SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF BIO AND CHEMICAL ENGINEERING
DEPARTMENT OF BIOMEDICAL ENGINEERING

UNIT – I – Introduction to discrete time signals and systems – SBMA1402

INTRODUCTION TO DISCRETE TIME SIGNALS AND SYSTEMS

A signal is a function of independent variables such as time, distance, position, temperature and pressure. A signal carries information, and the objective of signal processing is to extract useful information carried by the signal. Signal processing is concerned with the mathematical representation of the signal and the algorithmic operation carried out on it to extract the information present. For most purposes of description and analysis, a signal can be defined simply as a mathematical function, y where x is the independent variable .

$$y = f(x)$$

signal e.g.: $y=\sin(\omega t)$ is a function of a variable in the time domain and is thus a time signal

$X(\omega)=1/(-m\omega 2+ic\omega+k)$ is a frequency domain signal; An image I(x,y) is in the spatial domain

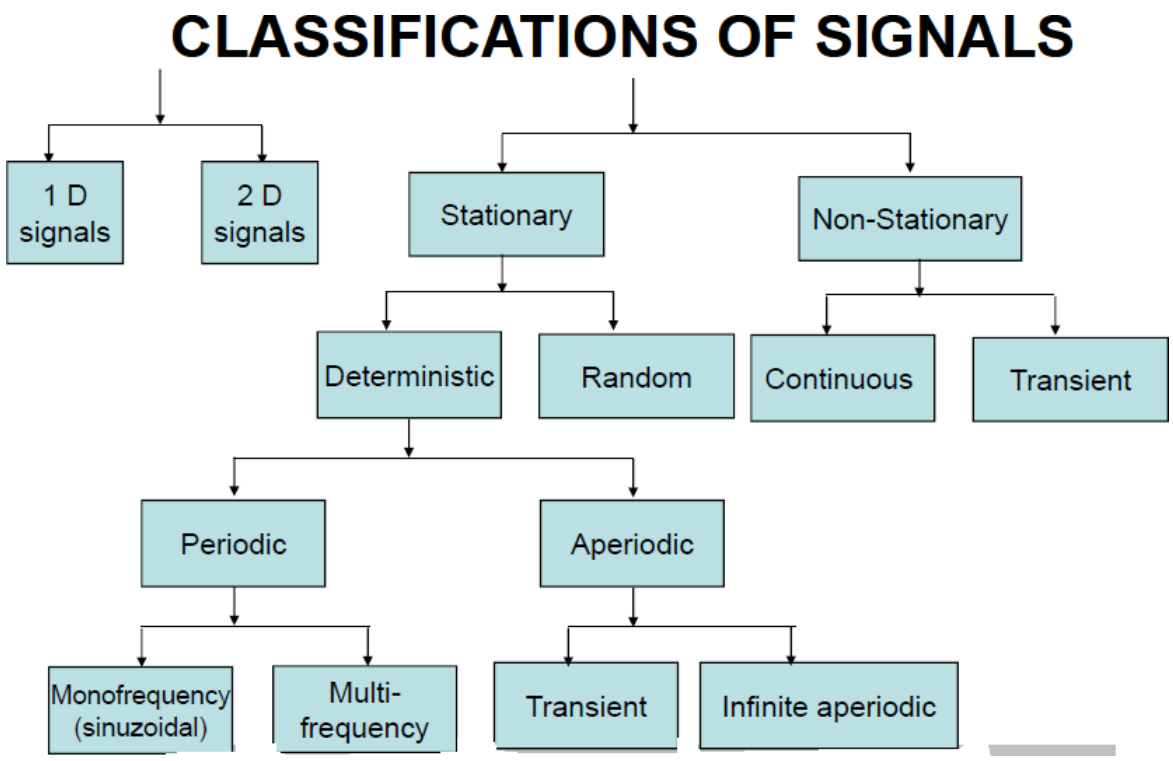


Fig.1:Classification

at t=0, will have the same motions at all time. There is no place for uncertainty here. If we can uniquely specify the value of θ for all time, *i.e.*, we know the underlying functional relationship between t andθ, the motion is **deterministic** or predictable. In other words, a signal that can be uniquely determined by a well defined process such as a mathematical expression or rule is called a **deterministic signal**. The opposite situation occurs if we know all the physics there is to know, but still cannot say what the signal will be at the next time instant-then the signal is **random** or **probabilistic.** In other words, a signal that is generated in a random fashion and can not be predicted ahead of time is called a **random signal.**

1.2 EXAMPLES OF SIGNALS

For a simple pendulum as shown, basic definition is: where θm is the peak amplitude of the motion and ω=$\sqrt{l/g}$ with l the length of the pendulum
and g the acceleration due to gravity. As the system has a constant amplitude (we assume no damping for now), a constant frequency (dictated by physics) and an initial condition (θ=0 when t=0), we know the value of θ(t) for all time

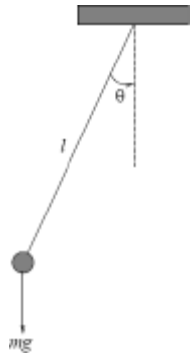
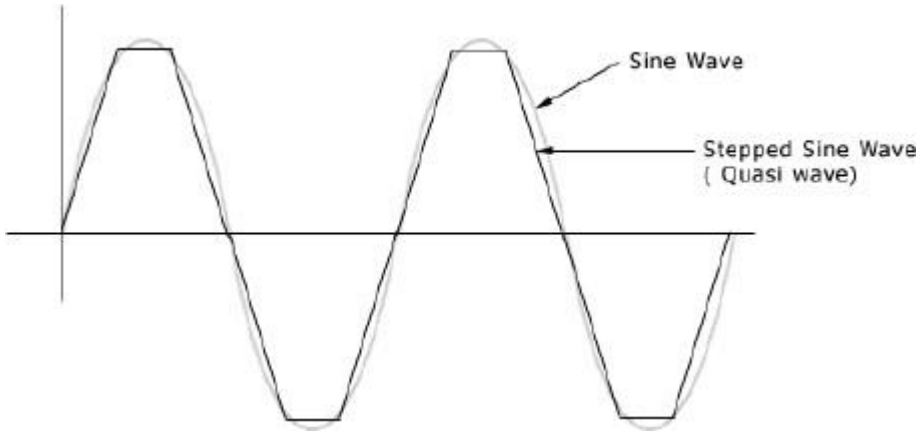


Fig. 2:Typical examples to deterministic signals are sine chirp and digital stepped sine.



1.3 Random signals are characterized by having many frequency components present over

a wide range of frequencies.The amplitude versus time appears to vary rapidly and unsteadily with time. The ‘shhhh’ sound is a good example that is rather easy to observe using a

microphone and oscillloscope. If the sound intensity is constant with time, the random signal is stationary, while if the sound intensity varies with time the signal is nonstationary. One can easily see and hear this variation while making the ‘shhhh’ sound.

