\,(-j) = 0$

$g_2(3) = [x(3) - x(7)]\, W_8^3 = [0 - 0]\,(-0.707 - j0.707) = 0$

Stage 2

$f_1(0) = g_1(0) + g_1(2) = -5 + 4 = -1$

$f_1(1) = g_1(1) + g_1(3) = 0 + 0 = 0$

$f_2(0) = [g_1(0) - g_1(2)]\, W_4^0 = [-5 - 4]\,(1) = -9$

$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1 = [0 - 0]\,(-j) = 0$

$f_3(0) = g_2(0) + g_2(2) = 3 + 0 = 3$

$f_3(1) = g_2(1) + g_2(3) = 0 + 0 = 0$

$f_4(0) = [g_2(0) - g_2(2)]\, W_4^0 = [3 - 0]\,(1) = 3$

$f_4(1) = [g_2(1) - g_2(3)]\, W_4^1 = [0 - 0]\,(-j) = 0$

Stage 3

$X(0) = f_1(0) + f_1(1) = -1 + 0 = -1$

$X(4) = f_1(0) - f_1(1) = -1 - 0 = -1$

$X(2) = f_2(0) + f_2(1) = -9 + 0 = -9$

$X(6) = f_2(0) - f_2(1) = -9 - 0 = -9$

$X(1) = f_3(0) + f_3(1) = 3 + 0 = 3$

$X(5) = f_3(0) - f_3(1) = 3 - 0 = 3$

$X(3) = f_4(0) + f_4(1) = 3 + 0 = 3$

$X(7) = f_4(0) - f_4(1) = 3 - 0 = 3$

We now arrange the output in the proper order

$\therefore \quad X(k) = \{-1, 3, -9, 3, -1, 3, -9, 3\}$

Ex. 4.5.3 : Given x(n) = n + 1 and N = 8, find DFT X(k) using DIF-FFT algorithm.

Soln. :

Given : $x(n) = n + 1 ;\ 0 \le n \le 7 \quad (\because N = 8)$

$x(n) = \{1, 2, 3, 4, 5, 6, 7, 8\}$

We draw a 8 – point DIF-FFT Butterfly diagram. In DIF-FFT, the input is ordered but the output obtained is in bit reversed order.

Here $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$

Similarly, $W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$

$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos \frac{\pi}{4} - j \sin \frac{\pi}{4} = 0.707 - j\,0.707$

$W_8^2 = e^{\frac{-j2\pi}{8} \cdot 2} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$

$W_8^3 = e^{\frac{-j2\pi}{8} \cdot 3} = \cos \frac{3\pi}{4} - j \sin \frac{3\pi}{4} = -0.707 - j\,0.707$

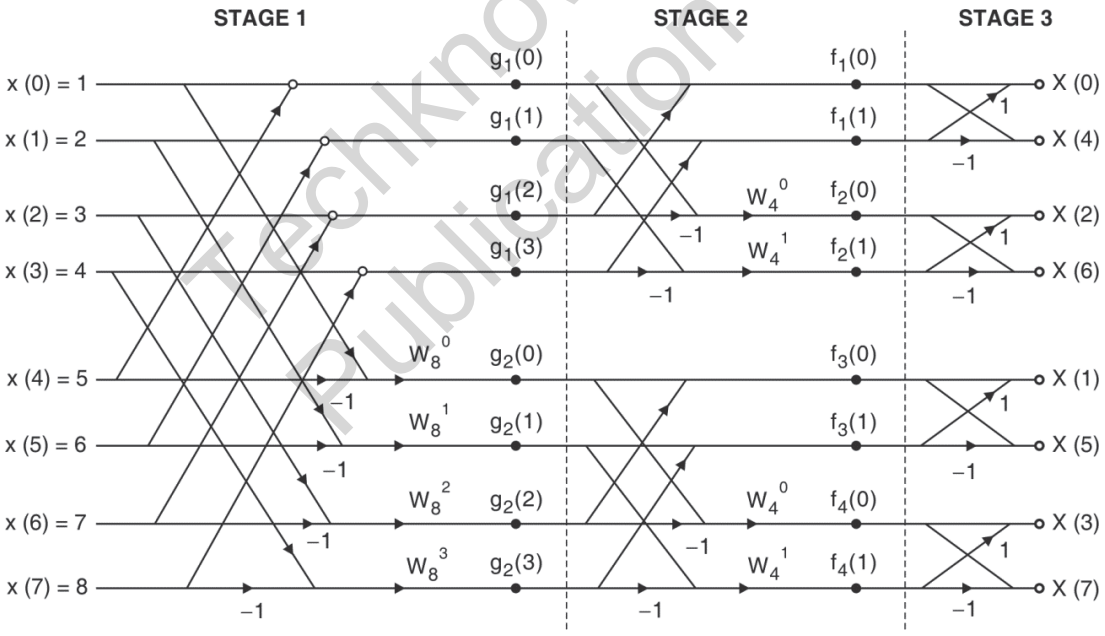


Fig. P. 4.5.3

Stage 1:

$g_1(0) = x(0) + x(4) = 1 + 5 = 6$

$g_1(1) = x(1) + x(5) = 2 + 6 = 8$

$g_1(2) = x(2) + x(6) = 3 + 7 = 10$

$g_1(3) = x(3) + x(7) = 4 + 8 = 12$

$g_2(0) = [x(0) - x(4)]\, W_8^0 = [1 - 5]\,(1) = -4$

$g_2(1) = [x(1) - x(5)]\, W_8^1 = [2 - 6]\,(0.707 - j0.707)$

$= -2.83 + j2.83$

$g_2(0) = [x(2) - x(6)]\, W_8^2 = [3 - 7]\,(-j) = +4j$

$g_2(3) = [x(3) - x(7)]\, W_8^3 = [4 - 8]\,(-0.707 - j0.707)$

$= 2.83 + j2.83$

Stage 2 :

$f_1(0) = g_1(0) + g_1(2) = 6 + 10 = 16$

$f_1(1) = g_1(1) + g_1(3) = 8 + 12 = 20$

$f_2(0) = [g_1(0) - g_1(2)]\ W_4^0 = [6 - 10]\ (1) = -4$

$f_2(1) = [g_1(1) - g_1(3)]\ W_4^1 = [8 - 12]\ (-j) = +4j$

$f_3(0) = g_2(0) + g_2(2) = -4 + 4j$

$f_3(1) = g_2(1) + g_2(3)$

$= (-2.83 + j2.83) + (2.83 + j2.83)$

$= j5.66$

$f_4(0) = [g_2(0) - g_2(2)]W_4^0 = [-4 - (+4j)]$

$= -4 - 4j$

$f_4(1) = [g_2(1) - g_2(3)]W_4^1$

$= (-2.83 + j2.83)\ (2.83 + j2.83)\ (-j) = j5.66$

Stage 3

$X(0) = f_1(0) + f_1(1) = 16 + 20 = 36$

$X(4) = f_1(0) - f_1(1) = 16 - 20 = -4$

$X(2) = f_2(0) + f_2(1) = -4 + 4j$

$X(6) = f_2(0) - f_2(1) = -4 - 4j$

$X(1) = f_3(0) + f_3(1) = (-4 + 4j) + (j5.66)$

$= -4 + j9.66$

$X(5) = f_3(0) - f_3(1) = (-4 + 4j) - (j5.66)$

$= -4 - j1.66$

$X(3) = f_4(0) + f_4(1) = (-4 - 4j) + (j5.66)$

$= -4 + j1.66$

$X(7) = f_4(0) - f_4(1) = (-4 - 4j) - (j5.66)$

$= -4 - j9.66$

We now rearrange the output in the proper order

$\therefore$ $X(k) = \{36, -4 + j9.66, -4 + 4j, -4 + 1.66j, -4, -4 - j1.66, -4 - 4j, -4 - j9.66\}$

Ex. 4.5.4 : Compute DFT of sequence x(n) = {1, 2, 2, 2, 1, 0, 0, 0} using DIF-FFT algorithm. Sketch its magnitude spectrum.

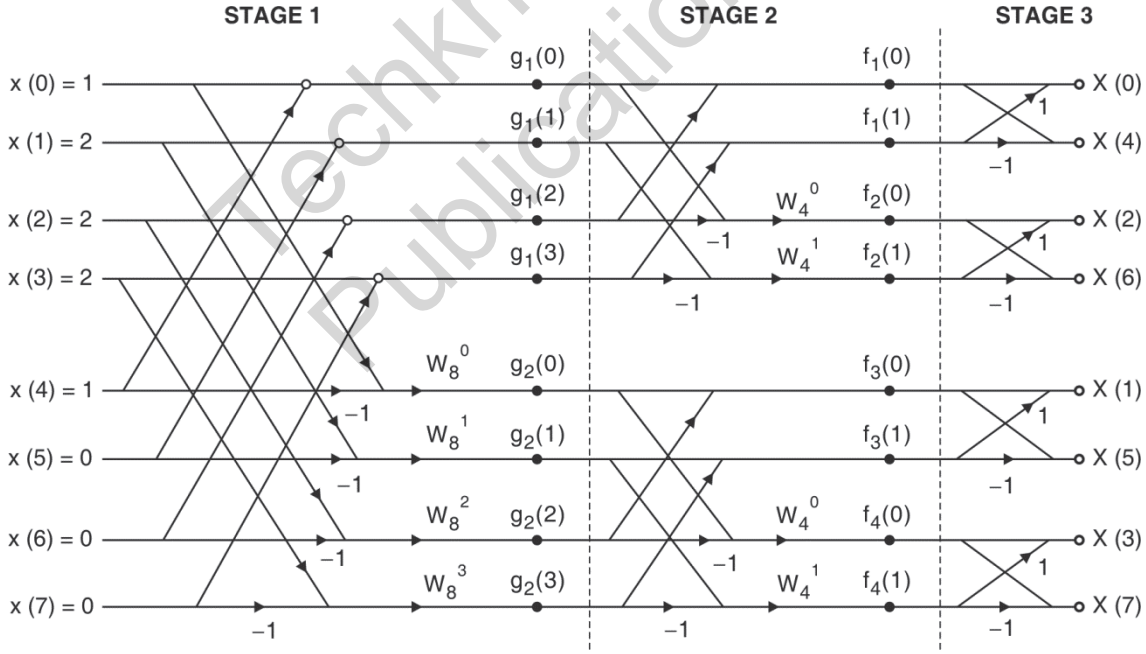


Fig. P. 4.5.4

Soln. :

$x(n) = \{1, 2, 2, 2, 1, 0, 0, 0\}$

We draw a 8-point DIF-FFT Butterfly diagram in DIF-FFT, the input is ordered but the output obtained in bit reversed order.

Here,

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{4} - j \sin\frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{4} \cdot 2} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{4} \cdot 3} = \cos\frac{3\pi}{4} - j \sin\frac{3\pi}{4} = -0.707 - j0.707$$

Stage 1

$$g_1(0) = x(0) + x(4) = 1 + 1 = 2$$

$$g_1(1) = x(1) + x(5) = 2 + 0 = 2$$

$$g_1(2) = x(2) + x(6) = 2 + 0 = 2$$

$$g_1(3) = x(3) + x(7) = 2 + 0 = 2$$

$$g_2(0) = [x(0) - x(4)]\, W_8^0 = [1 - 1]\,(1) = 0$$

$$g_2(1) = [x(1) - x(5)]\, W_8^1 = [2 - 0]\,(0.707 - j0.707)$$

$$= 1.414 - j1.414$$

$$g_2(2) = [x(2) - x(6)]\, W_8^2 = [2 - 0]\,(-j) = -2j$$

$$g_2(3) = [x(3) - x(7)]\, W_8^3 = [2 - 0]\,(-0.707 - j0.707)$$

$$= 1.414 - j1.414$$

Stage 2

$$f_1(0) = g_1(0) + g_1(2) = 2 + 2 = 4$$

$$f_1(1) = g_1(1) + g_1(3) = 2 + 2 = 4$$

$$f_2(0) = [g_1(0) - g_1(2)]\, W_4^0 = [2 - 2]\,(1) = 0$$

$$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1 = [2 - 2]\,(-j) = 0$$

$$f_3(0) = g_2(0) + g_2(2) = 0 + (-j2) = -j2$$

$$f_3(1) = g_2(1) + g_2(3)$$

$$= (1.414 - j1.414) + (-1.414 - j1.414)$$

$$= -j2.83$$

$$f_4(0) = [g_2(0) - g_2(2)]\, W_4^0 = 0 - (-j2) = +j2$$

$$f_4(1) = [g_2(1) - g_2(3)]\, W_4^1$$

$$= (1.414 - j1.414) - (-1.414 - j1.414)\,(-j)$$

$$= -j2.83$$

Stage 3

$$X(0) = f_1(0) + f_1(1) = 4 + 4 = 8$$

$$X(4) = f_1(0) - f_1(1) = 4 - 4 = 0$$

$$X(2) = f_2(0) + f_2(1) = 0 + 0 = 0$$

$$X(6) = f_2(0) - f_2(1) = 0 - 0 = 0$$

$$X(1) = f_3(0) + f_3(1) = -j2 + (-j2.83) = -j4.83$$

$$X(5) = f_3(0) - f_3(1) = -j2 - (-j2.83) = +j0.83$$

$$X(3) = f_4(0) + f_4(1) = j2 + (-j2.83) = -j0.83$$

$$X(7) = f_4(0) - f_4(1) = j2 - (-j2.83) = j4.83$$

We now rearrange the output in the proper order.

$$X(k) = \{8, -j4.83, 0, -j0.83, 0, +j0.83, 0, j4.83\}$$

To sketch the magnitude spectrum. We compute the magnitude of the Fourier coefficient X(k)

$\therefore$ $|X(k)| = \{8, 4.83, 0, 0.83, 0, 0.83, 0, 4.83\}$

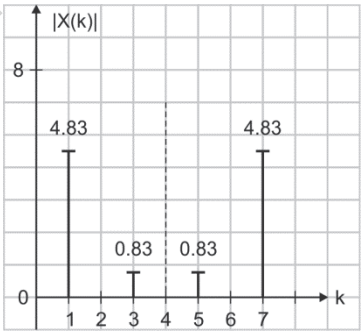


Fig. P. 4.5.4(a)

We observer that the magnitude spectrum is symmetric about the $\frac{N}{2}$ point, in this case 4.

Ex. 4.5.5 : N = 8 point DFT using Radix FFT for the given sequence x(n) = {1, 1, 1, 1, 0, 0, 0, 0}

Soln. :

$$x(n) = \{\ 1\ ,\ 1\ ,\ 1\ ,\ 1\ ,\ 0\ ,\ 0\ ,\ 0\ ,\ 0\ \}$$

$$x(0)\ \ x(1)\ \ x(2)\ \ x(3)\ \ x(4)\ \ x(5)\ \ x(6)\ \ x(7)$$

We draw a 8-point DIF-FFT Butterfly diagram

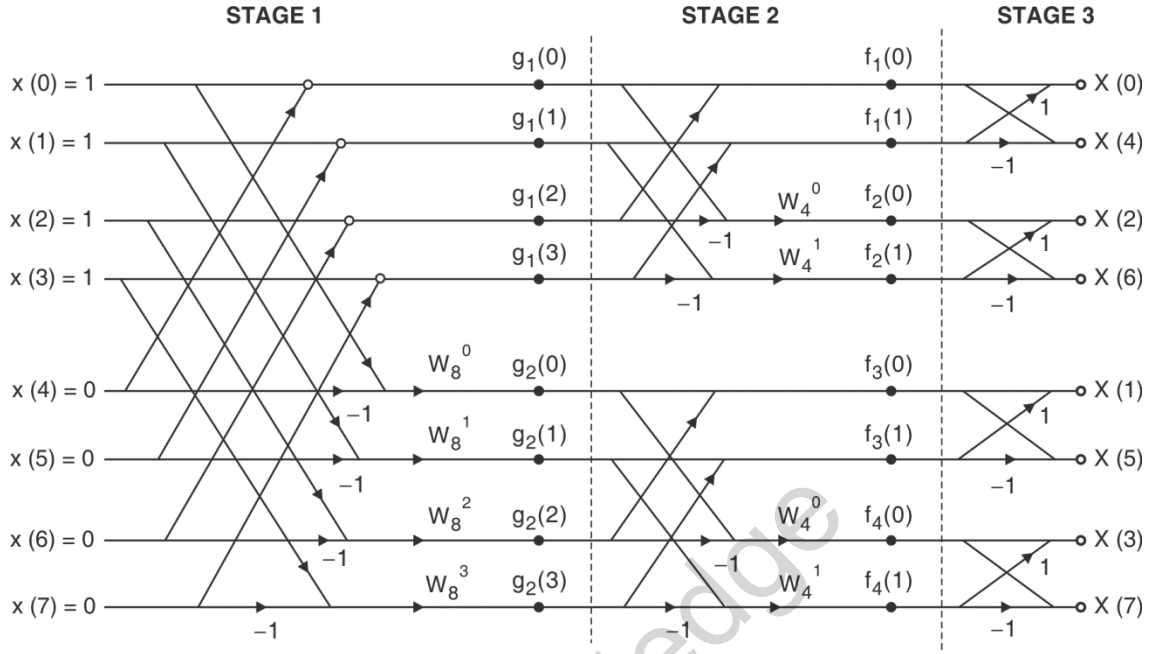


Fig. P. 4.5.5

Here,

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos \frac{\pi}{4} - j \sin \frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8} \cdot 2} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8} \cdot 3} = \cos \frac{3\pi}{4} - j \sin \frac{3\pi}{4} = -0.707 - j\,0.707$$

Stage 1

$g_1(0) = x(0) + x(4) = 1 + 0 = 1$

$g_1(1) = x(1) + x(5) = 1 + 0 = 1$

$g_1(2) = x(2) + x(6) = 1 + 0 = 1$

$g_1(3) = x(3) + x(7) = 1 + 0 = 1$

$g_2(0) = [x(0) - x(4)]\, W_8^0 = [1 - 0]\,(0) = 1$

$g_2(1) = [x(1) - x(5)]\, W_8^1$

$= [1 - 0]\,(0.707 - j0.707)$

$= 0.707 - j0.707$

$g_2(2) = [x(2) - x(6)]\, W_8^2 = [1 - 0]\,(-j) = -j$

$g_2(3) = [x(3) - x(7)]\, W_8^3$

$= [1 - 0]\,(-0.707 - j0.707)$

$= -0.707 - j0.707$

Stage 2

$f_1(0) = g_1(0) + g_1(2) = 1 + 1 = 2$

$f_1(1) = g_1(1) + g_1(3) = 1 + 1 = 2$

$f_2(0) = [g_1(0) - g_1(2)]\, W_4^0 = [1 - 1]\,(1) = 0$

$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1 = [1 - 1]\,(-j) = 0$

$f_3(0) = g_2(0) + g_2(2) = 1 + (-j) = 1 - j$

$f_3(1) = g_2(1) + g_2(3)$

$= (0.707 - j0.707) + (-0.707 - j0.707)$

$= -j1.414$

$f_4(0) = [g_2(0) - g_2(2)]\, W_4^0 = [1 - (-j)]\,(1) = 1 + j$

$f_4(1) = [g_2(1) - g_2(3)]\, W_4^1$

$= [(0.707 - j0.707) - (-0.707 - j0.707)]\,(-j)$

$= -j1.414$

Stage 3

$X(0) = f_1(0) + f_1(1) = 2 + 2 = 4$

$X(4) = f_1(0) - f_1(1) = 2 - 2 = 0$

$X(2) = f_2(0) + f_2(1) = 0 + 0 = 0$

$X(6) = f_2(0) - f_2(1) = 0 - 0 = 0$

$X(1) = f_3(0) + f_3(1) = (1 - j) + (- j1.414) = 1 - j2.414$

$X(5) = f_3(0) - f_3(1) = (1 - j) - (- j0.414) = 1 + j0.414$

$X(3) = f_4(0) + f_4(1) = (1 + j) + (- j0.414) = 1 - j0.414$

$X(7) = f_4(0) - f_4(1) = (1 + j) - (1 - j1.414) = 1 + j2.414$

We now rearrange the output in the proper order

$\therefore \quad X(k) = \{4 , 1 - j2.414, 0, 1 - j0.414, 0, 1 + j0.414, 0, 1 + j2.414\}$

Ex. 4.5.6 : Using DIF - FFT Find a

x(n) = {1, 2, 1, 2, 4, 2, 1, 2}

If $x_1(n) = x(- n)$, find $x_1(k)$ using earlier solution and not otherwise.

Soln. : We draw a 8-point DIF-FFT Butterfly diagram. In DIF-FFT, the input is ordered but the output obtained is in bit reversed order.

Here $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = - j$

Similarly,

$W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$

$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos\frac{\pi}{4} - j \sin\frac{\pi}{4} = -0.707 - j\, 0.707$

$W_8^2 = e^{\frac{-j2\pi}{8} \cdot 2} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = - j$

$W_8^3 = e^{\frac{-j2\pi}{8} \cdot 3} = \cos\frac{3\pi}{4} - j \sin\frac{3\pi}{4} = - 0.707 - j0.707$

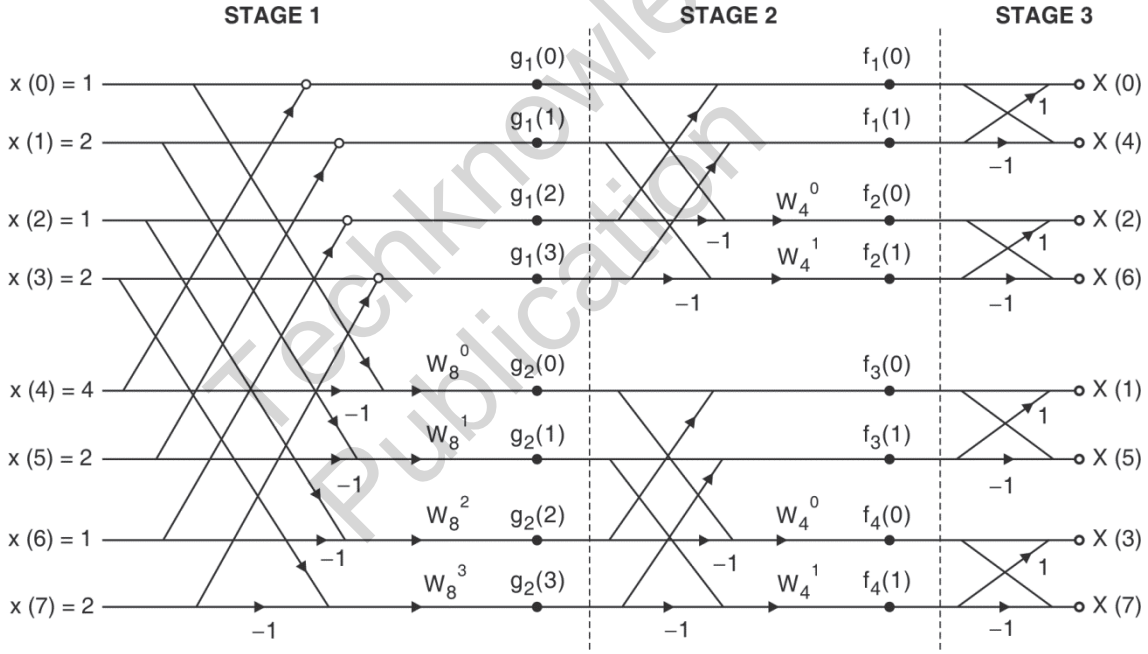


Fig. P. 4.5.6

Stage 1

$g_1(0) = x(0) + x(4) = 1 + 4 = 5$

$g_1(1) = x(1) + x(5) = 2 + 2 = 4$

$g_1(2) = x(2) + x(6) = 1 + 1 = 2$

$g_1(3) = x(3) + x(7) = 2 + 2 = 4$

$g_2(0) = [x(0) - x(4)]\, W_8^0 = [1 - 4]\, (1) = - 3$

$g_2(1) = [x(1) - x(5)]\, W_8^1 = [2 - 2]\, (0.707 - j0.707) = 0$

$g_2(2) = [x(2) - x(6)]\, W_8^2 = [1 - 1]\, (- j) = 0$

$g_2(3) = [x(3) - x(7)]\, W_8^3 = [2 - 2]\, (- 0.707 - j0.707) = 0$

Stage 2

$f_1(0) = g_1(0) + g_1(2) = 5 + 2 = 7$

$f_1(1) = g_1(1) + g_1(3) = 4 + 4 = 8$

$f_2(0) = [g_1(0) - g_1(2)]\, W_4^0 = [5 - 2]\,(1) = 3$

$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1 = [4 - 4]\,(-j) = 0$

$f_3(0) = g_2(0) + g_2(2) = -[-3 + 0] = -3$

$f_3(1) = g_2(1) + g_2(3) = [0 + 0] = 0$

$f_4(0) = [g_2(0) - g_2(2)]W_4^0 = [-3 - 0]\,(1) = -3$

$f_4(1) = [g_2(1) - g_2(3)]W_4^1 = [0 - 0]\,(-j) = 0$

Stage 3

$X(0) = f_1(0) + f_1(1) = 7 + 8 = 15$

$X(4) = f_1(0) - f_1(1) = 7 - 8 = -1$

$X(2) = f_2(0) + f_2(1) = 3 + 0 = 3$

$X(6) = f_2(0) - f_2(1) = 3 - 0 = 3$

$X(1) = f_3(0) + f_3(1) = -3 + 0 = -3$

$X(5) = f_3(0) - f_3(1) = -3 - 0 = 3$

$X(3) = f_4(0) + f_4(1) = -3 + 0 = -3$

$X(7) = f_4(0) - f_4(1) = -3 - 0 = -3$

We now rearrange the output in the proper order.

$X(k) = \{15, -3, 3, -3, -1, -3, 3, -3\}$

We now need to find $X_1(k)$.

Given $x_1(n) = x(-n)$

According to the time reversal properly of the DFT we have

$$x_1((-n))_N \xleftrightarrow{\text{DFT}} X((-k))_n$$

$$\therefore \quad x_1(n) = x((-n))_N \xleftrightarrow{\text{DFT}} X((-k))_N$$

$$= X_1(-k)$$

Hence $X_1(k)$ is obtained by circularly folding $X_1(k)$

$\therefore X_1(k) = \{15, -3, 3, -3, -1, -3, 3, -3\}$

$\therefore$ $X_1(k)$ is the same as $X(k)$.

Ex. 4.5.7 : Compute the DFT of
$x(n) = 2\delta(n) + 3\delta(n-1) + 4\delta(n-2) + 5\delta(n-3)$
Use DIF-FFT

Soln. : Here $\delta(n)$ is a impulse functions while $\delta(n-1)$, $\delta(n-2)$ and $\delta(n-3)$ are impulse shifted to the right.

$\therefore \quad x(n) = \{2, 3, 4, 5\}$

We draw a 4 – point DIF–FFT butterfly diagram. In DIF–FFT, the input is ordered but the output obtained is in the bit reversed order.

Here $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 0$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$

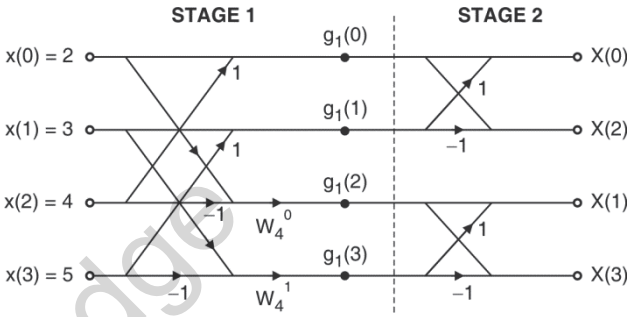


Fig. P. 4.5.7

Stage 1

$g_1(0) = x(0) + x(2) = 2 + 4 = 6$

$g_1(1) = x(1) + x(3) = 3 + 5 = 8$

$g_1(2) = [x(0) - x(2)]\, W_4^0 = [2 - 4]\,(1) = -2$

$g_1(3) = [x(1) - x(3)]\, W_4^1 = [3 - 5]\,(-j) = +2j$

Stage 2

$X(0) = g_1(0) + g_1(1) = 6 + 8 = 14$

$X(2) = g_1(0) - g_1(1) = 6 - 8 = -2$

$X(1) = g_1(2) + g_1(3) = -2 + (+2j) = -2 + 2j$

$X(3) = g_1(2) - g_1(3) = -2 - (+2j) = -2 - 2j$

We rearrange the output in the proper order

$\therefore \quad X(k) = \{14, -2 + 2j, -2, -2 - 2j\}$

Ex. 4.5.8 : Compute the DFT of $x(n) = \cos\frac{n\pi}{2}$ using DIF-FFT. Assume N = 4.

Soln. : Since N = 4, we have $x(n) = \cos\frac{n\pi}{2}$; $0 \le n \le 3$

$x(0) = \cos(0) = 1$

$x(1) = \cos\left(\frac{\pi}{2}\right) = 0$

$x(2) = \cos(\pi) = -1$

$x(3) = \cos\left(\frac{3\pi}{2}\right) = 0$

$\therefore \quad x(n) = \{1, 0, -1, 0\}$

We draw a 4 - point DIF-FFT Butterfly diagram

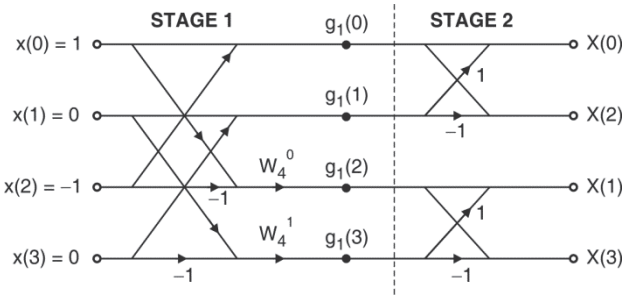


Fig. P. 4.5.8

Here $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$

$W_4^1 = e^{\frac{-j2\pi}{4}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$

Stage 1

$g_1(0) = x(0) + x(2) = 1 - 1 = 0$

$g_1(1) = x(1) + x(3) = 0 + 0 = 0$

$g_1(2) = [x(0) - x(2)]\, W_4^0 = [1 - (-1)]\,(1) = 2$

$g_1(3) = [x_1(1) - x(3)]\, W_4^1 = [0 - 0]\,(-j) = 0$

Stage 2

$X(0) = g_1(0) + g_1(1) = 0 + 0 = 0$

$X(2) = g_1(0) - g_1(1) = 0 - 0 = 0$

$X(1) = g_1(2) + g_1(3) = 2 + 0 = 2$

$X(2) = g_1(2) - g_1(3) = 2 - 0 = 2$

We rearrange the output in the proper order

$\therefore \quad X(k) = \{0, 2, 0, 2\}$

Ex. 4.5.9 : If x(n) = {1 + 2j, 3 + 4j, 5 + 6j, 7 + 8j}. Find DFT X(k) using DIF-FFT.

Soln.: Since N = 4, we draw a 4-point DIF-FFT Bufferfly diagram in DIF-FFT, the input is ordered but the output is in bit reversed order.

Here $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$

Stage 1

$g_1(0) = x(0) + x(2) = (1 + 2j) + (5 + 6j) = 6 + j8$

$g_1(1) = x(1) + x(3) = (3 + j4) + (7 + j8) = 10 + j12$

$g_2(2) = [x(0) - x(2)]\, W_4^0 = [(1 + 2j) - (5 + 6j)]\,(1)$

$= -4 - j4$

$g_1(3) = [x(1) - x(3)]\, W_8^1$

$= [(3 + j4) - (7 + j8)]\,(-j) = -4 + j4$

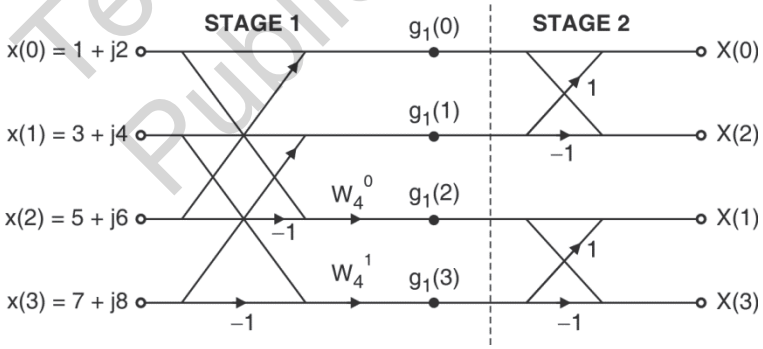


Fig. P. 4.5.9

Stage 2

$X(0) = g_1(0) + g_1(1) = (6 + j8) + (10 + j12) = 16 + j20$

$X(2) = g_1(0) - g_1(1) - (6 + j8) - (10 - j12) = -4 - j4$

$X(1) = g_1(2) + g_1(3) = (-4 - j4) + (-4 + j4) = -8$

$X(3) = g_1(2) - g_1(3) = (-4 - j4) + (-4 + j4) = -j8$

We rearrange the output in the proper order

$\therefore \quad X(k) = \{16 + j20, -8, -4 - j4, -j8\}$

4.6 Computational Complexity Comparison between DFT and FFT

- We now compare the number of computations required for DFT and FFT.
- It was mentioned earlier that FFT is merely a faster way of computing the DFT. We will now show how much is the reduction in computation when a FFT algorithm is used.

1. Computations required for DFT algorithm

We have shown at the beginning of the chapter that a N-point DFT required N^2 multiplication and N(N – 1) additions.

2. Computations required using FFT algorithm

- We begin by calculating the computation complexity required for one butterfly. Consider the general structure of butterfly as shown in Fig. 4.6.1.

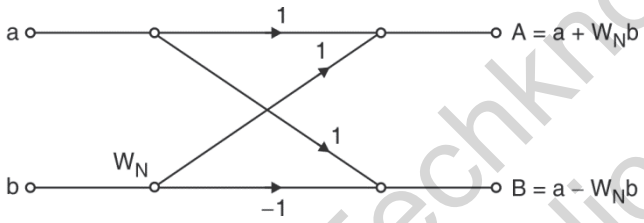


Fig. 4.6.1 : General structure of butterfly

- Here 'a' and 'b' are inputs and A and B are outputs of butterfly. The outputs are given by,

$$A = a + W_N b \quad \ldots(4.6.1)$$

$$\text{and } B = a - W_N b \quad \ldots(4.6.2)$$

- To calculate any output (A or B), we need to multiply each input by the twiddle factor W_N. So **one complex multiplication** is required for one butterfly.
- From Equations (4.6.1) and (4.6.2) we note that to calculate output 'A', one complex addition is required, while to calculate output 'B' one complex subtraction is required. But the computational complexity of addition and subtraction is same. So we can say that for one butterfly **two complex additions** are required.
- For a 8 point DFT, 4 butterflies are required at each stage. So for 'N' point DFT, at each stage, $\frac{'N'}{2}$ butterflies are required.
- Three such stages are required to compute 8–point DFT. In general, for 'N' point DFT, $\log_2 N$ stages are required.

Complex multiplications

- At each stage there are $\frac{N}{2}$ butterflies and the total number of stages are $\log_2 N$. For each butterfly, one complex multiplication is required.
- Hence total complex multiplications = $\frac{N}{2} \log_2 N$...(4.6.3)

Complex additions

- Total number of stages are $\log_2 N$ and at each stage, $\frac{N}{2}$ butterflies are required. For each butterfly, '2' complex additions are required.

$$\therefore \text{Total complex additions} = 2 \times \frac{N}{2} \log_2 N$$

$$\therefore \text{Total complex additions} = N \log_2 N$$

- Table 4.6.1 shows comparison of direct DFT computation and computation using FFT algorithms.

Table 4.6.1

Number of points 'N'	Direct Computation		Using FFT	
	Complex Multiplication N^2	Complex Additions $(N^2 - N)$	Complex Multiplication $\left(\frac{N}{2} \log_2 N\right)$	Complex Addition $(N \log_2 N)$
4	16	12	2	4
8	64	56	12	24
16	256	240	32	64
32	1024	992	40	80
64	4096	4032	96	192
128	16,384	16,256	224	448

The Table 4.6.1 shows that, by the use of FFT algorithms the number of complex multiplications and complex additions are reduced. `So there is tremendous improvement in the speed

4.6.1 Solved Examples on Computation of Inverse DFT using FFT Algorithms

Ex. 4.6.1 : How many computations are required to compute 16 point DFT using DFT and FFT algorithms

Soln. :

Here, N = 16

Using DFT

Number of multiplications = $N^2 = 16^2 = 256$

Number of addition = $N(N-1) = 16 \times 15 = 240$

Using FFT

Number of Multiplication = $\frac{N}{2}\log_2^N = 32$

Number of Addition = $N \log_2^N = 64$

Ex. 4.6.2 : How to find inverse one dimensional DFT using forward DIT FFT flow graph. Derive necessary formula.

Soln. : The 1-D-DFT formula is

$$X(k) = \sum_{k=0}^{N-1} x(n)\, W_N^{nk} \qquad ...(1)$$

The 1-D-IDFT formula is

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, W_N^{*\,nk} \qquad ...(2)$$

The DIT-FFT algorithm is computed using Equation (1). We now try to make Equation (2) appear like Equation (1)

We take the complex conjugate on both sides of Equation (2).

$$\therefore \quad x^*(n) = \frac{1}{N}\sum_{k=0}^{N-1} X^*(k)\, W_N^{**\,nk}$$

$$\therefore \quad x^*(n) = \frac{1}{N}\sum_{k=0}^{N-1} X^*(k)\, W_N^{nk} \qquad ...(3)$$

Equation (3) is identical to Equation (1) except for the complex sign and the $\frac{1}{N}$ term.

Hence in DIT-FFT, if we replace the input x(n) by $X^*(k)$ and divide it by $\frac{1}{N}$, we get $x^*(n)$. Taking the conjugate of this gives us x(n). Steps involved in computing DIT-IFFT.

1. Replace x(n) by $X^*(k)$ in the DIT FFT.
2. On obtaining $x^*(n)$, take the conjugate.

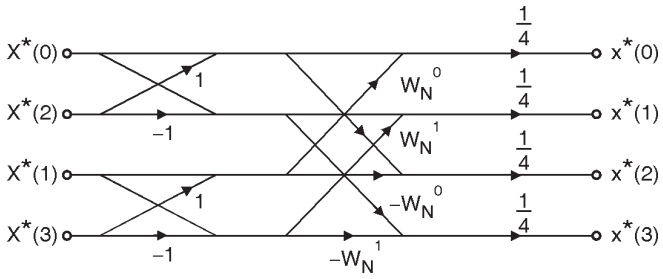


Fig. P. 4.6.2

Ex. 4.6.3 : Find the I-DFT using DIT-FFT of the given example.

$$X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$$

Soln. :

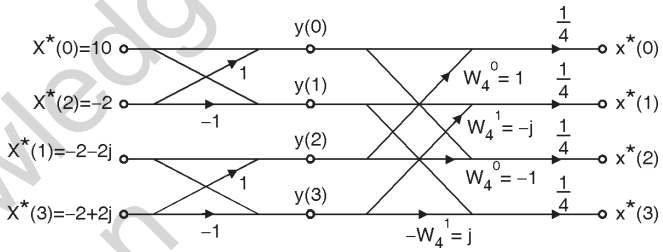


Fig. P. 4.6.3

$$y(0) = X^*(0) + X^*(2) = 8$$

$$y(1) = X^*(0) - X^*(2) = 12$$

$$y(2) = X^*(1) + X^*(3) = -4$$

$$y(3) = X^*(1) - X^*(3) = -4j$$

$$x^*(0) = \frac{1}{4}[Y(0) + Y(2) \cdot (1)] = 1$$

$$x^*(1) = \frac{1}{4}[Y(1) + Y(3) \cdot (-j)] = 2$$

$$x^*(2) = \frac{1}{4}[Y(0) + Y(2)\,(-1)] = 3$$

$$x^*(3) = \frac{1}{4}[Y(1) + Y(3) \cdot (j)] = 4$$

$$\therefore \quad x(n) = \{1, 2, 3, 4\}$$

We finally take the complex conjugate

$$x^*(n) = \{1, 2, 3, 4\}$$

We have thus found out the inverse DFT using the DIT-FFT.

Ex. 4.6.4 : Given :

$$X(k) = \{36, -4 + j9.656, -4 + j4, -4 + j1.656, -4, -4 - j1.656, -4 - j4, -4 - j9.656\}$$

Find x(n) using any IFFT algorithm.

Soln. : Let us use the DIF-FFT algorithm to compute the inverse FFT.

Here, N = 8

We draw a 8 point DIF-FFT butterfly diagram and make the required for computation of the inverse FFT. In required DIF-FFT, the input is in ordered but the output is in bit reversed order.

Here,

$$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

Similarly,

$$W_8^0 = e^{\frac{-j2\pi}{8} \cdot 0} = \cos 0 - j \sin 0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos \frac{\pi}{4} - j \sin \frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8} \cdot 2} = \cos \frac{\pi}{2} - j \sin \frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8} \cdot 3} = \cos \frac{3\pi}{4} - j \sin \frac{3\pi}{4}$$

$$= -0.707 - j\,0.707$$

Note: The input given to the network is the complex conjugate of the input and the output obtained needs to be divided by N.

Stage 1

$g_1(0) = X^*(0) + X^*(4) = 36 + (-4) = 32$

$g_1(1) = X^*(1) + X^*(5)$

$= (-4 - j9.656) + (-4 + j1.651)$

$= -8 - j8.005$

$g_1(2) = X^*(2) + X^*(6) = (-4 - j4) + (-4 + j4)$

$= -8$

$g_1(3) = X^*(3) + X^*(7)$

$= (-4 - j1.656) + (-4 + j9.656)]$

$= -8 + j8$

$g_2(0) = X^*[(0) - X^*(4)]\, W_8^0 = 36 - (-4)\,(1) = 40$

$g_2(1) = X^*[(1) - X^*(5)]\, W_8^1$

$= [(-4 - j9.656) - (-4 + j\,1.651)]\,(0.707 - j0.707)$

$= -8 - j\,8$

$g_2(2) = [\,X^*[(2) - X^*(6)]\, W_8^2$

$= [(-4 - j4) - (-4 + j\,4)]\,(-j)] = -8$

$g_2(3) = X^*[(3) - X^*(7)]\, W_8^3$

$= [(-4 - j1.656) - (-4 + j9.656)]\ (-0.707 - 0.707)$

$= -8 + j8$

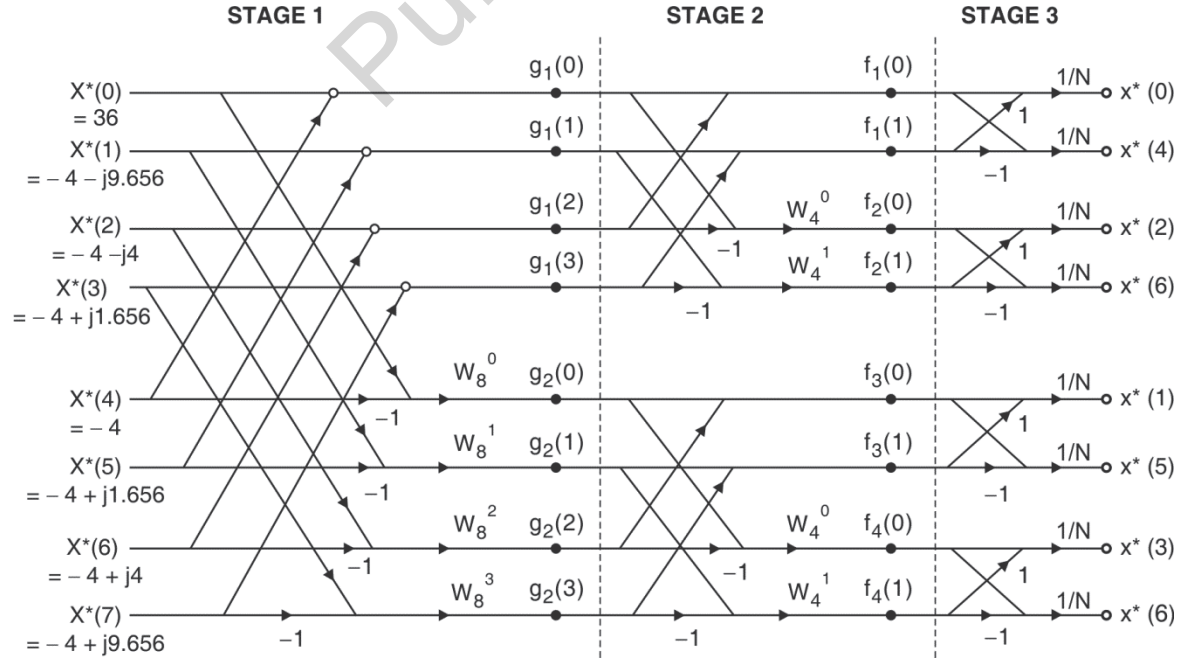


Fig. P. 4.6.4

Stage 2

$f_1(0) = g_1(0) + g_1(2) = 32 + (-8) = 24$

$f_1(1) = g_1(1) + g_1(3) = (-8 - j8) + (-8 + j8)$

$= -16$

$f_2(0) = [g_1(0) + g_1(2)]\, W_4^0 = [32 - (-8)](1) = 40$

$f_2(1) = [g_1(1) - g_1(3)]\, W_4^1$

$= [(-8 - j8) - (-8 + j8)](-j) = -16$

$f_3(0) = g_2(0) + g_2(2) = 40 + (-8) = 32$

$f_3(1) = g_2(1) + g_2(3) = (-8 - j8) + (-8 + j8)$

$= -16$

$f_4(0) = [g_2(0) - g_2(2)]W_4^0 = [40 - (-8)]\,(1) = 48$

$f_4(1) = [g_2(1) - g_2(3)]W_4^1$

$= [(-8 - j8) - (-8 + 8)](-j) = -16$

Stage 3

$x^*(0) = [f_1(0) + f_1(1)]/8 = [24 + (-16)]/8 = 1$

$x^*(4) = [f_1(0) - f_1(1)]/8 = [24 - (-16)]/8 = 5$

$x^*(2) = [f_2(0) + f_2(1)]/8 = [40 + (-16)]/8 = 3$

$x^*(6) = [f_2(0) - f_2(1)]/8 = [40 - (-16)]/8 = 7$

$x^*(1) = [f_3(0) + f_3(1)]/8 = [32 + (-16)]/8 = 2$

$x^*(5) = [f_3(0) - f_3(1)]/8 = [32 - (-16)]/8 = 6$

$x^*(3) = [f_4(0) + f_4(1)]/8 = [48 + (-16)]/8 = 4$

$x^*(7) = [f_4(0) - f_4(1)]/8 = [48 - (-16)]/8 = 8$

We rearrange the output in the proper order. Since x*(n) values are all real

$x(n) = x^*(n)$

$x(n) = \{1, 2, 3, 4, 5, 6, 7, 8\}$

Ex. 4.6.5 : Let x(n) be 8-point sequence. Its corresponding DFT X(k) is,

X(k) = { 0.5, 2 + j, 3 + j2, j, 3, - j, 3 – j2, 2 – j }

Soln. :

We will use the DIT – FFT algorithms since N = 8, we use the DIT – FFT algorithms we give X*(k) as the input and divide the output obtained by N. We also take the complex conjugate of the result obtained.

x(n) = {0.5, 2 + j ,3 +j2, j , 3 , – j , 3 – j2, 2 – j}

x(0) x(1) x(2) x(3) x(4) x(5) x(6) x(7)

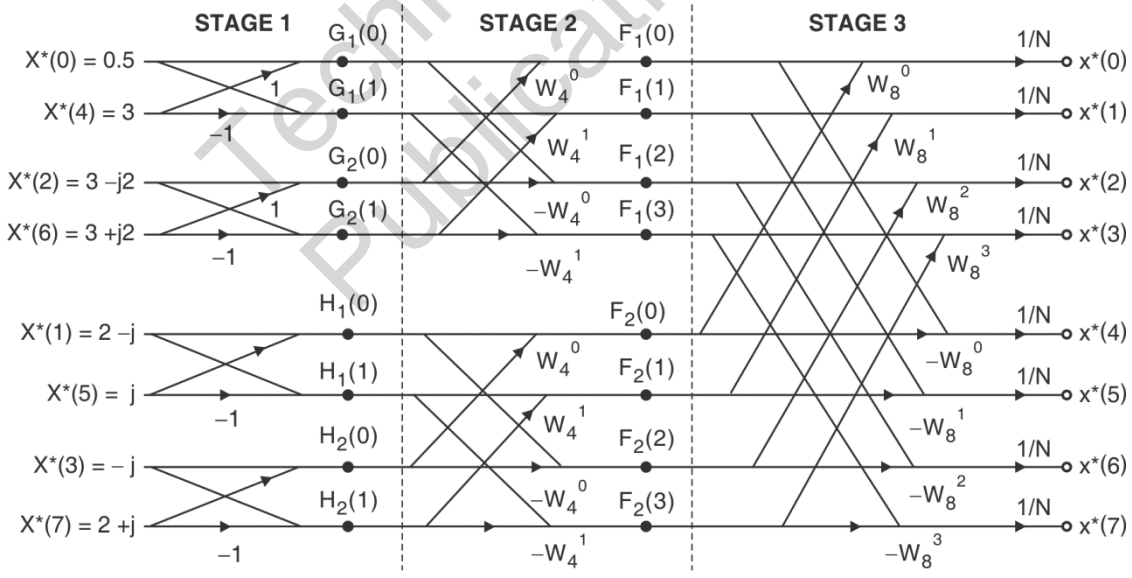


Fig. P. 4.6.5

Here, $W_4^0 = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$

Similarly, $W_8^0 = 1$

$W_8^1 = e^{\frac{-j2\pi}{8} \cdot 1} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j0.707$

$$W_8^2 = e^{\frac{-j2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - j\,0.707$$

We write down the equation at each stage and calculate the outputs

Stage 1

$$G_1(0) = X^*(0) + X^*(4)$$
$$= 0.5 + 3 = 3.5$$
$$G_1(1) = X^*(0) - X^*(4)$$
$$= 0.5 - 3 = -2.5$$
$$G_2(0) = X^*(2) + X^*(6)$$
$$= (3 - j2) + (3 + j2) = 6$$
$$G_2(1) = X^*(2) - X^*(6)$$
$$= (3 - j2) - (3 + j2) - j4$$
$$H_1(0) = X^*(1) + X^*(5)$$
$$= (2 - j) + (j) = 2$$
$$H_1(1) = X^*(1) - X^*(5) = (2 - j) - (j) = 2 - j2$$
$$H_2(0) = X^*(3) + X^*(7) = -j + (2 + j) = 2$$
$$H_2(1) = X^*(3) - X^*(7) = -j - (2 + j) = -2 - j2$$

Stage 2

$$F_1(0) = G_1(0) + W_4^0\, G_2(0)$$
$$= 3.5 + (1)\,(6) = 9.5$$
$$F_1(1) = G_1(1) + W_4^1\, G_2(1)$$
$$= -2.5 + (-j)\,(-j4) = -6.5$$
$$F_1(2) = G_1(0) - W_4^0\, G_2(0)$$
$$= 3.5 - (1)\,(6) = -2.5$$
$$F_1(3) = G_1(1) - W_4^1\, G_2(1)$$
$$= 2.5 - (-j)\,(-j4) = 1.5$$
$$F_2(0) = H_1(0) + W_4^0\, H_2(0)$$
$$= 2 + (1)\,(2) = 4$$
$$F_2(1) = H_1(1) + W_4^1\, H_2(1)$$
$$= (2 - j2) + (-j)\,(-2 - j2) = 0$$
$$F_2(2) = H_1(0) - W_4^0\, H_2(0)$$
$$= 2 - (1)\,(2) = 0$$
$$F_2(3) = H_1(1) - W_4^1\, H_2(1)$$
$$= (2 - j2) - (-j)\,(-2 - j2)$$
$$= 4 - j4$$

Stage 3

$$x^*(0) = [F_1(0) + W_8^0\, F_2(0)]/8 = [9.5 + (1)\,4]/8 = 1.69$$
$$x^*(1) = [F_1(1) + W_8^1\, F_2(1)]/8$$
$$= [-6.5 + (0.707 - j0.707\,(0)]/8 = -0.81$$
$$x^*(2) = [F_1(2) + W_8^2\, F_2(2)]/8$$
$$= [-2.5 + (-j)\,(0)]/8 = -0.31$$
$$x^*(3) = [F_1(3) + W_8^3\, F_2(3)]/8$$
$$= [1.5 + (-0.707 - j0.707)\,(4 - j4)]/8 = -0.52$$
$$x^*(4) = [F_1(0) - W_8^0\, F_2(0)]/8 = [9.5 - (1)\,4]/8 = 0.69$$
$$x^*(5) = [F_1(1) - W_8^1\, F_2(1)]/8$$
$$= [-6.5 - (0.707 - j0.707)\,(0)]/8 = -0.81$$
$$x^*(6) = [F_1(2) - W_8^2\, F_2(2)]/8$$
$$= [-2.5 - (-j)\,(0)]/8 = -0.31$$
$$x^*(7) = [F_1(3) - W_8^3\, F_2(3)]/8$$
$$= [1.5 - (-0.707 - j0.707)\,(4 - j4)]/8 = 0.89$$

We finally take the complex conjugate of the result.

$$\therefore\ x(n) = \{1.69, -0.81, -0.31, -0.52, 0.69, -0.81, -0.31, 0.89\}$$

Ex. 4.6.6 : Given

X(k) = {2, – 6j, 2 – 8j, 6j, 2 – 6j, 2 + 8j, 6j, 0} Find x(n) using any IFFT algorithm.