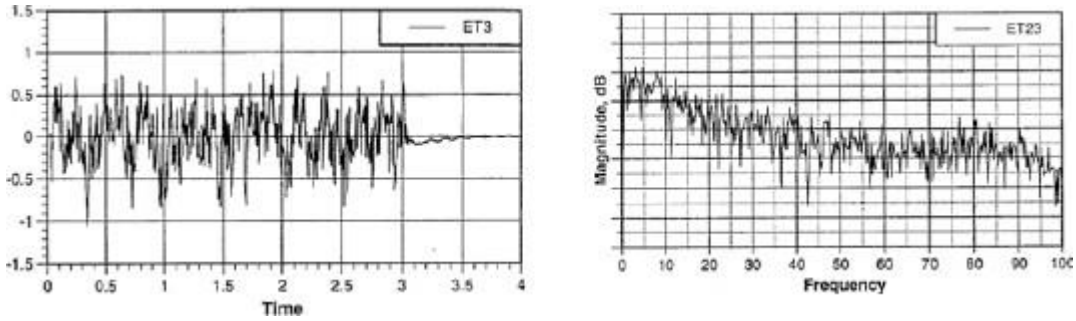


Fig. 3: Random signal

Random signals are characterized by analyzing the statistical characteristics across an ensemble of records. Then, if the process is ergodic, the time (temporal) statistical characteristics are the same as the ensemble statistical characteristics. The word temporal means that a time average definition is used in place of an ensemble statistical definition

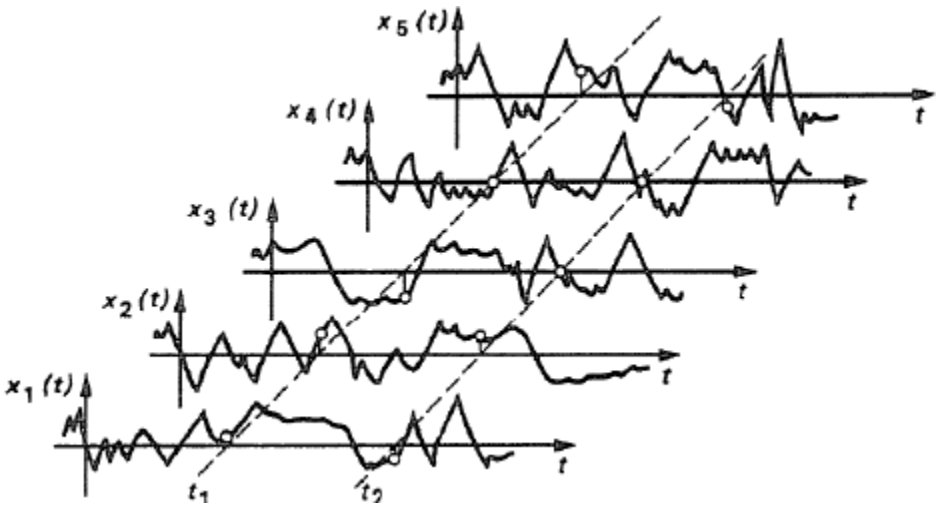


Fig. 4: Transient signal

1.4. Transient signals

may be defined as signals that exist for a finite range of time as shown in the figure. Typical examples are hammer excitation of systems, explosion and shock loading etc. It should be noted that periodicity does not necessarily mean a sinusoidal signal as shown in the figure.

$$S(t) = \sum_{i=1}^{\infty} s_i \sin\left(\frac{2\pi i}{\tau}.t\right)$$

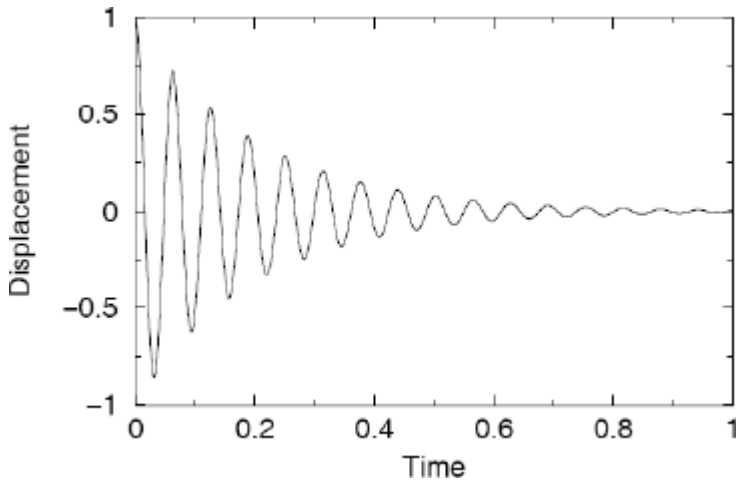
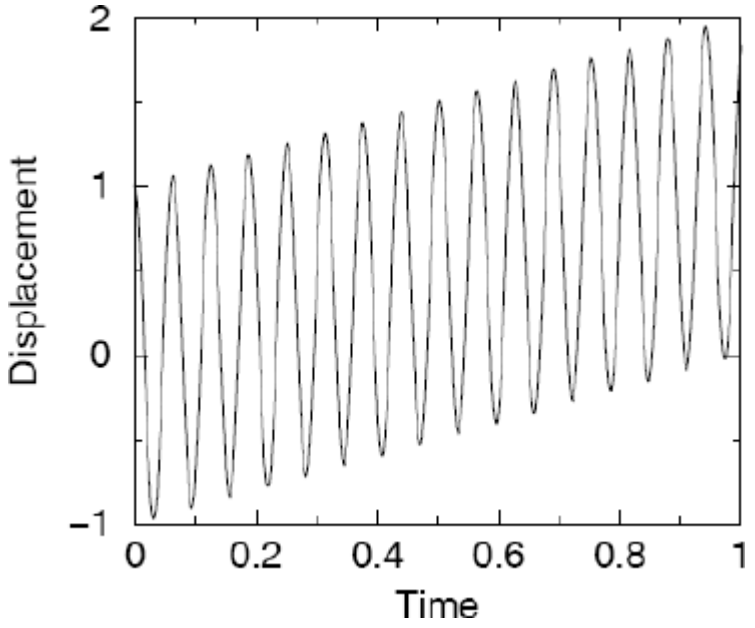
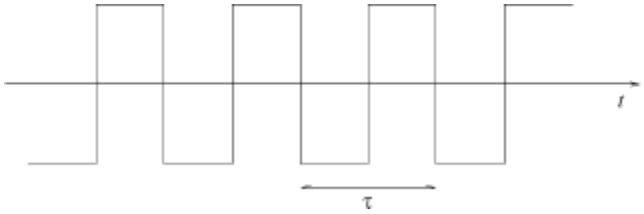
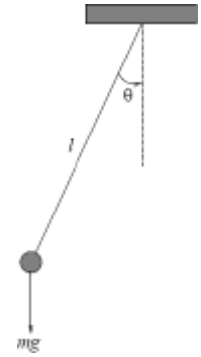


Fig. 6: A signal with a time varying mean is an **aperiodic** signal





For a simple pendulum as shown, if we define the period τ by
, then for the pendulum,and such signals are defined as periodic.



$$\theta(t) = \theta(t+\tau)$$

$$\tau = 2\pi/\omega = 2\pi\sqrt{l/g}$$

Fig. 7: Pendulum

A periodic signal is one that repeats itself in time and is a reasonable model for many real processes, especially those associated with constant speed machinery.

Stationary signals are those whose average properties do not change with time. **Stationary signals** have constant parameters to describe their behaviour.

Nonstationary signals have time dependent parameters. In an engine excited vibration where the engines speed varies with time; the fundamental period changes with time as well as with the corresponding dynamic loads that cause vibration.

1.5 Deterministic Vs Random Signal:

The signals can be further classified as **monofrequency** (sinusoidal) signals and **multifrequency** signals such as the square wave which has a functional form made up of an infinite superposition of different sine waves with periods τ,τ/2,τ/3,…

1 D signals are a function of a single independent variable. The speech signal is an example of a 1 D signal where the independent variable is time.

2D signals are a function of two independent variables. An image signal such as a photograph is an example of a 2D signal where the two independent variables are the two spatial variables

1.6 CONTINUOUS VERSUS DISCRETE SIGNALS

The value of a signal at a specific value of the independent variable is called its **amplitude**.

- The variation of the amplitude as a function of the independent variable is called its **waveform**.
- For a 1 D signal, the independent variable is usually labelled as time. If the independent variable is continuous, the signal is called a **continuous-time signal**. A continuous time signal is defined at every instant of time.
- If the independent variable is discrete, the signal is called a **discrete-time signal**. A discrete time signal takes certain numerical values at specified discrete instants of time, and between

these specified instants of time, the signal is not defined. Hence, a discrete time signal is basically a sequence of numbers.