Soln. : We will use the DIT-FFT algorithm. Since N = 8, we use a 8-point DIT-FFT Butterfly diagram. In DIT-FFT, the input is in bit reversed order. We give X*(k) as the input and divide the output - obtained by N. We also take complex conjugate of the result obtained.

$$X(k) = \{2, -j6, 2 - j8, j6, 2 - j6, 2 + j8, j6, 0\}$$
$$\therefore \quad \{X(0), X(1), X(2), X(3), X(4), X(5), X(6), X(7)\}$$

Here

$$W_4^0 = 1$$
$$W_4^1 = e^{\frac{-j2\pi}{4}\cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

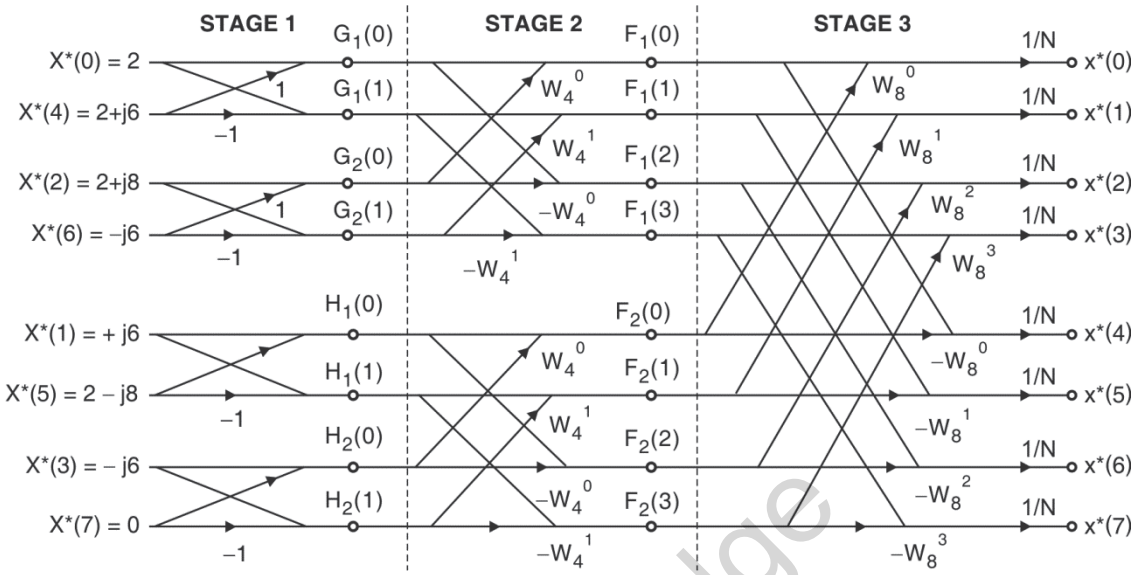


Fig. P. 4.6.6

Similarly,

$$W_8^0 = 1$$

$$W_8^1 = e^{\frac{-j2\pi}{8}\cdot 1} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4}$$

$$= 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4}$$

$$= -0.707 - j\,0.707$$

We write down the equation at each stage and calculate the outputs.

Stage 1

$$G_1(0) = X^*(0) + X^*(4) = 4 + j6$$

$$G_1(1) = X^*(0) - X^*(4) = -j6$$

$$G_2(0) = X^*(2) + X^*(6) = 2 + j2$$

$$G_2(1) = X^*(2) - X^*(6) = 2 + j14$$

$$H_1(0) = X^*(1) + X^*(5) = 2 - j2$$

$$H_1(1) = X^*(1) - X^*(5) = -2 + j14$$

$$H_2(0) = X^*(3) + X^*(7) = -j6$$

$$H_2(1) = X^*(3) - X^*(7) = -j6$$

Stage 2

$$F_1(0) = G_1(0) + W_4^0 G_2(0) = (4 + j6) + (1)(2 + j2)$$

$$= 6 + j8$$

$$F_1(1) = G_1(1) + W_4^1 G_2(1)$$

$$= -j6 + [(-j)(2 + j14)] = 14 - j8$$

$$F_1(2) = G_1(0) - W_4^0 G_2(0)$$

$$= (4 + j6) - [(1)(2 + j2)] = 2 + j4$$

$$F_1(3) = G_1(1) - W_4^1 G_2(1)$$

$$= -j6 - [(-j)(2 + j14)] = -14 - j4$$

$$F_2(0) = H_1(0) + W_4^0 H_2(0)$$

$$= (2 - j2) + [(1)(-j6)] = 2 - j8$$

$$F_2(1) = H_1(1) + W_4^1 H_2(1)$$

$$= (-2 + j14) + [(-j)(-j6)]$$

$$= -8 + j14$$

$$F_2(2) = H_1(0) - W_4^0 H_2(0)$$

$$= (-2 - j2) - [(1)(-j6)]$$

$$= 2 + j14$$

$$F_2(3) = H_1(1) - W_4^1 H_2(1) = (-2 + j14) - [(-j)(-j6)]$$

$$= 4 + j14$$

Stage 3

$$x^*(0) = [F_1(0) + W_8^0 F_2(0)]/N$$

$$= [(6 + j8)] + [(1)(2 - j8)]/8 = 1$$

$x^*(1) = [F_1(1) + W_8^1 F_2(1)]/N$

$= [(14 - j8) + [(0.707 - j0.707)(-8 + j4)]/8$

$= 2.28 + j0.95$

$x^*(2) = [F_1(2) + W_8^2 F_2(2)]/N$

$= [(2 + j4) + (-j)(2 + j4)]/8 = 0.75 + j0.25$

$x^*(3) = [F_1(3) + W_8^3 F_2(3)]/N = [(-14 - j4)$

$+ [(-0.707 - j0.707)(4 + j14)]/8$

$= -0.86 - j2.09$

$x^*(4) = [F_1(0) - W_8^0 F_2(0)]/N$

$= [(6 + j8) - [(1)(2 - j8)]/8 = 0.5 + j2$

$x^*(5) = [F_1(1) - W_8^1 F_2(1)]/N$

$= [(14 - j8) - [(0.707 - j0.707)(-8 + j4)]/8$

$= 1.22 - j2.94$

$x^*(6) = [F_1(2) - W_8^2 F_2(2)]/N$

$= [(2 + j4) - [(-j)(2 + j4)]]/8 = -0.25 + j0.75$

$x^*(7) = [F_1(3) - W_8^3 F_2(3)]/N$

$= [(-14 - j4) - [(-0.707 - j0.707)(4 + j14)]/8$

$= -2.63 + j1.09$

$\therefore \quad x^*(n) = \{1, 2.28 + j0.95, 0.75 + j0.25, -0.86 - j2.09, 0.5 + j2, 1.22 - j2.94, -0.25 + j0.75, -2.63 + j1.09\}$

What we have obtained is $x^*(n)$

$\therefore \quad x(n) = \{1, 2.28 - j0.95, 0.75 - j0.25, -0.866 + j2.09, 0.5 - j2, 1.22 + j2.94, -0.25 - j0.75, -2.63 - j1.09\}$

Ex. 4.6.7 : Find IDFT of X(k) = {10, – 2 + 2j, – 2, – 2 – 2j} using DIT-FFT algorithm.

Soln. :

$X(k) = \{10, -2 + 2j, -2, -2 - 2j\}$

$\{X(0), X(1), X(2), X(3)\}$

Since N = 4, we use a 4 – point DIT-FFT butterfly diagram.

In DIT-FFT, the input is in bit reversed order. We give X*(k) as input and divide the output obtained by N. We also take the complex conjugate of the result obtained.

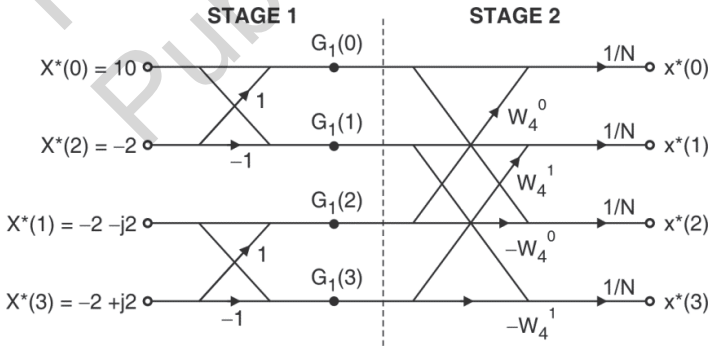


Fig. P. 4.6.7

Here, $W_4^0 = 1$

$W_4^1 = -j$

Stage 1

$G_1(0) = X^*(0) + X^*(2) = 10 + (-2) = 8$

$G_1(1) = X^*(1) - X^*(2) = 10 - (-2) = 12$

$G_1(2) = X^*(1) + X^*(3) = (-2 - j2) + (-2 + j2) = -4$

$G_1(3) = X^*(1) - X^*(3) = (-2 - j2) - (-2 + j2) = -j4$

Stage 2

$$x^*(0) = [G_1(0) + W_4^0 G_1(2)]/4$$
$$= [8 + (1)(-4)]/4 = 1$$
$$x^*(1) = [G_1(1) + W_4^1 G_1(3)]/4$$
$$= [12 + (-j)(-j4)]/4 = 2$$
$$x^*(2) = [G_1(0) - W_4^0 G_1(2)]/4 = [8 - (1)(-4)]/4 = 3$$
$$x^*(3) = [G_1(1) - W_4^1 G_1(3)]/4$$
$$= [12 - (-j)(-j4)]/4 = 4$$
$$\therefore \quad x^*(n) = \{1, 2, 3, 4\}$$
$$\therefore \quad x(n) = \{1, 2, 3, 4\}$$

4.6.2 Filtering using FFT Algorithms

Ex. 4.6.8 : Determine circular convolution of x(n) = {1, 2, 1, 4} and h(n) = {1, 2, 3, 2} using Radix-2 - FFT.

Soln. : From the circular convolution property of the DFT, it is known than,

$$y(n) = x(n) \circledast h(n) \underset{\text{IDFT}}{\overset{\text{DFT}}{\rightleftarrows}} X(k)\,H(k) = Y(k)$$

Hence to determine circular convolution using DFT, we follow the given steps

1. Compute DFT of x(n) to obtain X(k)
2. Compute DFT of h(n) to obtain H(k)
3. Obtain $Y(k) = X(k) \cdot H(k)$
4. Compute IDFT of Y(k) to obtain y(n)

We will use the DIT-FFT to compute the DFT of x(n) and h(n). Since both are of length 4, we will use a 4-point DIF-FFT butterfly diagram. In DIF-FFT the output is ordered and input is in bit reversed form.

Step 1 : Compute DFT of x(n)

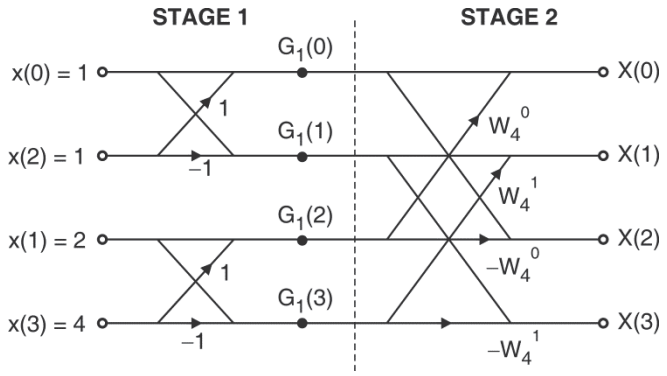


Fig. P. 4.6.8

Here, $W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = \cos 0 - j \sin 0 = 1$

$$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = \cos\frac{\pi}{2} - j \sin\frac{\pi}{2} = -j$$

Stage 1

$$G_1(0) = x(0) + x(2) = 1 + 1 = 2$$
$$G_1(1) = x(0) - x(2) = 1 - 1 = 0$$
$$G_1(2) = x(1) + x(3) = 2 + 4 = 6$$
$$G_1(3) = x(1) - x(3) = 2 - 4 = -2$$

Stage 2

$$X(0) = G_1(0) + W_4^0 G_1(2) = 2 + (1)(6) = 8$$
$$X(1) = G_1(1) + W_4^1 G_1(3) = 0 + (-j)(-2) = j2$$
$$X(2) = G_1(0) - W_4^0 G_1(2) = 2 - (1)(6) = -4$$
$$X(3) = G_1(1) - W_4^1 G_1(3) = 0 - (-j)(-2) = -j2$$
$$\therefore \quad X(k) = \{8, j2, -4, -j2\}$$

Step 2 : In a similar manner we compute the DFT of h(n)

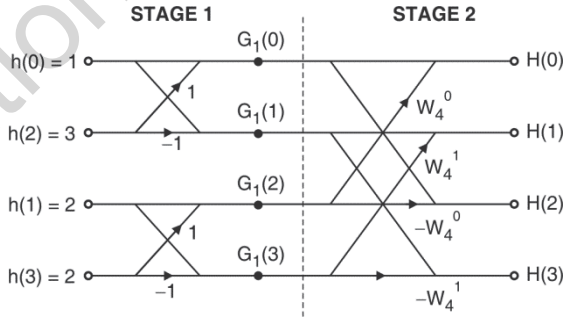


Fig. P. 4.6.8(a)

Stage 1

$$G_1(0) = h(0) + h(2) = 1 + 3 = 4$$
$$G_1(1) = h(0) - h(2) = 1 - 3 = -2$$
$$G_1(2) = h(1) + h(3) = 2 + 2 = 4$$
$$G_1(3) = h(1) - h(3) = 2 - 2 = 0$$

Stage 2

$$H_1(0) = G_1(0) + W_4^0 G_1(2) = 4 + (1)(4) = 8$$
$$H_1(1) = G_1(1) + W_4^1 G_1(3) = -2 + (-j)(0)$$
$$= -2$$
$$H_1(2) = G_1(0) - W_4^0 G_1(2) = 4 - (1)(4) = 0$$
$$H_1(3) = G_1(1) - W_4^1 G_1(3) = -2 - (-j)(0)$$
$$= -2$$
$$\therefore \quad H(k) = \{8, -2, 0, -2\}$$

Step 3 :

$$Y(k) = X(k) \cdot H(k)$$

$$Y(0) = X(0) \cdot H(0) = (8) \cdot (8) = 64$$

$$Y(1) = X(1) \cdot H(1) = (j2) \cdot (-2) = -j4$$

$$Y(2) = X(2) \cdot H(2) = (-4) \cdot (0) = 0$$

$$Y(3) = X(3) \cdot H(3) = (-j2)(-2) = j4$$

$$\therefore \quad Y(k) = \{64, -j4, 0, j4\}$$

Step 4 :

The final step is to obtain y(n). This is done by computing the IDFT of Y(k). We will use the DIT-FFT butterfly diagram with the required modification.

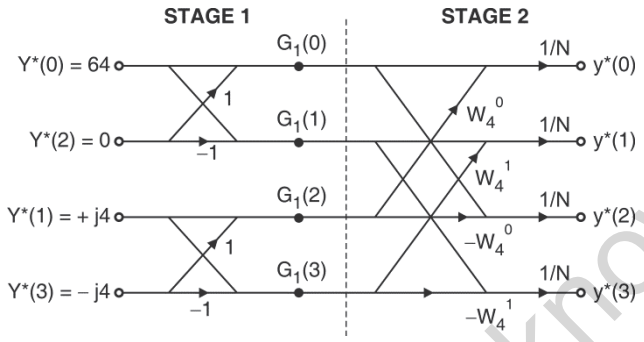


Fig. P. 4.6.8(b)

Stage 1

$$G_1(0) = y^*(0) + y^*(2) = 64 + 0 = 64$$

$$G_1(1) = y^*(0) - y^*(2) = 64 - 0 = 64$$

$$G_1(2) = y^*(1) + y^*(3) = j4 + (-j4) = 0$$

$$G_1(3) = y^*(1) - y^*(3) = j4 - (-j4) = j8$$

Stage 2

$$y^*(0) = [G_1(0) + W_4^0 G_1(2)]/4 = [64 + (1)(0)]/4 = 16$$

$$y^*(1) = [G_1(1) + W_4^1 G_1(3)]/4$$

$$= [64 + (-j)(j8)]/4 = 18$$

$$y^*(2) = [G_1(0) - W_4^0 G_1(2)/4 = [64 - (1)(0)]/4 = 16$$

$$y^*(3) = [G_1(1) - W_4^1 G_1(3)]/4$$

$$= [64 - (-j)(j8)/4 = 14$$

$$y^*(n) = \{16, 18, 16, 14\}$$

Since y*(n) has all real values

$$y(n) = y^*(n)$$

$$\therefore \quad y(n) = \{16, 18, 16, 14\}$$

We would have obtained the same result had we performed time domain circular convolution. Let us check the result.

$$y(n) = x(n) * h(0)$$

We generate a circular matrix of x(n)

$$\begin{matrix} & x(n) & & & & h(n) & \\ \begin{bmatrix} 1 & 4 & 1 & 2 \\ 2 & 1 & 4 & 1 \\ 1 & 2 & 1 & 4 \\ 4 & 1 & 2 & 1 \end{bmatrix} & & & & \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \end{bmatrix} & = & \begin{bmatrix} 16 \\ 18 \\ 16 \\ 14 \end{bmatrix} \end{matrix}$$

$$\therefore \quad y(n) = \{16, 18, 16, 14\}$$

Hence it is seen the results obtained are the same.

Ex. 4.6.9 : By means of FFT-IFFT technique, compute the linear convolution of x(n) = {2, 1, 2, 1}, h(n) = {1, 2, 3, 4}

Soln. : We are supposed to compute linear convolution using FFT-IFFT.

We know from the circular convolution property that,

$$y(n) = x(n) \circledast h(n) \underset{\text{IDFT}}{\overset{\text{DFT}}{\rightleftarrows}} X(k) \cdot H(k) = y(k)$$

We also know that to obtain linear convolution from circular convolution, the lengths of each of signals should be $N_1 + N_2 - 1$.

In this case length x(n) and h(n) = 4.

$\therefore$ If x(n) and h(n) are made equal to (4 + 4 – 1) = 7, then circular convolution and linear convolution will give us the same result. Then we can use the above mentioned property to obtain linear convolution using FFT-IFFT. We increase the size of x(n) and h(n) by Zero padding.

$$\therefore \quad x(n) = \{2,1, 2, 1, 0, 0, 0\}_{1\times 7}$$

$$h(n) = \{1, 2, 3, 4, 0, 0, 0\}_{1\times 7}$$

In order to use the radix-2 algorithm, the lengths of the signal must be powers of 2. We therefore pad one more zero to both x(n) and h(n) and make their lengths equal to 8.

Once the final result is obtained, we shall delete the last value.

$$x(n) = \{2, 1, 2, 1, 0, 0, 0\}_{1\times 8}$$

$$h(n) = \{1, 2, 3, 4, 0, 0, 0, 0\}_{1\times 8}$$

Steps involved are:

1. Compute DFT of x(n) to obtain X(k)
2. Compute DFT of h(n) obtain H(k)
3. Obtain Y(k) = X(k) · H(k)
4. Compute IDFT of y(k) to obtain y(n)

We shall use the DIT-FFT algorithm to compute the DFT of x(n) and h(n).

Step 1 : Computing X(k) from x(n)

$$X(n) = \{2,\ 1,\ 2,\ 1,\ 0,\ 0,\ 0,\ 0\}\ x(0)\ x(1)\ x(2)\ x(3)\ x(4)\ x(5)\ x(6)\ x(7)$$

Here $W_4^0 = 1$

$$W_4^1 = e^{\frac{-j2\pi}{4}\cdot 1} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

Similarly, $W_8^0 = 1$

$$W_8^1 = e^{\frac{-j2\pi}{8}\cdot 1} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} = 0.707 - j\,0.707$$

$$W_8^2 = e^{\frac{-j2\pi}{8}\cdot 2} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$$

$$W_8^3 = e^{\frac{-j2\pi}{8}\cdot 3} = \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} = -0.707 - j\,0.707$$

We write down the equation at each stage and calculate the outputs.

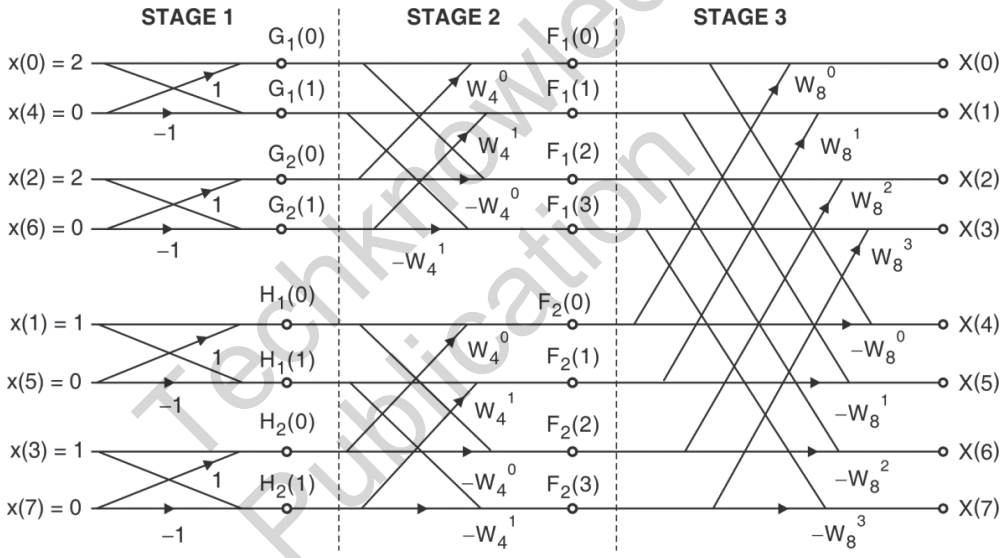


Fig. P. 4.6.9

Stage 1

$$G_1(0) = x(0) + x(4) = 2 + 0 = 2$$

$$G_1(1) = x(0) - x(4) = 2 - 0 = 2$$

$$G_2(0) = x(2) + x(6) = 2 + 0 = 2$$

$$G_2(1) = x(2) - x(6) = 2 - 0 = 2$$

$$H_1(0) = x(1) + x(5) = 1 + 0 = 1$$

$$H_1(1) = x(1) - x(5) = 1 - 0 = 1$$

$$H_2(0) = x(3) + x(7) = 1 + 0 = 1$$

$$H_2(1) = x(3) - x(7) = 1 - 0 = 1$$

Stage 2

$$F_1(0) = G_1(0) + W_4^0 G_2(0) = 2 + (1)\,(2) = 4$$

$$F_1(1) = G_1(1) + W_4^1 G_2(1) = 2 + (-j)\,(2) = 2 - j2$$

$$F_1(2) = G_1(0) - W_4^0 G_2(0) = 2 - (1)\,(2) = 0$$

$$F_1(3) = G_1(1) - W_4^1 G_2(1) = 2 - (-j)\,(2) = 2 + j2$$

$$F_2(0) = H_1(0) + W_4^0 H_2(0) = 1 + (1)\,(1) = 2$$

$$F_2(1) = H_1(1) + W_4^1 H_2(1) = 1 + (-j)\,(1) = 1 - j1$$

$$F_2(2) = H_1(0) - W_4^0 H_2(0) = 1 - (1)\,(1) = 0$$

$$F_2(3) = H_1(1) - W_4^1 H_2(1) = 1 - (-j)\,(1) = 1 + j1$$

Stage 3

$$X(0) = F_1(0) + W_8^0 F_2(0) = 4 + (1)\,(2) = 6$$

$$X(1) = F_1(1) + W_8^1 F_2(1)$$

$= (2 - j2) + [(0.0707 - j0.707)(1 - j1)]$

$= 2 - j3.41$

$X(2) = F_1(2) + W_8^2 F_2(2) = 0 + (-j)(0) = 0$

$X(3) = F_1(3) + W_8^3 F_2(3)$

$= (2 + j2) + [(-0.707 - j0.707(1 + j1)]$

$= 2 + j0.58$

$X(4) = F_1(0) - W_8^0 F_2(0) = 4 - (1)(2) = 2$

$X(5) = F_1(1) - W_8^1 F_2(1)$

$= (2 - j2) - [(0.707 - j0.707)(1 - j1)]$

$= 2 - j0.58$

$X(6) = F_1(2) - W_8^2 F_2(2) = 0 - (j)(0) = 0$

$X(7) = F_1(3) - W_8^3 F_2(3)$

$= (2 + j2) - [(-0.707 - j0.707)(1 + j1)]$

$= 2 + j3.41$

$\therefore \quad X(k) = \{6, 2 - j3.41, 0, 2 + j0.58, 2,$

$2 - j0.58, 0, 2 + j3.41\}$

Step 2 :Computing H (k) from h (n)

$h(n) = \{1, 2, 3, 4, 0, 0, 0, 0\}$

$\{h(0), h(1), h(2), h(3), h(4), h(5), h(6), h(7)\}$

We write down the equation at each stage and calculate the outputs

Stage 1

$G_1(0) = h(0) + h(4) = 1 + 0 = 1$

$G_1(1) = h(0) - h(4) = 1 - 0 = 1$

$G_2(0) = h(2) + h(6) = 3 + 0 = 3$

$G_2(1) = h(2) - h(6) = 3 - 0 = 3$

$H_1(0) = h(1) + h(5) = 2 + 0 = 2$

$H_1(1) = h(1) - h(5) = 2 - 0 = 2$

$H_2(0) = h(3) + h(7) = 4 + 0 = 4$

$H_2(1) = h(3) - h(7) = 4 - 0 = 4$

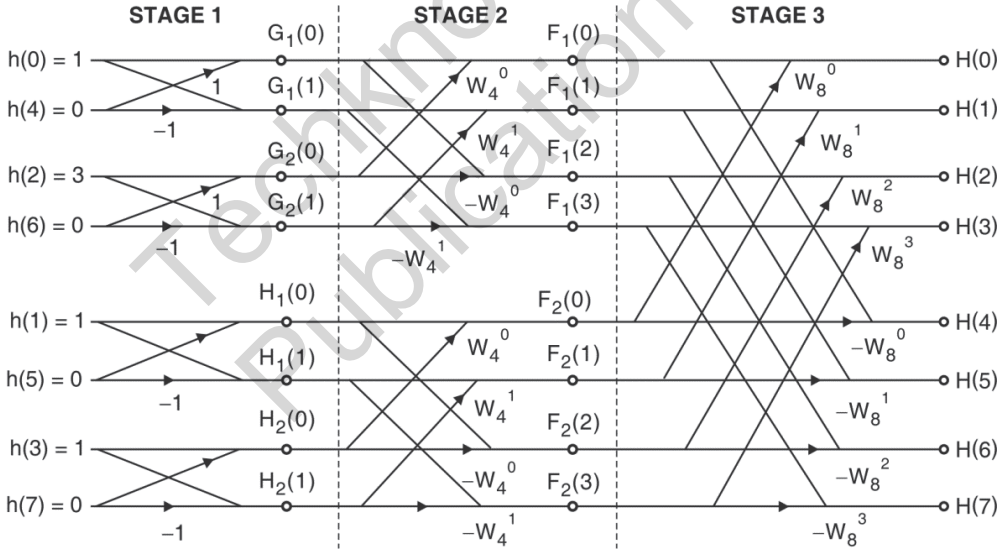


Fig. P.4.6.9(a)

Stage 2

$F_1(0) = G_1(0) + W_4^0 G_2(0) = 1 + (1)(3) = 4$

$F_1(1) = G_1(1) + W_4^1 G_2(1) = 1 + (-j)(3) = 1 - j3$

$F_1(2) = G_1(0) - W_4^0 G_2(0) = 1 - (1)(3) = -2$

$F_1(3) = G_1(1) - W_4^1 G_2(1) = 1 - (-j)(3) = 1 + j3$

$F_2(0) = H_1(0) + W_4^0 H_2(0) = 2 + (1)(4) = 6$

$F_2(1) = H_1(1) + W_4^1 H_2(1) = 2 + (-j)(4) = 2 - j4$

$F_2(2) = H_1(0) - W_4^0 H_2(0) = 2 - (1)(4) = -2$

$F_2(3) = H_1(1) - W_4^1 H_2(1) = 2 - (-j)(-4) = 2 + j4$

Stage 3

$$H(0) = F_1(0) + W_8^0 F_2(0) = 4 + (1)(6) = 10$$

$$H(1) = F_1(1) + W_8^1 F_2(1)$$

$$= (1 - j3) + [(0.707 - j0.707)(2 - j4)]$$

$$= -0.41 - j7.24$$

$$H(2) = F_1(2) + W_8^2 F_2(2) = -2 + [(-j)(-2)] = -2 + j2$$

$$H(3) = F_1(3) + W_8^3 F_2(3)$$

$$= (1 + j3) + [(-0.707 - j0.707)2 + j4]$$

$$= 2.41 - j1.24$$

$$H(4) = F_1(0) - W_8^0 F_2(0) = 4 - (1)(6) = -2$$

$$H(5) = F_1(1) - W_8^1 F_2(1)$$

$$= (1 - j3) - [(0.707 - j0.707)(2 - j4)]$$

$$= 2.41 + j1.24$$

$$H(6) = F_1(2) - W_8^2 F_2(2) = -2 - [(-j)(-2)]$$

$$= -2 - j2$$

$$H(7) = F_1(3) - W_8^3 F_2(3)$$

$$= (1 + j3) - [(-0.707 - j0.707)(2 + j4)]$$

$$= -0.41 + j7.24$$

$$H(k) = \{10, -0.41 - j7.24, -2 + j2, 2.41 - j1.24, -2, 2.41 + j1.24, -2 - j2, -0.41 + j7.24\}$$

Step3: Obtain $Y(k) = X(k) \cdot H(k)$

$$Y(0) = X(0) \cdot H(0) = 60$$

$$Y(1) = X(1) \cdot H(1) = -25.55 - j13.07$$

$$Y(2) = X(2) \cdot H(2) = 0$$

$$Y(3) = X(3) \cdot H(3) = 5.55 - j1.07$$

$$Y(4) = X(4) \cdot H(4) = -4$$

$$Y(5) = X(5) \cdot H(5) = 5.55 + j1.07$$

$$Y(6) = X(6) \cdot H(6) = 0$$

$$Y(7) = X(7) \cdot H(7) = -25.55 + j13.07$$

$$\therefore \quad Y(k) = \{60, -25.55 - j13.07, 0, 5.55 - j1.07, -4, 5.55 + j1.07, 0, -25.55 + j13.07\}$$

Step 4: Compute IDFT of y(k) to obtain y(n) we use the DIT–FFT algorithm to compute the IDFT. The final output obtained after computing the IDFT is,

$$y(n) = \{2, 5, 10, 16, 12, 11, 4, 0\}$$

Since an extra zero was added to x(n) and h(n) to increase their lengths to 8, we remove the last zero of y(n).

$$\therefore \quad y(n) = \{2, 5, 10, 16, 12, 11, 4\}$$

This is the same result that would be obtained of we perform linear convolutions in the time domain. Let us check the result,

$$y(n) = x(n) * h(n)$$

We use the matrix method

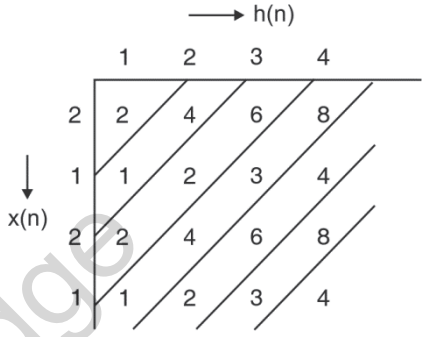


Fig. P.4.6.9(b)

$$y(n) = \{2, 5, 10, 16, 12, 11, 4\}$$

Hence it is seen that the results obtained one the same.

Summary

In this chapter we study the algorithms proposed by Cooley and Tukey. This method came to be known as the Fast Fourier Transform (F.F.T). The FFT requires only N $\log_2$ N calculations as opposed to N^2 used by the DFT. Two important algorithms viz. Decimation in Time and decimation in Frequency for Radix 2 were derived. We also learnt how to use the butterfly diagram to compute the Inverse DFT. Many examples were solved to understand how to compute the DFT using the FFT. Filtering using the FFT algorithms were also carried out.

Review Questions

Q. 1 Draw basic butterfly structure for DIT- FFT.

Q. 2 Draw total signal flow graph for DIT-FFT.

Q. 3 Obtain computational complexity of FFT algorithm.

Q. 4 Explain in place computation of FFT.

Q. 5 Derive the first stage of DIF-FFT algorithm. Draw the basic butterfly structure for the same.

Q. 6 Compare DIT-FFT and DIF-FFT algorithms.

Q. 7 Write a short note on bit reversal in FFT.

Introduction to Image Processing

Syllabus

Introduction to Digital Image, Digital Image Processing System, Image File Formats: BMP, TIFF and JPEG.

5.1 Introduction

Human beings are primarily visual creatures who depend on their eyes to gather information around them. Of the five senses that human beings have, sight is what we depend upon the most. Not many animals depend on their visual systems; the way human beings do.

Bats use high frequency sound waves. They emit sound waves which reflect back when they encounter some obstruction. Cats have poor vision but an excellent sense of smell. Snakes locate prey by heat emission and fish have organs that sense electrical fields.

5.2 What Do We Mean by Image Processing ?

What happens when we look at an object ?

- The eye records the scene and sends signals to the brain. These signals get processed in the brain and some meaningful information is obtained. Let us take a simple example : when we see fire, we immediately identify it as something hot. Two things have happened here.
  (1) The scene has been recorded by the eye.
  (2) The brain processed this scene and gave out a warning signal.

  This is image processing !!!
- We start processing images from the day we are born. Hence image processing is an integral part of us and we continue to process images till the day we die. So even if this subject seems to be new, we have been subconsciously doing this, all these years. The human eye-brain mechanism represents the ultimate imaging system.
- Apart from our vision, we have another important trait that is common to all human beings. We like to store information, analyse it, discuss it with others and try to better it. This trait of ours is responsible for the rapid development of the human race.
- Early human beings strove to record their world by carving crude diagrams on stone. All the drawings that we see in old caves is just that; storing images seen, trying to analyse them and discussing it with others in the tribe. Refer Fig. 5.2.1.
- This art developed through the ages by way of materials and skill. By the mid-nineteenth century, photography was well established. Image processing that we study starts from this era.

Fig. 5.2.1

- Though it was stated earlier that the human eye-brain mechanism represents the ultimate imaging system, image processing as a subject involves processing images obtained by a camera. With the advent of computers, image processing as a subject grew rapidly.
- Images from a camera are fed into a computer where algorithms are written to process these images. Here, the camera replaces the human eye and the computer does the processing.
- Hence image processing as an engineering subject is basically manipulation of images by a computer.

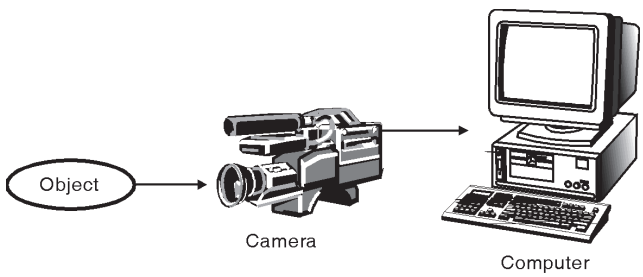


Fig. 5.2.2

5.3 Images

– Let us now define what we mean by an image. The world around us is 3-dimensional, while images obtained through a camera are 2-dimensional. Hence an image can be defined as a 2-dimensional representation of a 3-dimensional world. Consider the image shown in Fig. 5.3.1.

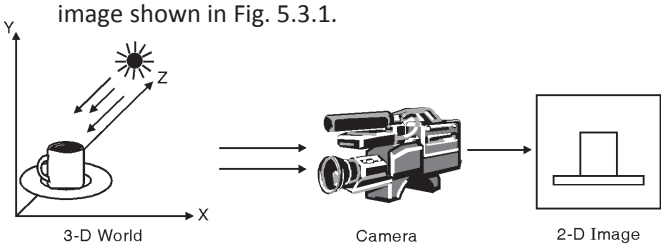


Fig. 5.3.1

– In moving from the 3-dimensional world to the 2-dimensional image, we loose one dimension. Depth information is lost. As can be seen from the Fig. 5.3.1, the handle of the tea cup is missing.

– All family pictures, photographs on identity cards etc. are 2-dimensional. If this statement is not clear, let us take a simple example.

Example of a 1-dimensional and a 2-dimensional signal

– Consider a voltage signal shown in Fig. 5.3.2. We are all familiar with a signal of this kind. Here the voltage is varying with respect to time. This is a typical 1-dimensional signal. If we want to locate a dot on the wave, all we need to know is its corresponding time.

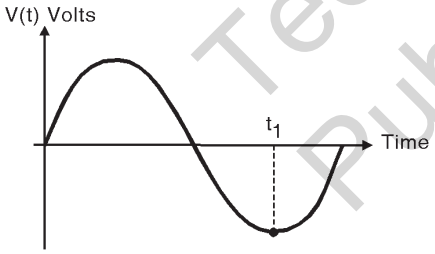


Fig. 5.3.2

– Let us see why images are 2-dimensional functions. Consider the image shown in Fig. 5.3.3.

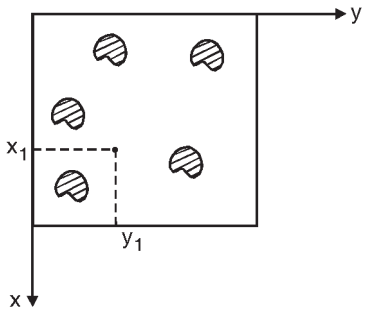


Fig. 5.3.3

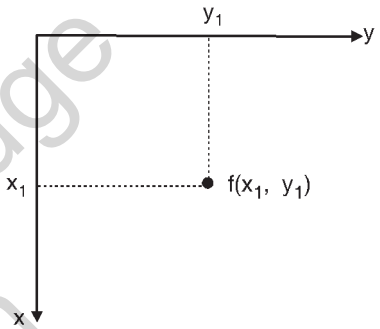


Fig. 5.3.4

– In this case, to locate the dot shown, we need to know its position in two directions (x and y) Fig. 5.3.4. Hence all images that we see are 2-dimensional functions. A typical image is represented as shown in Fig. 5.3.4. Here (x_1, y_1) are the spatial coordinates and f is the grey level (colour in the case of colour image) at that point. Hence grey level f varies with respect to the x and y coordinates.

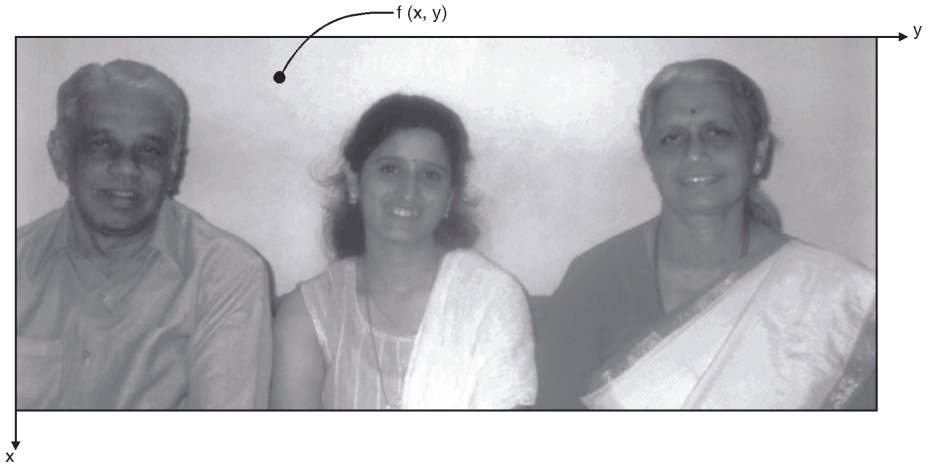


Fig. 5.3.5

5.4 The Electromagnetic Spectrum

- The apparatus shown in Fig. 5.2.2 will work only if light is incident on the object. What we call light is actually a very small section of the electromagnetic energy spectrum. The entire spectrum is shown in Fig. 5.4.1.

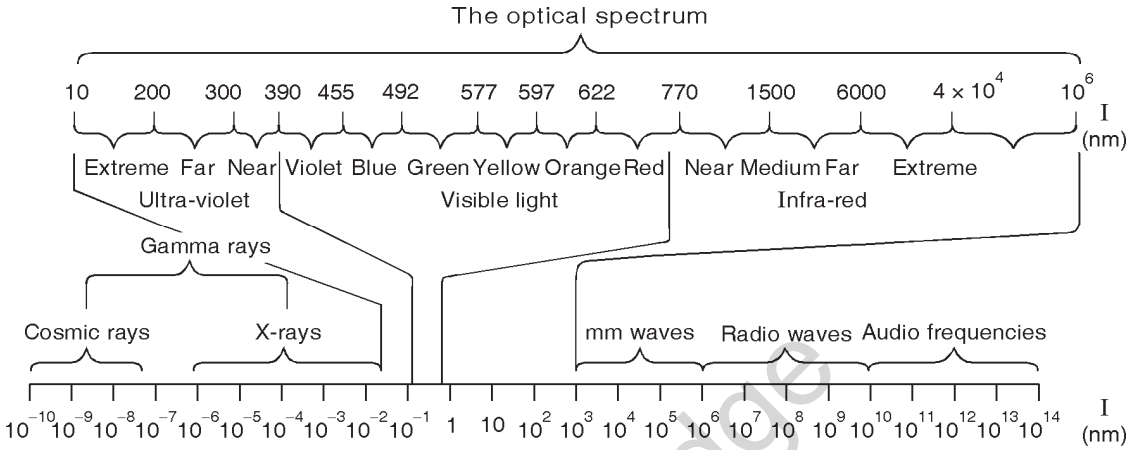


Fig. 5.4.1

- Electromagnetic energy, as the name suggests, exists in the simultaneous form of electricity and magnetism. These two forms of the same energy are transmitted together as electromagnetic radiation. One cannot exist without the other. A flow of electric current always produces magnetism, and magnetism is used to produce electricity. Electromagnetic radiation is propagated outwards from its source at a velocity of 300,00,0000 meters per second (3×10^8 m/sec).
- Although our natural source of electromagnetic radiation is the sun, there are also a number of man-made sources which, among many others, include tungsten filament lamps, gas discharge lamps and lasers. Light is a band of electromagnetic radiation mediated by the human eye and is limited to a spectrum extending from 380 nm to 760 nm.
- Most of the images that we encounter in our day to day life are taken from cameras which are sensitive to this range of the electromagnetic spectrum (380 - 760 nm). We must not forget though, that there are cameras which are capable of detecting infrared, ultraviolet light, X-rays and radio waves too.
- The electromagnetic spectrum can be expressed in terms of wavelength and frequency. The wavelength (λ) and the frequency (ν) are related by the expression

$$\lambda = \frac{c}{\nu} \qquad \ldots(5.4.1)$$

Here c is the speed of light = 3×10^8 m/sec

Solved Example

Ex. 5.4.1 : Calculate the frequency of oscillation of green light.

Soln. :

It has been known that green light has a wavelength of approximately 500 nm

(500×10^{-9}m)

Its frequency of oscillations can be calculated using Equation (5.4.1).

$$\lambda = \frac{c}{\nu}$$

$$\nu = \frac{c}{\lambda} = \frac{3 \times 10^8 \text{ m/sec}}{500 \times 10^{-9} \text{ m}}$$

$$\nu = 6 \times 10^{14} \text{ Hz}$$

i.e., the frequency of green light is 600,000,000,000,000 cycles/sec !

Hence it is more convenient to discuss electromagnetic radiation in terms of wavelengths (nm) rather than frequencies (Hz).

5.5 Units of Intensity

- We know that for an object to be seen, some amount of light has to fall on it.
- The unit of luminous intensity (I) is candela (cd) (SI unit) which by definition, is $(1/60)^{th}$ of a square centimetre of the surface of a black body radiator at the absolute temperature of 2045 K.

Note : A black body radiator is one that absorbs all the radiation incident upon it and has no reflecting power. The radiation from a heated black body source is called black body radiation and is always quoted in Kelvin.

The unit is called candela because initially it was defined as a standard candle burning a specific amount of wax per hour. There is another important unit apart from candela that we encounter. This unit is called lux.

5.6 Basic Elements/Steps of an Image Processing System

MU - Dec. 2018

Q. Explain Fundamental steps in Image processing. **(Dec. 2018, 5 Marks)**

Digital image processing is basically modification of images on a computer. The basic components of an image processing system are as follows :

(1) Image Acquisition.
(2) Image Storage.
(3) Image Processing.
(4) Display.
(5) Transmission (if required).

We shall discuss each one in detail.

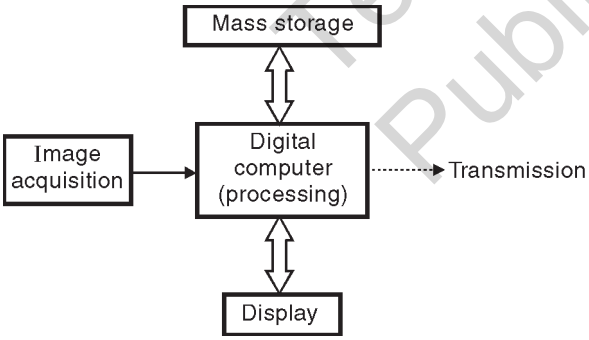


Fig. 5.6.1

(1) Image Acquisition

- Image acquisition is the first step in any image processing system. The general aim of image acquisition is to transform an optical image (real world data) into an array of numerical data which could be later manipulated on a computer.
- Image acquisition is achieved by suitable cameras. We use different cameras for different applications. If we need an X-ray image, we use a camera (film) that is sensitive to X-rays.
- If we want an infra-red image, we use cameras which are sensitive to infra-red radiations. For normal images (family pictures etc.) we use cameras which are sensitive to the visual spectrum. In this book we shall discuss cameras (sensors) which are sensitive only to the visual range.

Quantum detectors

- Quantum detection is the most important mechanism of image sensing and acquisition. It relies upon the energy of absorbed photons being used to promote electrons from their stable state to a higher state above an energy threshold.
- Whenever this occurs, the properties of that material get altered in some measurable way. Planck/Einstein came up with a relationship between the wavelength of the incident photon and the energy that it carries.

$$e = \frac{h\,c}{\lambda} \qquad \ldots(5.6.1)$$

Where, e = Energy carried by a photon
h = Planck's constant, 6.626×10^{-34} Js
c = Speed of light, 3×10^{8} m/sec
λ = Wavelength of the incident radiation.

- On collision, the photon transfers all or none of this quantum of energy to the electron.
- The Equation (5.6.1) is very important as it tells us that the maximum wavelength to which the quantum detector will respond is determined by the energy threshold and hence by the material selection.
- Of the several modes of operation of quantum detectors, the most important ones are

(a) Photoconductive
(b) Photovoltaic.

(a) Photoconductive : The resistance of photoconductive materials drop in the presence of light due to the generation of free charge carriers. An external bias is applied across the material to measure this change. This principle of photoconductivity is used in Vidicon imaging tubes.

Vidicon imaging tubes

- Vidicon is a vacuum tube used for image acquisition. Fig. 5.6.2 is the schematic diagram of the vidicon tube image sensor. The light image is focused on a transparent signal plate coated on the inside with a photoconductive target material.

- The front face of the signal plate is coated with SnO_2 (tin oxide) which forms the conducting layer. This plate is biased to a positive potential of about 50 V. The target coating is maintained at approximately 0 V by a mesh grid which is placed behind it.

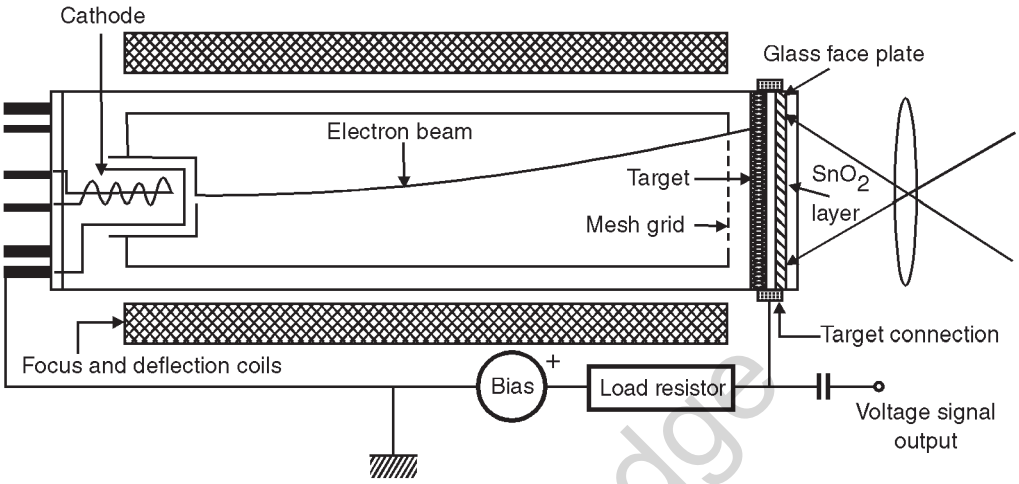


Fig. 5.6.2

- Due to this the signal plate and target behave as a continuous array of capacitors containing a photoconductive dielectric.
- When there is no light incident on the target, its resistance is very high due to which the charge across the capacitor is maintained. However when light strikes the target, the dielectric material becomes leaky and the target side voltage rises.
- The target is now scanned by a low velocity electron beam. When this beam strikes the discharged capacitor, current flows to restore the target side to its original voltage of 0 V.
- This current also flows in the external bias circuit, which produces a voltage drop in the load resistance.
- In order to produce the output necessary for broadcast video applications, the electron beam is scanned across the surface of the target by electromagnetic deflection coils. (Deflection coils were encountered when you studied CRO, (cathode ray oscilloscopes), in your lower semesters).
- The original vidicons use antimony-trisulphide, which is a photoconductive material, as their target coatings. Other forms of vidicon are made using more complex target materials.
- The silicon target vidicon achieves improved sensitivity and a spectral response similar to that of the human eye.
- The biggest drawback of the vidicon tube is that it uses the concept of deflection coils to position the electron beam. These deflection coils make the vidicon bulky and are responsible for major power drain. Since the coils are magnetic, the vidicon also becomes sensitive to external fields.