1.7 ANALOG VERSUS DIGITAL SIGNALS

A continuous-time signal with a continuous amplitude is usually called an **analog signal**.
A speech signal is an example of an analog signal.
A discrete time signal with discrete valued amplitudes represented by a finite number of digits is referred to as a **digital signal**

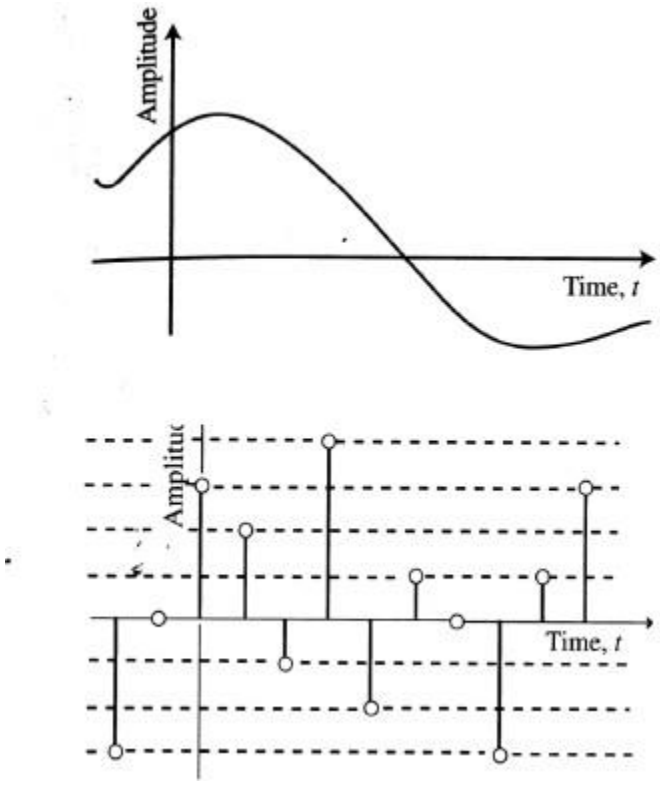


Fig. 8: ADC Conversion

1.8 CONVOLUTIONS

The convolution of *f* and *g* is written *f*∗*g*, using an asterisk or star. It is defined as the integral of the product of the two functions after one is reversed and shifted. As such, it is a particular kind
of integral transform:

$$(f * g)(t) \ \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$
$$= \int_{-\infty}^{\infty} f(t-\tau)\, g(\tau)\, d\tau.$$

While the symbol *t* is used above, it need not represent the time domain. But in that context, the convolution formula can be described as a weighted average of the function *f*(*τ*) at the
moment *t* where the weighting is given by *g*(−*τ*) simply shifted by amount *t*. As *t* changes, the weighting function emphasizes different parts of the input function.

For functions *f*, *g* supported on only $[0, \infty)$ (i.e., zero for negative arguments), the integration limits can be truncated, resulting in

$$(f * g)(t) = \int_0^t f(\tau)\, g(t-\tau)\, d\tau \quad \text{for} \quad f, g : [0, \infty) \to \mathbb{R}$$

In this case, the Laplace transform is more appropriate than the Fourier transform below and boundary terms become relevant.

1.8.1 Circular convolution

When a function g_T is periodic, with period *T*, then for functions, *f*, such that $f * g_T$

$$(f * g_T)(t) \equiv \int_{t_0}^{t_0+T} \left[\sum_{k=-\infty}^{\infty} f(\tau + kT) \right] g_T(t-\tau)\, d\tau,$$

exists, the convolution is also periodic and identical to:

where t_0 is an arbitrary choice. The summation is called a periodic summation of the function *f*.

When gT is a periodic summation of another function, g, then $f*gT$ is known as a *circular* or *cyclic* convolution of f and g.

And if the periodic summation above is replaced by fT, the operation is called a *periodic* convolution of fT and gT

1.9 SAMPLING AND QUANTIZATION

Nearly all data acquisition systems sample data with uniform time intervals. For evenly sampled data, time can be expressed as:

$T = (N\ 1)\Delta t$

where N is the sampling index which is the number of equally spaced samples. For most Fourier analyzers N is restricted to a power of 2.

• The sample rate or the sampling frequency is:

$f = 1 = (\ N - 1)\Delta f$

Sampling frequency is the reciprocal of the time elapsed Δt from one sample to the next.

• The unit of the sampling frequency is cycles per second or Hertz (Hz), if the sampling period is in seconds.

• The sampling theorem asserts that the uniformly spaced discrete samples are a complete representation of the signal if the bandwidth *fmax* is less than half the sampling rate. The sufficient condition for exact reconstructability from samples at a uniform sampling rate *fs* (in samples per unit time) ($fs \geq 2fmax$).

1.9.1 Aliasing

One problem encountered in A/D conversion is that a high frequency signal can be falsely confused as a low frequency signal when sufficient precautions have been avoided.

• This happens when the sample rate is not fast enough for the signal and one speaks of **aliasing**.

• Unfortunately, this problem can not always be resolved by just sampling faster, the signal's frequency content must also be limited.

• Furthermore, the costs involved with postprocessing and data analysis increase with the quantity of data obtained. Data acquisition systems have finite memory, speed and data storage capabilities. Highly oversampling a signal can necessitate shorter sample lengths, longer time on test, more storage medium and increased database management and archiving requirements The central concept to avoid aliasing is that the sample rate must be at least twice the highest frequency component of the signal

(fs≥2fmax).

We define the Nyquist or cut-off frequency

• The concept behind the cut-off frequency is often referred to as $2\Delta t$

Shannon's sampling criterion. Signal components with frequency content above the cut-off frequency are aliased and can not be distinguished from the frequency components below the cut-off frequency. Conversion of analog frequency into digital frequency during sampling is shown in the figure. Continuous signals with a frequency less **than** one-half of the sampling rate are directly converted into the corresponding digital frequency. Above one-half of the sampling rate, aliasing takes place, resulting in the frequency being misrepresented in the digital data. Aliasing always changes a higher frequency into a lower frequency between 0 and

0.5. In addition, aliasing may also change the phase of the signal by 180 degrees.

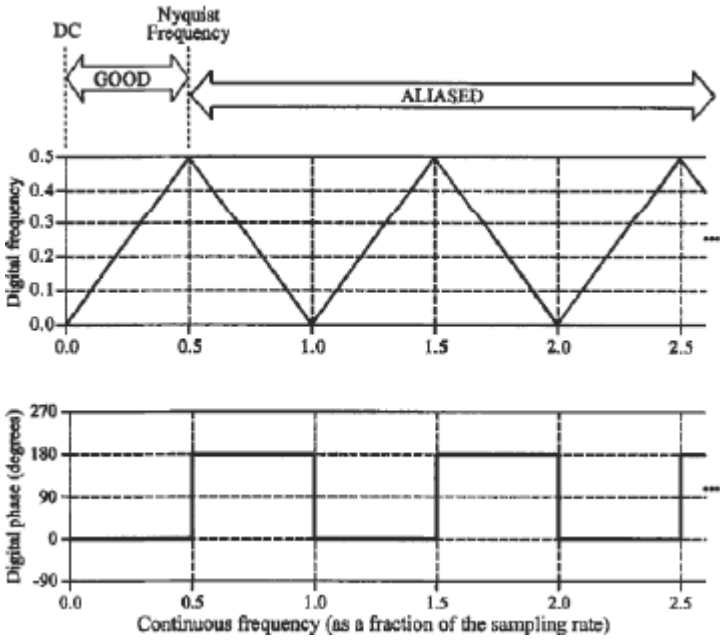
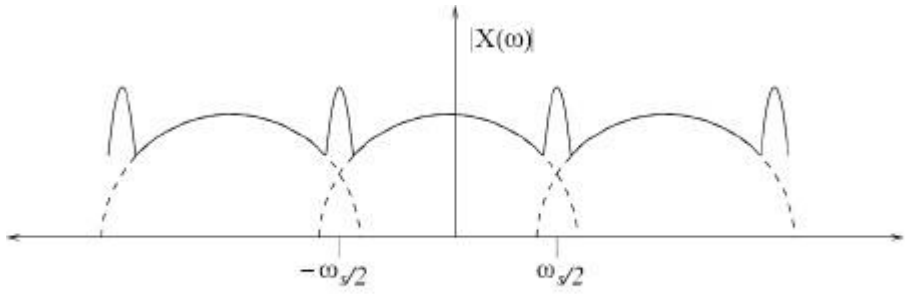


Fig. 9: Aliasing

If any energy in the original signal extends beyond the Nyquist frequency, it is folded back into the Nyquist interval in the spectrum of the sampled signal. This folding is called aliasing.



Spectrum of sampled signal $x_s(t)$ with overlap - aliasing.

1.9.2 Quantization

Quantization is involved to some degree in nearly all digital signal processing, as the process of representing a signal in digital form ordinarily involves rounding. Quantization also forms the core of essentially all lossy compression algorithms. The difference between an input value and its quantized value (such as round-off error) is referred to as **quantization error**. A device
or algorithmic function that performs quantization is called a **quantizer**. An analog-to-

digital converter is an example of a quantizer.

Because quantization is a many-to-few mapping, it is an inherently non-linear and irreversible process (i.e., because the same output value is shared by multiple input values, it is impossible in general to recover the exact input value when given only the output value).

The set of possible input values may be infinitely large, and may possibly be continuous and therefore uncountable (such as the set of all real numbers, or all real numbers within some limited range). The set of possible output values may be finite or countably infinite. The input and output sets involved in quantization can be defined in a rather general way. For example, *vector quantization* is the application of quantization to multi-dimensional (vector-valued) input data

1.10 CONCEPTS OF SIGNAL PROCESSING

In the case of **analog signals**, most signal processing operations are usually carried out in the **time domain**.

• In the case of **discrete time signals**, **both time domain and frequency domain** applications are employed.

• In either case, the desired operations are implemented by a combination of some **elementary operations** such as:

– Simple time domain operations
– Filtering
– Amplitude modulation

The three most basic time-domain signal operations are:

• **Scaling**
• **Delay**
• **Addition**

Scaling is simply the multiplication of a signal by a positive or a negative constant. In the case of analog signals, this operation is usually called **amplification** if the magnitude of the multiplying constant, called **gain**, is greater than one. If the magnitude of the multiplying constant is less than one, the operation is called **attenuation.** Thus, if $x(t)$ is an analog signal, the scaling operation generates a signal $y(t)=\alpha x(t)$, where α is the multiplying constant.

Delay operation generates a signal that is delayed replica of the original signal. For an analog signal $x(t)$, $y(t)=x(t-t0)$ is the signal obtained by delaying $x(t)$ by the amount $t0$, which is assumed to be a positive number. If $t0$ is negative, then it is an **advance** operation

Addition operation generates a new signal by the addition of signals. For instance, $y(t)=x1(t)+x2(t)-x3(t)$ is the signal generated by the addition of the three analog signals $x1(t)$, $x2(t)$ and $x3(t)$.

1.11 TYPICAL APPLICATIONS

The main applications of DSP are

audio signal processing,

sometimes referred to as audio processing, is the intentional alteration of auditory signals, or sound, often through an audio effect oreffects unit. As audio signals may be electronically represented in either digital or analog format, signal processing may occur in either domain. Analog processors operate directly on the electrical signal, while digital processors operate mathematically on the digital representation of that signal.

audio compression

bit-rate reduction involves encoding information using fewer bits than the original representation.[2]Compression can be either lossy or lossless. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by identifying unnecessary information and removing it.[3] The process of reducing the size of a data file is referred to as data compression. In the context of data transmission, it is called source coding (encoding done at the source of the data before it is stored or transmitted) in opposition to channel coding.[4]

digital image processing,

is the use of computer algorithms to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be model in the form of multidimensional systems

speech processing

s the study of speech signals and the processing methods of these signals. The signals are usually processed in a digital representation, so speech processing can be regarded as a special case of digital signal processing, applied to speech signal. Aspects of speech processing includes the acquisition, manipulation, storage, transfer and output of speech signals.

speech recognition,

is the inter-disciplinary sub-field of computational linguistics which incorporates knowledge and research in the linguistics, computer science, and electrical engineering fields to develop methodologies and technologies that enables the recognition and translation of spoken language into text by computers and computerized devices such as those categorized as Smart Technologies and robotics. It is also known as "automatic speech recognition" (ASR), "computer speech recognition", or just "speech to text" (STT).

digital communications, radar, sonar, financial signal processing seismology and biomedicine. Specific examples are speech compression and transmission in digitalmobile phones, room correction of sound in hi-fi and sound reinforcement applications, weather forecasting,economic forecasting, seismic data processing, analysis and control of industrial processes, medical imagingsuch as CAT scans and MRI, MP3 compression, computer graphics, image manipulation, hi- fi loudspeakercrossovers and equalization, and audio effects for use with electric guitar amplifiers

1.12 ADVANTAGES OF DIGITAL SIGNAL PROCESSING COMPARED WITH ANALOG SIGNAL PROCESSING

Accracy

Implimentation of sophisticated

algorithms Storage

Noise reduction

1.13 APPLICATIONS OF SIGNAL PROCESSING IN BIOMEDICAL ENGINEERING

- io signal processing – for electrical signals representing sound, such as speech or music
- Speech signal processing – for processing and interpreting spoken words
- Image processing – in digital cameras, computers and various imaging systems
- Video processing – for interpreting moving pictures
- Wireless communication - waveform generations, demodulation, filtering, equalization
- Control systems
- Array processing – for processing signals from arrays of sensors
- Seismology
- Financial signal processing – analyzing financial data using signal processing techniques, especially for prediction purposes.
- Feature extraction, such as image understanding and speech recognition.
- Quality improvement, such as noise reduction, image enhancement, and echo cancellation.
- (Source coding), including audio compression, image compression, and video compression

ANALYSIS OF DT LTI SYSTEM

The general form of difference equation of a Nth order system is given by

$$1 + \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \quad N > M$$

For input $x(n) = \delta(n)$, we obtain

$$1 + \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k \delta(n-k)$$

For $n > M$, Eq. reduces to homogeneous equation

$$\sum_{k=1}^{N} a_k y(n-k) = 0\,; \quad a_0 = 1.$$

If $N = M$, we have to add an impulse function to the homogeneous solution.