(b) Photovoltaic

- Photovoltaic devices consist of semiconductor junctions. They are solid state arrays composed of discrete silicon imaging elements known as *Photosites*.
- Photovoltaic devices give a voltage output signal that is proportional to the intensity of the incident light. No external bias is required as was in the case of photoconductive devices.
- The technology used in solid-state imaging sensors is based principally on charge-coupled devices, commonly known as Charged Coupled Devices CCDs. Hence the imaging sensors are called CCD sensors.
- The solid state array (CCD) can be arranged in two different configurations.

(i) Line array CCD.

(ii) Area array CCD.

(i) Line Arrays : The line array represents the simplest form of CCD imager and has been employed since the early 1970s. Line arrays consist of a one-dimensional array of photo sites.

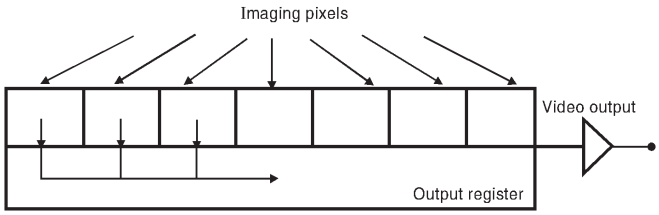


Fig. 5.6.3

- A single line of CCD pixels are clocked out into the parallel output register as shown in Fig. 5.6.3. The amplifier outputs a voltage signal proportional to the contents of the row of photosites.
- One thing to note is that line array CCD scans only one line (hence is one-dimensional). In order to produce a two-dimensional image, the line array CCD imager has to be used as a scanning device by moving this array over the object by some mechanical activity.

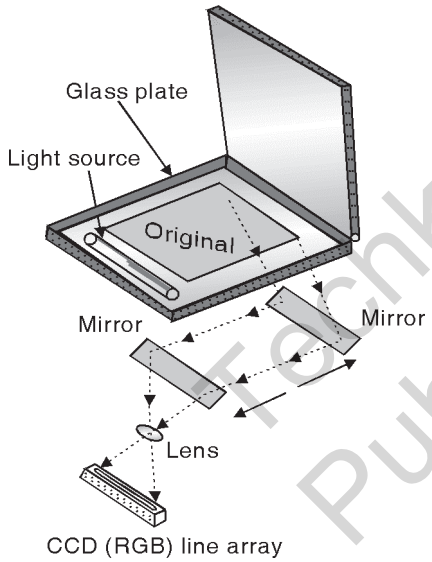


Fig. 5.6.4

- This technique is used in flat bed scanners (the scanners that you come across in your laboratory or in a cyber cafe). A line array CCD can have anything from a few elements to upto 6000 or more.

(ii) Area Arrays : The problem with line arrays is that it scans only one line. To get a two-dimensional image, we need to mechanically move the array over the entire image.

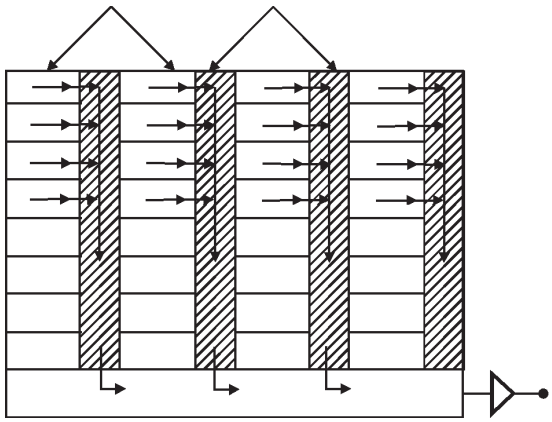


Fig. 5.6.5

- Area arrays or matrix arrays consist of a two-dimensional array of photosites. They make it possible to investigate static real world scenes without any mechanical scanning. Thus much more information can be deduced from a single real time glance than would be possible with line arrays.
- Area arrays can be seen in all the digital cameras that we use for video imaging. The area arrays are more versatile than the line arrays, but there is a price to be paid for this. Area arrays are higher on cost and complexity.
- Area sensors come in different ranges. i.e. 256×256, 490×380, 640×480, 780×575. CCD arrays are typically packaged as TV cameras.
- A significant advantage of solid state array sensors is that they can be shuttered at very high speeds (1/10,000 secs). This makes them ideal for applications in which freezing motion is required.

(2) Image Storage

- All video signals are essentially in analog form i.e. electrical signals convey luminance and colour with continuously variable voltage. The cameras are interfaced to a computer where the processing algorithms are written.
- This is done by a frame grabber card. Usually a frame grabber card is a Printed Circuit Board (PCB) fitted to the host computer with its analog entrance port matching the impedance of the incoming video signal.
- The A/D converter translates the video signals into digital values and a digital image is constructed. Frame grabber cards usually have a A/D card with resolution of 8 - 12-bits (256 to 4096 gray levels). Hence a frame grabber card is an interface between the camera and the computer.

- Frame grabber card has a block of memory, separate from the computers own memory, large enough to hold any image. This is known as the *frame buffer memory*. Image data, usually in the form of bytes (8-bits), is written into the frame grabber memory under the computer control, usually by DMA transfers.
- The contents of the memory are continuously read out at video rate (30 frames/sec), passed through the D/A converter and displayed on the monitor.
- The output has a colour map or a colour Look Up Table (LUT) to permit pseudo colouring. The table consists of 8-bit values for each of the red, green and blue guns of the monitor.
- The commercially available frame grabber cards have additional features such as ability to zoom regions of the image and pan the images.
- The frame buffer memory can be upto 5 MB. Images being 2-dimensional functions, occupy a lot of space and hence the storage space on the host computer has to be large. Let us try to find out as to how much space is actually taken by images.

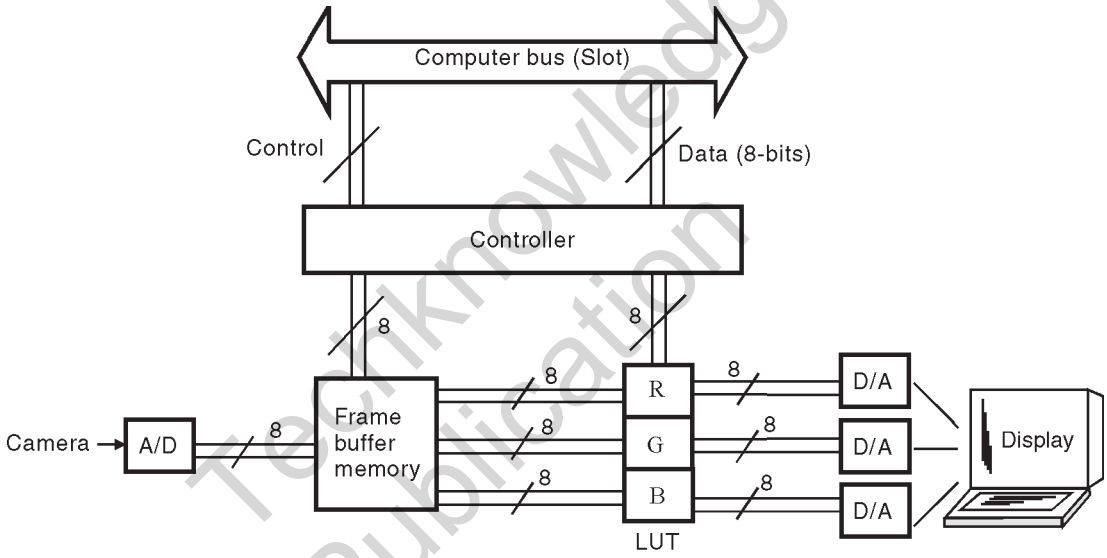


Fig. 5.6.6 : Simplified diagram of a frame grabber card

- Each image is stored as a matrix, where every value of the matrix represents the grey level at that point. Suppose the image (matrix) is of size A × B and suppose we require C bits to represent each element of the matrix. Memory space required to store this image is approximately equal to A × B × C bits.
- Suppose we have D such images then total number of bits ≈ A × B × C × D ...(5.6.2)

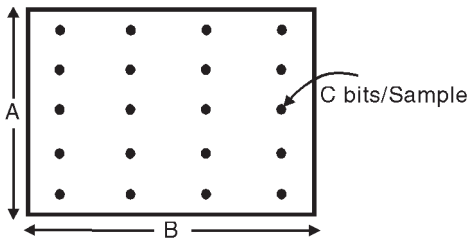


Fig. 5.6.7 (a)

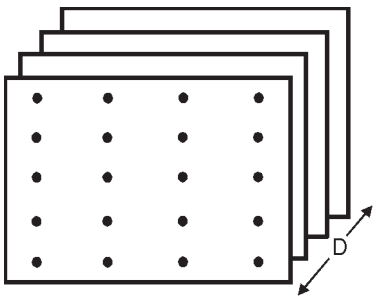


Fig. 5.6.7(b)

- Thus the formula ≈ ABCD gives us a fairly good idea of the amount of space required.

Table 5.6.1

	A	B	C	D	Number of bits
Low quality video phone	64	64	8	6	$\approx 0.2 \times 10^6$ bits
Digital broadcast TV	720	576	25	8	$\approx 83 \times 10^6$ bits
HDTV	1920	1150	50	8	$\approx 883 \times 10^6$ bits

- With the advent of the PCI bus, the host computers memory could now be used as the frame buffer memory. This is because the PCI bus facilitates fast communication.
- Hence the image could now directly be stored on the computer RAM. Due to this, the frame buffer memory has become more or less obsolete. Only the very high end frame grabber cards still have a small frame grabber memory embedded on them.
- Images being two-dimensional functions, occupy a lot of space and hence it is advisable to have a computer with a sizeable hard disk and also a good RAM. Processed images could be stored on magnetic floppies or CDs. Archival data are stored on magnetic tapes.

(3) Image Processing

- Systems ranging from microcomputers to general purpose large computers are used in image processing. Dedicated image processing systems connected to host computers are very popular now-a-days.
- Processing of digital images involve procedures that are usually expressed in algorithmic form due to which most image processing functions are implemented in software.
- The only reason for specialized image processing hardware is the need for speed in some applications or to overcome some fundamental computer limitations.
- The trend though is to merge general purpose small computers with image processing hardware. As stated in the earlier section, frame grabber cards play the important role of merging the image processing hardware with the host computer.
- One thing that we should remember is, image processing is characterized by specific solutions. Hence there is no one way to process images. A technique that works well in one area can be totally useless in some other applications.
- The image processing software that is used in this book is MATLAB.

(4) Display

- A display device produces and shows a visual form of numerical values stored in a computer as an image array. Principal display devices are printers, TV monitors and CRTs.
- Any erasable raster graphics display can be used as a display unit with an image processing system. However monochrome and colour TV monitors are the principal display devices used in modern image processing systems.
- These raster devices convert image data into a video frame. One major problem is, it must refresh the screen at a rate of about 30 frames per second to avoid flicker.
- Since some computers are unable to transfer data at such high speeds, it is a good idea to buy a high end frame grabber card which has frame buffer memory on it.
- If you want to build a image processing the system a home, you should have a Video Graphics Card (VGA) attached to a computer and a monitor that supports that VGA card. One could then use a simple camera that comes along with the system.

(5) Transmission

- There are a lot of applications where we need to transmit images. The stages in the transmission of an image over a channel or network are shown in Fig. 5.6.8.

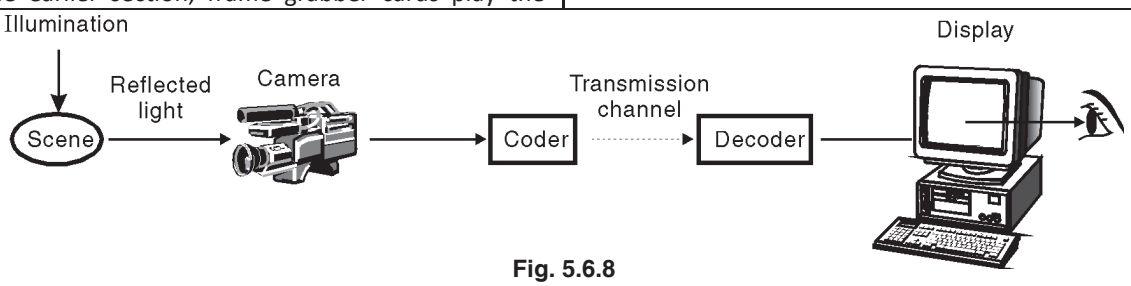


Fig. 5.6.8

- The image sequence from the camera is coded into as concise a representation as possible for transmission over the channel.
- Most transmission in broadcast television are still in analog form and analog coding methods are used to make it as efficient as possible. NTSE, PAL and SECAM are the 3 major coding systems used in various parts of the world (USA uses NTSE, while India uses PAL).
- Digital image coding is a high activity area. In image processing; it is concerned with efficient transmission of images over digital communication channels. A variety of indigenous ideas have been developed in recent years.
- Coding is influenced by the type of the channel used to carry the image signals. Several different types of transmission channels are encountered in practice including cables, terrestrial radio, satellites, and optical fibres. After decoding at the receiver, the image sequence is
- **Gamma :** In the image transmission chain, there are many elements which exhibit a non-linear behaviour. If x is the input and y is the output, then

$$y = cx^{\gamma}$$

where, c = Constant

γ = Gamma of the device

- The camera, the display device and eye all have non-unity gammas (all are non-linear). Hence to make sure that the perceived grey scale in the displayed image is correct, it is necessary to insert an additional compensating, non-linear device called the gamma corrector.
- We have discussed a lot of things so far and it is advisable at this stage to have a branch diagram of these topics.

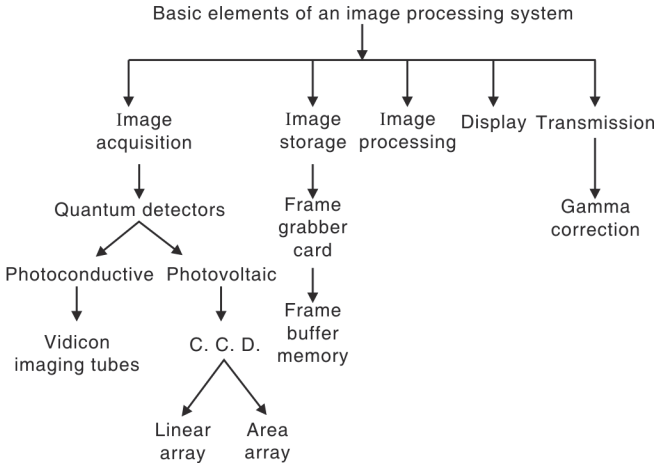


Fig. 5.6.9

5.7 Image Types

Images are 2-dimensional functions, as shown in Fig. 5.7.1.

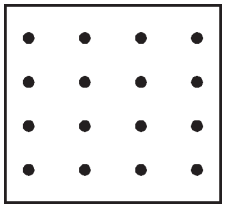


Fig. 5.7.1

Images can be classified as follows :

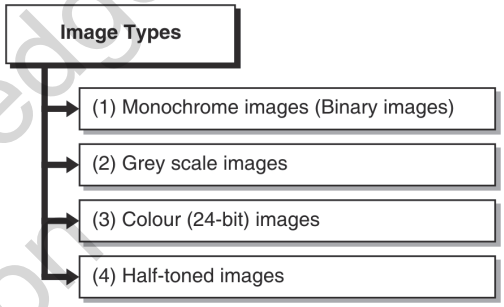


Fig. 5.7.2 : Image Types

(1) Monochrome Image : In this, each pixel is stored as a single bit (0 or 1). Here, 0 represents black while 1 represents white. It is a black and white image in the strictest sense. These images are also called bit mapped images. In such images, we have only black and white pixels and no other shades of grey. Refer Fig. 5.7.3.

Fig. 5.7.3

(2) Grey Scale Image : Here each pixel is usually stored as a byte (8-bits). Due to this, each pixel can have values ranging from 0 (black) to 255(white). Grey scale images, as the name suggests have black, white and various shades of grey present in the image. Refer Fig. 5.7.4.

Fig. 5.7.4

(3) Colour Image (24-bit)

- Colour images are based on the fact that a variety of colours can be generated by mixing the three primary colours viz, Red, Green and Blue, in proper proportions. In colour images, each pixel is composed of RGB values and each of these colours require 8-bits (one byte) for its representation.
- Hence each pixel is represented by 24-bits [R(8-bits), G(8-bits), B(8-bits)].

  A 24-bit colour image supports 16, 777, 216 different combination of colours.
- Colour images can be easily converted to grey scale images using the equation

$$X = 0.30\,R + 0.59\,G + 0.11\,B \quad \ldots(5.7.1)$$

- An easier formula that could achieve similar results is

$$X = \frac{R + G + B}{3} \quad \ldots(5.7.2)$$

MATLAB code for converting a colour image to a grey scale image is as follows :

```
%% Converting a colour image to a grey level image %%
clear
clc
im = imread('lily.tif');
[row col byt]=size(im);
a = im(:,:,1); % Red plane
b = im(:,:,2); % Green plane
c = im(:,:,3); % Blue plane
a = double(a);
b = double(b);
c = double(c);
for x = 1:1:row
for y = 1:1:col
        new(x,y)  = (a(x,y)+b(x,y)+c(x,y))/3;
        new1(x,y) = 3*a(x,y)+0.59*b(x,y)+0.11*c(x,y);
end
end
figure(1)
imshow(uint 8(im))
figure(2)
imshow(uint 8(new))
figure(3)
imshow(uint8(new1))
```

Matlab has an inbuilt command for conversion rgb2gray.

(4) Half Toning

- It is obvious that a grey scale image definitely looks better than the monochrome image as it utilizes more grey levels. But there is a problem in hand.
- Most of the printers that we use (inkjet, lasers, dot matrix) are all bi-level devices. i.e., they have only a black cartridge and can only produce two levels (black on a white back-ground). In fact, most of the printing jobs are done using bi-level devices.
- You have all read newspapers at some point of time (hopefully). The images do look like grey level images. But if you look closely, all the images generated are basically using black colour. Refer Fig. 5.7.5.

Fig. 5.7.5

- Even the images that you see in most of the books (including this one) are generated using black colour on a white background. In spite of this we do get an illusion of seeing grey levels. The technique to achieve an illusion of grey levels from only black and white levels is called half-toning.

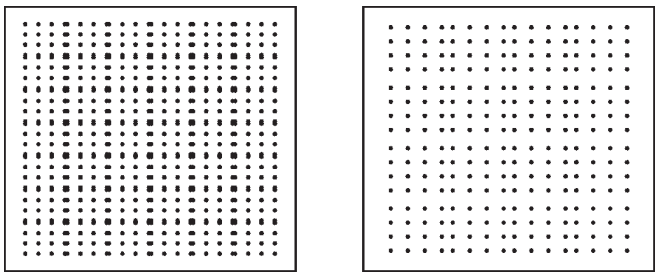


Fig. 5.7.6

- The human eye integrates the scene that it sees. Consider a simple example. Consider two squares of say 0.03×0.03 sq.inch.
- One of these squares contains a lot of black dots while the other square contains fewer black dots. When we look at these squares from a distance, the two squares give us a perception of 2 different grey levels.
- This integration property of the eye is the basis for half toning. In this, we take a matrix of a fixed size and depending on the grey level required, we fill the matrix with black pixels.

  Let us take an example.
- Consider a 3×3 matrix. This matrix can generate an illusion of 10 different grey levels when viewed from a distance.

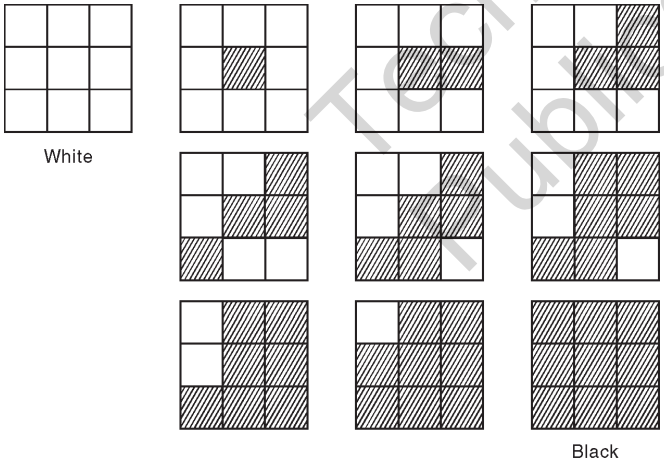


Fig. 5.7.7

- Here, the first block represents white, the last block represents black and all the other blocks represent intermediate values of grey. So in this case, while printing, if we encounter grey level 0, we plot 9 pixels which are all white. If we encounter grey level 1, we plot 9 pixels of which only one pixel is black and so on.
- As is evident a 3×3 matrix will generate 10 different grey levels.
- The dots that are placed in the 3×3 matrix example can be in any order. But we need to follow two rules.

(1) Dots should be arranged in such a manner so that they don't form straight lines. Lines are very obvious to the viewer and hence should be avoided.

Example, suppose in an image we have 4 consecutive grey levels of value 3, and if we use a code as shown in Fig. 5.7.8, the half-toned image would look like Fig. 5.7.9.

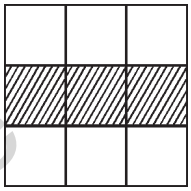


Fig. 5.7.8

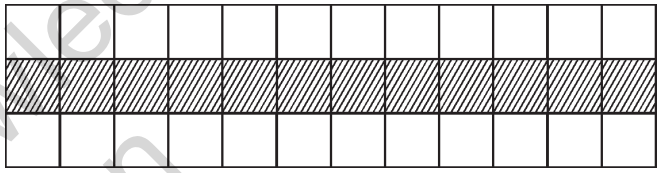


Fig. 5.7.9

It can be seen that it does not look like a grey level. It looks like a straight black line. Hence lines should be avoided while defining the matrices.

(2) If a pixel in a particular matrix is black for grey level i, it should be black for all further levels j > i. This reduces false contouring.

Half-toning gives excellent results and we do perceive a grey level image just by using black pixels on a white background.

The logic to implement a half-toned image from a grey level image is as follows :

(i) Define the size of the half-toned matrices based on the number of grey levels the original image has.

(ii) Generate the matrices starting from all white pixels to all black pixels.

(iii) Read the original image. For every grey level value read, plot the corresponding matrix.

Remember, the physical size of the half-toned image will always be bigger than the original image as for every single grey level value, we output an entire matrix.

5.8 Image File Formats

MU : May 2016, Dec. 2016

Q. Explain in detail any two types of image file formats. **(May 2016, 8 Marks)**

Q. What are various file formats ? Explain each in brief. **(Dec. 2016, 8 Marks)**

- Images obtained from the camera are stored on the host computer using different formats. A file format is a structure which defines how information is stored in the file and how that information is displayed on the monitor.
- There are numerous image file formats which are available. Of these only a few of them can be used universally. By universally it means, they can be understood by different operating systems.
- Some of the image formats that can be used on either Macintosh or PC platforms are;

BMP (Bit Mapped Graphic Image)	JPEG (Joint Photographic Expert Group)
TIFF (Tagged Image File Format)	EPS (Encapsulated Post Script)
GIF (Graphic Interchange Format)	PICT (Macintosh Supported)

- TIFF, which is one of the most well known formats was developed in 1986 and in its many versions is a standard image format for a bit-mapped graphics image. The TIFF format is also a data compression technique for monochrome as well as colour images. Images seen on the Internet sites are normally TIFF/GIF images simply because they occupy less space. GIF uses a form of Huffman coding.
- BMP images developed by Microsoft can save both monochrome as well as colour images. All the wall papers that your computer has are BMP images.
- Similarly when we work in Paint Brush, we can save our work as a BMP image. The quality of BMP files is very good but they occupy a lot of memory space. PICT is a general purpose format supported by Macintosh and used for storing bit-mapped images.
- EPS is file format of the post script page description language and is device independent. This simply means that images can readily be transferred from one application to another. However EPS images can only be printed on post script compatible printers.
- JPEG (Joint Photographic Expert Group) is the name of the committee that developed an image format which used compression algorithms.
- JPEG images are compressed images which occupy very little space. It is based on the Discrete Cosine Transform-DCT (explained in chapter of Image Transforms). JPEG images use lossy compression algorithms which result in some loss of original data and hence the quality of JPEG images is not as good as BMP images.
- Although these formats differ in technical details, they share structural similarities.
- The Fig. 5.8.1 shows the typical organisation of information encoded in an image file.

  The image file consists of two parts;

  (1) Header

  (2) Image data

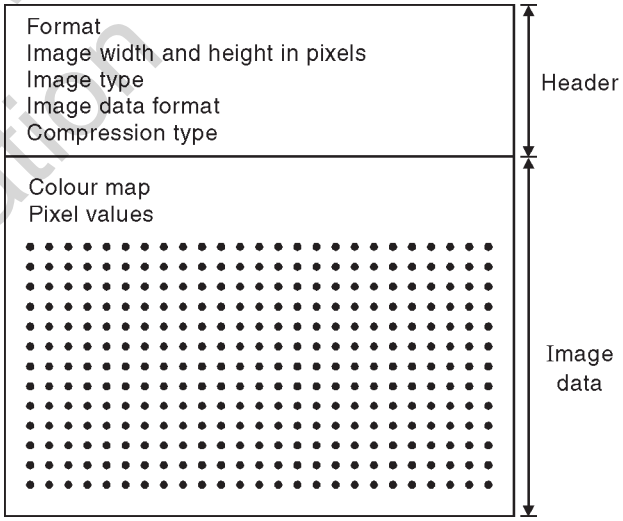


Fig. 5.8.1

- The header file gives us the information about the kind of image. The header file, begins with a binary code or ASCII string which identifies the format being used.
- The width and height of the image are given in number of pixels. Common image types include binary images, 8-bit grey scale images, 8-bit colour and 24-bit colour images.
- The image data format specifies the order in which pixel values are stored in the image data section. A commonly used order is left to right and top to bottom. Image data format also specifies if the RGB values in the image are interlaced.

- By interlaced we mean that the RGB values of each pixel stay together consecutively followed by the three colour components for the next pixel i.e.,

  R G B R G B R G B ……

  1st Pixel 2nd Pixel .....

- If the RGB values are not interlaced, the values of one primary colour for all pixels appear first, then the values of the next primary colour followed by the values of the third primary colour. Thus the image data is in the form.

  R R R R..... G G G G..... B B B B.....

- The compression type in the header indicates whether the image data is compressed using algorithms such as Run length encoding. Apart from these, there are a few more formats which we come across while dealing with images.

PGM (Portable Grey Map)	PGM is used to represent grey level images.
PBM (Portable Bit Map)	PBM is used to represent binary images
PPM (Portable Pixel Map)	PPM is used to represent RGB colour images

These formats are distinguished by 2 character signatures as follows :

Signature	Image type	Storage type
P1	Binary (PBM)	ASCII
P2	Grey scale (PGM)	ASCII
P3	RGB (PPM)	ASCII
P4	Binary (PBM)	Raw bytes
P5	Grey scale (PGM)	Raw bytes
P6	RGB (PPM)	Raw bytes

Most often the storage type is ASCII.

- The header of a PGM, PBM or PPM file begins with the appropriate signature followed by a blank line. There is a comment line after the signature which starts with the **#** character. Next in the header are the width and height of the image as ASCII decimal values.
- PBM files have no further header information. PGM and PPM files contain a third integer value, again in ASCII decimal form, representing the maximum allowable pixel value.
- From this value, we could find out the number of bits allocated for each pixel. For example, if the maximum allowable pixel value is 255 then we know that each bit is represented by 8 bits ($2^8 = 256$).
- Fig. 5.8.2 shows a 8 × 8 grey scale image and its representation as a ASCII PGM file

Fig. 5.8.2

Header
```
P2
#A PGM image
8  8  255
120 120 120 120 120 120 120 120
120 120 120 120 120 120 120 120
33  33  33  33  33  33  33  33
33  33  33  33  33  33  33  33
120 120 120 120 120 120 120 120
120 120 120 120 120 120 120 120
33  33  33  33  33  33  33  33
33  33  33  33  33  33  33  33
```

- Just by observing the header, we know that it is a PGM file. P2 indicates that it is a grey level image stored is ASCII. The 8 8 255 below the comment line indicates that the size of the image is 8 × 8 and can have 256 grey levels i.e., each value is represented by 8-bits.

5.9 Image Processing, Computer Graphics and Computer Vision

- Before we end, one issue needs to be sorted out. A very common question that people ask is what is the difference between image processing, computer graphics and computer vision ?
- Image processing deals with manipulation of images. A input image is modified into a new image. Computer graphics deals with creation of images. In computer graphics, models (2D or 3D) are created using mathematical functions (Descriptors).

– Computer vision which is also known as Machine vision deals with the analysis of image content. Computer vision is used to automate a process.

Table 5.9.1

Input \ Output	Image	Description (Mathematical function)
Image	I.P.	C.V.
Description (Mathematical function)	C. G.	–

From the Table 5.9.1,

Image processing → Input (Image) – Output (Image)

Computer graphics → Input (Description) – Output (Image)

Computer vision → Input (Image) – Output (Description)

The material provided in this chapter is primarily basic information which would be required in subsequent discussions. Our study of the human visual system, though not exhaustive, provides a basic idea of the capabilities of the eye in perceiving pictorial information.

Summary

In this chapter, preliminary concepts of digital image processing are presented. Difference between one-dimensional and two-dimensional signals is explained. Topics such as electromagnetic spectrum are discussed with examples.

The concepts explained here will be found useful in understanding image processing algorithms in subsequent chapters. This chapter forms the fundamental base required to understand image processing. Image acquisition forms the first step in any image processing system. In this chapter, a basic block diagram of the image processing system is introduced. Each block in the system is explained in detail. Important devices such as Vidicon cameras, CCD cameras and the frame grabber card are explained using simple illustrations. Importance of each is stated. Applications of line array CCD's and area array CCD's are presented. Basic concepts of image formats are also explained.

Review Questions

Q. 1 What do we mean by image processing ?

Q. 2 Why do we say that an image is a 2D-function ?

Q. 3 Calculate the frequency of oscillation of red light.

Q. 4 Distinguish between image processing and graphics.

Q. 5 List the basic elements of an image processing system.

Q. 6 Explain working of a Vidicon camera.

Q. 7 Explain line array and area array CCD cameras.

Q. 8 Write a short note on frame grabber card.

Q. 9 Explain the concept of half toning.

Q. 10 Explain image formats.

Q. 11 Advantages of CCD over Vidicon.

Q. 12 Explain the basics of quantum detector.

Q. 13 Explain gamma of a camera.

Q. 14 How many grey levels will a half toned image have ? Explain in detail.

❑❑❑

CHAPTER 6

Sampling and Quantization

Syllabus :

Sampling and Quantization. Representation of Digital Image

6.1 Introduction

- We know that an image is basically a 2-dimensional representation of the 3-dimensional world. We have also studied that images can be acquired using a Vidicon or a CCD camera or using scanners.
- The basic requirement for image processing is that images obtained be in the digital format. For example, one cannot work with or process photographs on identity cards unless he/she scans it using a scanner.
- The scanner digitizes the photograph and stores it on the hard disk of the computer. Once this is done, one can use image-processing techniques to modify the image as per requirement. In a Vidicon too, the output which is in analog form needs to be digitized in order to work with the images.
- To cut a long story short, to perform image processing, we need to have the images on the computer. This will only be possible when we digitize the analog pictures.
- Now that we have understood the importance of digitization, let us see what this term actually means.
- The process of digitization involves two steps :

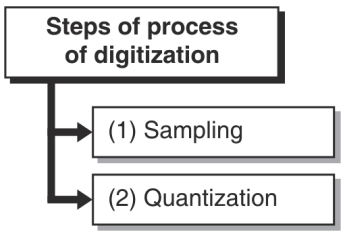


Fig. 6.1.1 : Steps of process of digitization

In other words, Digitization = Sampling + Quantization.

- We have had exposure to these terms in the lower semesters in subjects like Principles of Communication Engineering, Signals and Systems and Signal Processing which dealt with 1-dimensional signals. Let us take a brief look at these concepts and move ahead to the 2-dimensional domain.

6.2 Sampling

MU - May 2012, May 2013, Dec. 2013.

Q. Write short note on : Sampling and Quantization.

(May 2012, May 2013, Dec. 2013, 5 Marks)

- Consider a band-limited signal. Although real world signals are rarely band limited, we can produce one by passing the signal first through a low-pass filter known as an anti-aliasing filter.

Note : A signal is said to be band limited, if its Fourier transform (FT), F(u), is zero outside a bounded region.

- (To understand sampling, we first need to understand convolution involving impulse functions). If f(t) is a step signal shown in Fig. 6.2.1 and g(t) is a train of impulses (g(t) is also known as a COMB function), then the convolution of the two is given by f(t) * g(t). Let us see what happens when we convolve f(t) with g(t).

Proof : We know that the convolution of f(t) and g(t) is

$$y(t) = \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau = \int_{-\infty}^{\infty} g(\tau)\, f(t-\tau)\, d\tau$$

$$y(t) = \int_{-\infty}^{\infty} [\delta(t-T) + \delta(t) + \delta(t+T)]\, f(t-\tau) d\tau$$

$$y(t) = \int_{-\infty}^{\infty} \delta(t-T)\, f(t-\tau)\, d\tau + \int_{-\infty}^{\infty} \delta(t)\, f(t-\tau)\, d\tau + \int_{-\infty}^{\infty} \delta(t+T)\, f(t-\tau)\, d\tau$$

but, $\int_{-\infty}^{\infty} \delta(t-T)\, f(t-\tau)\, d\tau = \delta(t-T) * f(t)$

[This is impulse at t = –T convolved with f(t)]

and, $\int_{-\infty}^{\infty} \delta(t + T)\, f(t - \tau)\, d\tau = \delta(t + T) * f(t)$

[This is impulse at t = +T convolved with f(t)]

This gives us the figures as shown in Fig. 6.2.1.

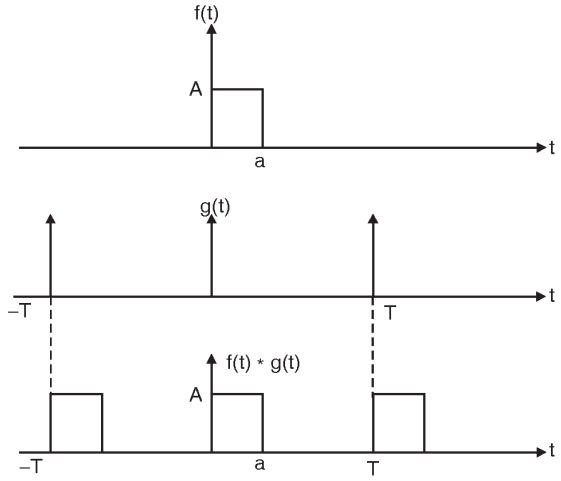


Fig. 6.2.1

- We see that convolution replicates the signal f(t) at every impulse g(t).
- Another important property that one needs to know is that convolution in one domain is equal to multiplication in the other i.e. convolution in the time domain is equal to multiplication in the frequency domain and similarly multiplication in the time domain is equal to convolution in the frequency domain.
- As is seen in Fig. 6.2.2, if we use Fig. 6.2.2(c) as the sampling period, we encounter overlapping or aliasing in the frequency domain [Fig. (f)]. This is due to undersampling. If we decrease the sampling period (increase the sampling frequency) as in Fig. 6.2.2(g), we can eliminate aliasing in the frequency domain [Fig. 6.2.2(h)].
- If we now multiply Fig. (h) with a gating function H(u) [Fig. 6.2.2(i)] we can select only the required part of the spectrum [Fig. 6.2.2(k)]. We take the inverse Fourier transform to get back the original signal f(t) [Fig. 6.2.2(j)].
- All this was possible because the sampling period was small (sampling frequency was large). Hence to avoid aliasing, the sampling frequency should be greater than or equal to 2 × Bandwidth (2 × w).
- In other words, $\Delta t \leq \frac{1}{2w}$. This is the Whittaker-Shannon sampling theorem. Only if this condition is met, can we recover the original signal. These 1-D concepts can be easily extended to the 2-D domain. A function f(x, y) is called a band limited function if its Fourier transform F(u, v) is zero outside a bounded region in the frequency plane i.e.

$$F(u, v) = 0 \quad u > u_0, v > v_0$$

- Let us consider a band limited signal as shown in Fig. 6.2.2.

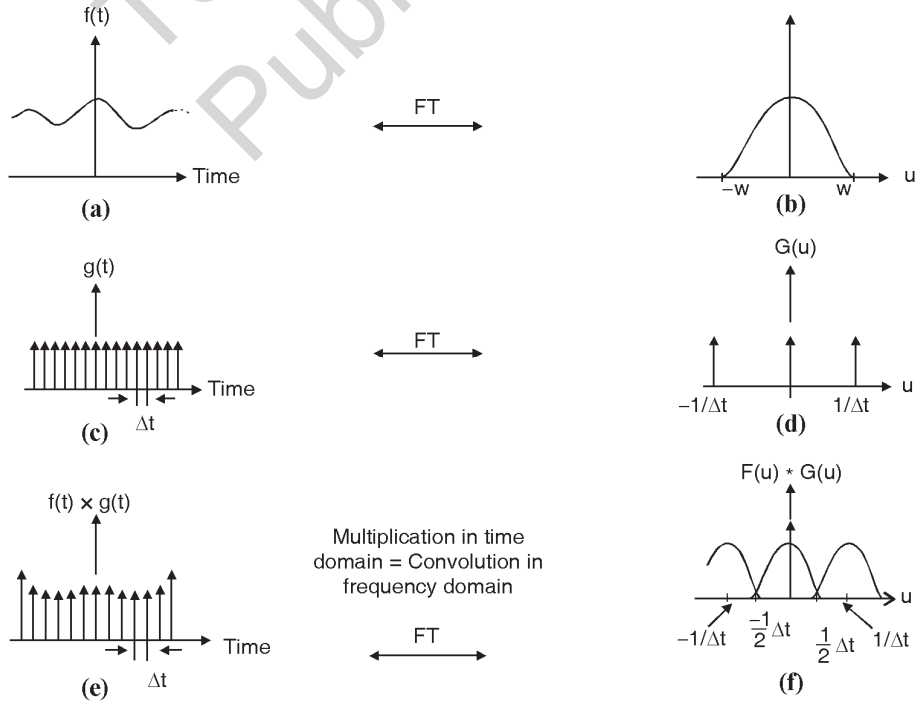


Fig. 6.2.2 contd....

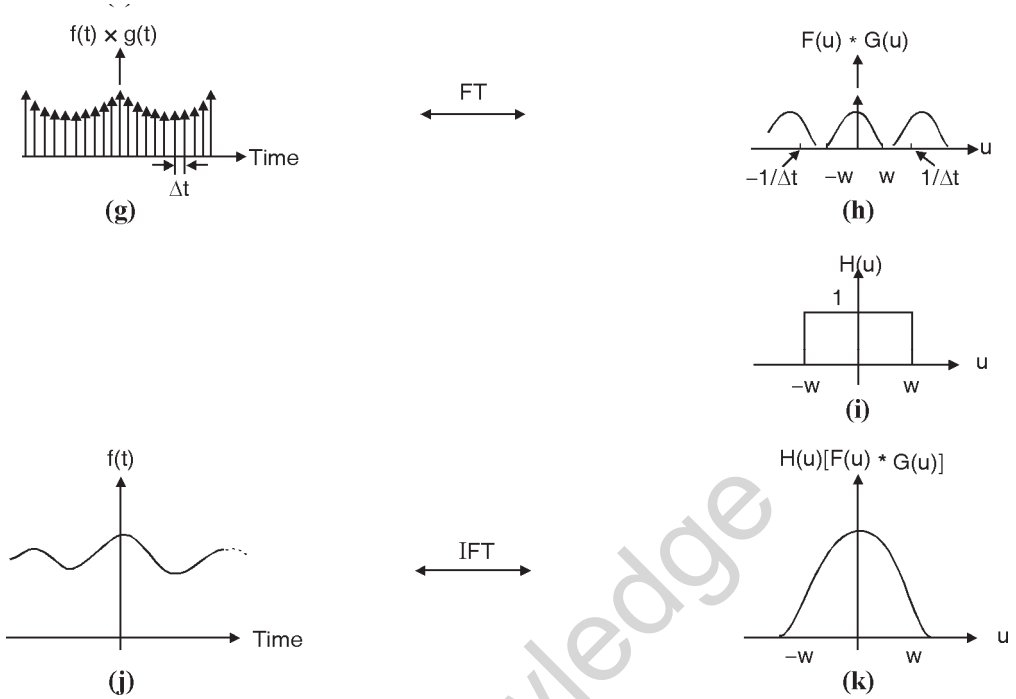


Fig. 6.2.2

– Fig. 6.2.3 (a) and (b) represent the Fourier Transform of a band limited signal and its region of support (the base) respectively.

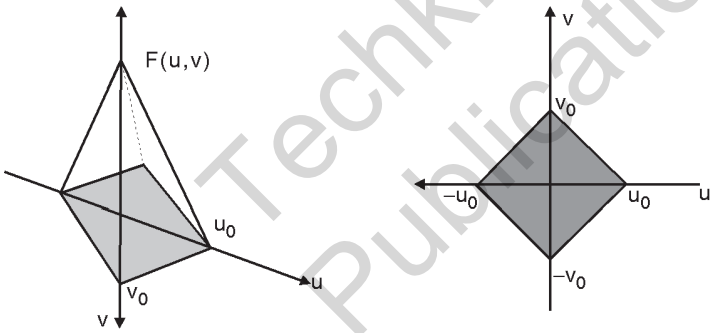


Fig. 6.2.3(a) : Fourier transform of a band limited signal

Fig. 6.2.3(b) : Its region of support (the base)

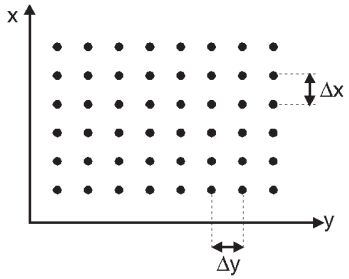
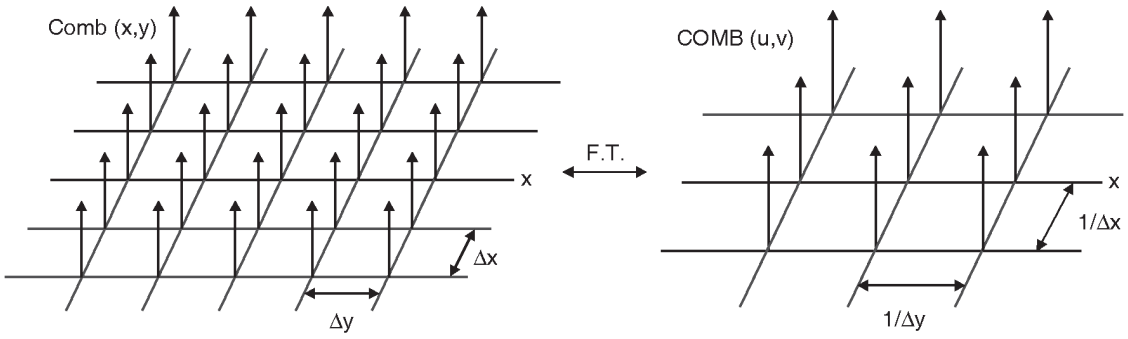


Fig. 6.2.4 : Two dimensional comb function



(a) 2-D comb function **(b) Fourier transform of COMB function**

Fig. 6.2.5

– Just as in the case of a 1-D function, consider a 2-D comb function (sampling function) [Refer Fig. 6.2.4]

A 2-dimensional comb function is defined as

$$\text{Comb}(x, y; \Delta x, \Delta y) \overset{\Delta}{=} \sum_{m, n = -\infty}^{+\infty} \sum \delta(x - m\,\Delta x, y - n\,\Delta y)$$

…(6.2.1)

– The top view of the comb function is given in Fig. 6.2.4. Another way of representing a 2-D function or a sampling function is shown in Fig. 6.2.5(a).

– The Fourier transform of the comb function with spacing of Δx and Δy is another COMB function with spacing $1/\Delta x$ and $1/\Delta y$.

$$\text{Comb}(u, v) = \text{FT}\{\text{comb}(x, y; \Delta x, \Delta y)\}$$
$$= \text{COMB}(u, v; 1/\Delta x, 1/\Delta y)$$

The subsampled image is defined as

$$f_s(x, y) = f(x, y) \times \text{comb}(x, y; \Delta x, \Delta y)$$

$$f_s(x, y) = \sum_{m, n = -\infty}^{+\infty} \sum f(m\,\Delta x, n\,\Delta y)\,\delta(x - m\,\Delta x, y - n\,\Delta y)$$

…(6.2.2)

– Since multiplication in the time domain is equal to convolution in the frequency domain, we get

$$F_s(u, v) = F(u, v) * \text{COMB}(u, v) \quad ...(6.2.3)$$

– We know from the 1-D case, $F_s(u, v)$ will simply be replicating F(u, v) at all impulses. Hence for a 2-D case, using Fig. 6.2.3(b), we have,

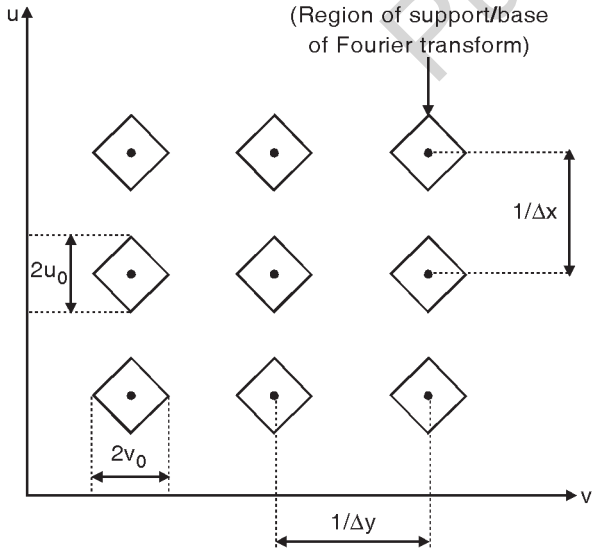


Fig. 6.2.6

– To avoid aliasing, the sampling frequency should be twice the maximum frequency in the signal (image)

$$\Delta x \leq \frac{1}{2u_0}$$

$$\text{and } \Delta y \leq \frac{1}{2v_0} \quad ...(6.2.4)$$

Where, $2u_0$ and $2v_0$ are the bandwidths in the u and v directions respectively. This is known as the Nyquist condition.

$\Delta x = \frac{1}{2u_0}$ and $\Delta y = \frac{1}{2v_0}$ are the lower bounds of the equation and are known as the Nyquist rate.

As seen from the Fig. 6.2.6, there is absolutely no aliasing. If the Nyquist condition is not met, overlapping of frequencies occurs and this is known as aliasing. This overlapping is shown in Fig. 6.2.7.

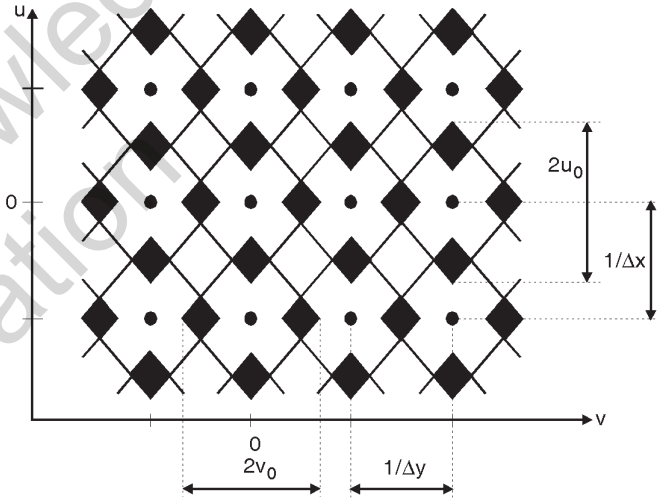


Fig. 6.2.7 : Aliasing due to low sampling frequency

– The shaded portions are where overlapping (aliasing) takes place.

– We observe aliasing when we go to watch movies. Whenever the hero of the movie (could be the villain too) races his car in the climax, have you ever observed the wheels of his car ?

– When the car is slow, the wheels move in the same direction as his car. But what happens when he speeds up his car ? The wheels start moving in the opposite direction. This is because; the camera, which is shooting this scene, has a particular maximum sampling frequency.

– As the car speeds up the Nyquist condition is not met and hence there is aliasing, i.e. high frequencies start looking as low frequencies.

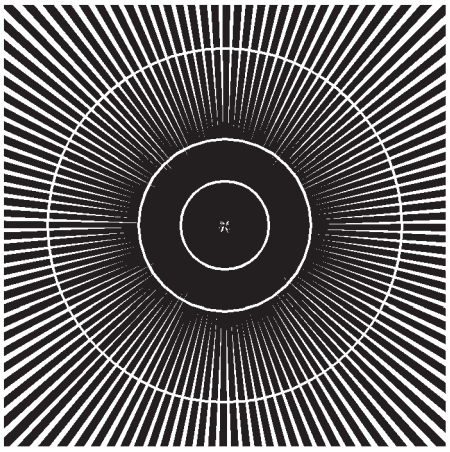


Fig. 6.2.8(a) : Original image

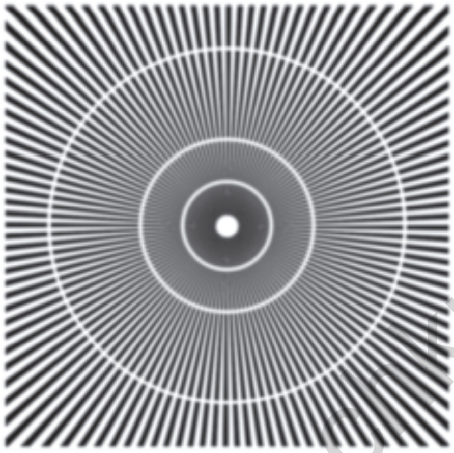


Fig. 6.2.8(b) : Sampled image

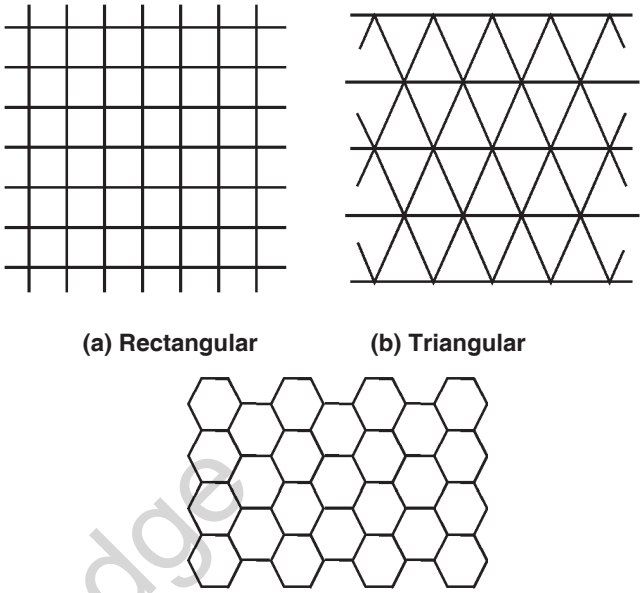
Coming back to the summation Equation (6.2.2),

$$f_s(x,y) = \sum_{m,\,n\,=-\infty}^{\infty} \sum f(m\,\Delta x,\, n\,\Delta y)\,\delta(x - m\,\Delta x,\, y - n\,\Delta y)$$

The equation suggests that a sampled 2-D function defined over a domain of size

M $\Delta x \times$ N Δy can be considered as a 2-D array {f(0,0), f(0,1),……}

- This means that the function is represented as an M $\times$ N uniformly spaced sample. Apart from the sampling interval in the x and y directions, another important choice relevant to the image sampling is the spatial arrangement of the sample points called Tessellation.
- We normally consider a comb function, which is a rectangular tessellation of grid. However other types of tessellations of the image plane, namely triangular and hexagonal, may also be used. Throughout this book though, we use rectangular tessellations.