Determine the impulse response $h(n)$ for the system described by the second-order difference equation $y(n) = 0.6y(n-1) - 0.08y(n-2) + x(n)$

Solution

Given

$$y(n) = 0.6y(n-1) - 0.08y(n-2) + x(n)$$

We know the total response

$$y(n) = y_h(n) + y_p(n).$$

For impulse $x(n) = \delta(n)$, the particular solution

$$y_p(n) = 0$$
$$\Rightarrow \quad y(n) = y_h(n).$$

The homogeneous solution can be found by substituting $x(n) = 0$

$$\Rightarrow \quad y(n) - 0.6y(n-1) + 0.08y(n-2) = 0$$

Let the solution

$$y_h(n) = \lambda^n$$

Substituting we obtain

$$\lambda^n - 0.6\lambda^{n-1} + 0.08\lambda^{n-2} = 0$$
$$\lambda^{n-2}[\lambda^2 - 0.6\lambda + 0.08] = 0$$
$$\Rightarrow \quad \lambda^2 - 0.6\lambda + 0.08 = 0$$

The roots of the characteristic equation are

$$\lambda_1 = 0.4; \lambda_2 = 0.2$$

The general form of the solution of the homogeneous equation is

$$y_h(n) = c_1\lambda_1^n + c_2\lambda_2^n$$
$$= c_1(0.4)^n + c_2(0.2)^n$$

$$y(0) = c_1 + c_2$$
$$y(1) = 0.4c_1 + 0.2c_2$$

From the difference equation we have

$$y(0) = 0.6y(-1) - 0.08y(-2) + x(0)$$
$$= 1$$

$$\boxed{\begin{array}{c} y(-1) = y(-2) = 0 \\ x(0) = \delta(0) = 1 \end{array}}$$

$$y(1) = 0.6y(0) - 0.08y(-1) + x(1)$$
$$= 0.6(1) - 0.08(0) + 0$$
$$= 0.6$$
$$\Rightarrow \quad y(0) = 1$$
$$y(1) = 0.6$$

Comparing

$$c_1 + c_2 = 1$$
$$0.4c_1 + 0.2c_2 = 0.6$$

and solving for c_1 and c_2 we get

$$c_1 = 2$$
$$c_2 = -1$$

Substituting the values in Eq. (1.235) yields

$$y(n) = 2(0.4)^n u(n) - (0.2)^n u(n)$$

Determine the impulse response $h(n)$ for the system described by difference equation $y(n) + y(n-1) - 2y(n-2) = x(n-1) + 2x(n-2)$

Solution

Given

$$y(n) + y(n-1) - 2y(n-2) = x(n-1) + 2x(n-2).$$

Since $M = N = 2$, the homogeneous solution includes an impulse term.

The total response is given by

$$y(n) = y_h(n) + y_p(n)$$

For input $x(n) = \delta(n)$, the particular solution $y_p(n) = 0$

$$\Rightarrow y(n) = y_h(n)$$

The homogeneous solution can be found by equating the input terms to zero, that is

$$y(n) + y(n-1) - 2y(n-2) = 0$$

Let the homogeneous solution $y_h(n) = \lambda^n$. Substituting this solution we obtain the characteristic equation

$$\lambda^n + \lambda^{n-1} - 2\lambda^{n-2} = 0$$
$$\lambda^{n-2}[\lambda^2 + \lambda - 2] = 0$$
$$\Rightarrow \quad \lambda^2 + \lambda - 2 = 0$$

Therefore, the roots are $1, -2$ and the general form of the solution to the homogeneous equation is

$$y_h(n) = c_1(1)^n + c_2(-2)^n + A\delta(n)$$

From the difference equation

$$\begin{aligned} y(0) + y(-1) - 2y(-2) &= x(-1) + 2x(-2) \\ y(0) &= 0 \\ y(1) + y(0) - 2y(-1) &= x(0) + 2x(-1) \\ y(1) &= 1 \\ \Rightarrow \quad y(0) &= 0 \\ y(1) &= 1 \\ y(2) &= 1 \end{aligned}$$

Substituting $n = 0, n = 1$ and $n = 2$ in Eq.

$$\begin{aligned} y(0) &= c_1 + c_2 + A \\ y(1) &= c_1 - 2c_2 \\ y(2) &= c_1 - 4c_2 \end{aligned}$$

from which $c_1 = 1; c_2 = 0; A = -1$.

Substituting these values in Eq.

$$\begin{aligned} y(n) &= u(n) - \delta(n) \\ &= u(n-1) \end{aligned}$$

Find the impulse response and step response of a discrete-time linear time invariant system whose difference equation is given by

$$y(n) = y(n-1) + 0.5y(n-2) + x(n) + x(n-1).$$

Solution

Given

$$y(n) = y(n-1) + 0.5y(n-2) + x(n) + x(n-1).$$

For impulse response the particular solution $y_p(n) = 0$.
Therefore

$$y(n) = y_h(n)$$

The homogeneous solution can be obtained by solving the homogeneous equation

$$\lambda^2 - \lambda - 0.5 = 0$$

from which

$$\lambda_1 = 1.366$$
$$\lambda_2 = -0.366$$
$$y_n(n) = c_1(1.366)^n + c_2(-0.366)^n$$

From the difference equation we can find

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

$$y(0) = c_1 + c_2$$
$$y(1) = 1.366c_1 - 0.366c_2$$

Comparing Eq. we get

$$c_1 + c_2 = 1$$
$$1.366c_1 - 0.366c_2 = 2$$
$$\Rightarrow \quad c_1 = 1.366$$
$$c_2 = -0.366$$
$$y(n) = 1.366(1.366)^n - 0.366(-0.366)^n$$

Step response

For step input $x(n) = u(n)$, the particular solution $y_p(n) = ku(n)$. Substituting $x(n)$ and $y_p(n)$ in difference equation

$$ku(n) = ku(n-1) + 0.5ku(n-2) + u(n) + u(n-1)$$

For $n = 2$ where none of the terms vanish we get

$$\begin{aligned} & k = k + 0.5k + 1 + 1 \\ \Rightarrow \quad & k = -4 \end{aligned}$$

Therefore

$$y_p(n) = -4u(n)$$

The total response

$$\begin{aligned} y(n) &= y_h(n) + y_p(n) \\ &= c_1(1.366)^n + c_2(-0.366)^n - 4u(n) \end{aligned}$$

For step input from the difference equation

$$\begin{aligned} y(0) &= 1 \\ y(1) &= 3 \end{aligned}$$

From Eq.

$$\begin{aligned} y(0) &= c_1 + c_2 - 4 \\ y(1) &= 1.366c_1 - 0.366c_2 - 4 \end{aligned}$$

Comparing Eq.

$$\begin{aligned} c_1 + c_2 &= 5 \\ 1.366c_1 - 0.366c_2 &= 7 \\ c_1 &= 5.098 \\ c_2 &= -0.098 \end{aligned}$$

The step response

$$\begin{aligned} y(n) &= 5.098(1.366)^n - 0.098(-0.366)^n - 4 \qquad n \geq 0 \\ &= 5.098(1.366)^n u(n) - 0.098(-0.366)^n u(n) - 4u(n). \end{aligned}$$

Z TRANSFORM

The z-transform of x(n) is denoted by X(z). It is defined as,

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

It is also known as Bilateral Z transform

Here z-is complex variable. x(n) and X(z) is called z-transform pair. It is represented as,

z-transform pair : $x(n) \xleftrightarrow{z} X(z)$

Unilateral or one sided z-transform : It is defined as,

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

Region of Convergence

Definition : ROC is the region where z-transform converges. From definition, it is clear that z-transform is an infinite power series. This series is not convergent all values of z. Hence ROC is useful in mentioning z-transform.

i) ROC gives an idea about values of z for which z-transform can be calculated.

ii) ROC can be used to determine causality of the system.

iii) ROC can be used to determine stability of the system.