(a) Rectangular **(b) Triangular**

(c) Hexagonal

Fig. 6.2.9 : Commonly used sampling grids

6.3 Quantization

MU - May 2012, May 2013, Dec. 2013

Q. Write short note on : Quantization.

(May 2012, May 2013, Dec. 2013, 5 Marks)

- This is also a topic that has been widely discussed in the lower semesters. The values obtained by sampling a continuous function usually comprise of an infinite set of real numbers ranging from a minimum to a maximum depending upon the sensors calibration.
- These values must be represented by a finite number of bits usually used by a computer to store or process any data. In practice, the sampled signal values are represented by a finite set of integer values. This is known as quantization. Rounding of a number is a simple example of quantization.
- With these concepts of sampling and quantization, we now need to understand what these terms mean when we look at an image on the computer monitor.
- Higher the spatial resolution of the image, greater is the sampling rate i.e. lower is the image area $\Delta x\ \Delta y$ represented by each sampled point. Similarly, higher the grey level resolution (tonal resolution) more are the number of quantized levels.
- Hence spatial resolution gives us an indication of the sampling while grey level resolution (tonal resolution) gives us an indication of the quantization.

Spatial resolution $\longrightarrow$ Sampling

Grey level resolution $\longrightarrow$ Quantization

– We have already stated that an image can be considered as a 2-D array. Image f(x,y) is arranged in the form of N × M array

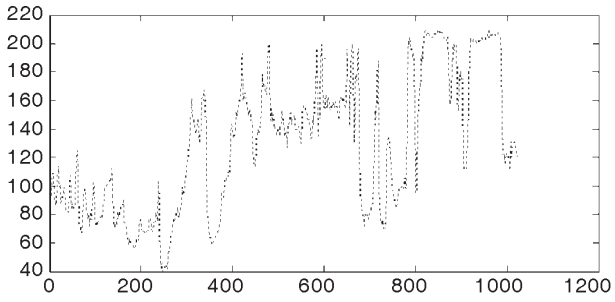
$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \ldots & f(0,M-1) \\ f(1,0) & f(1,1) & \ldots & f(1,M-1) \\ f(2,0) & f(2,1) & \ldots & f(2,M-1) \\ \vdots & & & \vdots \\ f(N-1,0) & f(N-1,1) & \ldots & f(N-1,M-1) \end{bmatrix}_{N \times M}$$

– Hence every image that is seen on the monitor, is actually this matrix. Each element of the matrix is called a *pixel*. Never forget this.

– Whenever we see an image on the screen of the computer it is actually a matrix which consists of N × M pixels and each pixel is considered to be a sample. Hence more the pixels, more the samples, higher the sampling rate and hence better the spatial resolution. The value of each pixel is known as the grey level.

– The computer understands only ones and zeros. Hence these grey levels need to be represented in terms of zeros and ones. If we have two bits to represent the grey levels, only 4 different grey levels (2^2) can be identified viz. 00, 01 10, 11, where 00 is black, 11 is white and the other two are different shades of grey.

Fig. 6.3.1(a)



(b)

Fig. 6.3.1 : The first row of the image is plotted.

– Similarly, if we have 8 bits to represent the grey levels, we will have 256 grey levels (2^8). Hence more the bits, more are the grey levels and better is the tonal clarity (quantization). The total size of the image is N × M × m, where m is the number of bits used.

– The x-axis is the number of samples (pixels) in the row (sampling) while the y-axis is the grey level of each sample (quantization)

– Now, comes the obvious question. As we know, more the samples and the bits, better is the image. What then should be the ideal values of sampling and quantization.

– This answer will vary from image to image. Given below is a table of sampling and the quantization values. As the sampling and the quantization increase, the number of bits required to store the image increases tremendously.

– The clarity increases, but storage space required increases too. We hence need to get a trade-off between the two. For simplicity, we consider a square matrix of size N × N.

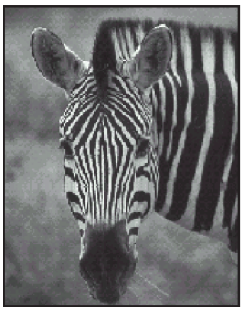
Table 6.3.1 : Number of storage bits for various values of N and m

N \ m	1	2	3	4	5	6	7	8
32	1,024	2,048	3,072	4,098	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,385	32,768	49,152	65,536	81,290	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1,024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608

We shall see the effects of reducing quantization levels and reducing spatial resolution separately.

MATLAB code for reducing quantization levels

```
%% Effects of reducing the quantization values %%
clear all
clc
a=imread('zebra.tif');
a=double(a);
a=a+1;
b=max(max(a)); %% gives us the maximum value in the image
i=input ('how many bits do you want 1 2 4 8');
j=b/(2^i); %% since total number of bits is equal to 2^i
F=floor(a/(j+1));
F1=(F*255)/max(max(F)); % normalizing %
figure(1)
imshow(uint8(a))
figure(2)
imshow(uint8(F1))
```



(a) Original image

(b) Image using 4-bits

(c) Image using 2-bits

(d) Image using 1-bit

Fig. 6.3.2

Comparing the images, we see that "false contouring" takes place as we reduce the number of grey levels.

MATLAB code for reducing spatial resolution.

```
%% Down sampling
%% To see the effects of reducing the number of samples %%
clear all
clc
    a= imread('deepa.tif')
    [row,col]=size(a);
    i=1; j=1;
for x=1:2:row
        for  y=1:2:col
            c(i,j)=a(x,y);
                j=j+1;
            end
    j= 1; %% This needs to be done else the value of j goes on increasing %%
    i= i+1;
end
figure(1),imshow(a)
figure(2),imshow(c)
figure(3),
imagesc(a),colormap(gray)
figure(4)
imagesc(c),colormap(gray)
```

(a) Original image

(b) Down sampled

Fig. 6.3.3

Fig. 6.3.3 (c) Down sampled image displayed after zooming to match the size of the original image

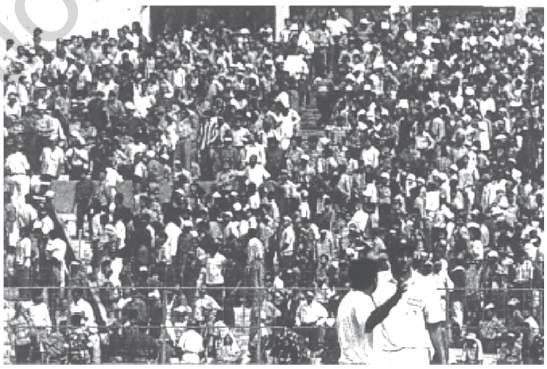
It is clear from the images that the resolution reduces as number of samples reduce. To compare and understand the actual effects, we plot them together. To make sure that they appear to be of the same size, we upsample the second image. Comparing Fig. 6.3.3(a) and Fig. 6.3.3(c) we see that the second image has a "Checker board" pattern due to the reduction of samples.

6.4 Isopreference Curves

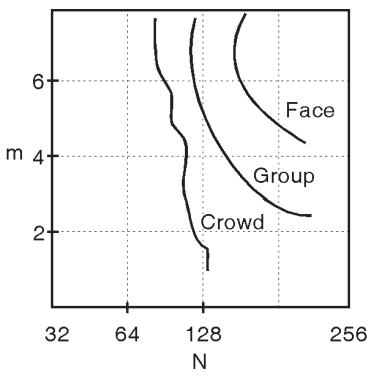
- We have seen the effects of reducing N and m in the preceding topic. We still do not know what is the ideal value of N (sampling resolution) and m (tonal resolution) for images.
- An early study by T.S. Huang in 1965 attempted to quantify experimentally the effects on image quality produced by varying N and m simultaneously. There, different types of images were shown to a group of people.
- The first image was that which did not have a lot of details for example a woman's face. The second image was one which had intermediate amount of information for example a small group standing together and the third image was one which had a lot of details example an image of a crowd.
- In these images, N and m were varied. Observers were then asked to rank them according to their subjective quality. These results were summarised in the form of isopreference *curves* in the N-m plane.
- Points lying on an isopreference curve corresponded to image of equal subjective quality. From the isopreference curves Huang concluded that images with large amount of details require few grey levels. Since the isopreference curve of the crowd is near vertical, it means that for a fixed value of N, the image is nearly independent of m.

(a)

(b)



(c)



(d)

Fig. 6.4.1

6.5 Non-Uniform Sampling

- We have already seen that as N and m increase, the spatial and the grey level resolution increase. But due to this the storage space required increases enormously.
- To address this problem, we use non-uniform sampling. In this, regions which have a lot of details i.e., were there are abrupt changes in grey levels, a lot of samples are taken while regions where the grey levels reduce very slowly (background) only few samples are used.
- Hence non-uniform sampling can be used to reduce the storage space required without distorting the subjective quality of the image.
- We have all seen non-uniform sampling been used. Whenever a person's identity needs to be concealed on T.V., non-uniform sampling is used.
- Very few samples are taken near the eye regions because of which we see blocks near the person's eye and hence can not make out who that person is.

6.6 Physical Resolution

- By now we know that a digital image, or image for short, is composed of discrete pixels. These pixels are arranged in a row and column fashion to form a rectangular picture area.
- Clearly, the total number of pixels in an image is a function of size of the image and the number of pixels per unit area (example: inch) in the horizontal as well as the vertical direction.
- The number of pixels per unit length is referred to as the resolution of the displaying device (most of the deskjet printers have 670 dpi i.e., 670 dots per inch.).
- Thus a 3×2 inch image at a resolution of 300 pixels per inch would have a total of 540,000 pixels!!
- In most books as well as in this book, image size is given as the total number of pixels in the horizontal direction times the total number of pixels in the vertical direction (example : 128×128, 512×512, 640×480 ....).
- Although this convention makes it relatively straight forward to gauge the total number of pixels in an image, it does not specify the physical size of the image or its resolution as defined in the paragraph above.
- A 640×480 image would measure 6.66 inches by 5 inches when displayed or printed at 96 pixels per inch. On the other hand, it would measure only 1.6 inch by 1.2 inch when displayed or printed at 400 pixels per inch.

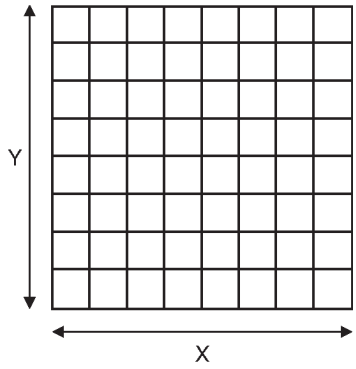


Fig. 6.6.1

- Another term that we need to understand is the Aspect ratio.
- The ratio of the image's width to its height, measured in unit length or number of pixels is referred to as its aspect ratio. Both, a 3×3 inch image and a 128×128 image have the same aspect ratio of 1.

$$\text{Aspect ratio} = \frac{X}{Y}$$

6.6.1 Solved Examples on Physical Resolution

Ex. 6.6.1 : Compute the physical size of a 640×480 image when printed by a printer at 240 pixels per inch.

Soln. : Since we have 240 pixels per inch, the physical size of the image is $\frac{640}{240}$ by $\frac{480}{240}$.

Ex. 6.6.2 : If we want to resize a 1024×768 image to one that is 600 pixels wide with the same aspect ratio as the original image, what should be the height of the resized image?

Soln. : We know

$$\text{Aspect ratio} = \frac{\text{Width}}{\text{Height}}$$

For the original image the

$$\text{Aspect ratio is} = 1024/768 = 1.33$$

Now for the resized image, we want the same aspect ratio but a width of 600 pixels.

$$\text{Height} = \frac{\text{Width}}{\text{Aspect ratio}} = \frac{600}{1.33} = 451$$

Hence the resized image will be 600×451.

Ex. 6.6.3 : How much storage capacity is required to store an image with size of 1024×768 and 256 gray levels ?

Soln. :

Storage capacity required = $A \times B \times C$

Here $A \times B$ = Physical size of image = 1024×768

C = Number of bits required to get 256 gray levels, we must have 8 bits ($2^8 = 256$)

Storage capacity required = $1024 \times 768 \times 8 = 6291456$ bits

Ex. 6.6.4 : A common measure of transmission for digital data is the baud rate, defined as the number of bits transmitted per second. Transmission is accomplished in packets consisting of a start bit, a byte (8-bits) of information and a stop bit.

(a) How many minutes would it take to transmit a 1024×1024 image with 256 grey levels if we use a 56 k baud modem ?

(b) What would be the time required if we use a 750 k band transmission line ?

Soln. :

(a) Since we have 256 grey levels, we need 8-bits for representing each pixel.

Along with these 8-bits, we also have a start bit and a stop bit.

Hence we have (8 + 2) bits per pixel.

$\therefore$ total number of bits for transmission are

$$N = 1024 \times 1024 \times 10$$

$$N = 10485760\text{-bits}$$

These bits are transmitted at 56 k baud.

$$\therefore \text{time taken} = \frac{N}{56 \times 10^3}$$

$$\therefore \text{time taken} = 187.25 \text{ sec} \approx 3.1 \text{ minutes}$$

(b) Now if we use a faster transmission line of 750 k baud rate, then

Time taken to transmit the image

$$\frac{N}{750 \times 10^3} = \frac{10485760}{750 \times 10^3}$$

$$= 13.98 \text{ sec} \approx 14 \text{ sec}$$

Ex. 6.6.5 : We have a C. T. image which we want to transmit through a voice grade telephone line. The CT image is of size 512×512 and has 256 grey levels. During transmission each byte is preceded by a start bit and succeeded by a stop bit. How many minutes will it take to transmit the image ? (Voice grade telephone lines can transmit 9600-bits/sec)

Soln. : 256 grey level required mean 8-bits.

$$\therefore \text{one packet} = [8 + 02] \text{ bits} = 10 \text{ bits}$$

$$\therefore \text{image size} = 512 \times 512 \times 10$$

$$\therefore \text{image size} = 2621440$$

$$\therefore \text{time taken} = \frac{512 \times 512 \times 10}{9600}$$

$$\therefore \text{time taken} = 273.06 \text{ sec} \approx 4.5 \text{ min}$$

Summary

This chapter deals with converting a scene that is continuous in time and space into an image that can be processed by computers. The technique of converting a continuous signal into a discrete one is called digitization and is explained here. Digitisation comprises of two steps viz. sampling and quantization. Concepts of sampling are explained in the one-dimensional as well as in the two dimensional domain. Relationship between the time domain and the frequency domain i.e., the Fourier domain are presented here.

Aliasing, which forms an important pitfall of sampling, has been introduced and the importance of the Nyquist criteria has been explained. The concept of an image being stored as a matrix has been introduced here. Spatial resolution and grey level resolution are also explained using MATLAB codes.

Review Questions

Q. 1 Explain the concept of aliasing for 1-dimensional signals.

Q. 2 Explain the concept of aliasing for 2-dimensional signals.

Q. 3 How does one avoid aliasing ?

Q. 4 Explain sampling and quantization.

Q. 5 Explain the effects of reducing sampling and quantization.

Q. 6 Explain isopreference curves.

Q. 7 Explain non-uniform sampling.

❑❑❑

Image Enhancement in Spatial Domain

Syllabus :

Gray Level Transformations, Zero Memory Point Operations, Neighborhood Processing, Spatial Filtering, Smoothing and Sharpening Filters, Median Filter.

7.1 Introduction

- Image enhancement is one of the first steps in image processing. As the name suggests, in this technique, the original image is processed so that the resultant image is more suitable than the original for specific applications i.e. the image is enhanced.
- Image enhancement is a purely subjective processing technique. By subjective we mean that the desired result varies from person to person.
- An image enhancement technique used to process images might be excellent for a person, but the same result might not be good enough for another. It is also important to know at the outset that image enhancement is a cosmetic procedure i.e. it does not add any extra information to the original image.
- It merely improves the subjective quality of the images by working with the existing data.
- Image enhancement can be done in two domains :
  (1) The spatial domain
  (2) The frequency domain.
- Let us start with explaining what is meant by the spatial domain, discuss the various methods of spatial domain enhancement and then move on to discuss the frequency domain technique in chapter of Image Enhancement in Frequency Domain.

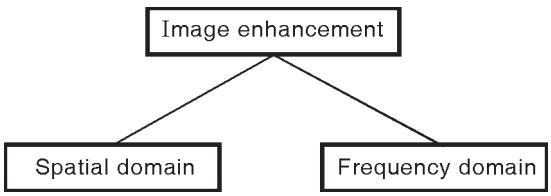


Fig. 7.1.1

7.2 Spatial Domain Methods

- The term spatial domain means working in the given space, in this case, the image. It implies working with the pixel values or in other words, working directly with the raw data.
- Let f(x, y) be the original image where f is the grey level value and (x, y) are the image coordinates. For a 8-bit image, f can take values from 0 - 255 where 0 represents black, 255 represents white and all the intermediate values represent shades of grey.

In a image of size 256 × 256 as shown in the Fig. 7.2.1.

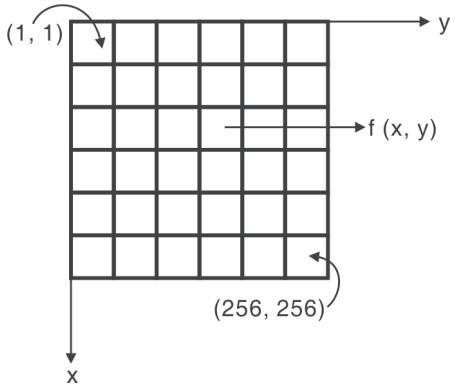


Fig. 7.2.1

Note : We always consider the axis as shown in Fig 7.2.1. The x-axis is taken in the downward vertical direction, while the y-axis is taken in the horizontal direction. The reason for taking this inverted coordinate system is to make sure that the image uses the standard matrix coordinates that we have been using right from our school. You will remember the first value of the matrix is always at the top left corner while the last value of the matrix is at the right bottom corner.

- The modified image can be expressed as,

$$g(x, y) = T[f(x, y)] \quad \ldots(7.2.1)$$

Here f(x, y) is the original image and T is the transformation applied to it to get a new modified image g(x, y). For all spatial domain techniques it is simply T that changes. The general equation remains the same.

- To summarize, spatial domain techniques are those, which directly work with the pixel values to get a new image based on Equation (7.2.1).
- Spatial domain enhancement can be carried out in two different ways :
  (1) Point processing
  (2) Neighbourhood processing

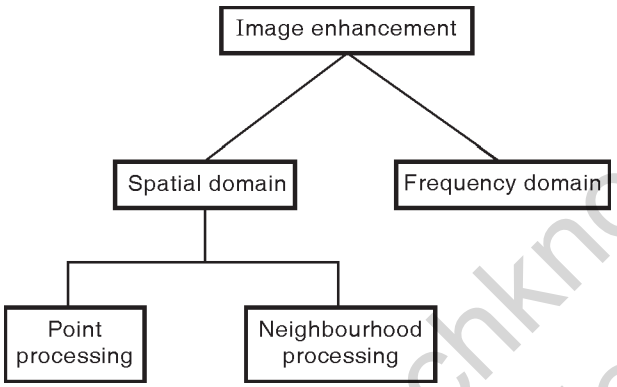


Fig. 7.2.2

7.3 Point Processing / Zero Memory Operation

MU - Dec. 2015, Dec 2016, May 2017, May 2018, Dec 2018

Q. What do you understand by zero memory operation. **(Dec. 2015, 5 Marks)**

Q. Explain any five zero memory operations. **(Dec. 2016, May 2017, 10 Marks)**

Q. Explain Thresholding. **(Dec. 2016, May 2017, 2 Marks)**

Q. Explain zero memory operations. **(May 2018, Dec. 2018, 5 Marks)**

- In point processing, we work with single pixels i.e. T is 1×1 operator. It means that the new value f(x, y) depends on the operator T and the present f(x, y). This statement will be clear as we start giving some examples.
- Since this operation does not require past pixel values, we do not need additional memory allocation. Hence point processing is also known as zero memory operation. Some of the common examples of point processing are :

(1) Digital negative
(2) Contrast stretching
(3) Thresholding
(4) Grey level slicing
(5) Bit plane slicing
(6) Dynamic range compression
(7) Power law transformation

- Before we proceed to explain the following examples, it is important to understand the identity transformation. The identity transformation is given in the Fig. 7.3.1.
- In the Fig. 7.3.1, the solid line is the transformation T. The horizontal axis represents the grey scale of the input image (r) and the vertical axis represents the grey scale of the output image (s). It is called an identity transformation because it does not modify the new image at all !!.
- As seen, the grey level 10 is modified to 10, 125 to 125 and finally 255 to 255. In general $s_1 = r_1$. Fig. 7.3.1 will help us understand the point processing techniques better.

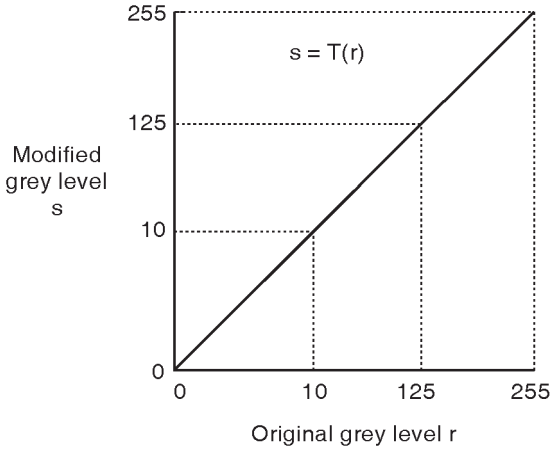


Fig. 7.3.1

(1) Digital negative

- Digital negatives are useful in a lot of applications. A common example of digital negative is the displaying of an X-ray image.

– As the name suggests, negative simply means inverting the grey levels i.e. black in the original image will now look white and vice versa. The Fig. 7.3.2 is the digital negative transformation for a 8-bit image.

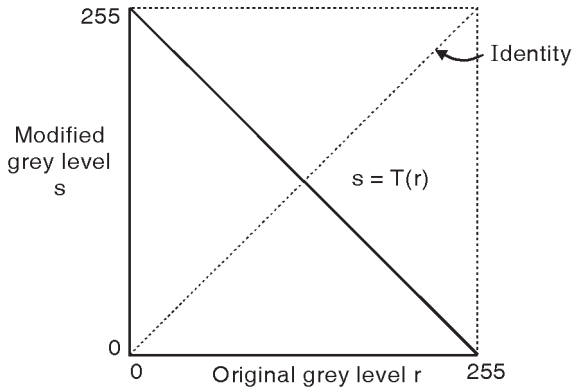


Fig. 7.3.2

– The digital negative image can be obtained by using a simple transformation given by,

$$s = 255 - r \ (r_{max} = 255)$$

Hence when

$$r = 0, s = 255 \text{ and when } r = 255, s = 0.$$

In general,

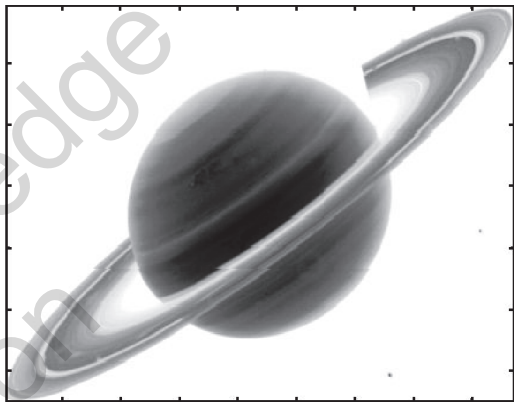
$$s = (L - 1) - r \qquad \ldots(7.3.1)$$

Here L is the number of grey levels. (256 in this case)

MATLAB program for finding the digital negative

```
%% MATLAB code to calculate negative %%
clear all
clc
aa=imread ('saturn.tif');
a=double (aa)
c=255; % for a 8-bit image %
b=c–a;
figure(1)
colormap(gray)
imagesc(a)
figure(2)
colormap(gray)
imagesc(b)
```

(a) Original image



(b) Digital negative

Fig. 7.3.3

(2) Contrast stretching

– Many times we obtain low contrast images due to poor illuminations or due to wrong setting of the lens aperture. The idea behind contrast stretching is to increase the contrast of the images by making the dark portions darker and the bright portions brighter.

– Fig. 7.3.4 shows the transformation used to achieve contrast stretching. In the Fig. 7.3.4, the dotted line indicates the identity transformation and the solid line is the contrast stretching transformation.

– As is evident from the Fig. 7.3.4, we make the dark grey levels darker by assigning a slope of less than one and make the bright grey levels brighter by assigning a slope greater than one.

– One can assign different slopes depending on the input image and the application. As was mentioned, image enhancement is a subjective technique and hence there is no one set of slope values that would yield the desired result.

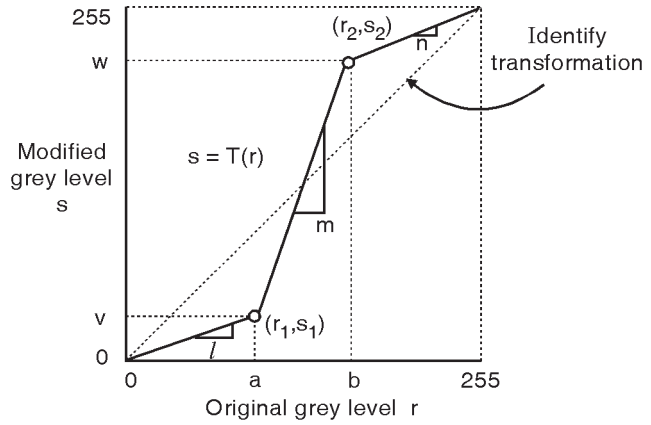


Fig. 7.3.4 : Original grey level r

– The formulation of the contrast-stretching algorithm is given as follows :

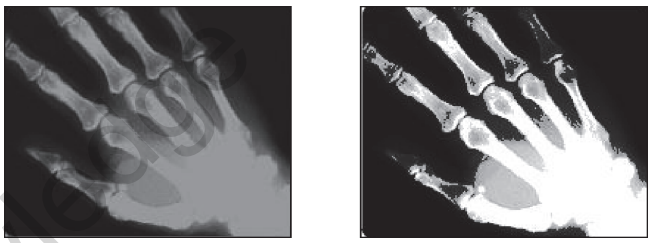
$$s = \begin{cases} l \cdot r & 0 \le r < a \\ m \cdot (r - a) + v & a \le r < b \\ n \cdot (r - b) + w & b \le r < L - 1 \end{cases} \quad \ldots(7.3.2)$$

where *l*, m and n are the slopes. It is clear from the Fig. 7.3.4 that *l* and n are less than one while m is greater than one. The contrast stretching transformation increases the dynamic range of the modified image.

MATLAB program for contrast stretching

```
%% Contrast stretching of an image %%
% Slopes taken are 0.5, 2 and 0.5 %
clear all;
clc;
a=imread('xray1.tif')
a=double(a);
[row col]=size(a);
LT=input('Enter the lower threshold value:');
UT = input('Enter the upper threshold value:');
for x=1:1:row
    for y=1:1:col
     if a(x,y)<=LT
        b(x,y)=0.5*a(x,y);
     else if a(x,y)<=UT
              b(x,y)=2*(a(x,y)-LT)+0.5*LT;
     else b(x,y)=0.5*(a(x,y)-UT)+0.5*LT+2*(UT-LT);
      end
    end
```

```
 end
end
subplot(2,1,1)
imshow(uint8(a))
title('Original Image');
subplot(2,1,2)
imshow(uint8(b))
title('Image after Contrast Stretching')
```



(a) Original image **(b) Contrast stretched image**

Fig. 7.3.5

(3) Thresholding

– Extreme contrast stretching yields thresholding. If we observe the contrast stretching diagram closely we notice that if the first and the last slope are made zero, and the centre slope is increased, we would get a thresholding transformation i.e. if $r_1 = r_2$, $s_1 = 0$ and $s_2 = L - 1$, we get a thresholding function. It is shown in Fig. 7.3.6.

– The formula for achieving thresholding is as follows,

$$\left.\begin{array}{ll} s = 0; & \text{if } r \le a \\ s = L - 1; & \text{if } r > a \end{array}\right\} \quad \ldots(7.3.3)$$

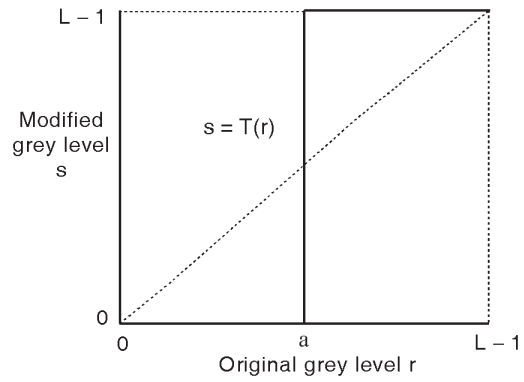


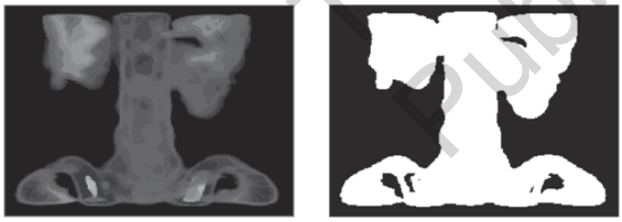
Fig. 7.3.6 : Original grey level r

Where, L is the number of grey levels.

– As mentioned earlier, image enhancement being a subjective phenomena, the value of a will vary from image to image and from person to person. The objective is to identify the region that he or she is interested in. An important thing to note is that a thresholded image has the maximum contrast as it has only black and white grey values.

MATLAB program for thresholding

```
%%% Thresholding %%%
clear all
clc
p=imread('spine.tif');
a=p;
[row col]=size(a);
T=input('Enter value of Threshold :')
for i=1:1:row
        for j=1:1:col
            if(p(i,j)<T)
                a(i,j)=0;
            else
                a(i,j)=255;
        end
    end
end
figure(1),imshow(p),
figure(2),imshow(a)
```



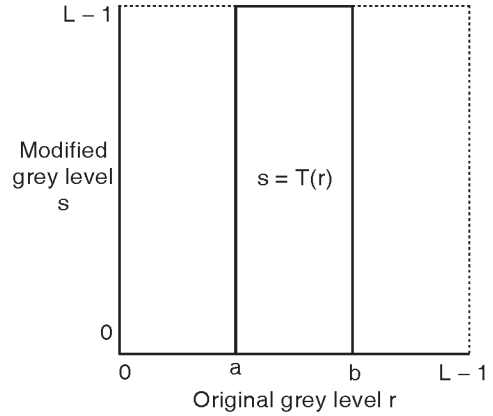
(a) Original image of spine **(b) Image obtained using threshold**

Fig. 7.3.7

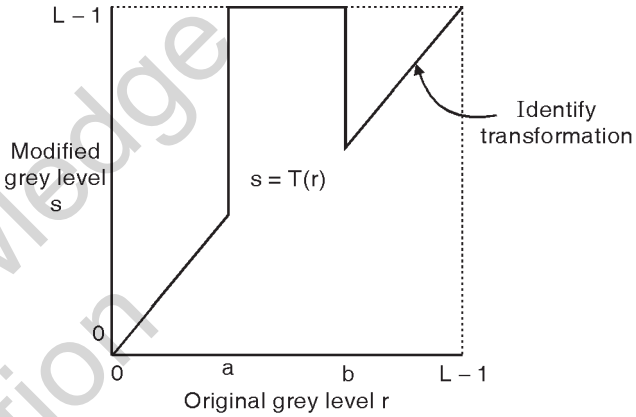
(4) Grey level slicing (Intensity slicing)

– What thresholding does is it splits the grey level into two parts. At times, we need to highlight a specific range of grey values like for example enhancing the flaws in an X-ray or a CT image.

– In such circumstances, we use a transformation known as grey level slicing. The transformation is shown in the Fig. 7.3.8(a). It looks similar to the thresholding function except that here we select a band of grey level values.



(a) Slicing without background



(b) Slicing with background

Fig. 7.3.8

This can be implemented using the formulation

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \le r \le b \\ s = 0; & \text{otherwise} \end{array}\right\} \quad \ldots(7.3.4)$$

– This method is known as Grey level slicing without background. This is because in this process, we have completely lost the background. In some applications, we not only need to enhance a band of grey levels but also need to retain the background.

– This technique of retaining the background is called as Grey-level slicing with background. The transformation is as shown in the Fig. 7.3.8(b).

The formulation for this is

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \le r \le b \\ s = r; & \text{otherwise} \end{array}\right\} \quad \ldots(7.3.5)$$

MATLAB program for grey level slicing with and without background is as follows :

```
%% Grey level slicing without background %%
clear all
clc
p=imread ('skull.tif');
z=double (p);
```

```
[row,col]=size(z)
for i=1:1:row
        for j=1:1:col
            if((z(i,j)>50))&&(z(i,j)<150)
            z(i,j)=255;
            else
                z(i,j)=0;
            end
        end
end
figure (1); % ……original image.
imahow (p)
figure (2);  % ……gray level slicing without background
imshow (uint8(z))
```

(a) Original image **(b) Gray level slicing without background**

Fig. 7.3.9

MATLAB program

```
%% Grey level slicing with backgound %%
clear all
clc
p=imread('skull.tif');
z=double(p);
[row col]=size(p);
for i=1:1:row
        for j=1:1:col
            if(z(i,j)>50))&&(z(i,j)<150)
                z(i,j)=255;
            else
                z(i,j)=p(i,j);
            end
        end
end
figure (1); %------- original image.
imshow(p)
figure (2); %------- grey level slicing with background
imshow (uint8(z))
```

(a) Original image **(b) Grey level slicing with background**

Fig. 7.3.10

(5) Bit plane slicing

- In this technique, we find out the contribution made by each bit to the final image. As mentioned earlier, an image is defined as say a $256 \times 256 \times 8$ image. In this, 256×256 is the number of pixels present in the image and 8 is the number of bits required to represent each pixel. 8-bits simply means 28 or 256 grey levels.
- Now each pixel will be represented by 8-bits. For example black is represented as 00000000 and white is represented as 11111111 and between them, 254 grey levels are accommodated. In bit plane slicing, we see the importance of each bit in the final image. This can be done as follows.
- Consider the LSB value of each pixel and plot the image using only the LSBs. Continue doing this for each bit till we come to the MSB. Note that we will get 8 different images and all the 8 images will be binary.

Solved Example

Ex. 7.3.1 : Given a 3×3 image, plot its bit planes.

1	2	0
4	3	2
7	5	2

Soln. : Since 7 is the maximum grey level, we need only 3-bits to represent the grey levels.

Hence we will have 3-bit planes. Converting the image to binary we get,

001	010	000
100	011	010
111	101	010

Binary image

1	0	0
0	1	0
1	1	0

LSB plane

0	1	0
0	1	1
1	0	1

Middle bit plane

0	0	0
1	0	0
1	1	0

MSB plane

MATLAB program

```
%% MATLAB code for bit extraction %%
clear all
clc
a=imread('warne.tif');
a=double(a);
r=input('which bit image do you want to see 1=MSB 8=LSB');
[row col]=size(a);
for x=1:1:row
        for y=1:1:col
        c=dec2bin(a(x,y),8);  % converts decimal to binary
        d=c(r);
        w(x,y)=double(d); %% since w is a char and cannot be plotted
    if w(x,y)==49 %% since double of d will be either 49 or 48
            w(x,y)=255;
          else
            w(x,y)=0;
        end
end
end
figure(1)
imshow(uint8(a))
figure(2)
imshow(uint8(w))
```

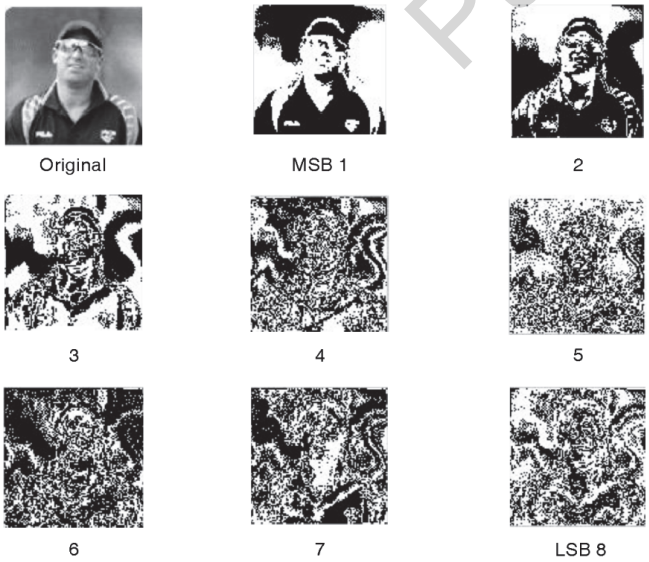


Fig. P. 7.3.1 : Eight images, each representing contribution of a single bit

Observing the images we come to the conclusion that the higher order bits contain majority of the visually significant data, while the lower bits contain the suitable details in the image. Bit plane slicing can hence be used in image compression. We can transmit only the higher order bits and remove the lower order bits. Bit plane slicing is also used in steganography.

(6) Dynamic range compression (Log transformation)

- At times, the dynamic range of the image exceeds the capability of the display device. What happens is that some pixel values are so large that the other low value pixels get obscured.
- A simple day-to-day example of such a phenomena is that during daytime, we cannot see the stars. The reason behind this is that the intensity of the sun is so large and that of the stars is so low that the eye cannot adjust to such a large dynamic range.
- In image processing, a classic example of such large differences in grey levels is the Fourier spectrum (will be discussed in detail in the frequency domain enhancement technique).
- In the Fourier spectrum only some of the values are very large while most of the values are too small. The dynamic range of pixels is of the order of 10^6. Hence, when we plot the Fourier spectrum, we see only small dots, which represent the large values.
- Something needs to be done to be able to see the small values as well. This technique of compressing the dynamic range is known as dynamic range compression.
- We all know that the log operator is an excellent compressing function. Hence the dynamic range compression is achieved by using a log operator. C is the normalization constant.

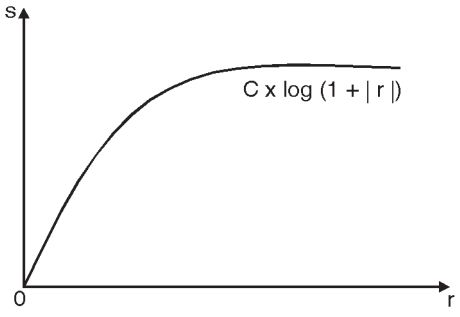
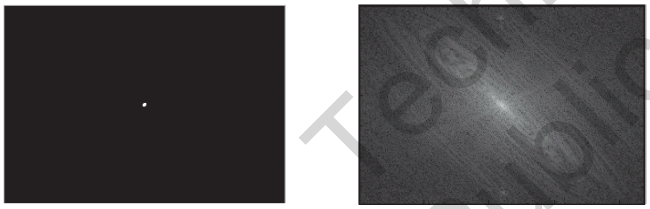


Fig. 7.3.11

MATLAB program for dynamic range compression

```
%% Dynamic range compression %%
clear all
clc
aa=imread('saturn.tif');
a=double(aa) ;
[row,col]=size(a);
for x=1:1:row
    for y=1:1:col
    c(x,y)=a(x,y)*((-1)^(x+y)); %% Needed to center the transform
  end
end
d=abs(fft2(c));
d_log=log(1+d);
%%% Plotting
figure(1)
colormap(gray)
imagesc(d)
figure(2)
colormap(gray)
imagesc(d_log)
```



(a) (b)

Fig. 7.3.12 : Dynamic range compression

(7) Power law transformation

– The basic formula for power-law transformation is

$$g(x, y) = c \times f(x, y)^{\gamma}$$

It can also be written as $s = c\, r^{\gamma}$ ...(7.3.6)

– Here c and γ are positive constants. The transformation is shown Fig. 7.3.13 for different values of γ which is also called the gamma correction factor. We observe that by changing the value of gamma, we obtain a family of transformation curves.

– Non-Linearities encountered during image capturing, printing and displaying can be corrected using gamma correction. Hence gamma correction is important if the image needs to be displayed on the computer.

– The power law transformation can also be used to improve the dynamic range of an image. Given below is the MATLAB code for power transformation. The final image has been normalized to 0 - 255 range.

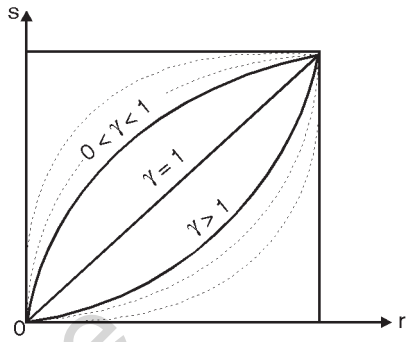


Fig. 7.3.13 : Gamma correction factor

```
%% Power transformation %%
clear all
clc
img1=imread('test.tif');
[row,col]=size(img1);
gamma=input('Enter the correction factor: ');
img=double(img1);

for i=1:row
        for j=1:col
        nuimg(i,j)=img(i,j)^gamma
        end
end
numax=max(max(nuimg));
numin=min(min(nuimg));
n=255/(numax-numin);
for i=1:row
    for j=1:col
    nuimg1(i,j)=n*(nuimg(i,j)-numin); % Normalisation
    end
end
nuimg2=uint8(nuimg1);
subplot(2,1,1)
imshow(img1)
title('Original image')
subplot(2,1,2)
imshow(nuimg2)
title('Image after power transformation')
```

Fig. 7.3.14

7.4 Neighbourhood Processing

- This, as mentioned before, is also a spatial domain technique in image enhancement. Unlike the point processing techniques where we consider one pixel at a time and modify it depending on our requirement, here we not only consider a pixel but also its immediate neighbours.
- To cut a long story short, we change the value of the pixel f(x, y) based on the values of its 8 neighbours as shown in Fig. 7.4.1. Instead of a 3×3 neighbourhood, we could also use a 5×5 or a 7×7..... neighbourhood.

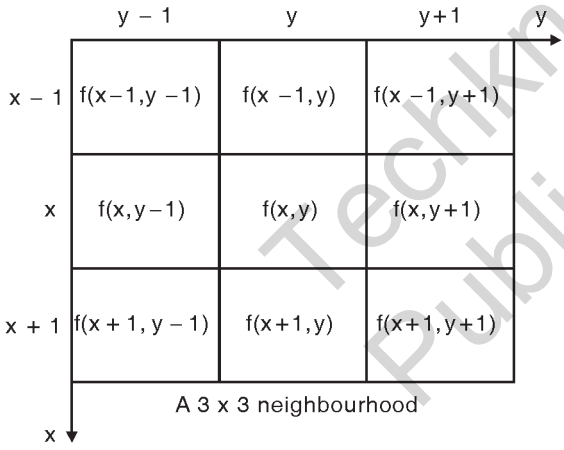


Fig. 7.4.1 : A 3×3 neighbourhood

w1	w2	w3
w4	w5	w6
w7	w8	w9

Fig. 7.4.2 : 3×3 mask

- There are a lot of things that can be achieved by neighbourhood processing which are not possible with point processing. Fig. 7.4.2 shown is called a mask or a window or a template.
- To achieve neighbourhood processing, we place this 3×3 (it could also be a 5×5 or a 7×7...) mask on the image, multiply each component of the mask with the corresponding value of the image, add them up and place the value that we get, at the center.
- This operation is the same as convolution. In the convolution operation, we have two signals. Of the two signals, we take one, flip it and then move it across the other signal step by step. The same thing is done here. We don't need to flip the mask as it is symmetric.
- If f is the original image and g is the modified image, then,

$$g(x, y) = f(x-1, y-1) \times w1 + f(x-1, y) \times w2 + f(x-1, y+1) \times w3 + f(x, y-1) \times w4 + f(x, y) \times w5 + \ldots\ldots\ldots + f(x+1, y+1) \times w9 \quad \ldots(7.4.1)$$

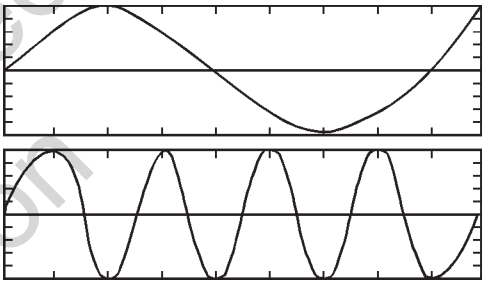


Fig. 7.4.3

- Once g(x, y) is calculated, we shift the mask by one step towards the right to the next pixel. Now w5 coincides with f(x, y + 1).
- One of the important operations that can be achieved using neighbourhood processing is that of image filtering.
- We can perform low pass, high pass and band pass filtering using neighbourhood operations.
- Before explaining the procedure of performing filtering on images, it is imperative to understand what we mean by frequencies in an image !! We are all well versed with frequencies in 1-dimensional signals. Given two signals, we can easily distinguish between the lower frequency signal and the higher frequency signal. Refer Fig. 7.4.3.

Fig. 7.4.4

- We can conclude that the lower signal is of a much higher frequency, by checking the number of oscillations. That is, higher the frequency, greater are the number of oscillations.
- If the two signals represent voltages, then, how fast the voltages change is an indication of the frequency. The same concepts can be applied to images.
- In images, instead of voltages, we have grey levels. If the grey levels change slowly over a region, it is considered to have low frequency, whereas, if the grey levels change very rapidly, that region is considered to have high frequencies.
- Hence in images, regions where the grey levels change slowly are low frequency areas and regions which have abrupt grey level changes, are high frequency areas. Always remember this.
- In most of the images, the background is considered to be a low frequency region, whereas the edges are considered to be high frequency regions. Hence low pass filtering implies removing (bluring) the edges while high pass filtering implies removing the background.

7.4.1 Low Pass Filtering (Smoothing)

- Low pass filtering as the name suggests removes the high frequency content from the image. It is used to remove noise present in the image.
- Noise, is normally a high frequency signal and low pass filtering eliminates the noise. Before proceeding to explain the low-pass filtering technique, we shall spend some time discussing the types of noise that are fairly common in images.

7.4.2 Noise

The principal sources of noise in a digital image arise during image acquisition and during transmission. No matter how much care one takes, some amount of noise always creeps in. Based on the shapes (Probability Density Functions) of the noise, they are classified as :

(1) Gaussian noise (2) Salt and pepper noise

(3) Rayleigh noise (4) Gamma noise

(5) Exponential noise (6) Uniform noise

Of these, the first two are more common than the others. We shall explain the Gaussian and salt and pepper noise in this chapter.

(1) Gaussian noise : The Probability Density Function (PDF) of Gaussian noise is given by the formula,

$$p(z) = \frac{1 \cdot e^{-(z-\mu)^2 / 2\sigma^2}}{\sqrt{2\pi}\,\sigma}$$

$z \rightarrow$ Grey level

$\mu \rightarrow$ Mean of average value of z

$\sigma \rightarrow$ Standard deviation

$\sigma^2 \rightarrow$ Variance

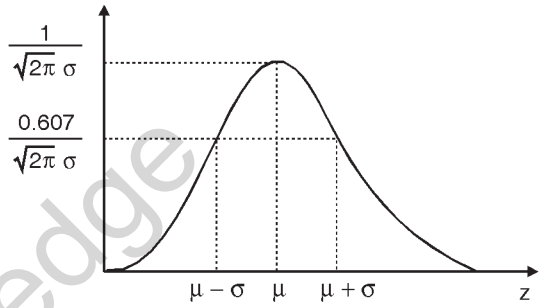


Fig. 7.4.5

If we plot this function, we notice that,

70% of its value lies in the range $[(\mu - \sigma), (\mu + \sigma)]$ and

95% of its value lies in the range $[(\mu - 2\sigma), (\mu + 2\sigma)]$

Gaussian noise has a maximum value at μ and then it starts falling off.

Consider the image shown in Fig. 7.4.6.

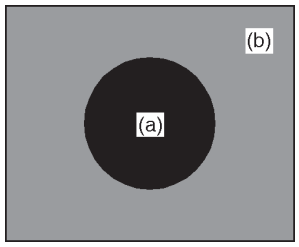


Fig. 7.4.6

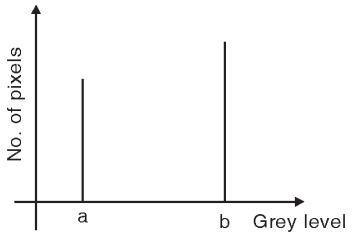


Fig. 7.4.7

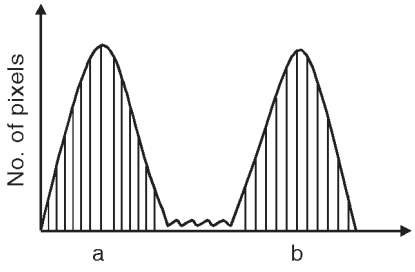


Fig. 7.4.8

There are two constant regions in the image. Hence, the histogram of the image is as shown in Fig. 7.4.7. If in this image, Gaussian noise creeps in, the histogram gets modified as shown in Fig. 7.4.8. Gaussian noise arise in an image due to factors such as circuit noise, sensor noise poor illumination and high temperature.

(2) Salt and Pepper Noise

The PDF of the salt and pepper noise (bipolar noise) is

$$p(z) = \begin{cases} P_a; & \text{for } z = a \\ P_b; & \text{for } z = b \\ 0; & \text{otherwise} \end{cases}$$

If P_a or P_b is zero, this noise is called unipolar noise. The PDF of salt and pepper is shown in Fig. 7.4.9.

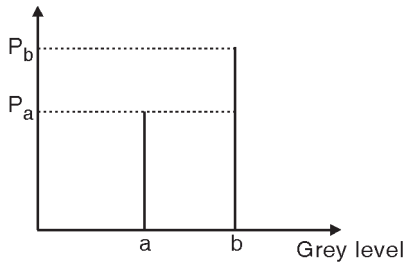


Fig. 7.4.9

Generally a and b are black and white grey levels respectively. Hence for a 8-bit image, a = 0, b = 255 because of which the noise is called salt (white) and pepper (black). Some books refer to it as speckle noise.

Take the same image as the one taken for the Gaussian example.

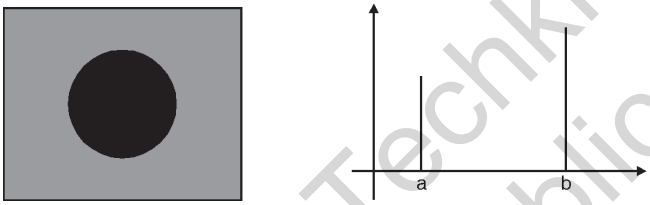


Fig. 7.4.10 (a)

When salt and pepper creeps in, the image looks like

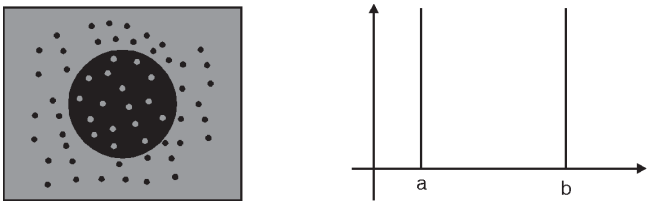


Fig. 7.4.10 (b)

Salt and pepper noise creeps into images in situations where quick transients, such as faulty switching take place.

7.4.3 Low Pass Averaging Filter

MU - May 2017

Q. Explain lowpass spatial filtering.

(May 2017, 5 Marks)

– If an image has Gaussian noise present in it, we use a low pass averaging filter to eliminate the noise. The frequency response of the low-pass filter along with its spatial response is shown in Fig. 7.4.11.