PROBLEMS

Determine z-transform of following sequences

i) $x_1(n) = \{1, 2, 3, 4, 5, 0, 7\}$

ii) $x_2(n) = \{1, 2, 3, \underset{\uparrow}{4}, 5, 0, 7\}$

$$x_1(n) = \{1, 2, 3, 4, 5, 0, 7\}$$

i.e. $x_1(0) = 1,\ x_1(1) = 2,\ x_1(2) = 3,\ x_1(3) = 4,\ x_1(4) = 5,\ x_1(5) = 0,\ x_1(6) = 7$

By definition, $X(z) = \sum_{n=0}^{\infty} x(n) z^{-n}$

$$\therefore \quad X_1(z) = \sum_{n=0}^{6} x_1(n) z^{-n}$$

Putting for $x_1(n)$, $= 1 + 2z^{-1} + 3z^{-2} + 4z^{-3} + 5z^{-4} + 0z^{-5} + 7z^{-6}$

$$X_1(z) = 1 + \frac{2}{z} + \frac{3}{z^2} + \frac{4}{z^3} + \frac{5}{z^4} + \frac{7}{z^6}$$

Result : i) $X_1(z)$ is as calculated above.

ii) $X_1(z) = \infty$ for $z = 0$, i.e. $X_1(z)$ is convergent for all values of z, except $z = 0$.

iii) Hence ROC : Entire z-plane except $z = 0$.

ii) $\boldsymbol{x_2(n) = \{1, 2, 3, \underset{\uparrow}{4}, 5, 0, 7\}}$

i.e. $x_2(0) = 4,\ x_2(1) = 5,\ x_2(2) = 0,\ x_2(3) = 7$ *and*

$$x_2(-1) = 3,\ x_2(-2) = 2,\ x_2(-3) = 1$$

$$\therefore \quad X_2(z) = \sum_{n=-3}^{3} x_2(n) z^{-n}$$

Putting for $x_2(n)$, $= 1 \cdot z^3 + 2 \cdot z^2 + 3z^1 + 4z^0 + 5z^{-1} + 0z^{-2} + 7z^{-3}$

$$= z^3 + 2z^2 + 3z + 4 + \frac{5}{z} + \frac{7}{z^3}$$

Result : i) Above equation gives $X_2(z)$.

ii) $X_2(z) = \infty$ for $z = 0$ and $z = \infty$.

iii) Hence ROC : Entire z-plane except $z = 0$ and ∞.

z-transform of $\delta(n)$.

$$\text{We know that } \delta(n) = \begin{cases} 1 & \text{for } n=0 \\ 0 & \text{for } n\neq 0 \end{cases}$$

$$\therefore \quad X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$= \sum_{n=0}^{\infty} \delta(n) z^{-n}$$

$$= 1 \cdot z^0 = 1$$

This is fixed value for any z. Hence ROC will be entire z-plane.

$$\delta(n) \xleftrightarrow{z} 1, \qquad ROC : \text{Entire z-plane}$$

z-transform of unit step sequence, u(n).

$$\text{Unit step sequence, } u(n) = \begin{cases} 1 & \text{for } n\geq 0 \\ 0 & \text{for } n<0 \end{cases}$$

$$\therefore \quad X(z) = \sum_{n=-\infty}^{\infty} u(n) z^{-n}$$

Putting u(n),

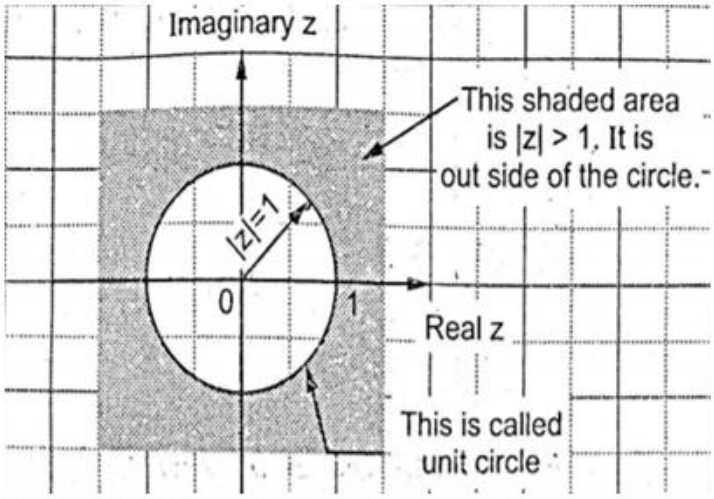
$$= \sum_{n=0}^{\infty} 1 \cdot z^{-n} = \sum_{n=0}^{\infty} (z^{-1})^n$$

$$= 1 + (z^{-1}) + (z^{-1})^2 + (z^{-1})^3 + (z^{-1})^4 + \cdots$$

Here use, $1 + A + A^2 + A^3 + A^4 + \cdots = \frac{1}{1-A}, |A| < 1$. Then above equation will be,

$$X(z) = \frac{1}{1-z^{-1}}, |z^{-1}| < 1$$

$$u(n) \xleftrightarrow{z} \frac{1}{1-z^{-1}}, \qquad ROC : |z| > 1$$



z-transform of right hand sided sequence $x(n) = a^n u(n)$.

By definition of z-transform, $X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$

$$= \sum_{n=-\infty}^{\infty} a^n u(n) z^{-n}$$

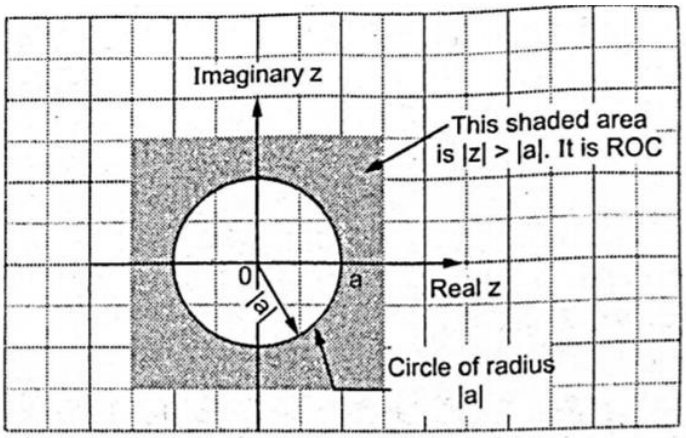
$$= \sum_{n=0}^{\infty} a^n z^{-n} \quad \textit{since } u(n) = 1 \textit{ for } n = 0 \textit{ to } \infty$$

$$= \sum_{n=0}^{\infty} (az^{-1})^n$$

$$= 1 + (az^{-1}) + (az^{-1})^2 + (az^{-1})^3 + (az^{-1})^4 + \cdots$$

Here use $1 + A + A^2 + A^3 + \cdots = \frac{1}{1-A}$, $|A| < 1$. Then above equation will be,

ROC of the right hand sided sequence (i.e. causal sequence) is outside the circle.



$$a^n u(n) \xleftrightarrow{z} \frac{1}{1-az^{-1}}, \qquad ROC : |z| > |a|$$

Z transform of left hand sequence

$$x(n) = -a^n u(-n-1)$$

Here $x(n) = \left\{ \begin{matrix} -a^n & for\ n \le -1 \\ 0 & for\ n \ge 0 \end{matrix} \right\} u(-n-1) = 1$ for n = − 1 to − ∞

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$= \sum_{n=-\infty}^{-1} -a^n z^{-n}$$

$n = -l$, $X(z) = \sum_{l=\infty}^{1} a^{-l} z^{l}$

Let

$$= -\sum_{l=1}^{\infty} (a^{-1} z)^l$$

$$= -\left\{ (a^{-1} z) + (a^{-1} z)^2 + (a^{-1} z)^3 + (a^{-1} z)^4 + \cdots \right\}$$

$$= -(a^{-1} z)\left\{ 1 + a^{-1} z + (a^{-1} z)^2 + (a^{-1} z)^3 + (a^{-1} z)^4 + \cdots \right\}$$

For the term in bracket use,

$$1+A+A^2+A^3+\cdots = \frac{1}{1-A}, |A| < 1. \text{ i.e.,}$$

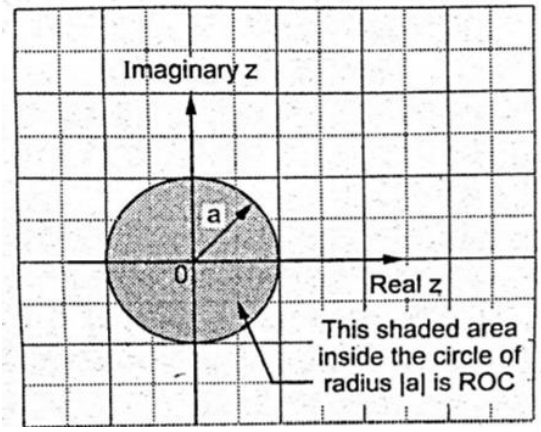
$$X(z) = -(a^{-1}z)\cdot\frac{1}{1-a^{-1}z}, |a^{-1}z| < 1$$

$$= \frac{1}{1-az^{-1}}, |a^{-1}z| < 1$$

Here $|a^{-1}z| < 1$ is equal to $|z| < |a|$. This ROC is the area that lies inside the circle of radius $|a|$. It is shown in Fig.