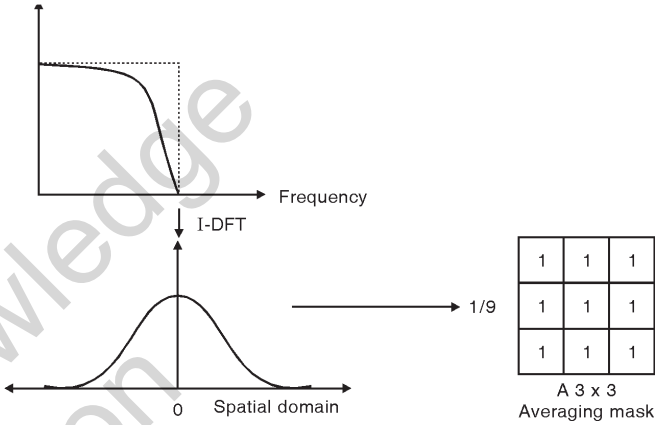


Fig. 7.4.11

– From the spatial response we generate the mask that would give us the low pass filtering operation. One important thing to note from the spatial response is that all the coefficients are positive. The standard low pass averaging mask (3 × 3) is given above. As the name suggests, each element of the mask is the average value.

– We could also use a 5 × 5 or a 7 × 7 mask as per our requirement.

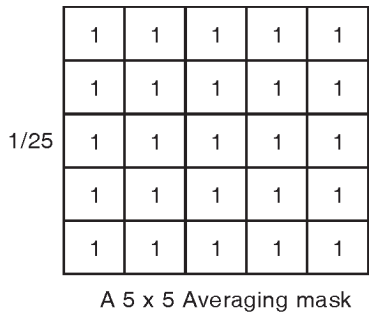


Fig. 7.4.12

– Let us take an example of a 3 × 3 mask on a pseudo-image and see how it eliminates the edges. Consider a 8 × 8 size image. It is clear that the image has a single edge between 10 and 50. To get rid of this edge (high frequency) we use a 3 × 3 averaging mask.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

– We place a 3×3 mask on this image. We start from the left hand top corner. We cannot work with the borders and hence are normally left as they are.

– We then multiply each component of the image with the corresponding value of the mask. Since all the nine values of the image are 10, the average is also ten and the centre pixel (the underlined pixel) remains ten.

– We now shift the mask towards the right till we reach the end of the line and then move it downwards. Some of the mask positions are shown here.

Note : The resultant should be written in a new matrix (image).

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

50	50	50	50	50	50	50	50
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

50	50	50	50	50	50	50	50
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

50	50	50	50	50	50	50	50
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

The last one in the sequence being

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

The result of convolving the image with the 3 × 3 averaging mask is shown.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
23.3	23.3	23.3	23.3	23.3	23.3	23.3	23.3
36.6	36.6	36.6	36.6	36.6	36.6	36.6	36.6
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

- As we notice, the low frequency regions have remained unchanged, but the sharp edge between 10 and 50 has become blurred.
- The transition has reduced. Now from 10, the grey level changes to 23.3 (23 after rounding off), from 23 it changes to 36.6 (36 after rounding off) and finally from 36 it changes to 50. Hence we can say that the sharp edge has become blurred. Check for yourself, what would happen if you took a bigger mask, say, 5 × 5. (You will have to take a bigger image to test your results).
- These kinds of averaging filters are excellent when the image contains Gaussian noise. It achieves filtering by blurring the noise. Some of the other low pass averaging masks are shown.

$$\frac{1}{6}\times\begin{bmatrix}0&1&0\\1&2&1\\0&1&0\end{bmatrix} \qquad \frac{1}{10}\times\begin{bmatrix}1&1&1\\1&2&1\\1&1&1\end{bmatrix}$$

MATLAB program for low pass filtering along with the images with Gaussian noise

```
%% Low pass filter used on an image with Gaussian noise
clear all
clc
aa=imread('xray.tif'); % size 600 × 800
f=double(aa);
ab=imnoise(aa,'gaussian'); % adding noise
a=double(ab);
w=[1 1 1; 1 1 1; 1 1 1]/9
        [row col]=size(a);
    for x=2:1:row-1
        for y=2:1:col-1
            a1(x,y)=w(1)*a(x-1,y-1)+w(2)*a
                    (x-1,y)+w(3)*…
                a(x-1,y+1)+w(4)*a
                    (x,y-1)+w(5)*a(x,y)+w(6)*…
            a(x,y+1)+w(7)*a(x+1,y-
1)+w(8)*a(x+1,y)+w(9)*…
                a(x+1,y+1);
        end
    end
figure (1)
imshow (uint8 (a))
figure (2)
imshow (uint8 (a1))
```

(a) Original image with Gaussian noise

(b) Low passed image

Fig. 7.4.13

Generalized MATLAB program for low pass averaging using any mask size

```
%% Low pass filtering of an image using averaging technique
clear all;
clc;
a=imread('blood.tif');
a=double(a);
[row col]=size(a)
m=input('Enter the mask size:');
n=m^2;
 for i=1:1:m
    for j=1:1:m
     w(i,j)=1/n;
    end
end
s=(m+1)/2;
for x=1:1:row
    for y=1:1:col
      b(x,y)=a(x,y);
            end
end
```

```
for x=s:1:row-s
    for y=s:1:col-s
     b(x,y)=0;
    end
end
for x=s:1:row-s
   for y=s:1:col-s
    for i=1:1:m
          for j=1:1:m
              b(x,y)=a(x-s+i,y-s+j)*w(i,j)+b(x,y);
          end
    end
end
end
subplot (3,1,1)
imshow (uint8(a))
title ('Original Image');
subplot (3,1,2)
imshow (uint8(b))
title ('Low Pass Filtered Image');
c=conv2(a,w);
subplot(3,1,3)
imshow(uint8(c))
title ('Low Pass Filtered Image using MATLAB');
```

7.4.4 Low Pass Median Filtering

MU - Dec. 2017

Q. Write short notes on : Median Filter.

(Dec. 2017, 5 Marks)

- The averaging filter removes the noise by blurring it till it is no longer seen. But in the process, it also blurs the edges. Bigger the averaging mask more is the blurring. There are times when the image contains salt and pepper noise.
- If we use an averaging filter to remove the same, it will blur the noise but it would also ruin the edges. Hence when we need to eliminate salt and pepper noise, we work with a non-linear filter known as the Median filter.
- They are also called order-statistic filters because their response is based on the ordering or ranking of the pixels contained within the mask.
- In this case, we use a mask similar to the averaging filter except that the mask has no values. So it's like working directly with the 8 neighbours of the centre pixel.
- The steps to perform median filtering are as follows :
  (1) Assume a 3×3 empty mask.
  (2) Place the empty mask at the left hand corner.
  (3) Arrange the 9 pixels in ascending or descending order.
  (4) Choose the median from these nine values.
  (5) Place this median at the centre.
  (6) Move the mask in a similar fashion to the averaging filter.
- In median filtering, the grey level of the centre pixel is replaced by the median value of the neighbourhood. Always write the resultant in a new matrix (image). We shall explain median filtering with an example.
- In the image shown below let 250 be the salt and pepper noise. If we use an averaging filter, the noise would spread out and the edges would also end up getting blurred. If we only want to remove the noise without disturbing the edges, we use a median filter.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	250	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

- Just like the earlier case, we start by placing the mask at the left hand corner. We again ignore the borders.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	250	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	250	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

- Arranging the 9 pixels in ascending or descending order we get,

  (10, 10, 10, 10, 10, 10, 10, 10, 250)
- The median is the 5^{th} value since the mask is 3×3 mask. This value is placed at the center. We now move the mask to the new location and continue the procedure. Performing this operation on the entire image, we get the final image.
- As can be seen, the salt and pepper noise gets eliminated without distorting the edges. Median filters give astonishing results even when the amount of noise is relatively large.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

Solved Example

Ex. 7.4.1 : If x = {2 3 4 3 4 5 6} and w = {–1 0 1}, perform median filtering.

Soln. : w = {–1, 0, 1} simply means that the size of the mask is 1×3 and the term 0 indicates the position from where the filtering starts.

Hence the final answer is y = {2 3 3 4 4 5 6}. The principal function of median filtering is to force points with distinct intensities to be more like their neighbours.

MATLAB program for median filtering along with the images with salt and pepper noise

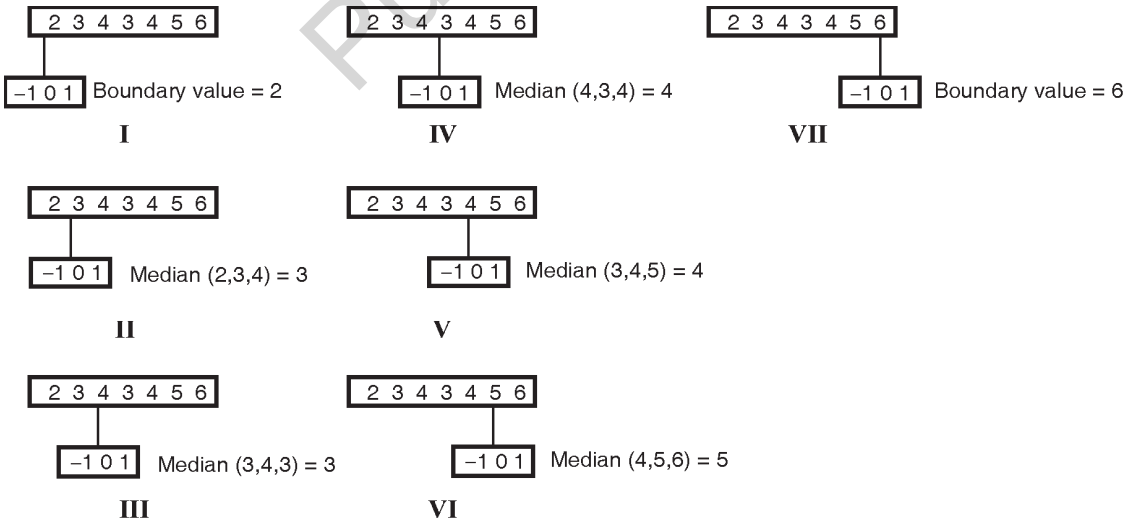


Fig. P. 7.4.1

```
%% Median filter on image with salt and pepper noise %%
clear all
clc
I=imread ('deepa.tif');
J=imnoise (I, 'salt & pepper',0.02); % Adding noise
a=double(J);
b=a;
[row col]=size(a);
for x=2:1:row-1;
    for y=2:1:col-1;
%% To make a 3 × 3 mask into a 1 × 9 mask
    a1=[a(x–1,y–1)    a(x–1,y)    a(x–1,y+1)    a(x,y–1)
a(x,y)…
    a(x,y+1)  a(x+1,y-1)  a(x+1,y)  a(x+1,y+1)];
    a2=sort(a1);
    med=a2(5); % the fifth value is the median
    b(x,y)=med
  end
end
figure(1)
Imshow(uint8(J))
figure(2)
imshow(uint8(b))
```

(a) Original image with salt **(b) Median filtered image and pepper noise**

Fig. P. 7.4.1

7.5 Highpass Filtering

- Highpass filtering eliminates the low frequency regions while retaining or enhancing the high frequency components.
- An image, which is high-passed, would have no background (as background are low frequency regions) and would have enhanced edges. Hence high pass filters are used to sharpen blurred images.
- Once we have understood as to how a mask moves over the entire image for the averaging filter, the same applies to the high pass filter as well. All that needs to be changed are the mask coefficients.
- The frequency response and the spatial response of a high pass filter are shown in Fig. 7.5.1.
- As seen from the spatial response, it is clear that the mask coefficients have to be such that they have a positive value at the centre and negative values at the periphery.
- One of the high pass masks is shown in Fig. 7.5.1.
- The important thing to note is that the sum of the coefficients of the high pass mask has to be equal to zero. This is because, when we place this mask over the low frequency regions, the result should be zero.

Let us take an example of a pseudo-image.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

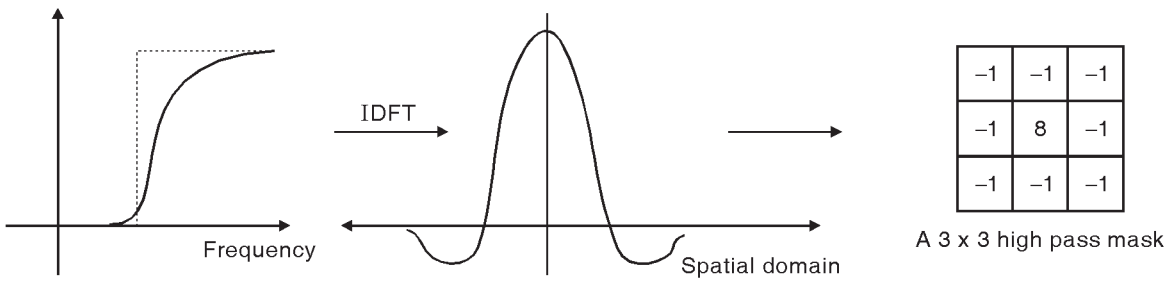


Fig. 7.5.1

- By now we know what to expect when we do a high pass operation. The background, which is low frequency, gets eliminated while the edges get enhanced.
- We move the mask in a similar fashion as in the low pass filter example. The output that we get is shown below. Note that the background has been eliminated and we get all zero values. This is because the sum of the mask coefficients is zero.

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
–270	–270	–270	–270	–270	–270	–270	–270
270	270	270	270	270	270	270	270
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

There are two issues to be dealt with, when working with high pass masks.

(1) As we can see, there are negative values in the output image. Pixel values *cannot be negative*. They always start with zero. Hence we need to get rid of the negative values. If we consider the mod operation, do you think it would solve the problem ? NO. The reason for that is –270 is a value that is lower than zero, that is, it is supposed to be darker than the darkest value i.e. zero. If we now take the mod value i.e. MOD [–270], we get +270, which is definitely not darker than the zero value. What would happen is that all large negative values would be shown as bright spots. Hence taking the mod of negative values will distort the image grey levels. A simple way to get rid of this problem is to let all negative values be zero. Hence –270 would be written as 0.

(2) The values of the original image at the edge are very large and there is a tendency of them going out of range due to the centre weight of +8. We always encounter this problem in practice. To eliminate this problem, we use a mask with a scaling function.

Hence for practical purposes we use a mask are as follows,

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

3×3

High pass mask

Note that 1/9 is simply a scaling function unlike the low pass mask in which the 1/9 term is a part of the averaging operation. In this case we could also work with 1/8 or 1/7 depending on the image. Some of the other high pass masks are as follows

0	–1	0
–1	4	–1
0	–1	0

–1	–2	–1
–2	12	–2
–1	–2	–1

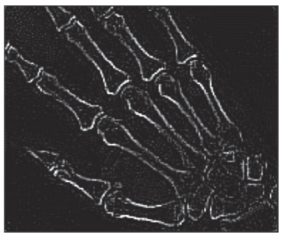
The result using the first mask and putting negative values as zeros is.

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
30	30	30	30	30	30	30	30
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

MATLAB program for high pass filtering

```
%% High pass filtering %%
clear all
clc
aa=imread ('xray.tif');
a=double (aa);
[row col]=size(a);
w=[-1 -1 -1; -1 8 -1, -1 -1 -1]
for x=2:1:row-1
        for y=2:1:col-1
            a1(x,y)= w(1)*a(x-1,y-1)+w(2)*
                     a(x-1,y)+w(3)* …
            a(x-1,y+1)+w(4)*a(x,y-
1)+w(5)*a(x,y)+w(6)* …
        a(x,y+1)+w(7)*a(x+1,y-
1)+w(8)*a(x+1,y)+w(9)* …
            a(x+1,y+1);
        end
end
figure(1)
imshow(uint8(a))
figure(2)
imshow(uint8(al))
```

(a) Original image



(b) High pass filtered image

Fig. 7.5.2

7.6 High-Boost Filtering

- As can be seen, the high pass filter gives great results. But there is one problem. It gets rid of the complete background.
- There are times, when we need to enhance the edges but also retain some of the background. To do that, we use a modified version of the high pass filter known as High-Boost filtering.
- In High-Boost filtering, we pass some of the background along with the high frequency content.

  We know that, High pass = Original – Low pass
- To pass some of the background, we multiply the original image with a multiplicative factor A. This gives us high boost filtering.

  Hence,

  High Boost = (A) original – Low pass

  = (A –1) Original + Original – Low pass

  High Boost = (A –1) Original + High pass

  If A = 1, then

  High Boost = High pass
- If A > 1, then some of the original signal is added back to the high pass result. This process restores some of the background into the high passed image. This technique is also known as unsharp masking. The mask coefficients for high boost filtering are as follows,

  Here X= 9A – 1

  Hence if A = 1, X = 8 which is a high pass mask.

$\frac{1}{9}$

–1	–1	–1
– 1	X	–1
–1	–1	–1

3×3 High boost mask

  If A = 1.1, X = 8.9.

  We have a mask which is as shown

$\frac{1}{9}$

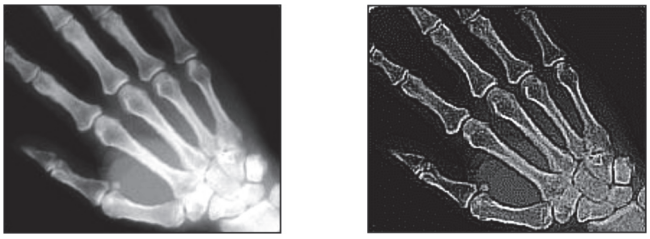
–1	–1	–1
– 1	8.9	–1
–1	–1	–1

3×3 High boost mask

- We can select different values of A and see the difference.
- Use the same pseudo image as the one used in the high pass example to see the results. What you will notice is that places which had a zero in the high pass example will now have some positive values. Hence it does not eliminate the background completely. This technique is one of the basic tools that is used in the printing industry.
- There are other neighbourhood techniques like Robert's filtering, Sobel's filtering and Prewitt's filtering, which are similar to the methods discussed above. They form a separate chapter called Segmentation and hence would be discussed later.

MATLAB program for high-boost filtering

```
%% High boost filtering
clear all
clc
aa=imread ('xray.tif');
a=double(aa);
[row col]=size (a);
w=[-1 -1 -1; -1 8.9 -1; -1 -1 -1]; %% High boost
mask
for x=2:1:row-1
        for y=:1:col-1
            a1(x,y)= w(1)*a(x-1,y-1)+w(2)*
                     a(x-1,y)+w(3)* …
             a(x-1,y+1)+w(4)*a(x,y-
1)+w(5)*a(x,y)+w(6)* …
        a(x,y+1)+w(7)*a(x+1,y-
1)+w(8)*a(x+1,y)+w(9)* …
             a(x+1,y+1);
        end
end
figure(1)
imshow(uint8(a))
figure(2)
imshow(uint8(a1))
```



(a) Original (b) High boost

Fig. 7.6.1

The generalized MATLAB program for high pass filtering using any mask size is given next.

```
%% High pass filtering of an image %%
clear all;
clc;
a=imread('mri2.tif');
a=double(a);
[row col]=size(a);
m = input('Enter the mask size:');
for i=1:1:m
        for j=1:1:m
            w(i,j)=-1
        end
end
s=(m+1)/2;
w(s,s)=m^2-1;
for x=1:1:row
        for y=s:1:col
            b(x,y)=a(x,y);
        end
end
for x=s:1:row-s
        for y=s:1:col-s
            b(x,y)=0;
        end
end
for x=s:1:row-s
        for y=s:1:col-s
            for i=1:1:m
                for j=1:1:m
                    b(x,y)=a(x-s+i,y-s+j)*w(i,j)+b(x,y);
                end
            end
            if b(x,y) < 0
                b(x,y)=0;
            end
        end
end
n=0
for x=s:1:row-s
        for y=s:1:col-s
            if b(x,y) > n
                n=b(x,y);
            end
        end
end
c=255/n;
for x=s:1:row-s
        for y=s:1:col-s
            b(x,y)=c*b(x,y);
        end
end
subplot(2,1,1)
imshow(uint8(a))
title('Original Image');
subplot(2,1,2)
imshow(uint8(b))
title('High Pass Filtered Image')
```

7.7 Zooming

- Another important application in image enhancement where the spatial domain neighbourhood operation is used image zooming.
- Zooming of images is not something that is new to you. If you have used Microsoft paint or Photoshop editor, you would be aware that there is an option which allows us to zoom an image by 25%, 50%, and 100% ……Have you ever imagined as to how an image that looked small can suddenly look so big ? The technique is quite simple and after reading the next couple of pages, you would be able to do it yourself.
- Zooming can be carried out using two different methods,

  (1) Replication (2) Interpolation
- We will discuss both these methods in detail. We shall first start with zooming using replication.

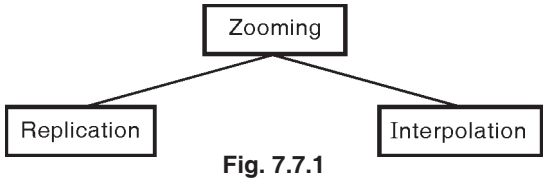


Fig. 7.7.1

7.7.1 Replication

- In replication, we simply replicate each pixel and then replicate each row. Consider the pseudo-image shown below.

1	2	3	4
5	6	7	8
9	8	6	7
0	1	2	3

- As stated earlier, we start from the first row. We replicate each pixel and then replicate each row. The first row now looks like

  1 1 2 2 3 3 4 4

  We now replicate this row to get

  1 1 2 2 3 3 4 4

  1 1 2 2 3 3 4 4

  Performing this operation on the entire image we get

1	1	2	2	3	3	4	4
1	1	2	2	3	3	4	4
5	5	6	6	7	7	8	8
5	5	6	6	7	7	8	8
9	9	8	8	6	6	7	7
9	9	8	8	6	6	7	7
0	0	1	1	2	2	3	3
0	0	1	1	2	2	3	3

- Hence a 4×4 image is zoomed to a 8×8 image. This method can be repeated to get bigger images. Remember, this is an image enhancement technique and hence no new data is added.
- In the zoomed pseudo-image, we observe that as we increase the size of the image, clusters of grey levels are formed which are disconcerting to the observer. Hence zooming increases the size of the image no doubt, but it also gives the image a patchy look.
- Zooming by replication can be implemented on the computer by using a replication mask. The first step is to interlace the original image with zeros. This is known as zero interlacing.
- In this we add zeros after every pixel and then add a complete row of zeros. Adding zeros to every other pixel of the first row we get,

  1 0 2 0 3 0 4 0

- This is also known as zero interlacing the columns, as we add zeros at every other column. Now inserting a row full of zeros gives us. This is known as zero interlacing the rows as we add zeros at every other row.

  1 0 2 0 3 0 4 0

  0 0 0 0 0 0 0 0

1	0	2	0	3	0	4	0
0	0	0	0	0	0	0	0
5	0	6	0	7	0	8	0
0	0	0	0	0	0	0	0
9	0	8	0	6	0	7	0
0	0	0	0	0	0	0	0
0	0	1	0	2	0	3	0
0	0	0	0	0	0	0	0

- This image is known as the zero interlaced image. On this image, we run a replication mask given below to get the zoomed image.

1	1
1	1

- The final zoomed image is as follows,

1	1	2	2	3	3	4	4
1	1	2	2	3	3	4	4
5	5	6	6	7	7	8	8
5	5	6	6	7	7	8	8
9	9	8	8	6	6	7	7
9	9	8	8	6	6	7	7
0	0	1	1	2	2	3	3
0	0	1	1	2	2	3	3

Note : To get the above result, we need to add a row of zeroes at the top and a column of zeroes at the beginning of the zero interlaced image making it 9×9.

– As mentioned earlier, zooming by replication gives the final image a patchy look since clusters of grey levels are formed. This can be substantially reduced by using a better method of zooming known as interpolation.

7.7.2 Linear Interpolation

– In this method, instead of replicating each pixel, average of the two adjacent pixels along the rows is taken and placed between the two pixels. The same operation is then performed along the columns. This operation is done on the interlaced image.

1	0	2	0	3	0	4	0
0	0	0	0	0	0	0	0
5	0	6	0	7	0	8	0
0	0	0	0	0	0	0	0
9	0	8	0	6	0	7	0
0	0	0	0	0	0	0	0
0	0	1	0	2	0	3	0
	0	0	0	0	0	0	0

Interpolation along rows is given by,

$$v_1(m, 2n) = u(m, n); \quad 0 \le m \le M-1, 0 \le n \le N-1$$

$$v_1(m, 2n+1) = \frac{1}{2}[u(m, n) + u(m, n+1)];$$

$$0 \le m \le M-1; \quad 0 \le n \le N-1$$

Interpolation of the column is given by,

$$v(2m, n) = v_1(m, n);$$

$$v(2m+1, n) = \frac{1}{2}[v_1(m, n) + v_1(m+1, n)];$$

$$0 \le m \le M-1; 0 \le n \le N-1$$

Hence the first row now becomes the modified image as follows :

1	1.5	2	2.5	3	3.5	4	2
0	0	0	0	0	0	0	0
5	5.5	6	6.5	7	7.5	8	4
0	0	0	0	0	0	0	0
9	8.5	8	7	6	6.5	7	3.5
0	0	0	0	0	0	0	0
0	0.5	1	1.5	2	2.5	3	1.5
0	0	0	0	0	0	0	0

– This is the first step. We now proceed to find the average values along the columns. Hence the final image is as shown. In practice, we need to round of the pixel values.

1	1.5	2	2.5	3	3.5	4	2
3	3.5	4	4.5	5	5.5	6	3
5	5.5	6	6.5	7	7.5	8	4
7	7	7	6.25	6.5	7	7.5	3.25
9	8.5	8	7	6	6.5	7	3.5
4.5	4.5	4.5	4.25	4	4.5	5	2.5
0	0.5	1	1.5	2	2.5	3	1.5
0	0.25	0.5	0.75	1	1.25	1.5	0.75

– If we compare the results of replication and interpolation, it is clear that the patchiness that was present in the replicated image is much less in the interpolated image. Hence we can conclude that zooming by interpolation is more effective than zooming by replication.

– To implement zooming by interpolation, we simply run an interpolation mask on the zero interlaced image.

1/4	1/2	1/4
1/2	1	1/2
1/4	1/2	1/4

MATLAB program for zooming by replication as well as interpolation

```
%% Zooming using replication as well as interpolation %%
clc
clear all
x=imread ('warne.tif');
s=size(x);
ss=s(1)+s(2);      %% To set the size of the output image %%
            y=zeros(ss);
              i=1;
              j=1;
for n=1:2:ss
                  for m=1:2:ss
                  y (m, n)= x (i, j);
```

```
                i=i+1;
                end
               i=1; %% Else the value of i goes on
                      increasing%%
               j=j+1;
end
h1=[1 1;1 1];                % Mask for replication
h2=[1/4  1/2  1/4;  1/2  1  1/2;  1/4  1/2  1/4];
                  %% Mask for interpolation %%
A1=conv2 (y,h1); %% We can use the formula that is used
in Low pass filtering %%
A2=conv2 (y,h2);
figure(1),imshow(x)
figure(2),imshow(uint8 (A1))
figure(3),imshow(uint8 (A2))
```

(a) Original

(b) Zooming by replication

(c) Zooming by interpolation

Fig. 7.7.2

7.8 Solved Examples

Ex. 7.8.1 : Obtain the digital negative of the following 8 Bits Per Pixel - BPP image.

121	205	217	156	151
139	127	157	117	125
252	117	236	138	142
227	182	178	197	242
201	106	119	251	240

MU - May 2014, 10 Marks

Soln. :

It is known that it is a 8-bit image. Hence the number of grey levels that this image can hold is $2^8 = 256$

$\therefore$ L = 256.

Hence the minimum grey level is 0

while the maximum grey level is 255

$$s(x,y) = (L-1) - r(x,y) = (256-1) - r(x,y)$$

$$s(x,y) = 255 - r(x,y)$$

We get the digital negative using the above equation.

134	50	38	99	104
116	128	98	138	130
3	138	19	117	113
28	73	77	58	13
54	149	136	4	15

Ex. 7.8.2 : For a given image find -

(i) Digital negative of an image.

(ii) Bit plane slicing.

4	3	2	1
3	1	2	4
5	1	6	2
2	3	5	6

Soln. : We assume the image to be 3-bit

(i) Digital negative

$$s = (L-1) - r$$

$$\text{or } g(x,y) = (L-1) - f(x,y)$$

Here $L = 2^3 = 8$

$$\therefore s = 7 - r$$

$$\text{i.e., } g(x,y) = 7 - f(x,y)$$

$\therefore$ The digital negative of the image is

3	4	5	6
4	6	5	3
2	6	1	5
5	4	2	1

(ii) Bit plane slicing

In this, we convert the given image into binary and then separate the planes.

We assume the image to be 3 bit.

100	011	010	001
011	001	010	100
101	001	110	010
010	011	101	110

Separating the bit planes we get,

1	0	0	0
0	0	0	1
1	0	1	0
0	0	1	1

MSB plane

0	1	1	0
1	0	1	0
0	0	1	1
1	1	0	1

Center bit plane

0	1	0	1
1	1	0	0
1	1	0	0
0	1	1	0

LSB plane

Ex. 7.8.3 : For following image find :

Contrast stretching $r_2 = 5$, $r_1 = 3$, $s_2 = 6$, $s_1 = 2$.

f (x, y) =

4	3	2	1
3	1	2	4
5	1	6	2
2	3	5	6

MU - May 2016, 10 Marks

Soln. :

Contrast stretching $r_2 = 5$, $r_1 = 3$, $s_2 = 6$, $s_1 = 2$:

f (x, y) =

4	3	2	1
3	1	2	4
5	1	6	2
2	3	5	6

The contrast stretching transformation is shown in Fig. P. 7.8.3.

From the values of r_1 and s_1 we can find the slope of α. In a similar manner, we can find the value of β using the values of r_1, r_2 and s_1, s_2.

From the values of r_2 and s_2, we can find the value of γ.

$$\alpha = \frac{s_1}{r_1} = \frac{2}{3} = 0.66$$

$$\beta = \frac{y_2 - y_1}{x_2 - x_1} = \frac{6-2}{5-3} = 2$$

Finally,

$$\gamma = \frac{y_2 - y_1}{x_2 - x_1} = \frac{7-6}{7-5} = 0.5$$

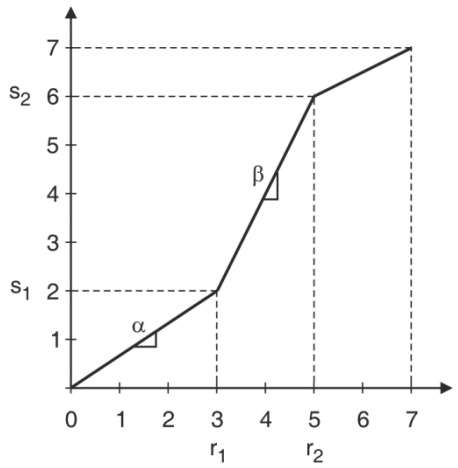


Fig. P. 7.8.3

Hence we have the required slopes to compute contrast stretching. The contrast stretching formula is given below.

$$s = \begin{cases} \alpha\, r & 0 \le r < 3 \\ \beta \cdot (r - r_1) + s_1 & 3 \le r < 5 \\ \gamma\,(r - r_2) + s_2 & 5 \le r < 7 \end{cases}$$

Here $r_1 = 3$, $r_2 = 5$, $s_1 = 2$, $s_2 = 6$

$\alpha = 0.66$, $\beta = 2$, $\gamma = 0.5$

We make a table of values for r and s using the contrast stretching formula.

r	s
0	$s = \alpha r = 0.66 \times 0 = 0$
1	$s = \alpha\, r = 0.66 \times 1 = 0.66$
2	$s = \alpha \cdot r = 0.66 \times 2 = 1.32$
3	$s = \beta\,(r - r_1) + s_1 = 2\,(3-3) + 2 = 2$
4	$s = \beta\,(r - r_1) + s_1 = 2\,(4-3) + 2 = 4$
5	$s = \gamma\,(r - r_2) + s_2 = 0.5\,(5-5) + 6 = 6$
6	$s = \gamma\,(r - r_2) + s_2 = 0.5\,(6-5) + 6 = 6.5$
7	$s = \gamma\,(r - r_2) + s_2 = 0.5\,(7-5) + 6 = 7$

Hence the contrast stretched image is

4	2	1.32	0.66
2	0.66	1.32	4
6	0.66	6.5	1.32
1.32	2	6	6.5

If we round off, the final image would be $\Rightarrow$

4	2	1	1
2	1	1	4
6	1	7	1
1	2	6	7

Ex. 7.8.4 : For the 3-bit 4 × 4 size image, perform the following operations :

(i) Negation

(ii) Thresholding with T = 4

(iii) Intensity level slicing with background $r_1 = 2$ and $r_2 = 5$

(iv) Bit plane slicing for MSB and LSB planes

(v) Clipping with $r_1 = 2$ and $r_2 = 5$

1	2	3	0
2	4	6	7
5	2	4	3
3	2	6	1

Soln. : Let the given image be f(x, y)

(i) Negation

$$s = (L - 1) - r$$

OR

$$g(x, y) = (L - 1) - f(x, y)$$

Since the image is 3-bit, $L = 2^3 = 8$

$$\therefore L - 1 = 7$$

$$\therefore g(x, y) = 7 - f(x, y)$$

Hence the digital negative of the image is

g(x, y) =

6	5	4	7
5	3	1	0
2	5	3	4
4	5	1	6

(ii) Thresholding with T = 4

The thresholding function is shown in Fig. P. 7.8.4 along with thresholding formula.

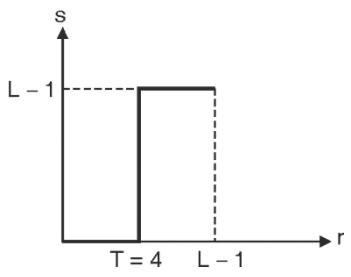


Fig. P. 7.8.4

$$g(x, y) = \begin{cases} 0 & \text{if } f(x, y) \leq 4 \\ L - 1 & \text{if } f(x, y) \geq 4 \end{cases}$$

Here

$$L - 1 = 7$$

$\therefore$ The thresholded image is as follows,

7	7	7	7
7	0	0	0
0	7	0	7
7	7	0	7

(iii) Bit plane slicing

We convert the given image into binary and then separate the plane. Since the given image is 3-bit, the binary image would be as follows,

001	010	011	000
010	100	110	111
101	010	100	011
011	010	110	001

Separating the bit planes we obtain.

0	0	0	0
0	1	1	1
1	0	1	0
0	0	1	0

MSB plane

0	1	1	0
1	0	1	1
0	1	0	1
1	1	1	1

Center bit plane

1	0	1	0
0	0	0	1
1	0	0	1
1	0	0	1

LSB plane

(iv) Clipping with $r_1 = 2$, $r_2 = 5$

The clipping transformation is shown in Fig. P. 7.8.4(a).

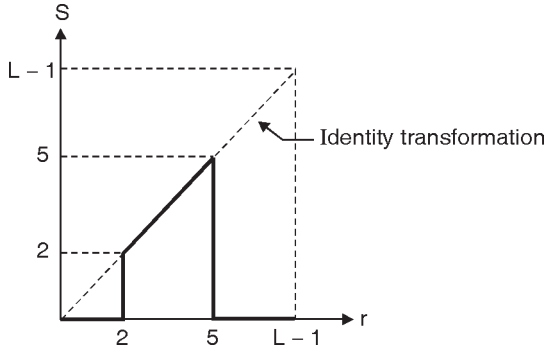


Fig. P. 7.8.4(a)

$$s = \begin{cases} r & 2 \le r \le 5 \\ 0 & \text{otherwise} \end{cases}$$

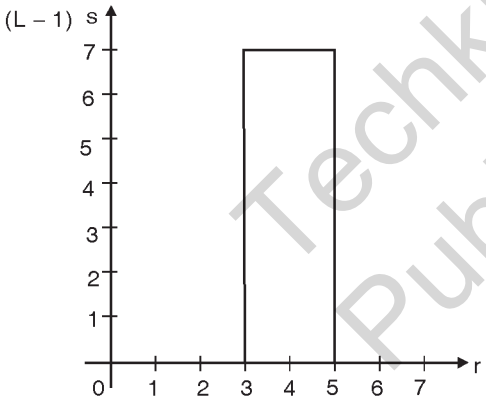
Hence the final clipped image is as follows,

0	2	3	0
2	4	0	0
5	2	4	3
3	2	0	0

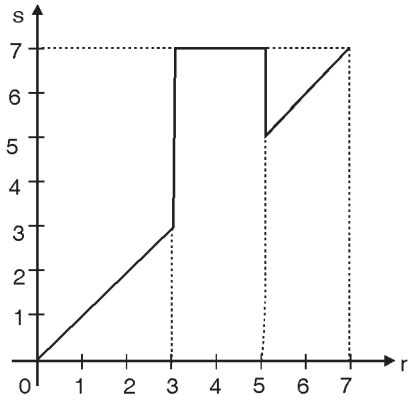
Ex. 7.8.5 : Perform intensity level (grey level) slicing on the 3 BPP image. Let $r_1 = 3$ and $r_2 = 5$. Draw the modified image using with background and without background transformations.

Soln. : Let us draw the grey level transformation.

2	1	2	2	1
2	3	4	5	2
6	2	7	6	0
2	6	6	5	1
0	3	2	2	1



(a)



(b)

Fig. P. 7.8.5

$s = L - 1 \quad r_1 \le r \le r_2$ $\qquad$ $s = L - 1 \quad r_1 \le r \le r_2$

$s = 0 \quad$ otherwise $\qquad$ $s = r \quad$ otherwise

0	0	0	0	0
0	7	7	7	0
0	0	0	0	0
0	0	0	7	0
0	7	0	0	0

Without background

2	1	2	2	1
2	7	7	7	2
6	2	7	6	0
2	6	6	7	1
0	7	2	2	1

With background

Ex. 7.8.6 : The grey levels in an image range from 10 to 50. We need to display the image on a device that has a grey level range of 0 to 255. Give a transformation which would accomplish this.

Soln. : The transformation that we use is known as the Range normalization.

It is given by,

$$s(x, y) = \frac{d - c}{b - a}[r(x, y) - a] + c$$

This transformation converts the original range [a, b] to the new range [c, d].

In this sum a = 10, b = 50, c = 0, d = 255.

$$s(x, y) = \frac{255}{40}[r(x, y) - 10] + 0$$

$$s(x, y) = \frac{51}{8} r(x, y) - \frac{255}{4}$$

$$\text{When } r = 10, \; s = 0$$

$$r = 50, \; s = 255$$

Ex. 7.8.7 : The image shown below has 8 different grey levels. Plot this image using only 4 grey levels.

0	1	1	1	1	4
1	1	2	3	2	2
1	1	2	2	3	3
1	2	4	6	2	3
1	2	4	2	4	4
1	2	3	7	2	5

Soln. :

This can be achieved using the Grey level reduction transformation which is given by,

$$s(x, y) = \left[\frac{Gr(x, y)}{K}\right]\frac{K}{G}$$

Here $K \rightarrow$ Original number of grey levels

$G \rightarrow$ Modified number of grey levels

$[\,.\,] \rightarrow$ Choosing the floor value.

$$s(x, y) = \left[\frac{4r(x, y)}{8}\right]\frac{8}{4}$$

∴ r(x, y)	s(x, y)	
0	0	1
1	0	
2	2	2
3	2	
4	4	3
5	4	
6	6	4
7	6	

∴ The modified image is

0	0	0	0	0	4
0	0	2	2	2	2
0	0	2	2	2	2
0	2	4	6	2	2
0	2	4	2	4	4
0	2	2	6	2	4

Ex. 7.8.8 : Show that a high pass filtered image can be obtained in the spatial domain as High pass = Original – Low pass.

Soln. :

Original image

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Low pass filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

High pass filter

–1/9	–1/9	–1/9
–1/9	8/9	–1/9
–1/9	–1/9	–1/9

When we apply the LPF on the image, the center pixel z_5 changes to

$$\frac{1}{9}[z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9]$$

$$\text{Original} - \text{Low pass} = z_5 - \frac{1}{9}[z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9]$$

$$= z_5 - \frac{z_1}{9} - \frac{z_2}{9} - \frac{z_3}{9} - \frac{z_4}{9} - \frac{z_5}{9} - \frac{z_6}{9} - \frac{z_7}{9} - \frac{z_8}{9} - \frac{z_9}{9}$$

$$= \frac{8z_5}{9} - \frac{1}{9}[z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9]$$

This is nothing but a high pass mask

$$\frac{1}{9} \times \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Note : We can also prove this by taking a pseudo-image. We move a low pass mask on the original image and subtract it from the original. We now move a high pass mask on the original image. The results of both these operations will be the same.

Ex. 7.8.9 : Filter the following image using 3 × 3 neighbourhood averaging by assuming (i) Zero padding (ii) Pixel replication

$$f(x, y) = \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 2 \\ \hline 4 & 2 & 5 & 1 \\ \hline 1 & 2 & 6 & 3 \\ \hline 2 & 4 & 6 & 7 \\ \hline \end{array}$$

Soln. :

A 3 × 3 averaging mask is shown below :

$$w(x, y) = \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filtering is performed using the convolution operation,

i.e., $g(x, y) = f(x, y) * w(x, y)$.

(i) Zero padding

In this, we zero pad the image before performing the filtering operation. This gives us a 6 × 6 image.

f (x, y) =

0	0	0	0	0	0
0	1	2	3	2	0
0	4	2	5	1	0
0	1	2	6	3	0
0	2	4	6	7	0
0	0	0	0	0	0

We move the averaging mask over this image starting from the top left corner as follows,

$\therefore$ g (x, y) = f (x, y) * w (x, y)

0	0	0	0	0	0
0	1	1.88	1.66	1.22	0
0	1.33	2.88	2.88	2.22	0
0	1.66	3.55	4	3.11	0
0	1	2.33	3.11	2.44	0
0	0	0	0	0	0

(ii) Pixel replication

In this we replicate the border pixels to generate a 6×6 image.

f (x, y) =

1	1	2	3	2	2
1	1	2	3	2	2
4	4	2	5	1	1
1	1	2	6	3	3
2	2	4	6	7	7
2	2	4	6	7	7

We move the averaging mask over this image starting from the top left corner as shown.

$\therefore$ g (x, y) = f (x, y) * w (x, y)

g (x, y) =

1	1	2	3	2	2
1	2	2.55	2.44	2.33	2
4	2	2.88	2.88	2.88	1
1	2.44	3.55	4	4.33	3
2	2.22	3.66	5	5.77	7
2	2	4	6	7	7

Ex. 7.8.10 : Perform discrete convolution on the following image arrays. Assume that the left bottom corner elements are at the origin in both cases. f_1 and f_2 are two images.

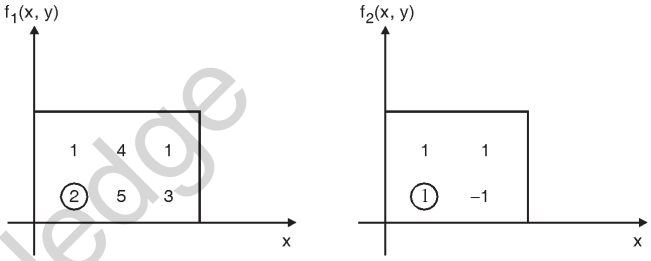


Fig. P. 7.8.10

Soln. :

$$f_1(x, y) * f_2(x, y) = \sum_m \sum_n f_1(m, n)\, f_2(x - m, y - n)$$

Let $f_1 (x, y) * f_2 (x, y) = g(x, y)$

$$g(x, y) = \sum_m \sum_n f_1(m, n)\, f_2(x - m, y - n)$$

$$g(0, 0) = \sum_m \sum_n f_1(m, n)\, f_2(-m, -n)$$

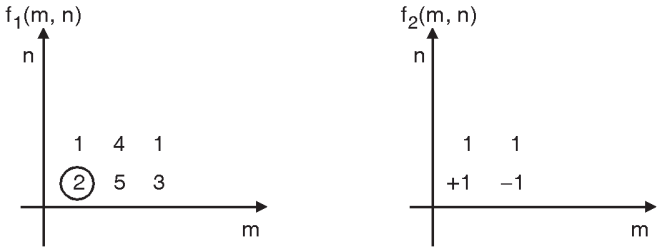


Fig. P. 7.8.10(a)

The basic shift operations are

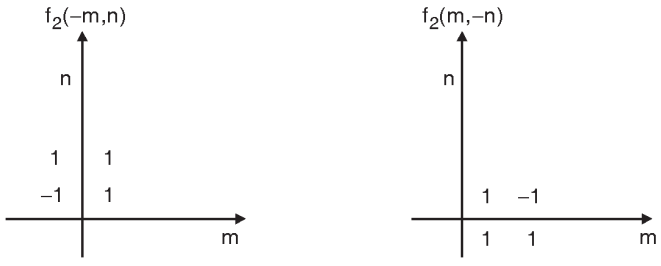


Fig. P. 7.8.10(b)

$f_2(-m, n) \Rightarrow$ fliping about m-axis $f_2(m, -n) \Rightarrow$ fliping about n-axis

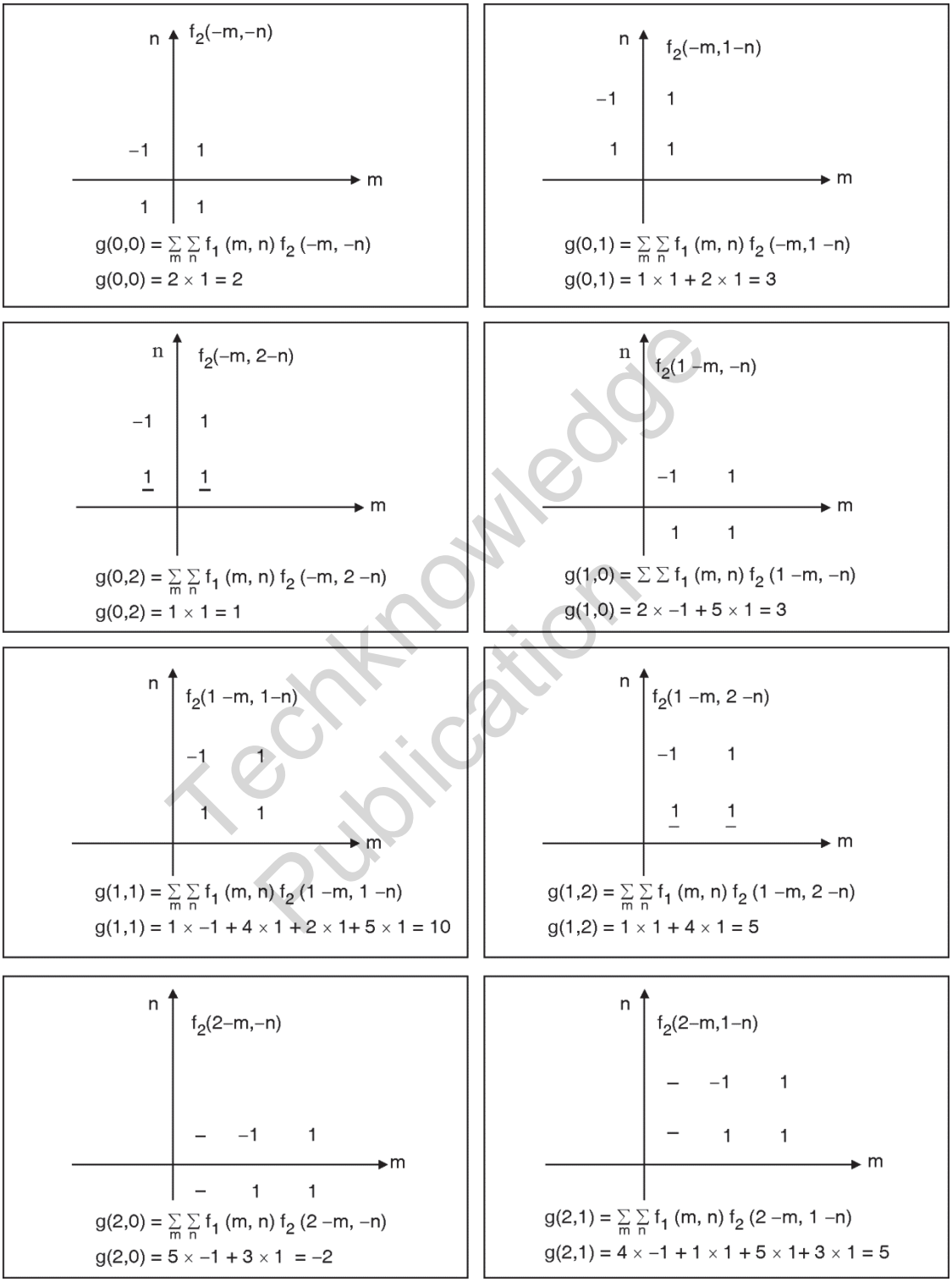


Fig. P. 7.8.10(c)cont...

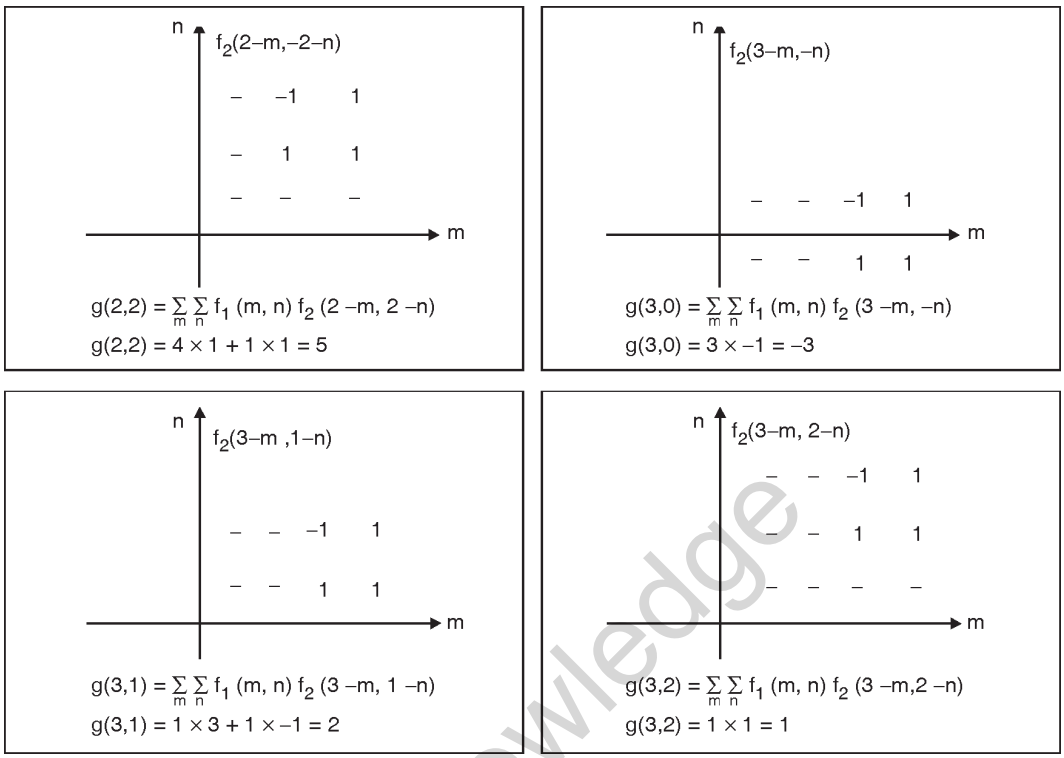


Fig. P. 7.8.10(c)

Another method : It is convenient to represent digital images as matrices and exploit the benefits of linear algebra. We modify the matrix so as to suit convolution.

Let F_1 and F_2 be the two images that need to be convolved :

Let F_1 be a $m_1 \times n_1$ image and F_2 be a $m_2 \times n_2$ image. We zero pad F_1 and F_2 so that both of them are of size $(m_1 + m_2 - 1) \times (n_1 + n_2 - 1)$.