ROC of left sided sequence

is an area inside the circle.



Z transform of both sided sequence

$$x(n) = a^n u(n) + b^n u(-n-1)$$

Here let $x_1(n) = a^n u(n)$ and $x_2(n) = b^n u(-n-1)$

$$x(n) = x_1(n) + x_2(n)$$

$$X(z) = \sum_{n=-\infty}^{\infty} [x_1(n) + x_2(n)] z^{-n}$$

$$= \sum_{n=-\infty}^{\infty} x_1(n) z^{-n} + \sum_{n=-\infty}^{\infty} x_2(n) z^{-n}$$

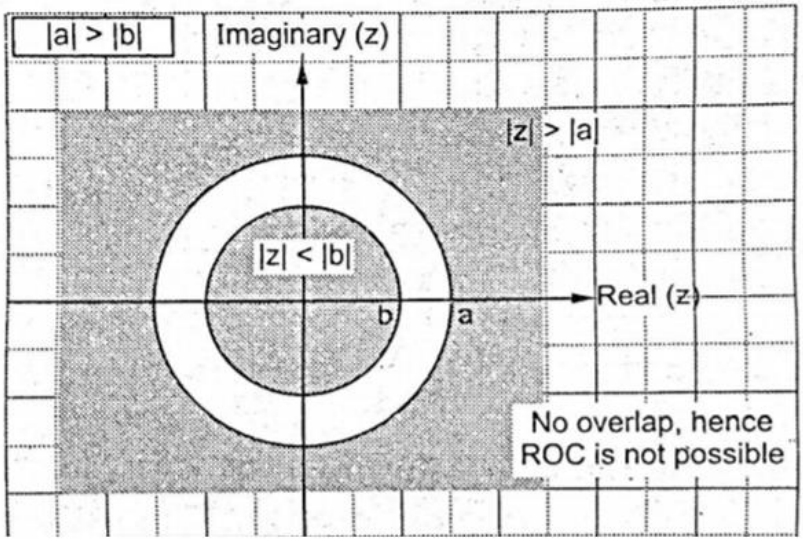
From the previous problem's result

$$X(z) = \frac{1}{1-az^{-1}} + \frac{1}{1-bz^{-1}}, \quad ROC : |z| > |a| \text{ and } |z| < |b|$$

i.e. $|a| < |z| < |b|$

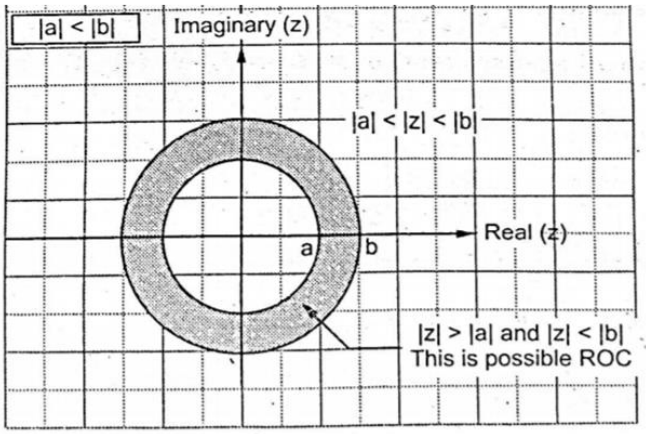
When |a|>|b|

As shown Fig. there is no overlap between the shaded areas for |z|>|a| and |z|<|a|. Hence both the terms of X(z) do not converge simultaneously. Therefore ROC is not possible.



When |a|<|b|

For this case, as shown in Fig. the shaded area shows the overlap of |z|>|a| and |z|<|b|. This area is |a|<|z|<|b|. In this area both the terms of X(z) converge simultaneously. Hence the ring shown by |a|<|z|<|b| is ROC of X(z).



PROPERTIES OF ROC

Property 1 : The ROC for a finite duration sequence includes entire z-plane, except z=0, and/or |z| = ∞.

Proof : Consider the finite duration sequence $x(n) = \{1\ 2\ \underset{\uparrow}{1}\ 2\}$

$$\therefore \quad X(z) = 1\cdot z^2 + 2\cdot z + 1z^0 + 2z^{-1} = z^2 + 2z + 1 + \frac{2}{z}$$

Here X(z) = ∞ for z = 0 and ∞. This proves first property.

Property 2 : ROC does not contain any poles.

Proof : The z-transform of $a^n u(n)$ is calculated as,

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

$$= \frac{z}{z-a} \quad \therefore ROC : |z| > |a|$$

This function has pole at z = a. Note that ROC is |z|>|a|. This means poles do not lie in ROC. Actually X(z) = ∞ at poles by definition of pole.

Property 3 : ROC is the ring in the z-plane centered about origin.

Proof : Consider $a^n u(n) \xleftrightarrow{z} \frac{1}{1-az^{-1}}$, ROC : $|z|>|a|$

or $-a^n u(-n-1) \xleftrightarrow{z} \frac{1}{1-az^{-1}}$, ROC : $|z|<|a|$

Here observe that $|z|$ is always a circular region (ring) centered around origin.

Property 4 : ROC of causal sequence (right hand sided sequence) is of the form $|z|>r$.

Proof : Consider right hand sided sequence $a^n u(n)$. It's ROC is $|z| > |a|$. Thus the ROC of right hand sided sequence is of the form of $|z|>r$ where 'r' is the radius of the circle.

Property 5 : ROC of left sided sequence is of the form $|z|<r$.

Proof : Consider left sided sequence $-a^n u(-n-1)$. Its ROC is $|z|<|a|$. Thus the ROC of left sided sequence is inside the circle of radius 'r'.

Property 6 : ROC of two sided sequence is the concentric ring in z-plane.

Proof : We know that ROC of $x(n) = a^n u(n) + b^n u(-n-1)$ is $|a|<|z|<|b|$, which is the concentric ring

PROPERTIES OF Z TRANSFORM

Linearity

$$a_1 x_1(n) + a_2 x_2(n) \xleftrightarrow{z} a_1 X_1(z) + a_2 X_2(z)$$

Proof :