Ex. 7.8.11 : Perform discrete convolution on the given image arrays

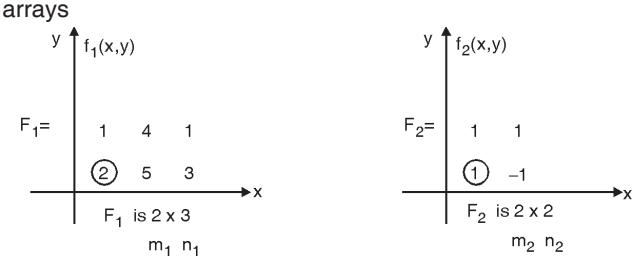


Fig. P. 7.8.11

Soln. :

Zero padding F_1 and F_2 so that both are of size

$$(m_1 + m_2 - 1) \times (n_1 + n_2 - 1) = 3 \times 4$$

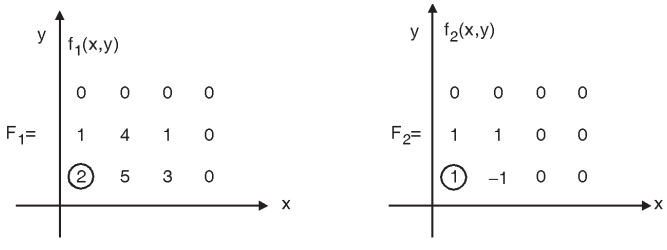


Fig. P. 7.8.11(a)

Now, of these two, arrange one as a circular matrix and the other one as a row stacking matrix. Let the first element of these be their respective origins, then

$$G(x, y) = F_1(x, y)\, F_2(x, y)$$

F_1 F_2 G

$$\begin{bmatrix} \textcircled{2} & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 & 3 & 5 \\ 5 & \textcircled{2} & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 & 3 \\ 3 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 \\ 0 & 3 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 \\ 1 & 0 & 3 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 4 \\ 4 & 1 & 0 & 3 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 4 & 1 & 0 & 3 & 5 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & 3 & 5 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 1 & 0 & 3 & 5 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 4 & 1 & 0 & 3 & 5 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 & 3 & 5 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 & 3 & 5 & 2 \end{bmatrix} \times \begin{bmatrix} \textcircled{1} \\ -1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \textcircled{2} \\ 3 \\ -2 \\ -3 \\ 3 \\ 10 \\ 5 \\ 2 \\ 1 \\ 5 \\ 5 \\ 1 \end{bmatrix}$$

$\therefore$ G (x, y)

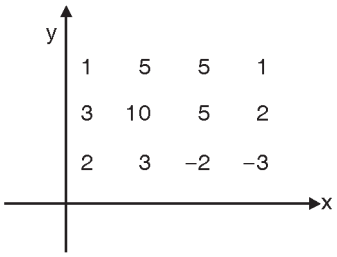


Fig. P. 7.8.11(b)

We see that the results are the same as obtained from the previous method.

Ex. 7.8.12 : Apply Low and High Pass Spatial masks on the following Fig. P. 7.8.12 matrix.
Prove that High pass = Original – Low pass. Assume virtual Rows and Columns.

f(x,y) =

30	31	32
33	120	30
32	32	31

Fig. P. 7.8.12

Soln. : We zero padd the image to make it 5×5

f(x,y) =

0	0	0	0	0
0	30	31	32	0
0	33	120	30	0
0	32	32	31	0
0	0	0	0	0

We convolve this image with a standard 3×3 Low pass and a High pass mask.

(i) Low Pass

Low pass mask h_1 (x,y) =

1	1	1
1	1	1
1	1	1

$$g_1(x,y) = f(x,y) * h_1(x,y) =$$

23.77	30.66	23.66
30.88	41.22	30.66
24.11	30.88	23.667

(ii) High Pass

$$\text{High pass mask } h_2(x,y) = \frac{1}{9}$$

– 1	– 1	– 1
– 1	8	–1
– 1	– 1	– 1

$$g_2(x,y) = f(x,y) * h_2(x,y) =$$

6.22	0.33	8.33
2.11	78.77	– 0.667
7.889	1.11	7.33

We now show that High pass = Original – Low pass

$$\text{Original} - \text{Lowpass} =$$

30	31	32
33	120	30
32	32	31

–

23.77	30.66	23.66
30.88	41.22	30.66
24.11	30.88	23.667

=

6.22	0.33	8.33
2.11	78.77	– 0.667
7.889	1.11	7.33

This is the same as the High pass result. Therefore, High pass = Original – Low pass

Ex. 7.8.13 : Given $f(x, y) = \begin{bmatrix} 5 & 6 & 7 \\ 8 & 9 & 10 \end{bmatrix}$, $h(x, y) = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$

Find linear convolution of input image f(x, y) with filter of h(x, y).

Soln. : Let the output image be g (x, y)

$$\text{Here, } g(x, y) = f(x, y) * h(x, y)$$

We shall solve this using the circular matrix method. (The other method is the graphical technique).

We zero pad f(x, y) and h(x, y) in such a way that both of them are of size.

$(m_1 + m_2 - 1) \times (n_1 + n_2 - 1)$

Here $m_1 \times n_1$ is the size of f(x, y) [2×3] and $m_2 \times n_2$ is the size of h(x, y) [2×2], Therefore $m_1 = 2$, $m_2 = 2$, $n_1 = 3$, $n_2 = 2$. Hence zero padding should be done in such a manner so that f_1 (x, y) and h(x, y) are of size $(2 + 2 - 1) \times (3 + 2 - 1)$.

$$= 3 \times 4$$

$$\therefore \quad f(x, y) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 5 & 6 & 7 & 0 \\ \textcircled{8} & 9 & 10 & 0 \end{bmatrix}; h(x, y)$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ \textcircled{3} & 4 & 0 & 0 \end{bmatrix}$$

We now arrange one of these as a circular matrix and the other one as a row slacking matrix. Let us convert f(x, y) to a circular matrix. Let the encircled terms be the starting points

$$\begin{bmatrix} 8 & 0 & 0 & 0 & 0 & 0 & 7 & 6 & 5 & 0 & 10 & 9 \\ 9 & 8 & 0 & 0 & 0 & 0 & 0 & 7 & 6 & 5 & 0 & 10 \\ 10 & 9 & 8 & 0 & 0 & 0 & 0 & 0 & 7 & 6 & 5 & 0 \\ 0 & 10 & 9 & 8 & 0 & 0 & 0 & 0 & 0 & 7 & 6 & 5 \\ 5 & 0 & 10 & 9 & 8 & 0 & 0 & 0 & 0 & 0 & 7 & 6 \\ 6 & 5 & 0 & 10 & 9 & 8 & 0 & 0 & 0 & 0 & 0 & 7 \\ 7 & 6 & 5 & 0 & 10 & 9 & 8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7 & 6 & 5 & 0 & 10 & 9 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 6 & 5 & 0 & 10 & 9 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 6 & 5 & 0 & 10 & 9 & 8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 6 & 5 & 0 & 10 & 9 & 8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 & 6 & 5 & 0 & 10 & 9 & 8 \end{bmatrix} \times \begin{bmatrix} 3 \\ 4 \\ 0 \\ 0 \\ 1 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Multiplying the two matrices gives us

$$\begin{bmatrix} (24) \\ 59 \\ 66 \\ 40 \\ \hline 23 \\ 63 \\ 73 \\ 48 \\ \hline 5 \\ 16 \\ 19 \\ 14 \end{bmatrix}$$

We re-arrange this matrix to generate a filtered image.

$$\therefore g(x, y) = \begin{bmatrix} 5 & 16 & 19 & 14 \\ 23 & 63 & 73 & 48 \\ 24 & 59 & 66 & 40 \end{bmatrix}$$

We would get the same answer if we make h(x, y) into a circular matrix instead of f(x, y). Go ahead and try it out yourself.

Ex. 7.8.14 : For the given 3 bit, 4×4 size image perform the following operations :

(i) Thresholding (T = 4)

(ii) Intensity level slicing with background for $r_1 = 2$, $r_2 = 5$

(iii) Bit plane slicing for LSB and MSB planes

(iv) Negation.

4	2	3	0
1	3	5	7
5	3	2	1
2	4	6	7

MU - Dec. 2011, 10 Marks

Soln. :

(i) Thresholding (T = 4)

Since the given image is 3-bit, $L = 2^3 = 8$

The transformation for thresholding is shown in Fig. P. 7.8.14.

Here,

$$s = L - 1 = 7 \quad r \geq 4$$

$$s = 0 \quad r < 4$$

∴ The final image would be,

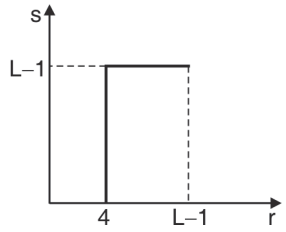


Fig. P. 7.8.14

7	0	0	0
0	0	7	7
7	0	0	0
0	7	7	7

(ii) Intensity level slicing with background for $r_1 = 2, r_2 = 5$

The transformation for intensity level slicing is shown in Fig. P. 7.8.14(a).

Here,

$$s = L - 1 = 7 \qquad 2 \le r \le 5$$

$$s = r \qquad \text{otherwise}$$

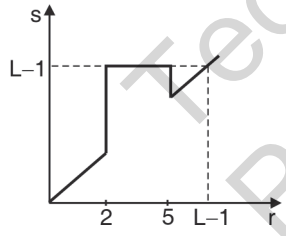


Fig. P. 7.8.14(a)

Hence all values between 2 and 5 are made equal to 7 and the remaining values remain unchanged. Hence the final result is,

7	7	7	0
1	7	7	7
7	7	7	1
7	7	6	7

(iii) Bit plane slicing

We convert the image to binary. Since there are 8 grey levels (0 to 7), we require 3 bits to represent each pixel.

4	2	3	0		100	010	011	000
1	3	5	7	Binary	001	011	101	111
5	3	2	1	⟶	101	011	010	001
2	4	6	7		010	100	110	111

The LSB and MSB planes are as follows,

0	0	1	0
1	1	1	1
1	1	0	1
0	0	0	1

LSB plane

1	0	0	0
0	0	1	1
1	0	0	0
0	1	1	1

MSB plane

(iv) Negation

The transformation for negation is shown in Fig. P. 7.8.14(b).

Here,

$$s = (L - 1) - r$$

$$s = 7 - r$$

Here the grey levels get inverted.

∴ The final image is as shown in Fig, 7.8.14(b)

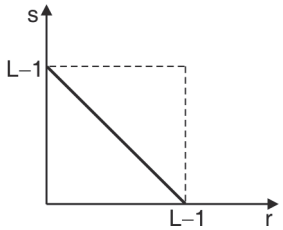
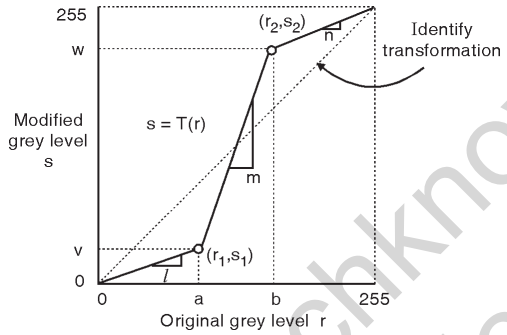
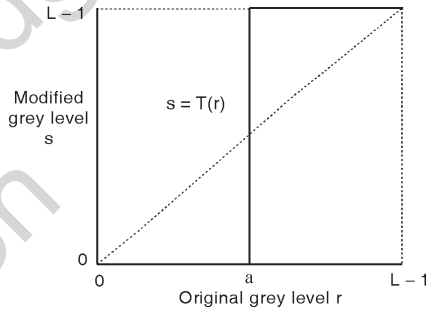


Fig. P. 7.8.14(b)

3	5	4	7
6	4	2	0
2	4	5	6
5	3	1	0

7.9 Comparison of Contrast Stretching and Thresholding

Sr. No.	Contrast Stretching	Thresholding
1.	Contrast stretching increases the dynamic range by making the dark pixels darker and bright pixels brighter.	Thresholding also increase that dynamic range by setting a threshold. All values less than the threshold are made black while values greater than the threshold are made equal to white.
2.	The formula of the contrast-stretching algorithm is given below : $s = \begin{cases} l \cdot r & 0 \le r < a \\ m \cdot (r-a) + v & a \le r < b \\ n \cdot (r-b) + w & b \le r < L-1 \end{cases}$	The formula for achieving thresholding is as follows : $s = 0$; if $r \le a$ $s = L-1$; if $r > a$
3.	The transformation for contrast stretching is shown below : 	The transformation for thresholding is shown below : 
4.	We can adjust the contrast in an image by changing the values of the slope.	Thresholding is extreme contrast stretching. In other words, a thresholded image has maximum contrast.
5.	The final result of contrast stretching gives us a grey scale image. The output image has different shades of grey.	The final result of thresholding gives us a binary image. Which means that the thresholded image has only black and white values.

Summary

In this chapter, we introduce the term spatial domain and image enhancement in the spatial domain. Different image enhancement techniques that improve the subjective quality of the image are explained. The methods described here can be classified into two broad groups : point processing and neighborhood processing. The concept of frequencies in an image is explained using simple illustrations. Basic filtering operations such low pass filtering and high pass filtering are presented. Advantages of each of the operations are stated using MATLAB codes. Any particular technique that may be applied depends on the subjective quality of the image as well as the purpose behind the processing. It must be noted that more than one technique can be used to get the desired result.

Review Questions

Q. 1 Explain spatial domain processing.

Q. 2 Compare and contrast average filtering and median filtering.

Q. 3 Explain the difference between operations involving 3×3 mask for median filtering and average filtering.

Q. 4 Develop a procedure for computing the median in $n \times n$ neighbourhood.

Q. 5 What do we mean by Gaussian noise and why is an averaging filter used to eliminate it ?

Q. 6 What is salt and pepper noise (binary noise) and how does a median filter remove it ?

Q. 7 Explain the output and application of the following zero memory enhancement techniques.

(1) Contrast stretching

(2) Thresholding

(3) Range compression

(4) Bit extraction.

Q.8 Compare and contrast the low pass and high pass spatial filters.

Q. 9 Explain why a median filter is called a salt and pepper filter.

Q. 10 Compare and contrast the smoothing and sharpening filters.

Q. 11 Low pass filter is a smoothing filter. Explain.

Q. 12 What do we mean by frequencies in an image ?

Q. 13 What is meant by unsharp masking and crisping ? Explain with relevant figures.

Q. 14 Given a image, what is the output using a 3×3 averaging filter and a medain filter ? Explain the result.

3 3 3 3 3 3
3 3 3 3 3 3
3 3 10 10 3 3
3 10 3 3 3 3
3 3 3 3 8 3
3 3 3 3 3 3
3 3 3 3 3 3
3 3 3 3 3 3

Q. 15 How is image subtraction carried out ? Give an example of image subtraction in the field of medicine.

Q. 16 Explain the procedure of zooming an image using replication and interpolation.

Q. 17 Why should the sum of any high pass mask be zero ?

Q. 18 Perform discrete convolution on the following image arrays.

1	3	2
②	5	6

2	4
①	3

Fig. Q. 18

Q. 19 Explain the power-law transformation.

❑❑❑

CHAPTER 8

Histogram Modelling

Syllabus

Histogram Processing, Histogram equalization

8.1 Introduction

- Histogram of images provide a global description of the appearance of an image. The information obtained from histograms is enormous and hence histogram modelling; though a spatial domain technique is introduced as a separate chapter.
- By definition, histogram of an image represents the relative frequency of occurrence of the various grey levels in an image. Histogram of an image can be plotted in two ways.
- In the first method, the x-axis has the grey levels and the y-axis has the number of pixels in each grey level, while in the second method, the x-axis represents the grey levels, while the y-axis represents the probability of the occurrence of that grey level.

Method 1

Grey level	Number of pixels (n_k)
0	40
1	20
2	10
3	15
4	10
5	3
6	2

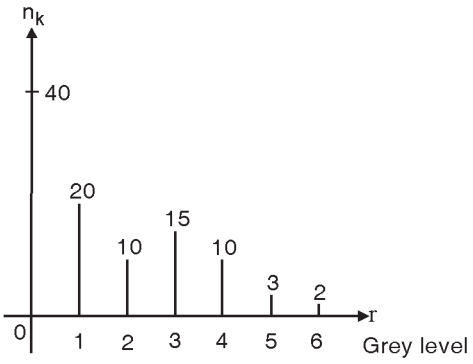


Fig. 8.1.1

Method 2

In the second method, instead of plotting the number of pixels directly, we plot their probability of occurrence i.e.,

$$p(r_k) = \frac{n_k}{n}$$

Where, $r_k \rightarrow k^{th}$ grey level

$n_k \rightarrow$ Number of pixels in the k^{th} grey level

$n \rightarrow$ Total number of pixels in an image

Grey level	Number of pixels (n_k)	$p(r_k) = \frac{n_k}{n}$
0	40	0.4
1	20	0.2
2	10	0.1
3	15	0.15
4	10	0.1
5	3	0.03
6	2	0.02
	n = 100	

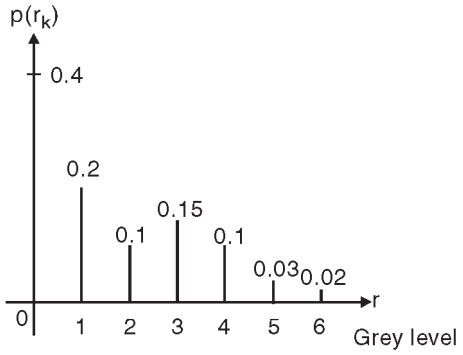


Fig. 8.1.2

- This is known as a normalized histogram. The advantage of the second method is that the maximum value to be plotted will always be 1. Generally black is considered as grey level 0 and white as the maximum.

– Just by looking at the histogram of the image, a great deal of information can be obtained. Some of the typical histograms are shown in Fig. 8.1.3.

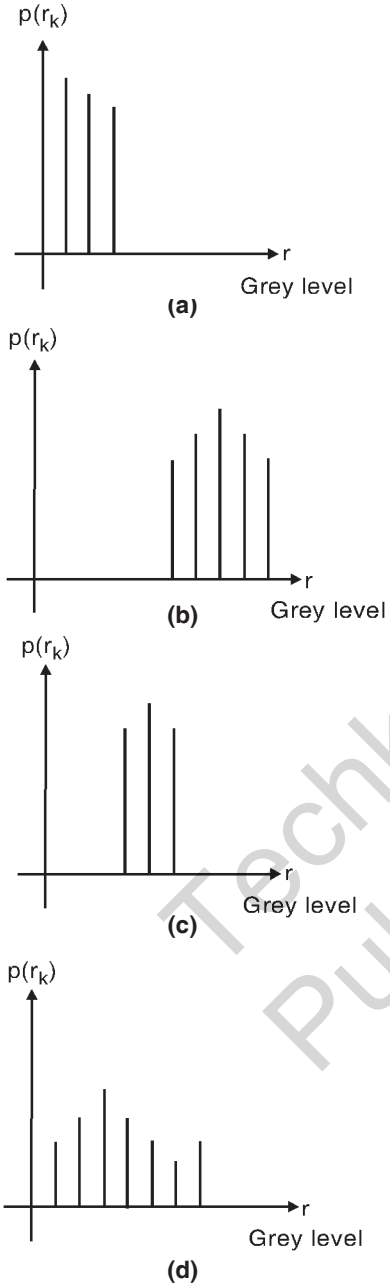


Fig. 8.1.3

– In Fig. 8.1.3(a) all the pixels belong to the lower grey levels 0, 1, ... and hence we can be sure that the image, represented by this histogram, is a dark image. Similarly, Fig. 8.1.3(b) is the histogram of a bright image. Fig. 8.1.3(c) is a low contrast image since only a small range of grey levels are present. Fig. 8.1.3(d) is a high contrast image of all the 4 histograms shown, the last histogram represents the best image.

– Our aim in this chapter would be to transform the first three histograms into the fourth histogram. In other words, we try to increase the dynamic range of the image. Though MATLAB has a inbuilt function to plot the histogram (hist) we shall write our own code.

MATLAB code for plotting the histogram of an image.

```
%% Program for histogram %%
clear all
clc
a=imread('test.tif');
a=double(a);
[row col]=size(a);
h=zeros(1,300);
for n=1:1:row
for m=1:1:col
if a(n,m)==0
%% To ensure that the values of a are not zero
            a(n,m)=1;
        end
    end
end
for n=1:1:row
    for m=1:1:col
        t=a(n,m);
%% Takes the value of the pixel ex. 12
        h(t)=h(t)+1;
%% Incrementing each counter h(12)
    end
end
%% PLOTTING %%
figure (1)
imshow(uint8(a))
figure(2)
bar(h)
```

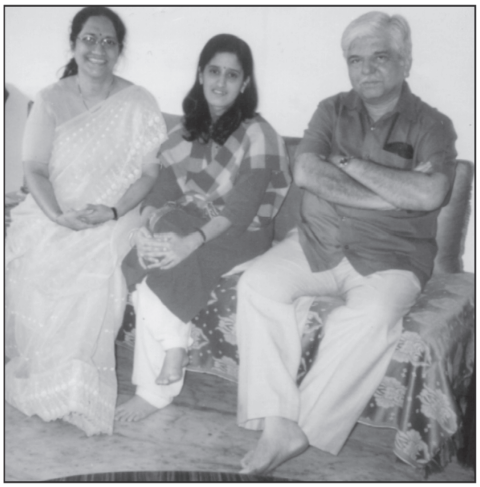


Fig. 8.1.4(a) : Original image

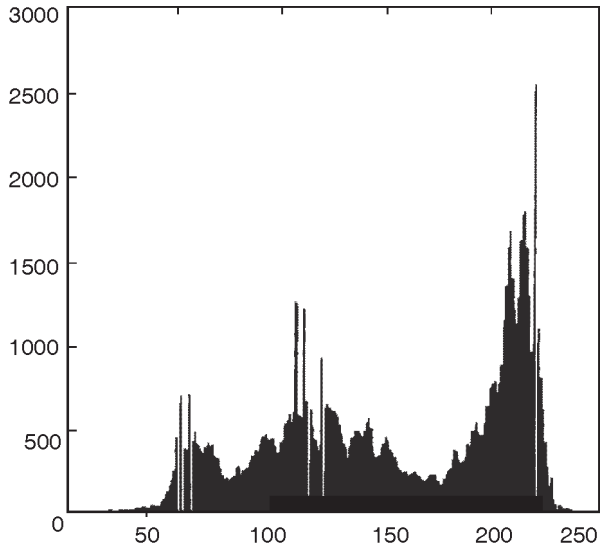


Fig. 8.1.4(b) : Histogram of image

8.1.1 Mean and Standard Deviation of Histogram

Histogram of an image provides a global description of the appearance of an image. By definition, histogram of an image represents the relative frequency of occurrence of the various grey levels in an image.

1. **Mean (μ)** : The Mean is a measure of central tendency. Mean is calculated by multiplying and adding the columns of the frequency table and diving it by the total number of elements present in the image. The mean value best reflects the typical score of a data set. Mean is also known as the Expected value.

   Consider the given frequency table.

Grey level	1	2	3	4
Number of pixels	5	12	10	5

The mean in this case will be = $\big((1 \times 5) + (2 \times 12) + (3 \times 10) + (4 \times 5)\big) / 32 = 2.46$

2. **Standard deviation (σ) :** In image processing, standard deviation is a measure that is used to measure the amount of variation of grey levels from the mean. A standard deviation close to 0 indicates that the data points tend to be very close to the mean of the image, while a high standard deviation indicates that the data points are spread out over a wider range of grey level values.

8.2 Linear Stretching

– One way to increase the dynamic range is by using a technique known as *histogram stretching*.

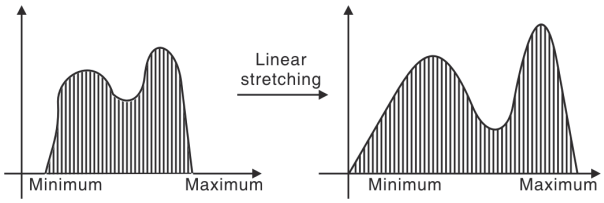


Fig. 8.2.1

– In this method, we do not alter the basic shape of the histogram, but we spread it so as to cover the entire dynamic range. We do this by using a straight line equation having slope $(s_{max} - s_{min}) / (r_{max} - r_{min})$.

Where, $s_{max} \rightarrow$ Maximum grey level of output image

$s_{min} \rightarrow$ Minimum grey level of output image

$r_{max} \rightarrow$ Maximum grey level of input image

$r_{min} \rightarrow$ Minimum grey level of input image

$$s = T(r) = \frac{s_{max} - s_{min}}{r_{max} - r_{min}} (r - r_{min}) + s_{min} \quad \ldots(8.2.1)$$

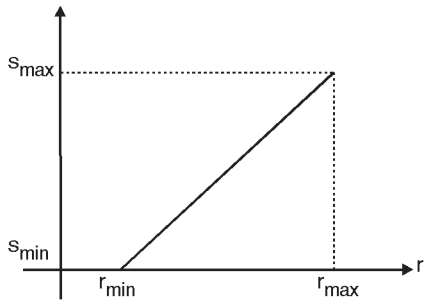


Fig. 8.2.2

– This transformation function shifts and stretches the grey level range of the input image to occupy the entire dynamic range (s_{min}, s_{max}).

8.2.1 Solved Examples on Linear Stretching

Ex. 8.2.1 : Perform histogram stretching so that the new image has a dynamic range of [0, 7].

Grey level	0	1	2	3	4	5	6	7
Number of pixels	0	0	50	60	50	20	10	0

Soln. :

Here, $r_{min} = 2$ $r_{max} = 6$

$s_{min} = 0$ $s_{max} = 7$

$$\therefore \quad s = \frac{7}{4}(r-2) + 0$$

When, $r = 2,$ $s = 0$

$r = 3,$ $s = 1.75 \approx 2$

$r = 4,$ $s = 3.5 \approx 4$

$r = 5,$ $s = 5.2 \approx 5$

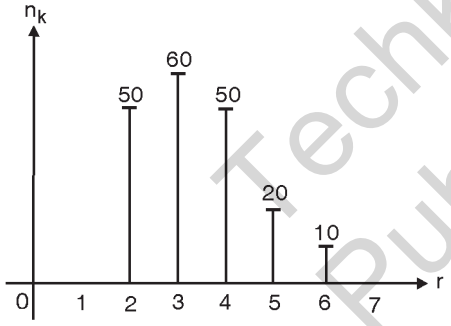
$r = 6,$ $s = 7$

∴ We have,

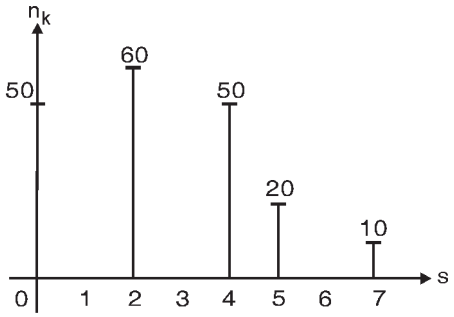
r = 2	s = 0
r = 3	s = 2
r = 4	s = 4
r = 5	s = 5
r = 6	s = 7

∴ Modified histogram is,

Grey level	0	1	2	3	4	5	6	7
Number of pixels	50	0	60	0	50	20	0	10



Original histogram



Histogram after linear stretching

Fig. P. 8.2.1

We see that the shape of the histogram is retained but it is spread out or stretched.

Let us take another example of histogram linear stretching.

Ex. 8.2.2 : Perform histogram stretching so that the new image has a dynamic range of [0, 7].

Grey level	0	1	2	3	4	5	6	7
Number of pixels	100	90	85	70	0	0	0	0

Soln. :

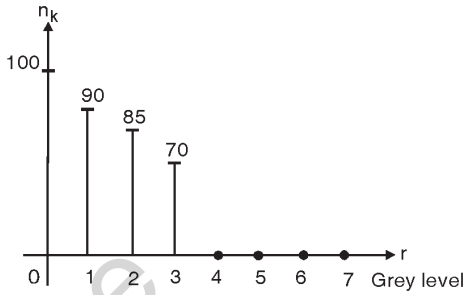


Fig. P. 8.2.2

This histogram of the image is shown. As can be concluded, it is a dark image.

Here, $r_{min} = 0,$ $r_{max} = 3,$

$s_{min} = 0,$ $s_{max} = 7$

$$s = \frac{s_{max} - s_{min}}{r_{max} - r_{min}} (r - r_{min}) + s_{min}$$

$$\therefore \quad s = \frac{7}{3}(r-0) + 0$$

When, $r = 0, s = 0$

$$r = 1, s = \frac{7}{3}(1-0) = 2.3 \approx 2$$

$$r = 2, s = \frac{7}{3}(2-0) = 4.67 \approx 5$$

$$r = 3, s = \frac{7}{3}(3-0) = 7$$

∴ We have,

r = 0	s = 0
r = 1	s = 2
r = 2	s = 5
r = 3	s = 7

Hence the histogram tends to spread out as shown in Fig. P. 8.2.2 (a).

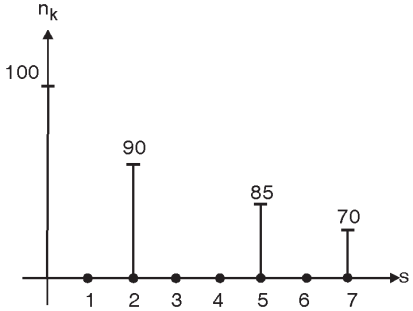


Fig. P. 8.2.2(a)

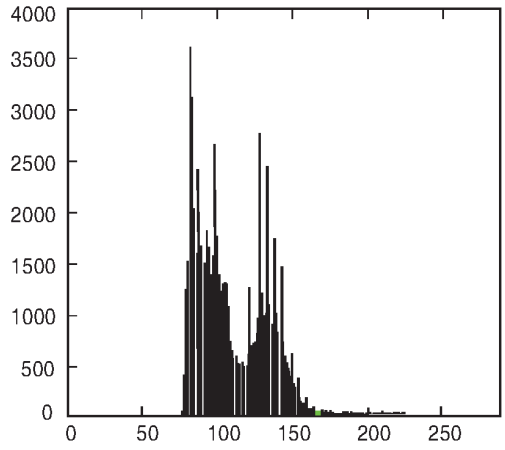
MATLAB code for histogram stretching %%

```
%% Histogram stretching %%
clear all
clc
a=imread('pout.tif');
a=double(a);
[row col]=size(a);
a=a+1;
w=min(min(a));
constant=255/(max(max(a))-min(min(a)));
cmin=0;
for x1=1:1:row
    for y1=1:1:col
        c(x1,y1)=constant*(a(x1,y1)-w)+cmin;
 end
end

%% Plotting histogram of the original image %%
h=zeros(1,300);
for n=1:1:row
for m=1:1:col
t=a(n,m);  %% Takes the value of the pixel for example 12
h(t)=h(t)+1; %% Incrementing the corresponding counter h(12)
end
end
figure(1), bar(h)
    % Plotting histogram of the modified image %
c=c+1;
h=zeros(1,400);
for n=1:1:row
  for m=1:1:col
    t=round(c(n,m));
    h(t)=h(t)+1;
  end
end
figure(2),bar(h)
figure(3),imshow(uint8(a))
figure(4),imshow(uint8(c))
```

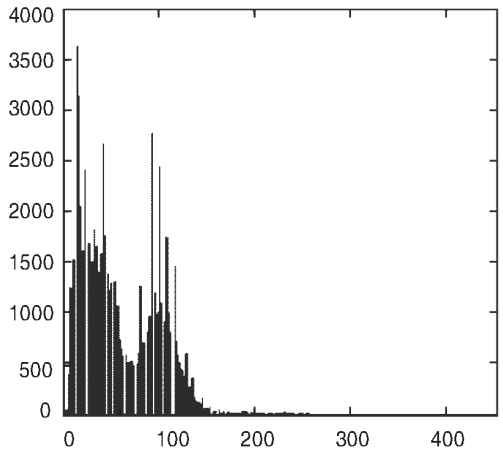
(a) Original image

(b) Image after histogram stretching



(c) Original histogram

Fig. 8.2.3 : Cont…



(d) Linear stretched histogram

Fig. 8.2.3

8.3 Histogram Equalization

- Linear stretching is a good technique but as mentioned earlier, the shape remains the same.
- There are many applications, wherein we need a flat histogram. This cannot be achieved by histogram stretching.
- We now introduce a new technique known as histogram equalization. A perfect image is one which has equal number of pixels in all its grey levels. Hence our objective is not only to spread the dynamic range, but also to have equal pixels in all the grey levels.
- This technique is known as histogram equalization. We now search for a transformation that would transform a bad histogram to a flat histogram

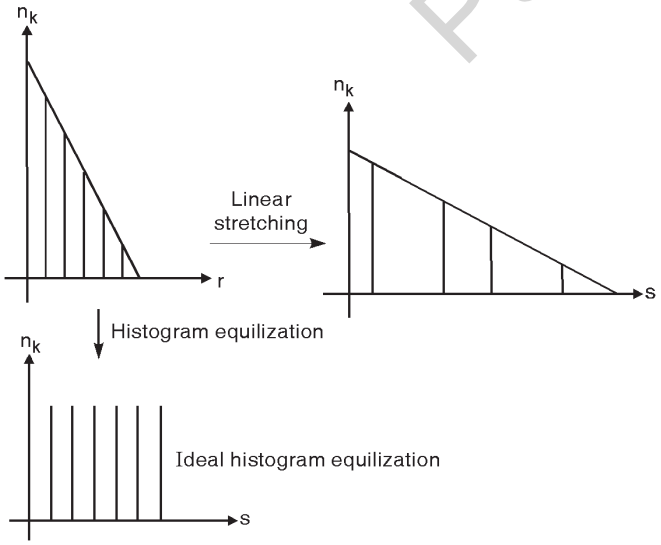


Fig. 8.3.1

We know, $s = T(r)$

What is this T which produces equal values in each grey level ?

- The transformation that we are looking for must satisfy the following two conditions :

(a) T(r) must be single valued and monotonically increasing in the interval $0 \le r \le 1$ and

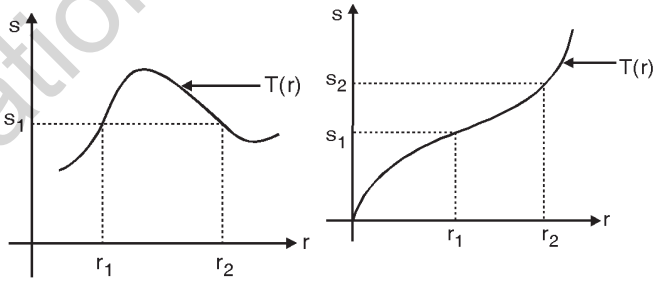
(b) $0 \le T(r) \le 1$ for $0 \le r \le 1$ i.e., $0 \le s \le 1$ for $0 \le r \le 1$

Here the range of r is taken as [0, 1]. This is called the normalized range. This range is taken for simplicity. So instead of r being in the range [0, 255] we take [0, 1]

Let us see what these two conditions :

(1) If T (r) is not single valued. Refer Fig. 8.3.2(a). Here r_1 and r_2 are mapped as a single grey level s_1 i.e., two different grey levels look the same in the modified image. This is a big drawback, hence the transformation has to be single valued. This preserves the order from black to white.

A grey level transformation that is both single valued and monotonically increasing is shown in Fig. 8.3.2(b).



(a) **(b)**

Fig. 8.3.2

(2) If condition (b) is not satisfied, then the mapping will not be consistent with the allowed range of pixel values. In other words, s will go beyond the valid range.

Since the transformation is single valued and monotonically increasing, the inverse transformation exists.

$\therefore \quad r = T^{-1}(s) \; ; 0 \le s \le 1$

- The grey levels for continuous variables can be characterized by their probability density functions $p_r(r)$ and $p_s(s)$.
- From probability theory we know that if $p_r(r)$ and T(r) are known and if $T^{-1}(s)$ satisfies condition (a), then the probability density of the transformed grey level is

$$p_s(s) = \left[p_r(r) \frac{dr}{ds} \right]_{r = T^{-1}(s)} \quad \ldots(8.3.1)$$

i.e., the probability density of the transformed image is equal to the probability density of the original image multiplied by the inverse slope of the transformation.

– We now need to find a transformation which would give us a flat histogram. Let us consider the Cumulative Density Function (CDF). Cumulative density function is obtained by simply adding up all the Probability Density Functions (PDF).

$$s = T(r) = \int_0^r p_r(r)\,dr \; ; \quad 0 \le r \le 1 \quad \ldots(8.3.2)$$

Differentiating with respect to r, we get

$$\frac{ds}{dr} = p_r(r) \quad \ldots(8.3.3)$$

Using Equation (8.3.3) in Equation (8.3.1)

$$p_s(s) = [\,1\,] = 1; \quad 0 \le s \le 1$$

i.e., $p_s(s) = 1$

This is nothing but a uniform density function !!

Let us see what we have got here.

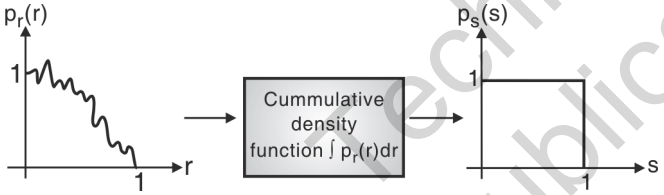


Fig. 8.3.3

A bad histogram becomes a flat histogram when we find the cumulative density function!! Let us cement this concept by taking an examples.

8.3.1 Solved Examples on Histogram Equalization

Ex. 8.3.1 For the given histogram. Perform histogram equalization.

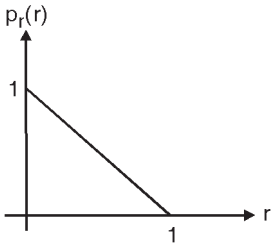


Fig. P. 8.3.1

Note : $p_r(r)$ is the PDF and hence cannot be greater than 1.

Soln. :

From the diagram, we know that the above histogram represents a dark image.

$$p_r(r) = \begin{cases} -r + 1 & ; 0 \le r \le 1 \\ 0 & ; \text{otherwise} \end{cases}$$

We shall use the CDF that we just discovered.

$$s = T(r) = \int_0^r p_r(r)dr = \int_0^r (-r + 1)\,dr$$

$$s = \frac{-r^2}{2} + r$$

$$2s = -r^2 + 2r$$

$$2s - 1 = -r^2 + 2r - 1$$

$$(1 - 2s) = (r^2 - 2r + 1)$$

$$(1 - 2s) = (r - 1)^2$$

$$(r - 1) = \pm\sqrt{1 - 2s}$$

$$r = 1 \pm \sqrt{1 - 2s} \qquad \because\ 0 \le r \le 1$$

$$\therefore\ r = 1 - \sqrt{1 - 2s}$$

We know,

$$p_s(s) = \left[p_r(r) \frac{dr}{ds} \right] = \left[(-r + 1) \frac{dr}{ds} \right]$$

Substituting the value of r,

$$p_s(s) = (-1 + \sqrt{1 - 2s} + 1) \frac{d}{ds} \left\{ 1 - (1 - 2s)^{1/2} \right\}$$

$$p_s(s) = \sqrt{1 - 2s} \cdot \frac{d}{ds} \left\{ -(1 - 2s)^{1/2} \right\}$$

$$\left(\because \frac{d}{ds}(1) = 0 \right)$$

$$= \sqrt{1 - 2s} \cdot \left\{ \frac{-1}{2} (1 - 2s)^{-1/2} (-2) \right\}$$

$$= \sqrt{1 - 2s} \cdot \frac{1}{\sqrt{1 - 2s}}$$

$$\therefore\ p_s(s) = 1;$$

Hence $p_s(s) = 1; \quad 0 \le r \le 1.$

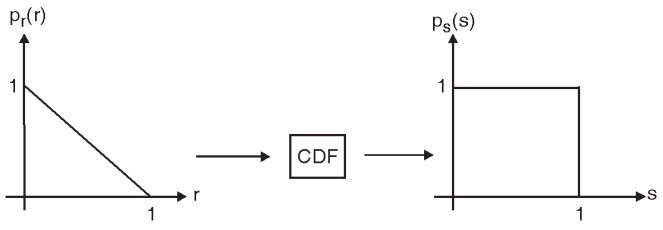


Fig. P. 8.3.1(a)

We see that $p_r(r)$ which represented a dark histogram has been transformed to an equalized histogram.

Ex. 8.3.2 : Though $p_r(r)$ cannot be greater than 1, you could just consider it as another problem and solve it

$$p_r(s) = \begin{cases} -2r + 2 & ; 0 \le r \le 1 \\ 0 & ; \text{otherwise} \end{cases}$$

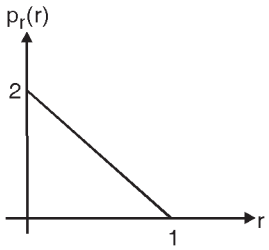


Fig. P. 8.3.2

Soln. :

Try out this problem yourself and see what you get.

We see that CDF gives us a flat response no matter what the shape of the original histogram is.

So far, we have developed the technique of histogram equalization using continuous domain mathematics. For image processing, we need to work in the discrete domain.

Hence if n is the total number of pixels in an image, then the PDF is

$$p_r(r_k) = \frac{n_k}{n}$$

We know, $$s = T(r) = \int_0^r p_r(r)$$

In the discrete domain,

$$s_k = T(r_k) = \sum_0^r p_r(r). \text{ This is the C. D. F}$$

Let us take up an example to see how this works in the discrete domain. Some additional steps need to be taken into account while working in the discrete domain.

Ex. 8.3.3 : Equalize the given histogram

Grey level	0	1	2	3	4	5	6	7
Number of pixels	790	1023	850	656	329	245	122	81

Soln. :

L = 8 (Number of grey levels)

We first plot the original histogram

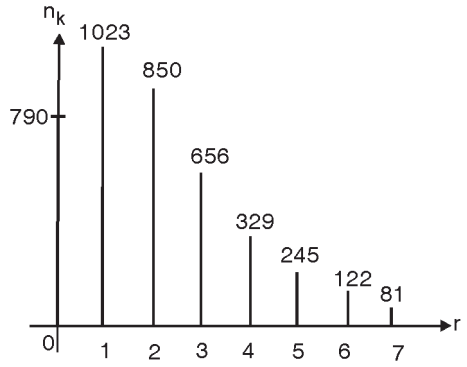


Fig. P. 8.3.3 : Original dark histogram

Grey level	n_k	PDF $p_r(r_k) = \frac{n_k}{n}$	CDF $s_k = \sum p_r(r_k)$	$(L-1) \times s_k$ i.e. $7 \times s_k$	Rounding off
0	790	0.19	0.19	1.33	1
1	1023	0.25	0.44	3.08	3
2	850	0.21	0.65	4.55	5
3	656	0.16	0.81	5.67	6
4	329	0.08	0.89	6.23	6
5	245	0.06	0.95	6.65	7
6	122	0.03	0.98	6.86	7
7	81	0.02	1	7	7
	N = 4096				

We take 1st, 2nd and the last column

Old grey level	Equalized grey level	New grey level
0	790	$\rightarrow 1$
1	1023	$\rightarrow 3$
2	850	$\rightarrow 5$
3	656	$\rightarrow 6$
4	329	$\rightarrow 6$
5	245	$\rightarrow 7$
6	122	$\rightarrow 7$
7	81	$\rightarrow 7$

We notice that the new grey levels have pixels only at 1, 3, 5, 6, 7. There are no pixels in grey levels 0, 2 and 4.

Equalized grey level	Number of pixels
0	0
1	790
2	0
3	1023
4	0
5	850
6	656 + 329 = 985
7	245 + 122 + 81 = 448

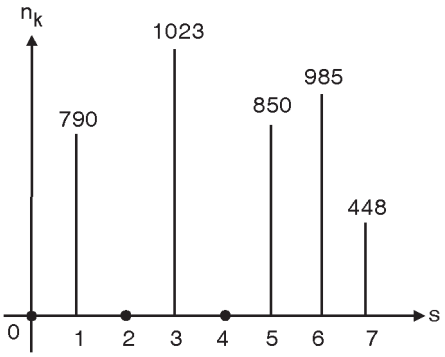


Fig. P. 8.3.3(a) : Equalized histogram

So here we are. A dark histogram becomes an evenly spaced histogram. But wait a minute, we had started out to get a flat histogram as in the continuous case but the histogram obtained is not flat; why?

Discrete domain is an approximation of the continuous domain i.e., values between 0 and 1, 1 and 2 and so on are not known. Due to this reason, perfectly flat results are never obtained.

Ex. 8.3.4

Grey level	n_k
0	100
1	90
2	50
3	20
4	0
5	0
6	0
7	0

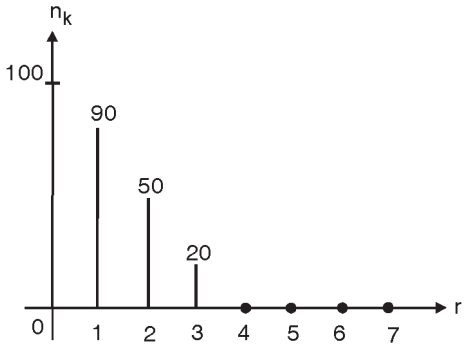


Fig. P. 8.3.4

Equalize the above histogram (Here L = 8)

Soln. :

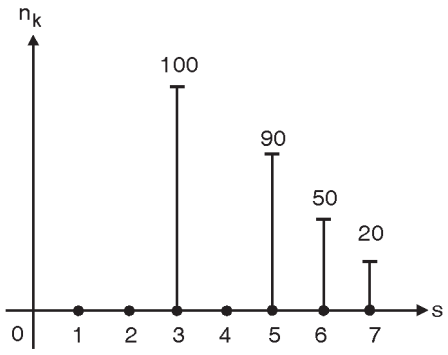
Grey level	n_k	$p_r(r_k) = \frac{n_k}{n}$	$s_k = \sum p_r(r_k)$	$s_k \times (L-1)$ i.e. $s_k \times 7$	Rounding off
0	100	0.384	0.384	2.688	3
1	90	0.346	0.73	5.11	5
2	50	0.1923	0.9223	6.456	6
3	20	0.0769	1	7	7
4	0	0	1	7	7
5	0	0	1	7	7
6	0	0	1	7	7
7	0	0	1	7	7
	n = 260				

We take the 1st, 2nd and the last column.

Old grey levels	n_k	New grey levels	New n_k	Modified grey levels
0	100 →	3 →	100	0 → 0
1	90 →	5 →	90	1 → 0
2	50 →	6 →	50	2 → 0
3	20	7		3 → 100
4	0	7		4 → 0
5	0	7 (20 + 0 + 0 + 0 + 0 = 20)	20	5 → 90
6	0	7		6 → 50
7	0	7		7 → 20

The original histogram showed that the image had only 0-3 grey levels. After histogram equalization we see that the dynamic range has increased. Hence histogram equalization does increase the dynamic range.

Next question that is usually asked is what happens when we equalize an already equalized histogram. How many times can we equalize an image ?



Equalized histogram

Fig. P. 8.3.4(a)

i.e. Dark historgram $\xrightarrow{\text{Equalize}}$ Flat historgram $\xrightarrow{\text{Equalize again ?}}$

We shall see what happens by considering an example.

Ex. 8.3.5 : Given a histogram, see what happens when we equalize it twice.

Grey level	0	1	2	3
n_k	70	20	7	3

Soln. :

Grey level	n_k	$p_r(r_k) = \frac{n_k}{n}$	$s_k = \sum p_r(r_k)$	$L = 4$ $s_k \times (L-1)$	Rounding off	New n_k
0	70	0.7	0.7	2.1	2 →	70
1	20	0.2	0.9	2.7	3	
2	7	0.07	0.97	2.91	3 →	20 + 7 + 3
3	3	0.03	1	3	3	= 30
L = 4		100				

∴ The modified grey levels are

Grey level	0	1	2	3
n_k	0	0	70	30

We now equalize this again.

Grey level	n_k	$p_r(r_k)$	s_k	$L = 4$ $s_k \times (L-1)$	Rounded off	Modified grey level
0	0	0	0	0	0	
1	0	0	0	0	0	
2	70	0.7	0.7	2.1	2 →	70
3	30	0.3	1	3	3 →	30
	100					

∴ we get,

Grey level	0	1	2	3
n_k	0	0	70	30

Which is the same as what we had got after the first equalization.

Hence, equalizing again gives us the same result i.e., equalizing an already equalized histogram causes no change in the histogram. Go ahead, try out your own examples. They will all give similar results.

Histogram equalization is guaranteed to yield exact results only in the continuous case. As the number of levels decrease, the error between the specified and resulting histogram increase and hence we do not get a flat histogram.

MATLAB code for histogram equalization

```
%% Histogram equalization %%
clear all
clc
a=imread('kids2.tif');
% You can try the program with 'pout.tif'%
a=double(a);
big=max((max(a)));
%% Gives maximum value in the image
[row col]=size(a);
C=row*col;   %% Gives the total number of pixels required
h=zeros(1,300);
%% Defining two arrays (h and z) to store the histogram
values
z=zeros(1,300);
for n=1:1:row
        for m=1:1:col
            if a(n,m)==0
          %% To ensure that the values of 'a' are not zero
    a(n,m)=1;
        end
 end
end
%% Plotting the histogram of the original image %%
for n=1:1:row
        for m=1:1:col
            t=a(n,m);
%% Takes the value of the pixel example 12 %%
         h(t)=h(t)+1;
%% Incrementing the corresponding counter h(12)%%
        end
end
pdf=h/C;              %% Probability function %%
cdf(1)=pdf(1);
%% So that we don't have the condition x(0) in the
                        for loop below %
for x=2:1:big
cdf(x)=pdf(x)+cdf(x-1);   %% Calculating the cdf %%
```

```
end
new=round(cdf*big);        %% New is the new gray level
new=new+1;      %% if new = 0, it is not to be taken in the loop
for p=1:1:row
for q=1:1:col
            temp=a(p,q);
             b(p,q)=new(temp);
%% b has the equalized image %%
%% Plotting the equalised histogram %%
    t=b(p,q);
%% Takes the value of the pixel example 12 %%
            z(t)=z(t)+1; %% Incrementing the corresponding counter z(12)%%
    end
end
b=b-1;
%% Since we had initially incremented the value of n %%
%% Plotting %%
figure (1), imshow(uint8(a))
figure (2), bar(h)
figure (3), imshow(uint8(b))
figure (4), bar(z)
```

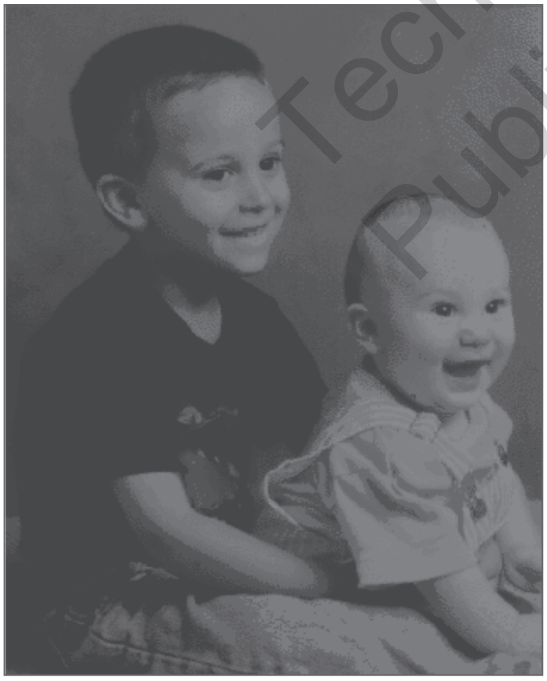
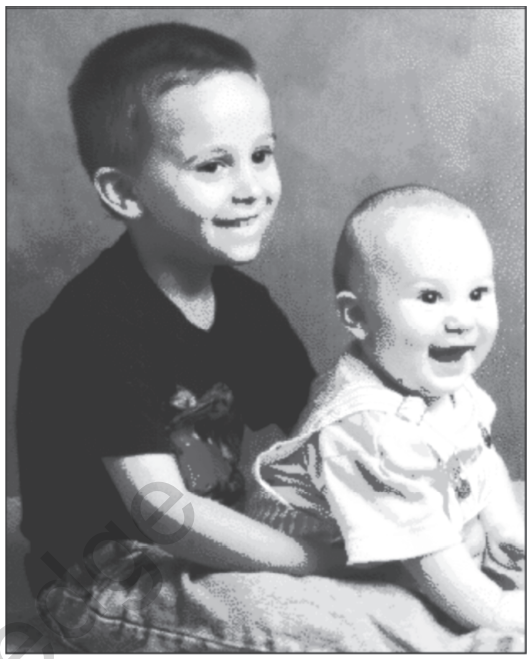
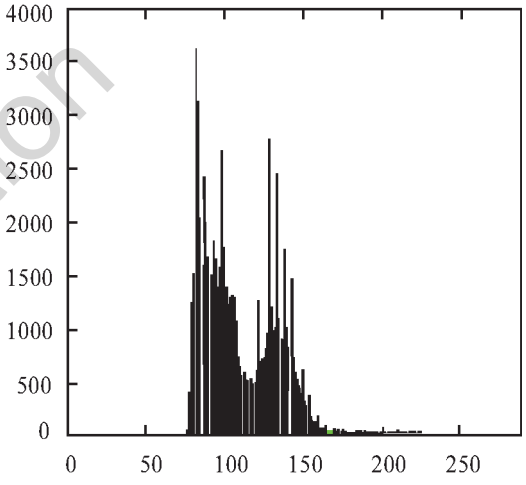


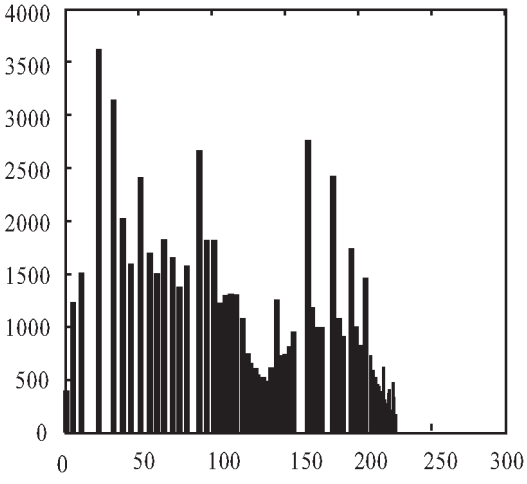
Fig. P. 8.3.5(a) : Original image



(b) Image after histogram equalization



(c) Original histogram



(d) Histogram equalized

Fig. P. 8.3.5

8.4 Histogram Specification

- From the earlier method (Histogram equalization) we note that histogram equalization is automatic. It is not interactive. i.e., it always gives one result - an approximation to an uniform histogram. It is at times desirable to have an interactive method in which certain grey level are highlighted.
- It should be noted that if we modify the grey level of an image that has a uniform PDF, $p_s(s)$, using the inverse transformation $r = T^{-1}(s)$, we get back the original histogram $p_r(r)$. Exploiting this knowledge, we can obtain any shape of the grey level histogram by processing the given image in the following way.
- Suppose $p_r(r)$ and $p_s(s)$ represent grey level PDFs of input and output images and r and s are their respective grey levels. Suppose k represents grey level of some intermediate image result i.e.,

$$k = T_1(r) = \int_0^r p_r(r)\,dr \quad \text{and} \quad k = T_2(r) = \int_0^s p_s(s)\,ds$$

- In both these cases $p_k(k)$ is uniform. Hence the transformation $s = T_2^{-1}(T_1(r))$ achieves the desired result. The procedure of histogram specification can be computed as follows :
  (1) Equalize the levels of the original image.
  (2) Specify the desired density function and obtain the transformation function.
  (3) Apply the inverse transformation function.

8.4.1 Solved Example Histogram

Ex. 8.4.1 : Given histograms (a) and (b), modify histogram a as given by histogram (b).

Histogram (a)

Grey	0	1	2	3	4	5	6	7
Number of pixels	790	1023	850	656	329	245	122	81

Histogram (b)

Grey	0	1	2	3	4	5	6	7
Number of pixels	0	0	0	614	819	1230	819	614

Soln. :

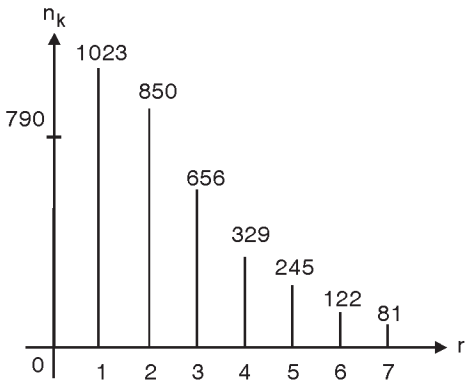


Fig. P. 8.4.1(a)

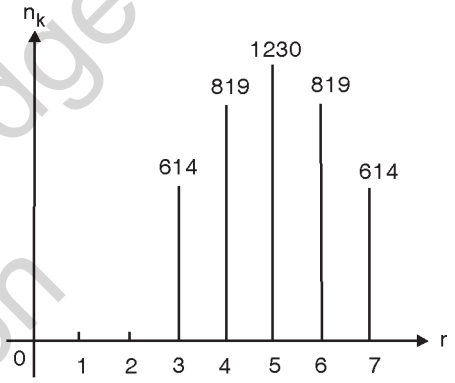


Fig. P. 8.4.1(b)

Now equalize (a) as well as (b)

Histogram (a) : L = 8

Grey level	n_k	$p_r(r_k) = \frac{n_k}{n}$	CDF = $\sum p_r(r_k)$	CDF × (L – 1)	Rounding off		New grey level
0	790	0.19	0.19	1.33	1	→	790
1	1023	0.25	0.44	3.08	3	→	1023
2	850	0.21	0.65	4.55	5	→	850
3	656	0.16	0.81	5.67	6	→	656 +
4	329	0.08	0.89	6.23	6		329 = 985
5	245	0.06	0.95	6.25	7	→	
6	122	0.03	0.98	6.86	7	→	245 + 122 + 81
7	81	0.02	1	7	7	→	= 448
	4096						

Note that there are 5 distinct levels after histogram equilization of histogram (a)

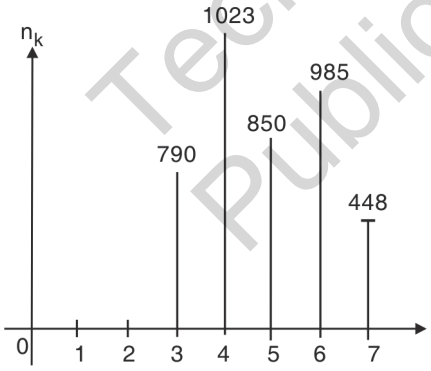
Grey level	0	1	2	3	4	5	6	7
Number of pixels	0	790	0	1023	0	850	985	448

Now equalize (b) : L = 8

Grey level	n_k	$p_r(r_k) = \frac{n_k}{n}$	CDF	CDF × (L – 1)	New grey level
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	614	0.149	0.149	1.05	1
4	819	0.20	0.35	2.45	3
5	1230	0.30	0.65	4.55	5
6	819	0.20	0.85	5.97	6
7	614	0.15	1	7	7
	4096				

To obtain histogram specification, we apply the inverse transform comparing equalized (a) and (b).

1 → 3
3 → 4
5 → 5
6 → 6
7 → 7



Resultant histogram

Fig. P. 8.4.1(c)

Hence the final result is as shown

Grey	0	1	2	3	4	5	6	7
Number of pixels	0	0	0	790	1023	850	985	448

Hence we see that histogram (a) has been equalized and mapped in such a way that it resembles the histogram (b). This is histogram specification.

8.5 Solved Examples on Histogram Modelling

Ex. 8.5.1 : Two images can have the same histogram (Justify/ Contradict with reason). **MU - Dec 2017, 5 Marks**

Soln. :

Histograms give us information about the numbers of grey levels present in the image, it does not give us any spatial information. This means, histograms do not tell us where the grey levels lie in an image. If all pixels in an image are shuffled, there will be no change in the histogram. For example, the given two images are completely different but their histograms will be equal as the total number of black pixels and white pixels are the same in both the images. Hence two images can have the same histogram.

Fig. P. 8.5.1

Ex. 8.5.2 : What effect would setting to zero the lower order bit planes have on the histogram of an image in general ?

Soln. : The number of pixels having different grey level values would decrease, thus causing the number of components in the histogram to decrease. Since the number of pixels is constant, this would cause the height of the remaining histogram peaks to increase. Typically, less variability in grey level values will reduce the contrast.

Let us check this out with an example.

Given below is a 4 × 4 image having grey levels 0-18.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Histogram of the original image is as shown in Fig. P. 8.5.2.

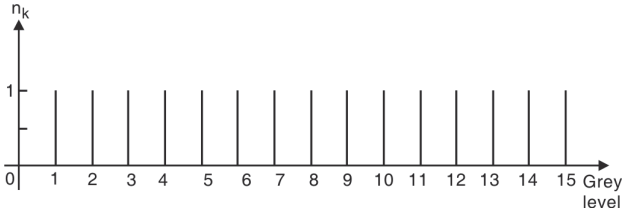


Fig. P. 8.5.2

We convert the original image to a binary image. Since the maximum grey level is 15, we need 4-bits for its representation.

0000	0001	0010	0011
0100	0101	0110	0111
1000	1001	1010	1011
1100	1101	1110	1111

Suppose we make the last two bits equal to zero, we get

0000	0000	0000	0000
0100	0100	0100	0100
1000	1000	1000	1000
1100	1100	1100	1100

Converting back to the grey level format we get,

0	0	0	0
4	4	4	4
8	8	8	8
12	12	12	12

∴ its histogram is as shown in Fig. P. 8.5.2(a).

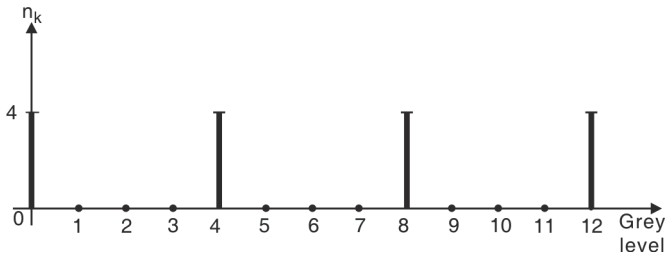


Fig. P. 8.5.2(a)

Comparing this histogram with the original histogram, we see that the variability is reduced, i.e. number of grey levels are reduced and the height of these grey levels has increased.

Ex. 8.5.3 : What would be the effect on the histogram if we set to zero, the higher order bit planes.

Soln. :

Proceeding in a similar manner as in Ex. 8.5.2 and working with the same image, we get

Converting to binary

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

(a)

0000	0001	0010	0011
0100	0101	0110	0111
1000	1001	1010	1011
1100	1101	1110	1111

(b)

Converting the upper two bits to zero

0000	0000	0000	0000
0100	0100	0100	0100
1000	1000	1000	1000
1100	1100	1100	1100

(c)

Converting back to grey levels we get

0	1	2	3
0	1	2	3
0	1	2	3
0	1	2	3

(d)

Fig. P.8.5.3

Plotting the histogram as shown in Fig. P. 8.5.3(e).

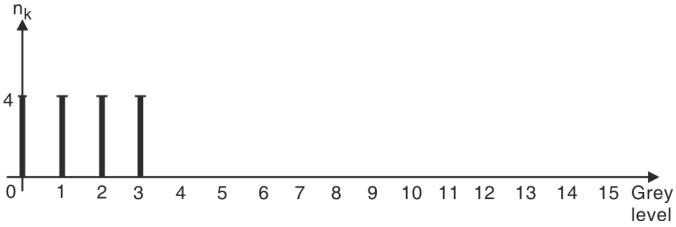


Fig. P. 8.5.3 (e)

In this case too, the grey levels reduce. But one important thing that happens is that the image becomes much darker.

Ex. 8.5.4 : A digital image with 8 quantization levels is given below :

Perform histogram equalization.

f(x,y) = |x – y| for x = 0 to 7 and y = 0 to 7 **MU - Dec. 2018, 10 Marks**

Soln. :

In this we form an image by taking the modulus of the difference of the spatial coordinates. The image that is formed using the equation f(x,y) = |x – y| is shown below :

f(x, y) =

	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	0	1	2	3	4	5	6
2	2	1	0	1	2	3	4	5
3	3	2	1	0	1	2	3	4
4	4	3	2	1	0	1	2	3
5	5	4	3	2	1	0	1	2
6	6	5	4	3	2	1	0	1
7	7	6	5	4	3	2	1	0

We generate the frequency table / histogram of the original image.

Grey levels	0	1	2	3	4	5	6	7
Number of pitch	8	14	12	10	8	6	4	2

The original histogram is shown in Fig. P.8.5.4.

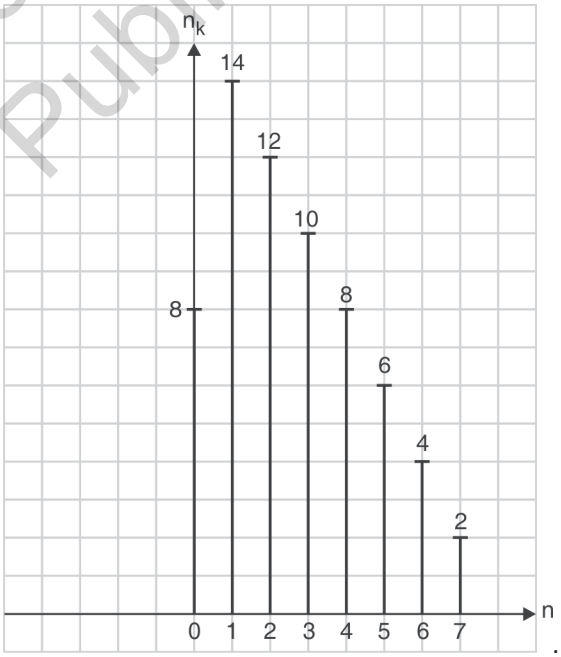


Fig. P. 8.5.4

We now, generate a table to perform histogram equalization

Grey level	n_k	$p_r(r_k) = \frac{n_k}{n}$	$S_k = \Sigma p_r(r_k)$	$S_k \times (L-1)$ $S_k \times 7$	Rounding off
0	8	0.125	0.125	0.875	1
1	14	0.218	0.343	2.401	2
2	12	0.187	0.53	3.71	4
3	10	0.156	0.686	4.802	5
4	8	0.125	0.811	5.677	6
5	6	0.093	0.904	6.328	6
6	4	0.062	0.966	6.762	7
7	2	0.031	1	7	7
	$\sum n_k = 64$				

We consider the 1st, 2nd and last column

Greg level	n_k	Rounding off	New n_k
0	8	1→	8
1	14	2→	14
2	12	4→	12
3	10	5→	10
4	8	6 } 8 + 6→	14
5	6	6 }	
6	4	7 } 4 + 2 →	6
7	2	7 }	

Hence the modified frequency table is

Grey levels	0	1	2	3	4	5	6	7
n_k	0	8	14	0	12	10	14	6

The equalized histogram is shown in Fig. P. 8.5.4(a).

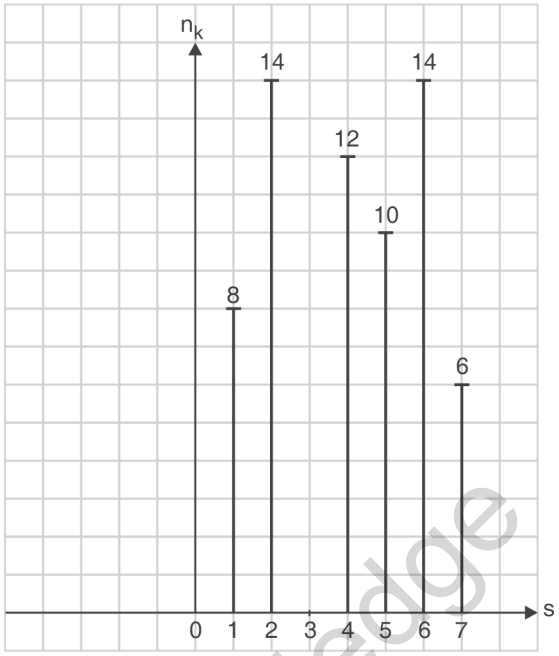


Fig. P. 8.5.4(a)

Ex. 8.5.5 : Given below is the slope transformation and the image. Draw the frequency tables for original and new image. Assume $l_{min} = 0$ and $l_{max} = 7$.

2	3	4	2
5	5	2	4
3	6	3	5
5	3	5	5

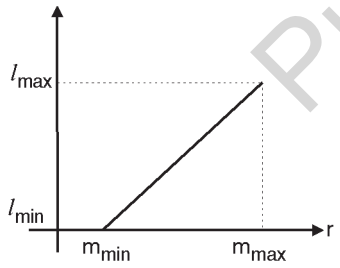


Fig. P. 8.5.5

Soln. : This is a sum of histogram linear stretching. By frequency tables, we mean histogram. From the image, it is clear that $m_{min} = 2$, $m_{max} = 6$, $l_{min} = 0$ and $l_{max} = 7$.

Using the formula

$$l = \frac{l_{max} - l_{min}}{m_{max} - m_{min}} (m - m_{min}) + l_{min}$$

$$\therefore \quad l = \frac{7 - 0}{6 - 2} (m - 2) + 0$$

$$\therefore \quad l = \frac{7}{4} (m - 2)$$

From the above formulation

when, $m = 2, \quad l = 0$

$m = 3, \quad l = 1.75 \approx 2$

$m = 4, \quad l = 3.5 \approx 4$

$m = 5, \quad l = 5.25 \approx 5$

$m = 6, \quad l = 7$

$\therefore$ The modified image is

0	2	4	0
5	5	0	4
2	7	2	5
5	2	5	5

Histogram of the original and modified images are shown in Fig. P. 8.5.5(a).

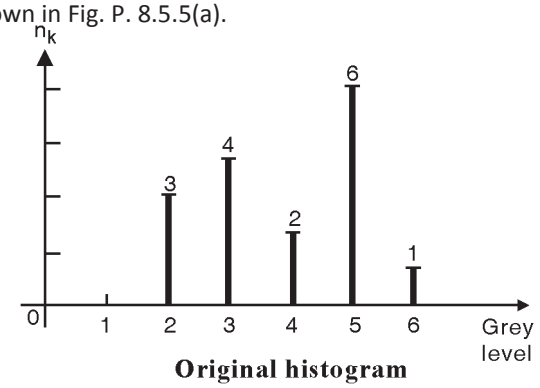


Fig. P. 8.5.5(a)

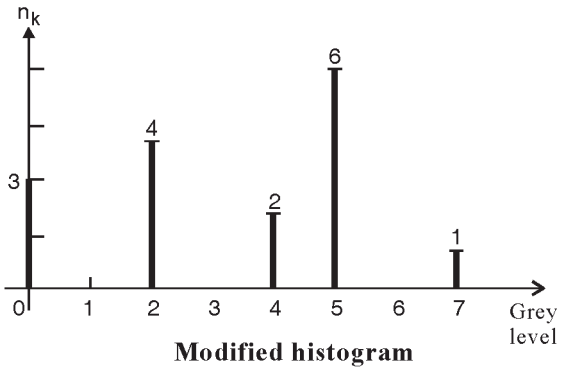


Fig. P. 8.5.5(b)

Ex. 8.5.6 : A popular procedure for image enhancement combines high frequency emphasis and histogram equalization to achieve edge sharpening and contrast enhancement. Does it matter which of the two processes are applied first ?

Soln. :

High frequency emphasis is the same as a high boost filter. Let us see what we get when we apply a high frequency emphasis filter.

$$f(x,y) \longrightarrow \boxed{\begin{matrix} h(x,y) \\ \text{High boost} \end{matrix}} \longrightarrow f(x,y) * h(x,y) = g(x,y)$$

i.e., the output g(x, y) is a convolution of the input image and the impulse response of the high boost filter.

If we now perform histogram equalization we get

$$s = T[g(x, y)]$$

$$s = T[f(x, y) * h(x, y)] \quad \ldots(1)$$

If we decide to apply histogram equalization prior to high boost filter, we get

$$T[f(x,y)] \longrightarrow \boxed{\begin{matrix} h(x,y) \\ \text{High boost} \end{matrix}} \longrightarrow T[f(x,y)] * h(x,y)$$

$$\ldots(2)$$

We see that Equation (1) ≠ Equation (2).

∴ The order of the operation does matter i.e., the final result varies if we interchange the two operations.

Ex. 8.5.7 : The grey level distribution of an image is shown in the table below. Perform histogram equalization and plot the original and equalized histograms.

Grey level	0	1	2	3	4	5	6	7
Frequency of occurrence	0	50	100	200	400	200	50	0

MU - Dec. 2013, 10 Marks

Soln. :

We draw the original histogram of the image.

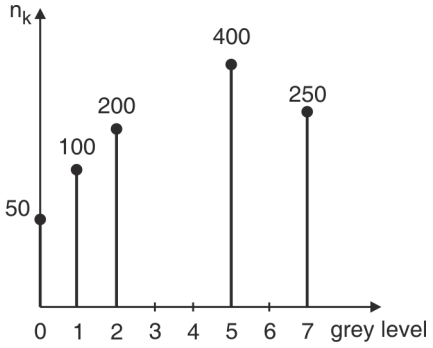


Fig. P. 8.5.7

Here L = 8 We now perform histogram equalization.

Grey level	n_k	PDF $p_r(k) = \frac{n_k}{\sum n}$	CDF i.e., $s_k = \sum p_r(k)$	$s_k \times (L-1)$ i.e., $s_k \times 7$	Round off
0	0	0	0	0	0
1	50	0.05	0.05	0.35	0
2	100	0.1	0.15	1.05	1
3	200	0.2	0.35	2.45	2
4	400	0.4	0.75	5.25	5
5	200	0.2	0.95	6.65	7
6	50	0.05	1	7	7
7	0	0	1	7	7
	$\sum n = 1000$				

Consider the 1st, 2nd and last column

Grey level	n_k	New grey level	
0	0	0	0 + 50
1	50	0	= 50
2	100	1	100
3	200	2	200
4	400	5	400
5	200	7	200 + 50 = 250
6	50	7	
7	0	7	

Hence the equalized frequency table is,

Grey level	0	1	2	3	4	5	6	7
n_k	50	100	200	0	0	400	0	250

The equalized histogram is shown in Fig. P. 8.5.7(a)

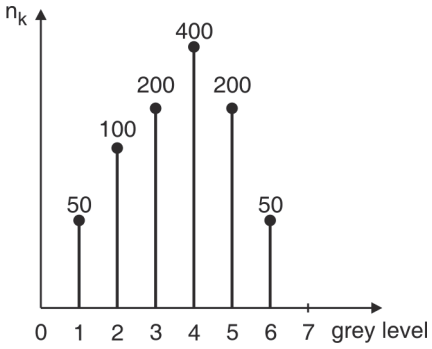


Fig. P. 8.5.7(a)

Ex. 8.5.8 : A 64 × 64 image, represented by 3 bit / pixel has the following grey level distribution :

Gray level	0	1	2	3	4	5	6	7
No. of Pixel	128	75	280	416	635	1058	820	684

Perform histogram equalization and give new distribution of gray level. Show plots of original and equalized image.

MU - May 2011, Dec. 2012, 10 Marks

Soln. :

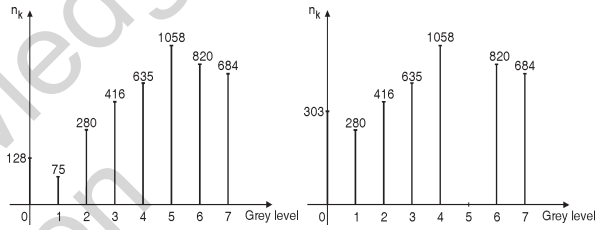
We generate the histogram table

Grey level	n_k	PDF $p_r(r_k) = \frac{n_k}{n}$	CDF $s_k = \Sigma p_r(r_k)$	$(L-1) \times s_k$	Rounding off
0	128	0.031	0.031	0.217	0
1	75	0.018	0.0496	0.347	0
2	280	0.068	0.1179	0.825	1
3	416	0.101	0.2195	1.536	2
4	635	0.155	0.3745	2.622	3
5	1058	0.258	0.6328	4.430	4
6	820	0.201	0.8330	5.831	6
7	684	0.1670	1	7.00	7
	n = 4096				

We take the 1st, 2nd and the last column

Old grey level	n_k	New grey level	Modified grey level
0	128	0 → 128 +	0 → 303
1	75	0 → 175	1 → 280
2	280	1 → 280	2 → 416
3	416	2 → 416	3 → 635
4	635	3 → 635	4 → 1058
5	1058	4 → 1058	5 → 0
6	820	6 → 820	6 → 820
7	684	7 → 684	7 → 684

We now draw the original as well as the equalized histogram.



(a) Original histogram (b) Equalized histogram

Fig. P. 8.5.8

Summary

In this chapter, the concept of histograms is introduced. Though histogram processing is a spatial domain technique, it is presented here as a separate chapter because of its wide spread applications. Histograms give us an idea about the quality of an image. One of the most common defects found in recorded images is poor contrast. This degradation may be caused by inadequate lighting, aperture size and/or shutter speed. Procedures of modifying the histogram to get a high contrast image are discussed. Emphasis has been laid on the two important techniques: linear stretching and histogram equalization.

Review Questions

Q. 1 Plot the graph of a cumulative density function (CDF).

Q. 2 Explain how histogram can aid in the process of thresholding and contrast stretching.

Q. 3 Equalize the given histogram.

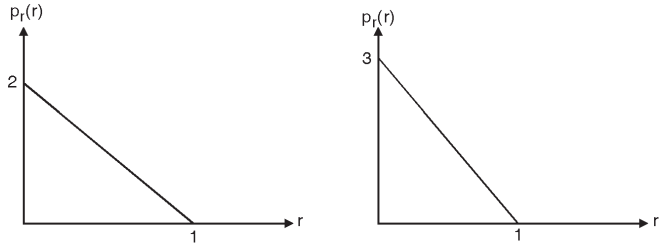


Fig. Q. 3

Note : $p_r(r)$ cannot be greater than 1 but solve this as any other mathematical problem.

Q. 4 Explain why a discrete histogram technique does not, in general, yield a flat histogram.

Q. 5 Suppose that a image is subjected to a histogram equalization. Show that a second pass of histogram equalization will produce exactly the same result as the first pass.

Q. 6 What are the two conditions that a transformation has to satisfy ?

Q. 7 Write down an algorithm to plot the histogram of an image.

Q. 8 Equalize the given histogram.

Grey	0	1	2	3	4	5	6	7
Number of pixels	50	0	50	0	50	0	50	0

Q. 9 Compare between contrast stretching and histogram equalization.

Q. 10 Suppose we have a dark image which needs to be compressed and also equalized. Which operation would we use first ? Will the results be the same if the order of operations is reserved ?

Q. 11 Equalize the given image.

0	1	2
3	4	5
6	7	8

Q. 12 Given the frequency table,

Grey	0	1	2	3	4	5	6	7
Number of pixels	0	0	a	b	c	d	e	0

Perform linear stretching.

❑❑❑

Segmentation

Syllabus :
Segmentation based on Discontinuities (point, Line, Edge), Image Edge detection using Robert, Sobel, Previtt masks, Image Edge detection using Laplacian Mask. Connectivity

9.1 Introduction

May 2016, Dec. 2016, Dec. 2018, 2 Marks

Q. What is image segmentation

(May 2016, Dec. 2016, Dec. 2018, 2 Marks)

- Segmentation, as the name suggests, subdivides (segments) an image into its constituent regions or objects. Unlike image enhancement, were our primary concern was to improve the subjective quality of images for display, in segmentation, we address some aspects of analyzing the content of an image.
- This simply means we endeavour to find out what is in the picture. Segmentation forms a section of computer vision i.e., we use segmentation, when we want the computer to make decisions.
- Segmentation is used when we need to automate a particular activity. The ultimate aim in an automated system is to extract important features from image data, from which description, interpretation, or understanding of the scene can be provided by the machine.
- Segmentation is not required when the images have to be shown to a human being. This is because a human visual system has an inherent quality to segment the image shown to it.

Let us take two simple examples where image segmentation is used.

(1) Automated blood cell counting :

- We all know blood is made up of RBC's, WBC's, platelets etc.
- In manual machines, the technician looks at the slide and based on various parameters classifies them as RBC's, WBC's etc. As stated earlier, the human visual system automatically segments the image as RBC's, WBC's... etc.
- In an automated blood cell counting machine, this classification has to be made by the machine (computer). The computer scans the slide and based on certain algorithms classifies them as RBC's, WBC's....etc.
- We hence needed to do image segmentation to divide the image into various parts.

(2) Finger print matching in forensic studies

- In this, there exists a data base of finger prints of people who have been convicted of crime at some point in time.
- Now if there has been a crime at a place and some finger prints have been found, images are taken, and these images are run through an algorithm to see if this particular finger print matches with the one in the data base.
- Here too, since the system is automated we need to use segmentation to divided the image into various parts.
- We hence say that image segmentation is used for computer decision making (machine vision) and not for human interpretation and the main purpose of segmentation is to partition the image.
- Image segmentation algorithms are generally based on one of the two basic properties i.e. image segmentation can be achieved in any one of the two ways viz.
  (1) Segmentation based on discontinuities in intensity.
  (2) Segmentation based on similarities in intensity.

  In the first method, the approach is to partition an image based on abrupt changes in intensity, such as edges, while in the second method, we partition an image into regions that are similar according to a set of predefined criteria.

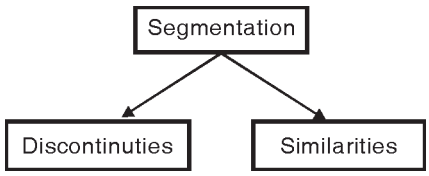


Fig. 9.1.1

– The syllabus only contains Segmentation based on discontinuities

9.2 Image Segmentation based on Discontinuities

When we look at an image, we immediately observe 3 basic types of discontinuities in the grey level viz. points, lines and edges. The easiest way is to use spatial masks which have properties to detect these discontinuities.

9.2.1 Point Detection

– Remember one thing, points, lines and edges are all high frequency components and hence we need masks which are basically high pass. Hence the masks that we present here for point, line and edge detection have the properties of a high mask mainly, the sum of the coefficients of the mask has to be equal to zero in all the three cases.

– Detecting points is fairly simple. We use the standard high pass mask for it. And so that this mask detects only points and not lines, we set a threshold value i.e., we say a point has been detected at the location on which the mask is centered only if

– 1	– 1	– 1
– 1	8	– 1
– 1	– 1	– 1

$$|R| \geq T \qquad \ldots(9.2.1)$$

Where R is derived from the standard convolution formula

$$R = w_1 z_1 + w_2 z_2 + \ldots\ldots\ldots + w_9 z_9$$

$$R = \sum_{i=1}^{9} w_i z_i$$

– We take |R| because we want to detect both the kinds of points i.e. white points on a black background as well as black points on a white background.

T is a non negative threshold which is defined by the user.

9.2.2 Line Detection

– Detection of lines can be done using the masks shown below. In an image, lines can be in any direction and detecting these lines would need different masks.

– 1	– 1	– 1
2	2	2
– 1	– 1	– 1

Horizontal

– 1	– 1	2
– 1	2	– 1
2	– 1	– 1

+ 45°

– 1	2	– 1
– 1	2	– 1
– 1	2	– 1

Vertical

2	– 1	– 1
– 1	2	– 1
– 1	– 1	2

– 45°

– We notice that all these masks have a sum equal to zero, and hence all of them are high pass masks. The first mask would respond strongly to lines that are oriented horizontally. The second mask would respond to lines at an angle of + 45°, the third mask would respond strongly to vertical lines and the last mask would respond strongly to lines at an angle of – 45°.

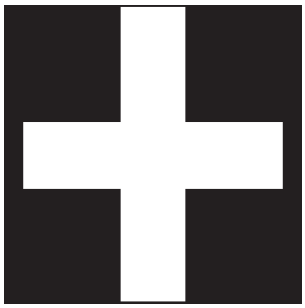


Fig. 9.2.1

Let us take a simple example of a 8×8 pseudo image.

0	0	0	10	0	0	0	0
0	0	0	10	0	0	0	0
0	0	0	10	0	0	0	0
0	0	0	10	0	0	0	0
0	10	10	10	10	10	10	0
0	0	0	10	0	0	0	0
0	0	0	10	0	0	0	0
0	0	0	10	0	0	0	0

- It is quite evident that in this image we have two lines, one horizontal and the second vertical.
- Let us see what happens when we move the first (Horizontal) mask over the image. By now you should be quite comfortable with moving masks on the entire image. We leave the borders and hence start from the encircled pixel.

  The result that we obtain is as shown

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	– 20	– 20	– 20	– 20	– 30	– 20	0
0	+ 40	+ 40	+ 40	+ 40	+ 60	+ 40	0
0	– 20	– 20	– 20	– 20	– 30	– 20	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

- Since line detection masks are high pass masks, we encounter negative values.
- These values are made zero as was discussed in chapter of Image Enhancement in Spatial Domain. Hence the final image obtained as follows,

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	40	40	40	40	60	40	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

- We see that the horizontal mask detects lines only in the horizontal direction and cleans up the other lines (vertical line in this case).
- In a similar fashion we can show that vertical, + 45°, and – 45° masks respond strongly to lines in their respective directions. It would be a good idea to try this out yourself.

9.2.3 Edge Detection

- More than isolated points and lines, it is the detection of edges that form an important part of image segmentation. An edge can be defined as a set of connected pixels that form a boundary between two disjoint regions.
- A typical example of an edge is shown in Fig. 9.2.2.

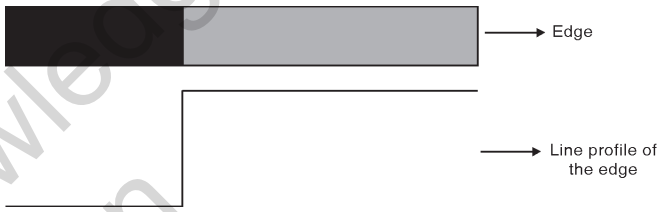


Fig. 9.2.2

- Here the set of pixels that separate the black region from the grey region is called an *edge*. The line profile of such an edge is shown along with edge.
- The image shown has a step edge. Such step edges occur only in artificially generated images such as test patterns. Images obtained from digitization of optical images of real scenes do not possess step edges. Recollect what happens when we digitize an image.
- We first sample the image using the Nyquist condition and then quantize it. Real world signals are not band-limited i.e. F(u, v) ≠ 0 outside an interval and maximum frequency present in it is infinite. Hence the Nyquist criteria is not met. We therefore pass the original signal through an anti-aliasing filter (which is a low pass filter) to remove the very high frequencies.

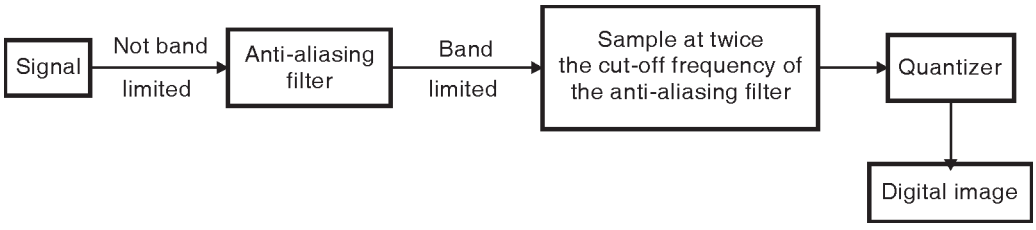


Fig. 9.2.3

- It is this anti-aliasing filter which reduces the slope of the edge. Due to this, real world images when captured through a camera will not have step edges. In practice, edges are modelled as having a ramp like profile.

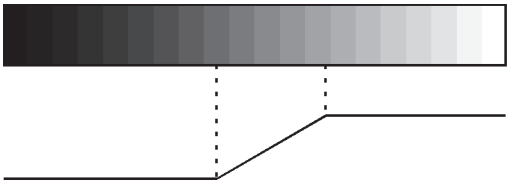


Fig. 9.2.4

Here the slope of the ramp is inversely proportional to the degree of blurring.

Given below are some of the two dimensional, discrete domain edge models.

a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b

Vertical step edge

a	a	a	b	b	b	b	b	b	b
a	a	a	a	b	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	a	b	b	b	b
a	a	a	a	a	a	a	b	b	b

Diagonal step edge

a	a	a	a	a	a	a	a	a	a
a	a	a	a	a	a	a	a	a	a
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b
a	a	a	a	a	b	b	b	b	b

Corner step edge

a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Vertical ramp edge

a	a	c	b	b	b	b	b	b	b
a	a	a	c	b	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	a	c	b	b	b	b
a	a	a	a	a	a	c	b	b	b

Diagonal ramp edge Single pixel transition

a	a	a	a	a	a	a	a	a	a
a	a	a	a	c	c	c	c	c	c
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Corner ramp edge

a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Vertical ramp edge

a	a	d	e	b	b	b	b	b	b
a	a	a	d	e	b	b	b	b	b
a	a	a	a	d	e	b	b	b	b
a	a	a	a	a	d	e	b	b	b
a	a	a	a	a	a	d	e	b	b

Diagonal ramp edge Smoothed transition

a	a	a	a	a	a	a	a	a	a
a	a	a	a	d	c	c	c	c	c
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b

Corner ramp edge

$$c = \frac{a+b}{2}, \quad d = \frac{3a+b}{4}, \quad e = \frac{a+3b}{4}, \quad b > a$$

Fig. 9.2.5

9.3 Detection of Edges

How do these edges come in an image ?

- Variation of scene features, usually brightness, give rise to edges. In other words, edges are representations of the discontinuities of the scene intensity function.
- There could be various reasons such as type of material, surface texture, lighting conditions, etc., which play an important role in forming these discontinuities.
- Our aim in this chapter is to detect these discontinuities as edges. The process of edge detection is carried out by the derivative approach.
- The basic idea of the derivative approach is the computation of the local derivative operator.

Consider the image shown in Fig. 9.3.1.

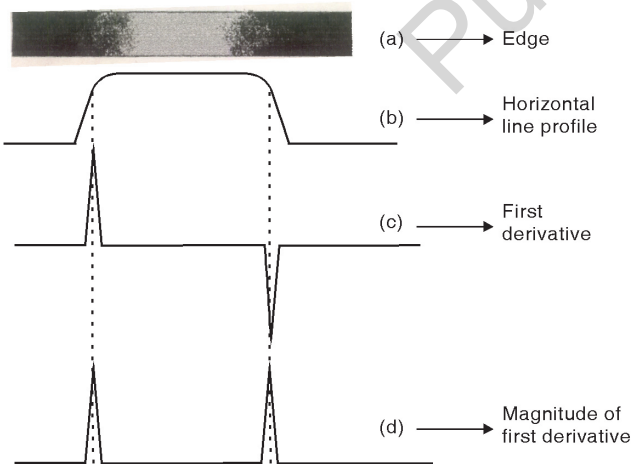


Fig. 9.3.1

- Fig. 9.3.1 shows that the first derivative of the grey line profile is positive at the leading edge of a transition, negative at the trailing edge and zero in areas of constant grey levels.
- If we now want to plot the derivative image, we need to take the magnitude of Fig. 9.3.1(c). This gives us Fig. 9.3.1(d). As can be seen, we get non-negative values only at the two edges of the image. We can thus conclude that computing the derivative helps us in detecting the edges.
- Although this discussions has been limited to the 1-D horizontal profile, a similar argument applies to an edge of any orientation in an image.
- The first derivative at any point in an image is obtained by using the magnitude of the gradient at that point.

9.3.1 Computing the Gradient

Let us spend some time understanding the basics of the gradient. This discussion will help us understand edge detection better.

Case 1 : Output is a function of one variable

(**Example :** 1-dimensional signal) y = f(x).

We know that the derivative of y w.r.t. x is

$$\frac{dy}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} = f'(x)$$

The slope is $\frac{dy}{dx}$

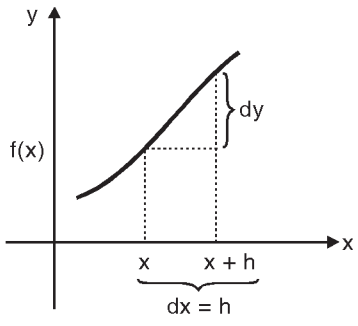


Fig. 9.3.2

Case 2 : Output is a function of two variables (2-dimensional example), f(x, y). In the two variable case, x and y are the variables. Both these variables change to say x + h and y + k. To find the gradient, we work with each separately i.e., we take partial derivatives.

Let us first keep y fixed and alter only x.

$$\frac{\partial f}{\partial x} = \lim_{h \to 0} \frac{f(x + h, y) - f(x, y)}{h}$$

Here $\frac{\partial f}{\partial x}$ is the rate of change of f w.r.t. x keeping y fixed.

Similarly, keeping x fixed and changing y, we get

$$\frac{\partial f}{\partial y} = \lim_{k \to 0} \frac{f(x, y + k) - f(x, y)}{k}$$

$\frac{\partial f}{\partial y}$ is the rate of change of f w.r.t. y keeping x constant.

Hence the final gradient is

$$\nabla F = \hat{i} \frac{\partial f}{\partial x} + \hat{j} \frac{\partial f}{\partial y} \quad ...(9.3.1)$$

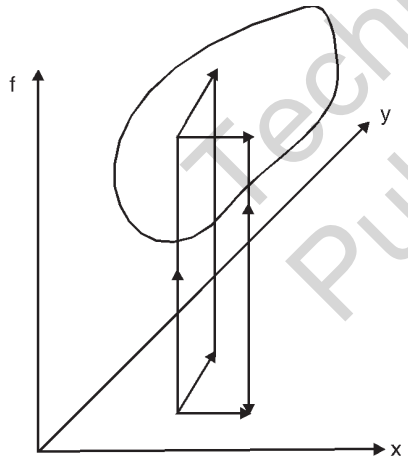


Fig. 9.3.3

Case 3 : Output is a function of three variables (3-dimensional, e.g.. Density at different points in the atmosphere or pressure at different points) (Not useful in image processing)

$$P = f(x, y, z)$$

$\therefore$ the three partial derivatives are

$$\frac{\partial f}{\partial x} = \lim_{h \to 0} \frac{f(x + h, y, z) - f(x, y, z)}{h}$$

$$\frac{\partial f}{\partial y} = \lim_{k \to 0} \frac{f(x, y + k, z) - f(x, y, z)}{k}$$

$$\frac{\partial f}{\partial z} = \lim_{l \to 0} \frac{f(x, y, z + l) - f(x, y, z)}{l}$$

$$\therefore \text{ gradient } \nabla F = \hat{i} \frac{\partial f}{\partial x} + \hat{j} \frac{\partial f}{\partial y} + \hat{k} \frac{\partial f}{\partial z} \quad ...(9.3.2)$$

Case 3 will not be used in this book

Now, magnitude of a vector is given by ,

$$A^2 = A_x^2 + A_y^2$$

$$|A| = \sqrt{A_x^2 + A_y^2}$$

Similarly, magnitude of a 2-D gradient is,

$$|\nabla F| = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2} \quad ...(9.3.3)$$

Here $\frac{\partial f}{\partial x}$ = Rate of change of 'f' w.r.t the x-direction

$\frac{\partial f}{\partial y}$ = Rate of change of 'f' w.r.t the y-direction

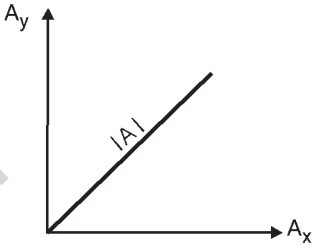
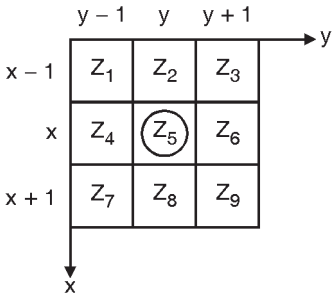


Fig. 9.3.4

9.4 Finding Gradients using Masks

Consider a 3 × 3 neighbourhood with Z_5 as the origin.



We know, $\frac{\partial f}{\partial x} = \lim_{h \to 0} \frac{f(x + h, y) - f(x, y)}{h}$

$$\frac{\partial f}{\partial y} = \lim_{k \to 0} \frac{f(x, y + k) - f(x, y)}{k}$$

In the discrete domain, h = k = 1

$$\therefore \frac{\partial f}{\partial x} = f(x + 1, y) - f(x, y)$$

$$\text{and } \frac{\partial f}{\partial y} = f(x, y + 1) - f(x, y)$$

From the 3 × 3 neighbourhood we have,

$$\frac{\partial f}{\partial x} = Z_8 - Z_5 \quad \text{and} \quad \frac{\partial f}{\partial y} = Z_6 - Z_5$$

$$\text{Since, } |\nabla F| = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}$$

$$\therefore \quad |\nabla F| = [(Z_8 - Z_5)^2 + (Z_6 - Z_5)^2]^{1/2}$$

So as to make it easier for implementation,

$$|\nabla F| = |Z_8 - Z_5| + |Z_6 - Z_5|$$

$$\text{i.e. } |\nabla F| = |Z_5 - Z_8| + |Z_5 - Z_6| \qquad ...(9.4.1)$$

This is the first order difference gradient. This can be implemented using two masks.

$|Z_5 - Z_8| =$

(1)	0
−1	0

→ mask 1

and $|Z_5 - Z_6| =$

(1)	−1
0	0

→ mask 2

This is known as the Ordinary operator.

Steps to compute the gradient of an image are as follows

(1) Convolve the original image with mask 1. This gives us the gradient along the x-direction.

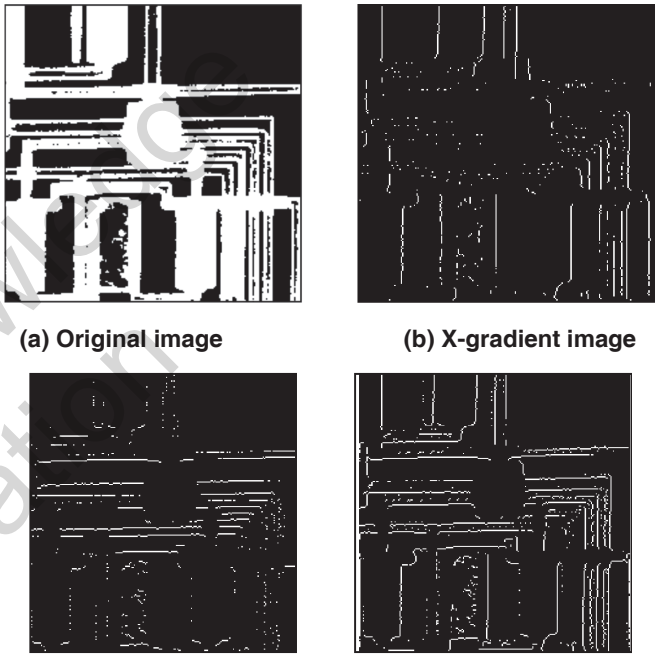
(2) Convolve the original image with mask 2. This gives us the gradient along the y-direction.

(3) Add the results of 1 and 2.

The advantage of this prolonged method is that, we can see the effect of each mask separately.

MATLAB code for ordinary operator

```
%% Ordinary operator %%
clear all
clc
aa = imread('test.jpg');
a = double(aa);
[row col] = size(a);
w1=[1  0;-1 0];
w2=[1 -1; 0 0];
for x=2:1:row-1;
for y=2:1:col-1;
a1(x,y)=w1(1)*a(x,y)+w1(2)*a(x,y+1)+w1(3)*a(x+1,y)
+w1(4)*a(x+1,y+1);
a2(x,y)=w2(1)*a(x,y)+w2(2)*a(x,y+1)+w2(3)*a(x+1,y)
+w2(4)*a(x+1,y+1);
        end
    end
    a3=a1+a2;    %% To find the resultant gradient image
figure(1)
imshow(uint8(a1)) %% The x-gradient image, you might need normalization
figure(2)
imshow(uint8(a2)) %% The y-gradient image, you might need normalization
figure(3)
imshow(uint8(a3)) %% Final image
```



(a) Original image **(b) X-gradient image**

(c) Y-gradient image **(d) Resultant gradient image**

Fig. 9.4.1

There is another direct method of implementing the ordinary operator practically. It gives results that are almost similar.

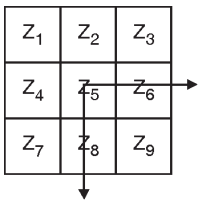
(1) Add masks 1 and 2

(1)	0	+	(1)	−1	=	2	−1
−1	0		0	0		−1	0

(2) Convolve the original image with this resultant mask to get the gradient image.

9.4.1 Roberts Mask

Robert L.G in 1965 published a paper in which he stated that better results could be obtained if cross differences were taken instead of the straight difference.

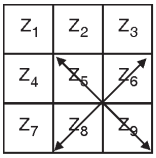


Ordinary masks

$$|\nabla F| = |Z_5 - Z_8| + |Z_5 - Z_6|$$

Roberts masks

$$|\nabla F| = |Z_5 - Z_9| + |Z_6 - Z_8| \quad \ldots(9.4.2)$$



∴ Roberts masks are

1	0
0	–1

0	1
–1	0

The results of the Roberts mask are shown along with the program.

The sum of the two Roberts masks is

1	1
– 1	– 1

MATLAB code for Roberts operator

```
% Roberts operator %%
clear all
clc
aa=imread('cameraman.jpg');
a=double(aa);
[row col]=size(a);
w1= [1 0;  0 -1];
w2= [0 1; -1, 0];
for x=2:1:row-1;
for y=2:1:col-1;
a1(x,y)=w1(1)*a(x,y)+w1(2)*a(x,y+1)+w1(3)*a(x+1,y)
+w1(4)*a(x+1,y+1);
a2(x,y)=w2(1)*a(x,y)+w2(2)*a(x,y+1)+w2(3)*a(x+1,y)
+w2(4)*a(x+1,y+1);
        end
end
a3=a1+a2;      %% Final Gradient Image %%
figure(1)
imshow(uint8(a1)) %% The x-gradient image, you might
need normalization
figure(2)
imshow(uint8(a2)) %% The y-gradient image, you might
need normalization
figure(3)
imshow(uint8(a3)) %% Final image %%
```

(a) Original image

(b) X-gradient image

(c) Y-gradient image

(d) Resultant image

Fig. 9.4.2

9.4.2 Prewitts and Sobel Operators

Roberts mask, as is evident, is an even sized mask (2 × 2). Masks of even sizes are awkward to implement. Note that the differential along the diagonals of a 2 × 2 mask is used and the edge value after the convolution corresponds to the central point $\left(r - \frac{1}{2}, c - \frac{1}{2}\right)$.

This problem can be avoided using 3 × 3 masks.

Prewitts Operator

Prewitt J.M.S in his paper "Object Enhancement and Extraction" in 1970 came up with a 3 × 3 mask. The Prewitt operator, as is now called, while approximating the first derivative, assigns similar weights to all the neighbours of the candidate pixel whose edge strength is being calculated.

$$\nabla F \approx \underbrace{|(Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)|}_{\text{x-gradient}} + \underbrace{|(Z_3 + Z_6 + Z_9) - (Z_1 + Z_4 + Z_7)|}_{\text{y-gradient}} \quad \ldots(9.4.3)$$

From this equation, the masks that we obtain are

These masks are known as the Prewitts masks.

–1	–1	–1
0	0	0
1	1	1

F_x

–1	0	1
–1	0	1
–1	0	1

F_y

Sobel Operator

Duda R.O and Hart P.E in 1973 published a paper "Pattern Classification and Scene Analysis" in which they used a new operator known as the Sobel Operator.

In the Sobel Operator higher weights are assigned to the pixels close to the candidate pixels.

$$\nabla F \approx |(Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3)| + |(Z_3 + 2Z_6 + Z_9) - (Z_1 + 2Z_4 + Z_7)|$$

x-gradient y-gradient

...(9.4.4)

From this equation, the masks that we obtain are

–1	–2	–1
0	0	0
1	2	1

F_x

–1	0	1
–2	0	2
–1	0	1

F_y

Implementation of Prewitt and Sobel Operators

(1) Convolve original image with the Fx mask to get the x-gradient image.

(2) Convolve original image with Fy mask to get the y-gradient image.

(3) Add the results of step (1) and (2)

As mentioned earlier, the advantage of using three steps is that we can see the results of the Fx mask and the Fy mask separately.

A shorter way of doing this which would give almost similar results is

(i) Add the Fx and Fy masks first.

(ii) Convolve the new mask with the original image.

Prewitt mask

– 1	– 1	– 1
0	0	0
1	1	1

+

– 1	0	1
– 1	0	1
– 1	0	1

=

– 2	– 1	0
– 1	0	1
0	1	2

F_x F_y $F_x + F_y$

Sobel mask

– 1	– 2	– 1
0	0	0
1	2	1

+

– 1	0	1
– 2	0	2
– 1	0	1

=

– 2	– 2	0
– 2	0	2
0	2	2

F_x F_y $F_x + F_y$

MATLAB code for Prewitts operator

```
%% Prewitts operator %%
clear all
clc
aa=imread('test.tif');
a=double(aa);
[row col]=size(a);
w2=[-1  0  1;-1  0  1;-1  0  1];
w1=[-1  -1  -1; 0  0  0; 1  1  1];

for x=2:1:row-1;
for y=2:1:col-1;
a1(x,y)=w1(1)*a(x-1,y-1)+w1(2)*a(x-1,y)
+w1(3)*a(x-1,y+1)+w1(4)* ...
a(x,y-1)+w1(5)*a(x,y)+w1(6)*a(x,y+1)
+w1(7)*a(x+1,y-1)+w1(8)* ...
a(x+1,y)+w1(9)*a(x+1,y+1);
a2(x,y)=w2(1)*a(x-1,y-1)+w2(2)*a(x-1,y)
+w2(3)*a(x-1,y+1)+w2(4)* ...
a(x,y-1)+w2(5)*a(x,y)+w2(6)*a(x,y+1)
+w2(7)*a(x+1,y-1)+w2(8)* ...
a(x+1,y)+w2(9)*a(x+1,y+1);
    end
end
a3=a1+a2;        %% The final gradient value %%
figure(1)
imshow(uint8(a1))%% The x-gradient image,
Normalisation might be required
figure(2)
imshow(uint8(a2))%% The y-gradient image %%
figure(3)
imshow(uint8(a3))%% Final gradient image,
Normalisation might be required
```

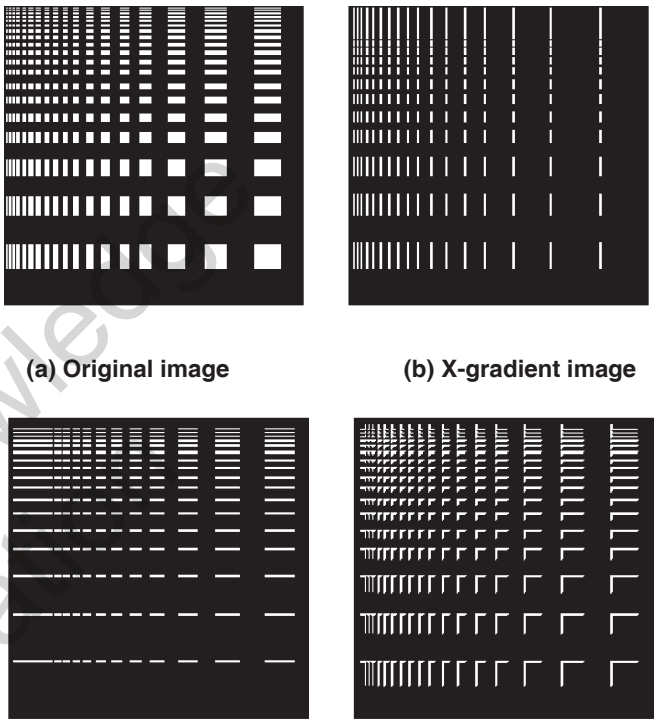
(a) Original image (b) X-gradient image

(c) Y-gradient image (d) Resultant gradient image

Fig. 9.4.3

MATLAB code for Sobel operator

```
%% Sobel operator %%
clear all
clc
aa=imread('warne.tif');
a=double(aa);
[row col]=size(a);
w1=[-1  -2  -1;  0  0   0;        1   2   1  ];
w2=[-1   0   1; -2  0   2;       -1   0   1  ];
for x=2:1:row-1;
for y=2:1:col-1;
a1(x,y)=w1(1)*a(x-1,y-1)+w1(2)*a(x-1,y)
+w1(3)*a(x-1,y+1)+w1(4)* …
a(x,y-1)+w1(5)*a(x,y)+w1(6)*a(x,y+1)
+w1(7)*a(x+1,y-1)+w1(8)* …
a(x+1,y)+w1(9)*a(x+1,y+1);
a2(x,y)=w2(1)*a(x-1,y-1)+w2(2)*a(x-1,y)
+w2(3)*a(x-1,y+1)+w2(4)* …
a(x,y-1)+w2(5)*a(x,y)+w2(6)*a(x,y+1)
+w2(7)*a(x+1,y-1)+w2(8)*
…a(x+1,y)+w2(9)*a(x+1,y+1);
        end
end
    a3=a1+a2;      %% Final gradient values %%
figure(1)
imshow(uint8(a1)) %% The x-gradient image,
Normalisation might be required
figure(2)
imshow(uint8(a2)) %% The y-gradient image,
Normalisation might be required
figure(3)
imshow(uint8(a3)) %% Final gradient image,
Normalisation might be required
```



(a) Original image (b) X-gradient image

(c) Y-gradient image (d) Resultant gradient image

Fig. 9.4.4

Both results can be obtained using absolute values.

Have you noticed one thing that is common to all the edge detection masks that we have studied so far ?

Let us draw all the masks together.

1	0
−1	0

1	−1
0	0

Ordinary operator

1	0
0	−1

0	1
−1	0

Robert's operator

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt mask

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel mask

Sum of the coefficients of each of these masks is zero !!

- This is a very important property to note. Edges are abrupt discontinuities in the gray levels and hence are high frequency regions. Since the sum of the coefficients of all these masks is zero, they eliminate all the low frequency components of the image i.e., when these masks are placed on low frequency regions, the output is zero. Hence these masks give edges without any low frequency regions in the final output image.
- Derivative filters (Gradient filters), as the name suggests, calculate the gradient of the image.
- Since noise is also high frequency, the derivative of the noise terms will always be large values and hence derivative filters are very sensitive to noise.
- There is a huge advantage in using Prewitt and Sobel masks for edge detection. Both these operators provide a smoothing effect along with providing differentiation.
- Hence Prewitt and Sobel operators perform well even when the image is noisy because they smoothen the noise !!

How do they achieve this ?

Let us consider the Prewitt operator F_x and F_y

−1	−1	−1
0	0	0
1	1	1

and

−1	0	1
−1	0	1
−1	0	1

These operators can be factored into the successive application of two simpler operators.

i.e.

−1	−1	−1
0	0	0
1	1	1

=

−1
0
1

×

1	1	1

and

−1	0	1
−1	0	0
−1	0	1

=

1
1
1

×

−1	0	1

In this

1	1	1

is a low pass or a smoothing operator, while

−1	0	1

is a high pass operator.

- Hence the Prewitt operator performs uniform smoothing in one direction with edge detection in the perpendicular direction.
- In a similar manner, we can split up the Sobel operator into smaller and simpler operators.

i.e

−1	−2	−1
0	0	0
1	2	1

=

−1
0
1

HP

×

1	2	1

LP

and

−1	0	−1
−2	0	2
1	0	1

LP

=

1
2
1

×

−1	0	1

HP

- One can verify this by comparing the results of a Prewitt/ Sobel filter with that of standard highpass filter on an image corrupted by Gaussian noise. (Use **imnoise** command to corrupt the image).

9.4.3 Compass Operators

It is seen that edges in the horizontal as well as in the vertical direction are enhanced when Prewitt's or Sobel's operator is used. There are applications, in which we need edges in all the directions. A simple method would be to rotate the Prewitts or Sobel's mask in all the possible directions.

Considering a Prewitts operator

−1	−1	−1
0	0	0
1	1	1

We move this mask in the anticlockwise direction to get all other masks

−1	−1	−1
0	0	0
1	1	1

−1	−1	0
−1	0	1
0	1	1

−1	0	1
−1	0	1
−1	0	1

0	1	1
−1	0	1
−1	−1	0

1	1	1
0	0	0
−1	−1	−1

1	1	0
1	0	−1
0	−1	−1

1	0	−1
1	0	−1
1	0	−1

0	−1	−1
1	0	−1
1	1	0

– 1	– 1	– 1
0	0	0
1	1	1

This operator is known as the compass operator and is very useful for detecting weak edges.

Compass operators can also be implemented using the Sobel operator.

```
%% Program of compass operator %%
clear all, clc
aa=imread('lily.tif');
I=double(aa);
s=size(I);
a=[-1 -1 -1; 0 0 0; 1 1 1]'
b=[a(1,2) a(1,3) a(2,3); a(1,1) a(2,2) a(3,3); a(2,1) a(3,1)
a(3,2)]
c=[b(1,2) b(1,3) b(2,3); b(1,1) b(2,2) b(3,3); b(2,1) b(3,1)
b(3,2)]
d=[c(1,2) c(1,3) c(2,3); c(1,1) c(2,2) c(3,3); c(2,1) c(3,1)
c(3,2)]
e=[d(1,2) d(1,3) d(2,3); d(1,1) d(2,2) d(3,3); d(2,1) d(3,1)
d(3,2)]
f=[e(1,2) e(1,3) e(2,3); e(1,1) e(2,2) e(3,3); e(2,1) e(3,1)
e(3,2)]
g=[f(1,2) f(1,3) f(2,3); f(1,1) f(2,2) f(3,3); f(2,1) f(3,1)
f(3,2)]
h=[g(1,2) g(1,3) g(2,3); g(1,1) g(2,2) g(3,3); g(2,1) g(3,1)
g(3,2)]
for x=2:1:s(1)-1;
for y=2:1:s(2)-1;
A(x,y)=[a(1)*I(x-1,y-1)+a(2)*I(x-1,y)
+a(3)*I(x-1,y+1)+a(4)*I(x,y-1)+ …
a(5)*I(x,y)+a(6)*I(x,y+1)+a(7)*I(x+1,y-1)
+a(8)*I(x+1,y)+a(9)*I(x+1,y+1)];
B(x,y)=[b(1)*I(x-1,y-1)+b(2)*I(x-1,y)
+b(3)*I(x-1,y+1)+b(4)*I(x,y–1) …
+b(5)*I(x,y)+b(6)*I(x,y+1)+b(7)*I(x+1,y-1)
+ b(8)*I(x+1,y)+b(9)*I(x+1,y+1)];
C(x,y)=[c(1)*I(x-1,y-1)+c(2)*I(x-1,y)
+c(3)*I(x-1,y+1)+c(4)*I(x,y-1) …
+c(5)*I(x,y)+c(6)*I(x,y+1)+c(7)*I(x+1,y-1)
+c(8)*I(x+1,y)+c(9)*I(x+1,y+1)];
D(x,y)=[d(1)*I(x-1,y-1)+d(2)*I(x-1,y)
+d(3)*I(x-1,y+1)+d(4)*I(x,y-1)+ …
d(5)*I(x,y)+d(6)*I(x,y+1)+d(7)*I(x+1,y-1)
+d(8)*I(x+1,y)+d(9)*I(x+1,y+1)];
E(x,y)=[e(1)*I(x-1,y-1)+e(2)*I(x-1,y)
+e(3)*I(x-1,y+1)+e(4)*I(x,y-1) …
+e(5)*I(x,y)+e(6)*I(x,y+1)+e(7)*I(x+1,y-1)
+e(8)*I(x+1, y)+e(9)*I(x+1,y+1)];
F(x,y)=[f(1)*I(x-1,y-1)+f(2)*I(x-1,y)
+f(3)*I(x-1,y+1)+f(4)*I(x,y-1) …
+f(5)*I(x,y)+f(6)*I(x,y+1)+f(7)*I(x+1,y-1)
+f(8)*I(x+1,y)+f(9)*I(x+1,y+1)];
G(x,y)=[g(1)*I(x-1,y-1)+g(2)*I(x-1,y)
+g(3)*I(x-1,y+1)+g(4)*I(x,y-1) …
+g(5)*I(x,y)+g(6)*I(x,y+1)+g(7)*I(x+1,y-1)
+g(8)*I(x+1,y)+g(9)*I(x+1, y+1)];
H(x,y)=[h(1)*I(x-1,y-1)+h(2)*I(x-1,y)
+h(3)*I(x-1,y+1)+h(4)*I(x,y-1) …
+h(5)*I(x,y)+h(6)*I(x,y+1)+h(7)*I(x+1,y-
1)+h(8)*I(x+1,y)+h(9)*I(x+1, y+1)];
        end
end
W=max(max(max(max(max(max(A,B),C),D),E),F),G),
H); %% To calculate maximum
%% Plotting %%
figure(1),imshow(uint8(A)),figure(2),imshow(uint8(B))
figure(3),imshow(uint8(C)),figure(4),imshow(uint8(D))
figure(5),imshow(uint8(E)),figure(6),imshow(uint8(F))
figure(7),imshow(uint8(G)),figure(8),imshow(uint8(H))
figure(9),imshow(uint8(W)),figure(10),imshow(uint8(I))
%% Normalisation may be required %%
```

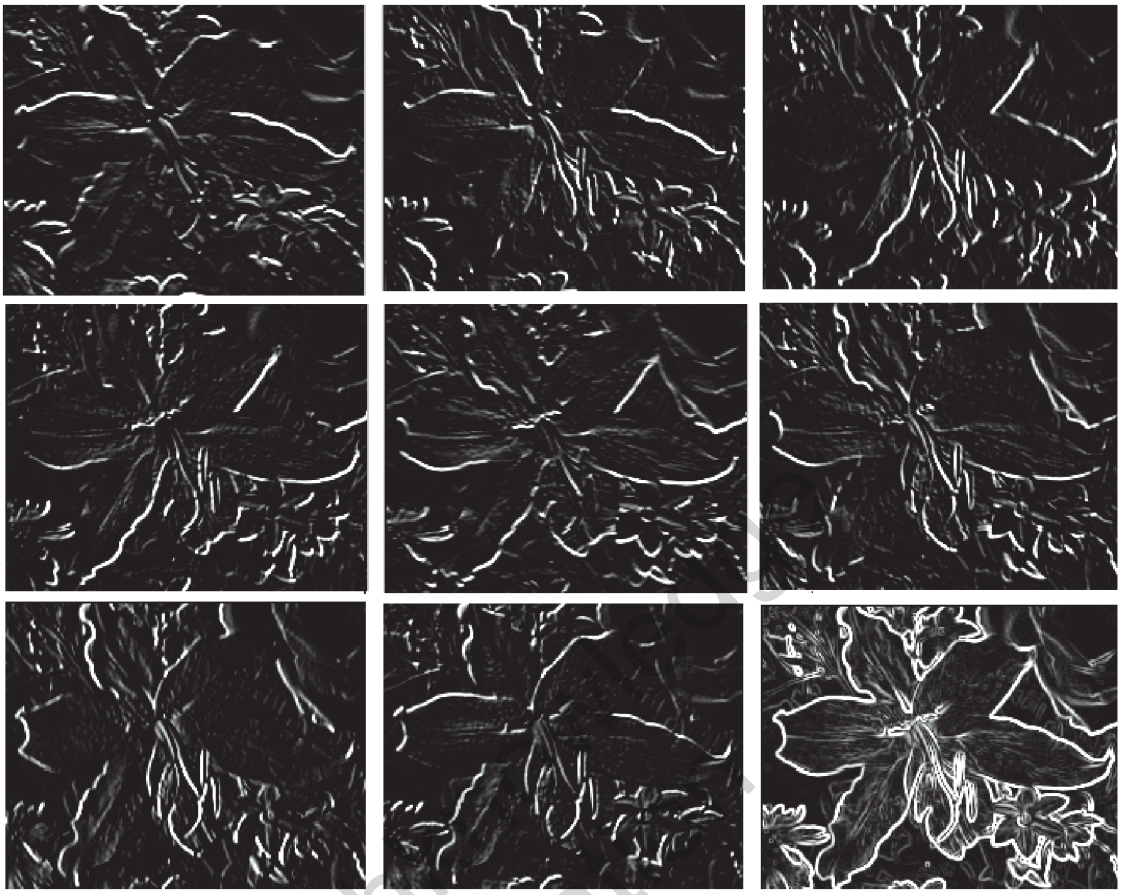


Fig. 9.4.5 : Compass operated images

9.5 Image Segmentation using the Second Derivative - the Laplacian

– We have already seen the effects of using the first derivative of the image. It was shown that the first derivative does enhance the edges of the image. We now try to find out the effects of computing the second derivative on the edges.

We know,

$$\nabla F = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}$$

$$\therefore \quad \frac{\partial f}{\partial x} = f(x+1, y) - f(x, y) \quad \text{and}$$

$$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$

The second derivative is given by

$$\nabla^2 F = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Where $$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \quad \text{and}$$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\nabla^2 F = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\therefore \quad |\nabla^2 F| = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)] \quad ...(9.5.1)$$

Considering the 3×3 neighbourhood

	y – 1	y	y + 1
x – 1	Z_1	Z_2	Z_3
x	Z_4	Z_5	Z_6
x + 1	Z_7	Z_8	Z_9

This equation in the discrete form reduces to,

$$|\nabla^2 F| = [Z_8 + Z_2 + Z_6 + Z_4 - 4Z_5]$$

– This equation can be implemented using a mask i.e.

0	1	0
1	(–4)	1
0	1	0

– This is known as the Laplacian operator. Some books reverse the signs of the coefficients of the mask i.e.,

0	–1	0
–1	4	–1
0	–1	0

– There might be cases when, we would need to add higher weights at the center pixel (>4). While doing this we must make sure that the sum of the coefficients of the mask is zero since edges are high frequency components. Hence if we increase the weights at the centre, we would also have to change the values of the coefficients at the borders.

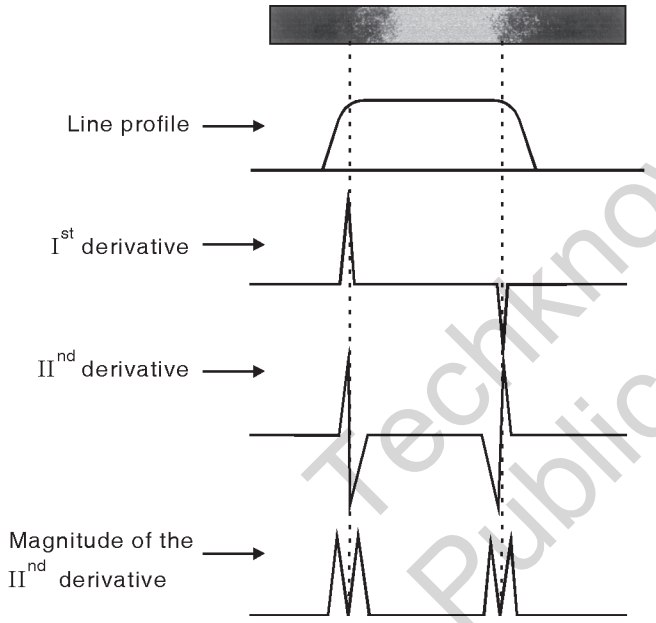


Fig. 9.5.1

– One important thing to note about the Laplacian mask is that unlike the Sobel and Prewitt operator, they are isotropic filters. i.e., their response is independent of the direction of the discontinuties in the image. In other words, isotropic filters are rotation invarient. i.e., rotating the image and then applying the mask gives the same result as applying the mask to the image first and then rotating the result.

– Now that we have the second derivative Laplacian mask with us, can we use it directly on the image ? the answer is No. The Laplacian is not generally used in its original form as an edge detector for mainly two reasons.

(1) Since the Laplacian is a second derivative filter, it is very sensitive to noise (much more than the first derivative). Hence in an image, if there is any noise present, the Laplacian gives very large values and ruins the entire image.

(2) The magnitude of the Laplacian produces double edges, which is an undesireable effect. Let us explain what we mean by double edges. Consider a strip of image shown in Fig. 9.5.1. It is seen that the magnitude of the 2nd derivative produces two peaks for a single edge.

The zero crossing property of the Laplacian is used to detect edges.

9.5.1 Laplacian of Gaussian Operator

– As stated earlier, the Laplacian mask evokes very strong response to stray noise pixels. Hence if some kind of noise cleaning is done prior to the application of the Laplacian operator, better results could be obtained. In practice the Laplacian is preceded by a smoothing operation (Gaussian smoothing). The resultant algorithm is commonly known as a Laplacian of Gaussian (LoG) or Marr-Hildreth operator. (Marr D.C. and Hildreth presented a paper "Theory of Edge Detection" in 1980 which explains in this concept)

Let us explain the mathematics of this.

– What we just said was that, we first smoothen the image using a Gaussian function (Remember low pass Gaussian filter?) and then take the Laplacian (second derivative of the image). A simple method to implement this practically, is to combine the two i.e. combine the action of a gaussian function and a second derivative function.

Consider the gaussian function

$$h(x, y) = e^{-\left(\frac{x^2 + y^2}{2\sigma^2}\right)} \qquad \ldots(9.5.2)$$

where σ is the standard deviation and x and y are the two variables (spatial coordinates). It is σ that determines the degree of blurring.

$$\text{Let } x^2 + y^2 = r^2$$

$$\therefore \quad h(r) = e^{-\left(\frac{r^2}{2\sigma^2}\right)}$$

– The Laplacian of h i.e., the second derivative of h with respect to r is now calculated. We first calculate the first derivative w.r.t. r

$$\frac{\partial}{\partial r} h(r) = \frac{\partial}{\partial r} e^{-r^2/2\sigma^2} = e^{-r^2/2\sigma^2} \cdot \frac{\partial}{\partial r}\left(\frac{-r^2}{2\sigma^2}\right)$$

$$\frac{\partial}{\partial r} h(r) = e^{-r^2/2\sigma^2} \cdot \left(\frac{-2r}{2\sigma^2}\right) = e^{-r^2/2\sigma^2} \cdot \left(\frac{-r}{\sigma^2}\right)$$

Now taking the second derivative, we get

$$\frac{\partial^2}{\partial r^2} h(r) = \frac{\partial}{\partial r}\left(\frac{-r}{\sigma^2} e^{-r^2/2\sigma^2}\right) = \frac{-1}{\sigma^2} \frac{\partial}{\partial r}\left(re^{-r^2/2\sigma^2}\right)$$

$$\frac{\partial^2}{\partial r^2} h(r) = \frac{-1}{\sigma^2}\left(re^{-r^2/2\sigma^2}\left(\frac{-r}{\sigma^2}\right) + e^{-r^2/2\sigma^2}\right)$$

$$= \frac{1}{\sigma^2}\left(\frac{r^2 e^{-r^2/2\sigma^2}}{\sigma^2} - e^{-r^2/2\sigma^2}\right)$$

$$= \frac{1}{\sigma^2} e^{-r^2/2\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right)$$

$$\frac{\partial^2}{\partial r^2} h(r) = \frac{r^2 - \sigma^2}{\sigma^4} e^{-r^2/2\sigma^2}$$

$$\therefore \nabla^2 h = \left(\frac{r^2 - \sigma^2}{\sigma^4}\right) e^{-r^2/2\sigma^2} \quad \ldots(9.5.3)$$

- This is the Laplacian of a Gaussian (LoG). $\nabla^2 h$ can be easily plotted using MATLAB. Due to the shape of the LoG, it commonly referred to as the Mexican Hat function.

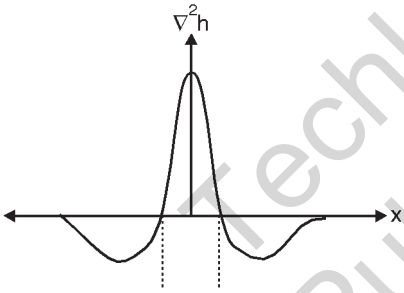


Fig. 9.5.2

- It can be seen that $\nabla^2 h$ has a large positive value at the centre and negative values at the peripheries.

We can approximate this by using a 5 × 5 mask as shown.

- Remember, this mask is not unique. We could use any values that approximate the shape of the LoG.
- One thing that needs to be kept in mind is that since our intention is to enhance the edges, the sum of coefficients of the mask should be zero.

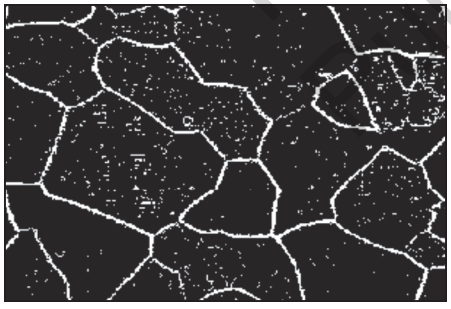
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

MATLAB code for obtaining Laplacian of a Gaussian image

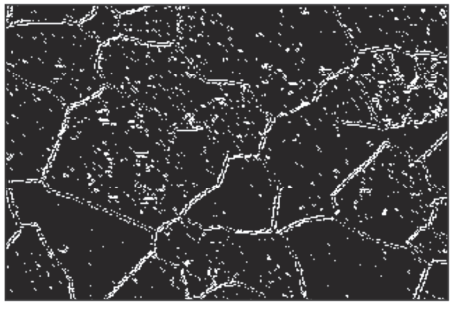
```
clear all
clc
a=imread('alumgrns.tif');
a=double(a);
    [row col]=size(a);
    lap=[0 1  0;1  -4  1;0  1  0];    %% Laplacian mask
    log=[0 0 -1 0 0;0 -1 -2 -1 0;-1 -2 16 -2 -1;0 -1 -2 -1
0;0 0 -1 0 0];%% LOG
    for x=3:1:row-2;
    for y=3:1:col-2;
%% Calculating only the Laplacian %%
c(x,y)=[lap(1)*a(x-1,y-1)+lap(2)*a(x-1,y)
+lap(3)*a(x-1,y+1)+lap(4)*a(x,y-1)+...
    lap(5)*a(x,y)+lap(6)*a(x,y+1)
+lap(7)*(x+1,y-1)+lop(8)*a(x+1,y)+...
    lap(9)*a(x+1,y+1)];
%% Calculating the Laplacian of a Gaussian  %%
dee(x,y)=log(1)*a(x-2,y-2)+log(2)*a(x-2,y-1)
+log(3)*a(x-2,y)+log(4)*a(x-2,y+1)+...
    log(5)*a(x-2,y+2)+log(6)*a(x-1,y-2)
+log(7)*a(x-1,y-1)+log(8)*a(x-1,y)+...
    log(9)*a(x-1,y+1)+log(10)*a(x-1,y+2)
+log(11)*a(x,y 2)+log(12)*a(x,y-1)+...
    log(13)*a(x,y)+log(14)*a(x,y+1)+log(15)*a(x,y+2)
+log(16)*a(x+1,y-2)+...
    log(17)*a(x+1,y-1)+log(18)*a(x+1,y)
+log(19)*a(x+1, y+1)+log(20)*a(x+1,y+2)+...
    log(21)*a(x+2,y-2)+log(22)*a(x+2,y-
1)+log(23)*a(x+2,y)+log(24)*a(x+2,y+1)+...
    log(25)*a(x+2,y+2);
    end
end
%% Thresholding the output of the LoG operator %
for x=1:1:row-2
    for y=1:1:col–2
        if dee(x,y)<40
            deep(x,y)=0;
        else
            deep(x,y)=255
    end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%ZERO CROSSINGS%%%%%%%%%%%
for x=2:1:row-2
    for y=2:1:col-2
        a1=deep(x,y);
        a2=deep(x-1,y-1);
```

```
        if a1==0 && a2==0
        deepa(x,y)=0;
        deepa(x-1,y-1)=0;
    else if a1==255 && a2==255
            deepa(x,y)=0;
            deepa(x-1,y-1)=0;
        else
            deepa(x,y)=255;
            deepa(x-1,y-1)=255;
            end
            end
        end
        end
    end
    figure(1),imshow(uint8(a))        %% Original image
    figure(2),imshow(uint8(c))        %% Only Laplacian
    figure(3),imshow(uint8(dee))      %% LoG
    figure(4),imshow(uint8(deep))     %% LoG+Threshold
    figure(5),imshow(uint8(deepa)) %% Zero crossings
```

(a) Original image



(b) Image using LoG operator



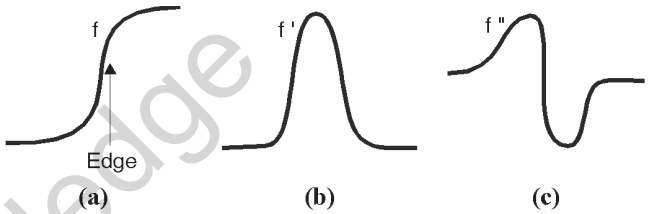
(c) Zero crossing image

Fig. 9.5.3

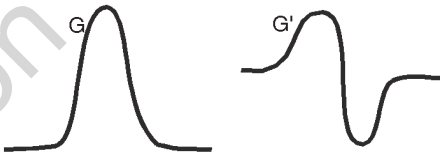
9.5.2 Canny Edge Detector

- This is the last of the edge detectors that we would study here. Canny operator (Named after J. K. Canny) is a first derivative edge detector coupled with noise cleaning. Canny has, over the years, become one of the most popular derivative operators. Like in the LoG, a Gaussian function is used to smoothen the noise.
- In the Canny edge detector, we first smoothen the image using a Gaussian low pass filter and then take the first derivative.

Consider the Fig. 9.5.4,



(a) A ramp edge, (b) First derivative of ramp edge, (c) Second derivative of ramp edge



(d) Gaussian function, **(e) First derivative of Gaussian function**

Fig. 9.5.4

Compare Fig. 9.5.4(c) and Fig. 9.5.4(e). Notice the similarity between the shape of the first derivative of the Gaussian and the second derivative of the ramp edge.

Hence the derivative of the bell shape of the Gaussian function approximates the second derivative or zero crossing operator (LoG).

9.6 Connectivity

MU : Dec. 2016, Dec. 2017

Q. Write short note on : 4, 8 and m-connectivity.
(Dec. 2016, Dec. 2017, 5 Marks)

- Consider a pixel p(x, y). This pixel p has two horizontal and two vertical neighbours which share a surface with p(x, y).
- This set of four pixels is known as the 4-neighbourhood of pixel p(x, y), where each pixel is said to be at a unit distance from p(x, y). This set of pixels is denoted as $N_4(p)$. In a similar manner, apart from these 4-neighbours, there are another four diagonal pixels which touch p(x, y) at the corners $N_D(p)$.

	p(x – 1, y)	
p(x, y – 1)	p(x, y)	p(x, y + 1)
	p(x + 1, y)	

$N_4(p)$

- This set of eight pixels (The 4-neighbours as well as the diagonal pixels) is called the 8-neighbourhood of pixel p(x, y) and is denoted as $N_8(p)$

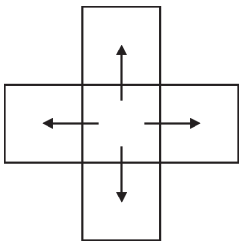
$\therefore N_8(p) = N_4(p) + N_D(p)$.

- Connectivity is an important concept that is used in segmentation. When we started off with this chapter, it was stated that segmentation is used for machine vision.

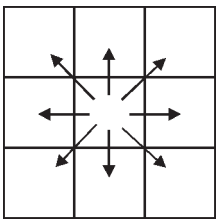
p(x –1, y – 1)		p(x –1, y + 1)
	p(x, y)	
p(x + 1, y – 1)		p(x + 1, y + 1)

$N_D(p)$

- In segmentation, the machine (computer) scans the image to form different regions. This scanning of the image by the machine is based on the connectivity that the user or the code specifies. To establish if two pixels are connected, it must be determined whether they are neighbours (4-neighbours or 8-neighbours) and if their grey levels satisfy a specified criteria.
- Two regions that touch only at a corner can be considered to be a single region or two distinct regions : How they are considered depends on the definition of connectivity used ?
- Two pixels p and q with some common criteria are said to be 4-connected if they share a side. i.e., two pixels p and q with some common criteria are 4-connected if q is in the set $N_4(p)$.



- Two pixels p and q with some common criteria are said to be 8-connected if they share either a side or a corner. i.e., two pixels p and q with some common criteria are 8-connected if q is in the set $N_8(p)$.



S_1 S_2

0	0
1 (p)	0
0	1 (q)

- Let us take an example. Given the criteria that p = q = 1 for the region to be connected, check whether the two region S_1 and S_2 are connected.
- Even though p and q are both equal to one, these two regions will not be shown connected, if we use a 4-connectivity algorithm. This is because, in the 4-connectivity algorithm, we only check the 4-neighbours and see if any of these four neighbours has a pixel with the same value 1, in this case
- If we use an 8-connectivity algorithm, S_1 and S_2 would be shown as a single connected region. Apart from 4-connectivity and 8-connectivity, we also have m-adjacent or m-connectivity (mixed connectivity). It is used to eliminate the double paths that arise when we use 8-connectivity.

Let us take an example.

Consider the image shown.

0	2	2
0	2	0
0	0	2

Here the condition used is that pixels belong to a region if their value is 2.

When we use 8-connectivity here, we get

0	2	2
0	2	0
0	0	2

We notice that the pixel at the right hand upper corner gets connected twice due to 8-connectivity. The ambiguity is removed by using m-connectivity.

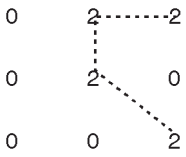
Hence m-connectivity can be defined as follows.

Two pixels p and q are said to be m-connected if

(i) q is in $N_4(p)$ or

(ii) q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ is empty

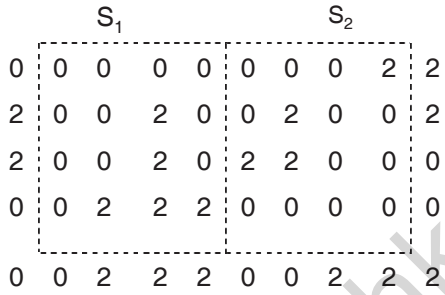
Using this definition we get



Any connectivity can be used as long as one is consistent. Often 8-connectivity yields results that lie closer to one's intuition.

9.6.1 Solved Example on Connectivity

Ex. 9.6.1 : Consider two image subsets S_1 and S_2

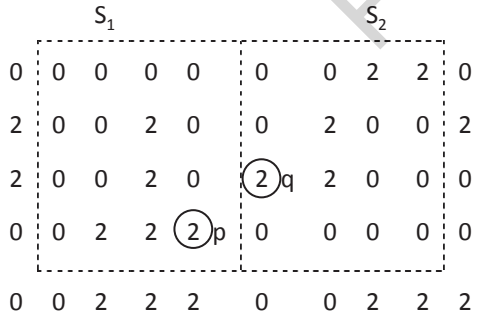


For V = {2}, determine whether S_1 and S_2 are

(a) 4-connected (b) 8-connected (c) m-connected

Soln. :

V = {2} simply means that two pixels p and q are connected if their values are equal to 2.



(a) S_1 and S_2 are not 4-connected because q is not in the set of $N_4(p)$.

(b) S_1 and S_2 are eight connected because q is in the set of $N_8(p)$.

(c) S_1 and S_2 are m-connected because

(i) q is in the set $N_D(p)$ and

(ii) the set $N_4(p) \cap N_4(q)$ is empty

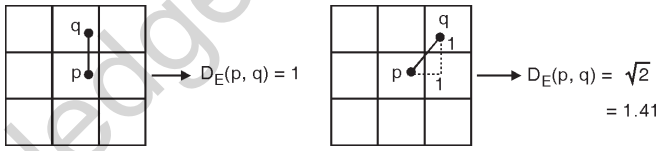
9.7 Distance Transform

Though not directly related to segmentation, Distance transform is introduced here as it is related to connectivity. Distance transforms are required in image processing projects where measurements have to be made. The distance transform provides a measure of the separation of points in an image.

(1) Euclidean Distance : Euclidean distance is the straight line distance between two pixels. If p and q are the two pixels with coordinates (x_1, y_1) and (x_2, y_2), then

$$D_E = [(x_1 - x_2)^2 + (y_1 - y_2)^2]^{1/2}$$

Diagrammatically it can be shown as



$\therefore$ for a 3×3 region, we have the Euclidian distance as

(2) City Block Distance (D_4 Distance) : For the same points $p(x_1, y_1)$ and $q(x_2, y_2)$, the city block distance is defined as

$$D_{CITY}(p, q) = D_4(p, q) = |x_1 - x_2| + |y_1 - y_2|$$

The city block distance measures the path between the pixels based on a 4-connected neighbourhood. Pixels whose edges touch are 1 unit apart and pixels touching diagonally are 2 units apart.

2	1	2
1	0	1
2	1	2

(3) Chess Board Distance (D_8 Distance) : For pixels $p(x_1, y_1)$ and $q(x_2, y_2)$, the chess board distance is defined as

$$D_{CHESS}(p, q) = D_8(p, q) = \max(|x_1 - x_2|, |y_1 - y_2|)$$

The chessboard distance measures the path between the pixels based on a 8-connected neighbourhood. Pixels whose edges or corners touch are 1 unit apart.

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

(4) D_m Distance (m-adjacency Distance) : This distance measure is based on m-adjacency. Pixels p and q are m-adjacent if

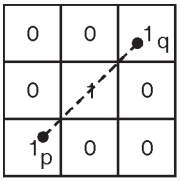
(a) q is in $N_4(p)$ or

(b) q is in $N_D(p)$ and $N_4(p) \cap N_4(q)$ is empty.

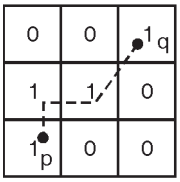
9.7.1 Solved Examples on Distance Transform

Ex. 9.7.1 : Measure the D_m distance between p and q for the following images.

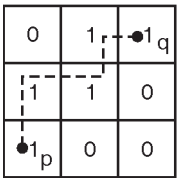
Soln. :



→ The path from p to q is shown by dotted line
$D_m = 2$



→ Since we consider m-connectivity, the path that needs to be taken is shown by dotted line
$\therefore D_m = 3$



→ The path taken from p to q under m-connectivity is shown
$\therefore D_m = 4$

Ex. 9.7.2 : Given below is a 5 × 5 image. Find out D_4, D_8.

Soln. :

	0	1	2	3	4
0	3	2	4	3	(1q)
1	0	4	4	3	2
2	2	2	2	0	2
3	2	2	1	1	1
4	(1p)	0	1	0	3

$p(x_1, y_1) = p(4, 0)$

$q(x_2, y_2) = q(0, 4)$

(a) $D_4(p, q) = |x_1 - x_2| + |y_1 - y_2|$

$= |4 - 0| + |0 - 4|$

$= 4 + 4 = 8$

(b) $D_8(p, q) = \max(|x_1 - x_2|, |y_1 - y_2|)$

$= \max(|4 - 0|, |0 - 4|)$

$= \max(4, 4) = 4$

Ex. 9.7.3 : Consider the image segment shown

3	1	2	1q
2	2	0	2
1	2	1	1
1p	0	1	2

for V = {0, 1}, compute the length of the shortest m-path between p and q.

Soln. : The path taken is shown below :

3	1	2	1
2	2	0	2
1	2	1	1
1	0	1	2

Hence the shortest m-distance is equal to 5.

9.8 Additional Solved Examples

Ex. 9.8.1 : Develop an algorithm for converting a one-pixel thick 8-path to a 4-path.

Soln. : This can be done by defining all possible neighbourhood shapes to go from a diagonal segment to a corresponding 4-connected segment, as shown in Fig. P. 9.8.1.

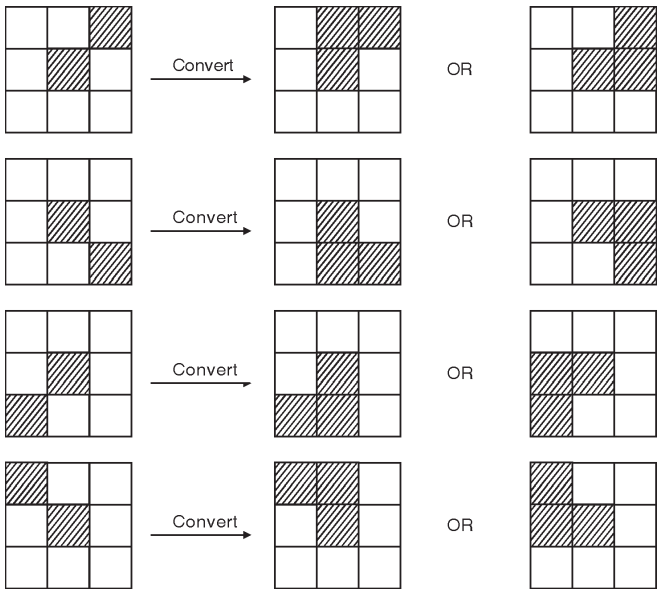


Fig. P. 9.8.1

The algorithm then simply looks for the appropriate match everytime a diagonal segment is encountered.

Ex. 9.8.2 : A binary image contains straight lines oriented horizontally, vertically, at 45° and at – 45°. Give a set of 3 × 3 masks that can be used to detect 1 pixel long breaks in these lines. Assume that the grey level of the lines is 1 and that the grey level of the background is 0.

Soln. :

The 4 masks that we could use are

0	0	0
1	–2	1
0	0	0

Horizontal

0	1	0
0	–2	0
0	1	0

Vertical

0	0	1
0	–2	0
1	0	0

+45°

1	0	0
0	–2	0
0	0	1

–45°

Note that the sum of the coefficients along the required direction is zero.

Hence each mask would yield a value of 0 when centered on a pixel of an unbroken 3-pixel segment oriented in the direction of the required mask. The response of the mask would be +2 when the mask is centered on a one pixel gap in a 3-pixel segment oriented in the direction of the required mask.

Ex. 9.8.3 : Show that the average value of the Laplacian operator $\nabla^2 h$ is zero.

Soln. :

We have seen that

$$\nabla^2 h(r) = + \left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

What we need to prove is

$$\int_{-\infty}^{\infty} \left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}} dr = 0$$

$$\frac{1}{\sigma^4} \int_{-\infty}^{\infty} r^2 e^{-\frac{r^2}{2\sigma^2}} dr - \frac{1}{\sigma^2} \int_{-\infty}^{\infty} e^{-\frac{r^2}{2\sigma^2}} dr = 0$$

We know from the definition of the Guassian density that

$$\frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} e^{-\frac{r^2}{2\sigma^2}} dr = 1$$

We also know that the variance of a Guassian random variable is

$$\text{Variance}(r) = \sigma^2 = \int_{-\infty}^{\infty} r^2 e^{-\frac{r^2}{2\sigma^2}} dr$$

∴ We get

$$\int_{-\infty}^{\infty} \left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}} dr = \frac{\sqrt{2\pi\sigma^2}}{\sigma^4} \sigma^2 - \frac{\sqrt{2\pi\sigma^2}}{\sigma^2} = 0$$

Ex. 9.8.4 : Show that the Laplacian mask is a Isotropic filter. (Hint : use a 3 × 3 mask)

Soln. :

By Isotropic filters we mean, filters that are rotation invariant. It means that rotating the image and then applying the filter gives the same result as applying the filter to the image first and then rotating the result.

A 3 × 3 Laplacian mask is given

0	– 1	0
– 1	4	– 1
0	– 1	0

Consider a 3 × 3 image

1	2	1
2	4	2
4	8	4

Case 1: Image → Convolve with Laplacian mask → Rotate

Applying the mask we get,

0	2	0
–1	2	–1
6	20	6

Rotate →

6	20	6
–1	2	–1
0	2	0

Case 2 : Image → Rotate → Convolve with Laplacian mask

Rotating image we get,

4	8	4
2	4	2
1	2	1

Apply mask →

6	20	6
–1	2	–1
0	2	0

We see that (1) = (2)

Hence we conclude that the Laplacian mask is an Isotropic filter (Invariant to rotation).

Note : Convolution was performed after zero padding the image.

Ex. 9.8.5 : Show that the Laplacian operation defined by $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$ is istotropic

Soln. :

We use the following equations relating coordinates after axis rotation by angle θ.

$$x = x' \cos\theta - y' \sin\theta$$
$$y = x' \sin\theta + y' \cos\theta$$

where (x, y) are the unrotated and (x′, y′) are the rotated coordinates.

Laplacian operation for the unrotated coordinates are,

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

and the Laplacian operation for the rotated coordinates are,

$$\nabla^2 f = \frac{\partial^2 f}{\partial x'^2} + \frac{\partial^2 f}{\partial y'^2}$$

Let us start with

$$\frac{\partial f}{\partial x'} = \frac{\partial f}{\partial x} \cdot \frac{\partial x}{\partial x'} + \frac{\partial f}{\partial y} \cdot \frac{\partial y}{\partial x'} = \frac{\partial f}{\partial x} \cos\theta + \frac{\partial f}{\partial y} \sin\theta$$

We take the partial derivative of this expression again with respect to x′

$$\frac{\partial^2 f}{\partial x'^2} = \frac{\partial^2 f}{\partial x^2} \cos^2\theta + \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right) \sin\theta \cos\theta + \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right) \cos\theta \sin\theta + \frac{\partial^2 f}{\partial y^2} \sin^2\theta \quad \ldots(1)$$

We now find $\frac{\partial f}{\partial y'}$

$$\frac{\partial f}{\partial y'} = \frac{\partial f}{\partial x} \cdot \frac{\partial x}{\partial y'} + \frac{\partial f}{\partial y} \cdot \frac{\partial y}{\partial y'} = -\frac{\partial f}{\partial x} \sin\theta + \frac{\partial f}{\partial y} \cos\theta$$

Taking the partial derivatives of this expression with respect to y′ gives us

$$\frac{\partial^2 f}{\partial y'^2} = \frac{\partial^2 f}{\partial x^2} \sin^2\theta - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right) \cos\theta \sin\theta - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right) \sin\theta \cos\theta + \frac{\partial^2 f}{\partial y^2} \cos^2\theta$$

Adding the two equations we get

$$\frac{\partial^2 f}{\partial x'^2} + \frac{\partial^2 f}{\partial y'^2} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

This proves that the Laplacian operation is independent of rotation.

Ex. 9.8.6 : Show that subtracting the Laplacian from an image is proportional to unsharp masking.

Soln. :

Let f(x, y) be the original image.

The Laplacian mask is given by

0	1	0
1	– 4	1
0	1	0

$$\therefore f(x, y) - \nabla^2 f(x, y) = f(x, y) - [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)]$$

$$= 6\, f(x, y) - [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) + f(x, y)]$$

$$= 5\left[1.2\, f(x, y) - \frac{1}{5}[f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) + f(x, y)]\right]$$

$$f(x, y) - \nabla^2 f(x, y) = 5\,\{1.2\, f(x, y) - \bar{f}\,(x, y)\}$$

Here $\bar{f}\,(x, y)$ denotes the average of f(x, y) .

Treating the constants as factors of proportionality we get.

$$f(x, y) - \nabla^2 f(x, y) \approx f(x, y) - \bar{f}\,(x, y)$$

The RHS is the equation for unsharp masking. Hence subtracting the Laplacian from an image is equivalent to unsharp masking.

Ex. 9.8.7 : Segment the following image into two regions based on the edge-orientations derived through a gradient operator clearly depicting the boundary.

27	27	27	27	27
25	25	25	25	25
23	23	30	33	36
21	21	33	36	39
19	19	36	39	42

Soln. : The easiest way to segment this image into two regions would be use a thresholding function with T ≥ 28. But since we have been asked to do this using gradient operators, let us use a Prewitts compass operator.

We use the following Prewitts masks and convolve each one of them with the image.

We finally calculate the maximum.

M_1 =

– 1	– 1	– 1
0	0	0
1	1	1

M_2 =

– 1	– 1	0
– 1	0	1
0	1	1

$M_3 =$

– 1	0	1
– 1	0	1
– 1	0	1

$M_4 =$

0	1	1
– 1	0	1
– 1	– 1	0

$M_5 =$

1	1	1
0	0	0
– 1	– 1	– 1

$M_6 =$

1	1	0
1	0	– 1
0	– 1	– 1

$M_7 =$

1	0	– 1
1	0	– 1
1	0	– 1

$M_8 =$

0	– 1	– 1
1	0	– 1
1	1	0

Let the original image be f (x, y).

While convolving we ignore the corner pixels.

$A = f * M_1 =$

0	0	0	0
0	– 5	5	18
0	0	15	33
0	– 2	8	18

$B = f * M_2 =$

0	0	0	0
0	– 8	– 1	9
0	– 15	– 6	13
0	– 27	– 23	0

$C = f * M_3 =$

0	0	0	0
0	– 7	– 10	– 6
0	– 19	– 25	– 12
0	– 36	– 45	– 18

$D = f * M_4 =$

0	0	0	0
0	1	– 9	– 15
0	– 11	– 29	– 31
0	– 21	– 37	– 24

$E = f * M_5 =$

0	0	0	0
0	5	– 5	– 18
0	0	– 15	– 33
0	2	– 8	– 18

$F = f * M_6 =$

0	0	0	0
0	8	1	– 9
0	15	6	– 13
0	27	23	0

$G = f * M_7 =$

0	0	0	0
0	7	10	6
0	19	25	12
0	36	45	18

$H = f * M_8 =$

0	0	0	0
0	– 1	9	15
0	11	29	31
0	21	37	24

We now take the maximum from each of the pixel locations.

Hence we have

$W =$

0	0	0	0
0	8	10	18
0	19	29	33
0	36	45	24

We zero pad one row and one column to get the final image.

Final =

0	0	0	0	0
0	0	0	0	0
0	0	8	10	18
0	0	19	29	33
0	0	36	45	24

Hence we have two regions, one being the zero region (say a) and the other being the non-zero region (say b).

a	a	a	a	a
a	a	a	a	a
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

Summary

Segmentation is an essential step in almost all image analysis systems. It is imperative to understand the applications of image segmentation. Segmentation partitions an image into meaningful regions having certain characteristics unique to that region. In this, the output is an abstract representation of the input image. There exists no general segmentation algorithm which will work satisfactorily for all images. Users have to choose a particular segmentation algorithm which would be suitable for the problem in hand. Segmentation based on dissimilarities as well as segmentation based on similarities have been discussed in detail. MATLAB codes have been provided for gradient operators such as Prewitts and Sobel. Special emphasis has been given to Hough transforms which forms an important topic of edge linking.

Review Questions

Q. 1 Explain the term Image Segmentation.

Q. 2 Explain, segmentation based on discontinuties and segmentation based on similarities.

Q. 3 Explain edge detection in detail.

Q. 4 Show that the Laplacian operator is invariant to rotation.

Q. 5 Explain the need of a LoG operator.

Q. 6 Explain the following edge extraction operators :

(a) Sobel (b) Prewitt

(c) Roberts (d) Laplacian

Use these masks on an image of your choice.

Q. 7 A binary image contains straight lines oriented horizontally, vertically, at + 45° and at − 45°. Give a set of 3×3 masks that can be used to detect 1-pixel long breaks in these lines. Assume that the grey level of the lines is 1 and that of the background is 0.

Q. 8 Explain the concept of compass operators.

Q. 9 Explain whether poorly illuminated image can be easily segmented.

Q. 10 Write in detail on connectivity of pixels.

Q. 11 What is the distance transform ?

Q. 12 Explain :

(a) Euclidean distance

(b) City block distance

(c) Chess board distance

(d) D_m distance.

❑❑❑

Note