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$= \sum_{n=-\infty}^{\infty} [a_1 x_1(n) + a_2 x_2(n)] z^{-n}$$

$$= \sum_{n=-\infty}^{\infty} a_1 x_1(n) z^{-n} + \sum_{n=-\infty}^{\infty} a_2 x_2(n) z^{-n}$$

$$= a_1 \sum_{n=-\infty}^{\infty} x_1(n) z^{-n} + a_2 \sum_{n=-\infty}^{\infty} x_2(n) z^{-n}$$ Since a_1 and a_2 are constants

$$= a_1 X_1(z) + a_2 X_2(z)$$

Time Shifting

$$x(n-k) \xleftrightarrow{z} z^{-k} X(z)$$

Proof : $Z\{x(n-k)\} = \sum_{n=-\infty}^{\infty} x(n-k) z^{-n}$

Let $n-k = m$. Hence $n = k+m$ and $m = -\infty$ to $+\infty$. i.e.,

$$Z\{x(n-k)\} = \sum_{m=-\infty}^{\infty} x(m) z^{-(k+m)}$$

$$= \sum_{m=-\infty}^{\infty} x(m) z^{-k} \cdot z^{-m} = z^{-k} \sum_{m=-\infty}^{\infty} x(m) z^{-m}$$

$$= z^{-k} X(z)$$

Scaling in z-Domain

Let $x(n) \xleftrightarrow{z} X(z), \quad ROC : r_1 < |z| < r_2$

then $$a^n x(n) \xleftrightarrow{z} X\left(\frac{z}{a}\right), \qquad \text{ROC} : |a| r_1 < |z| < |a| r_2$$

Proof : $Z\{a^n x(n)\} = \sum_{n=-\infty}^{\infty} a^n x(n) z^{-n}$

$$= \sum_{n=-\infty}^{\infty} x(n) (a^{-1} z)$$

$$= X(a^{-1} z)$$

$$= X\left(\frac{z}{a}\right), \qquad ROC : r_1 < \left|\frac{z}{a}\right| < r_2 \ \text{i.e.} \ |a| r_1 < |z| < |a| r_2$$

Time Reversal

Let $x(n) \xleftrightarrow{z} X(z)$, $ROC : r_1 < |z| < r_2$

then

$$\boxed{x(-n) \xleftrightarrow{z} X\left(z^{-1}\right), \qquad \text{ROC} : \frac{1}{r_2} < |z| < \frac{1}{r_1}}$$

Proof : $Z\{x(-n)\} = \sum_{n=-\infty}^{\infty} x(-n) z^{-n}$

with $n = -m$, $= \sum_{m=\infty}^{-\infty} x(m) z^{m} = \sum_{m=-\infty}^{\infty} x(m) (z^{-1})^{-m}$

$= X(z^{-1})$ $\quad ROC : r_1 < |z^{-1}| < r_2$ *i.e.* $\frac{1}{r_2} < |z| < \frac{1}{r_1}$

Differentiation in z-Domain

$$\boxed{n\,x(n) \xleftrightarrow{z} -z \frac{d}{dz} X(z)}$$

Proof : $X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$

$\therefore$ $\frac{d}{dz} X(z) = \sum_{n=-\infty}^{\infty} \frac{d}{dz} [x(n) z^{-n}] = \sum_{n=-\infty}^{\infty} x(n) \frac{d}{dz} z^{-n}$

$= \sum_{n=-\infty}^{\infty} x(n) \cdot (-n) \cdot z^{-n-1} = - \sum_{n=-\infty}^{\infty} n\,x(n) z^{-n} \cdot z^{-1}$

$= -z^{-1} \sum_{n=-\infty}^{\infty} [n\,x(n)] z^{-n} = -z^{-1} \cdot Z\{n\,x(n)\}$

or $Z\{n\,x(n)\} = -z \frac{d}{dz} X(z)$, $\quad ROC$: Same as that of $x(n)$

Convolution in Time Domain

$$x_1(n) * x_2(n) \xleftrightarrow{z} X_1(z) \cdot X_2(z)$$

Proof : $x_1(n) * x_2(n) = \sum_{k=-\infty}^{\infty} x_1(k)\, x_2(n-k)$

$$\therefore \quad Z\{x_1(n) * x_2(n)\} = \sum_{n=-\infty}^{\infty} \left[\sum_{k=-\infty}^{\infty} x_1(k)\, x_2(n-k) \right] z^{-n}$$

Interchanging orders of summation,

$$= \sum_{k=-\infty}^{\infty} x_1(k) \left\{ \sum_{n=-\infty}^{\infty} x_2(n-k)\, z^{-n} \right\}$$

Since $x_2(n-k) \xleftrightarrow{z} z^{-k} X_2(z)$,

$$= \sum_{k=-\infty}^{\infty} x_1(k) \left\{ z^{-k} X_2(z) \right\}$$

$$= \left\{ \sum_{k=-\infty}^{\infty} x_1(k)\, z^{-k} \right\} \cdot X_2(z) = X_1(z) \cdot X_2(z)$$

Correlation of Two Sequences

$$\sum_{n=-\infty}^{\infty} x_1(n)\, x_2(n-l) \xleftrightarrow{z} X_1(z) X_2(z^{-1})$$

Proof : Correlation of two sequences is given as,

$$r_{x_1 x_2}(l) = \sum_{n=-\infty}^{\infty} x_1(n)\, x_2(n-l)$$

$$= \sum_{n=-\infty}^{\infty} x_1(n)\, x_2[-(l-n)] = x_1(l) * x_2(-l)$$

$$\therefore \quad Z\left\{\sum_{n=-\infty}^{\infty} x_1(n)\, x_2(n-l)\right\} = Z\{x_1(l) * x_2(-l)\}$$

$$= Z\{x_1(l)\} \cdot Z\{x_2(-l)\}$$

$$= X_1(z) \cdot X_2(z^{-1})$$

Multiplication of Two Sequences

$$x_1(n)\, x_2(n) \xleftrightarrow{z} \frac{1}{2\pi j} \oint_c X_1(v) X_2\left(\frac{z}{v}\right) v^{-1}\, dv$$

Here 'c' is the closed contour. It encloses the origin and lies in the ROC which is common to both $X_1(v)$ and $X_2\left(\frac{1}{v}\right)$.

Proof : Inverse z-transform is given as, $x(n) = \frac{1}{2\pi j} \oint X(v) v^{n-1}\, dv$

Let $x(n) = x_1(n)\, x_2(n)$

Putting inverse z-transform of $x_1(n)$ in above equation,

$$x(n) = \frac{1}{2\pi j} \oint_c X_1(v) v^{n-1}\, dv \cdot x_2(n)$$

$$\therefore \quad X(z) = \sum_{n=-\infty}^{\infty} \left\{\frac{1}{2\pi j} \oint_c X_1(v) v^{n-1}\, dv \cdot x_2(n)\right\} z^{-n}$$

Interchanging the order of integration and summation,

$$X(z) = \frac{1}{2\pi j} \oint_c X_1(v) \sum_{n=-\infty}^{\infty} v^n \cdot v^{-1}\, x_2(n) z^{-n}\, dv$$

$$= \frac{1}{2\pi j} \oint_c X_1(v) \left\{\sum_{n=-\infty}^{\infty} x_2(n) \left(\frac{z}{v}\right)^{-n}\right\} v^{-1}\, dv$$

$$= \frac{1}{2\pi j} \oint_c X_1(v) \cdot X_2\left(\frac{z}{v}\right) \cdot v^{-1}\, dv$$

Conjugation of a Complex Sequence

$$x^*(n) \xleftrightarrow{z} X^*(z^*)$$

Proof : $Z\{x^*(n)\} = \sum_{n=-\infty}^{\infty} x^*(n) z^{-n} = \sum_{n=-\infty}^{\infty} [x(n)(z^*)^{-n}]^*$

$$= \left[\sum_{n=-\infty}^{\infty} x(n)(z^*)^{-n}\right]^*$$

$$= [X(z^*)]^* = X^*(z^*)$$

z-Transform of Real Part of a Sequence

$$\text{Re}[x(n)] \xleftrightarrow{z} \frac{1}{2}[X(z) + X^*(z^*)]$$

Proof : $x(n) = Re[x(n)] + j\ Im\ [x(n)]$ *and* $x^*(n) = Re\ [x(n)] - j\ Im\ [x(n)]$

$\therefore$ $Re[x(n)] = \frac{1}{2}[x(n) + x^*(n)]$

$\therefore$ $Z\{\text{Re}[x(n)] = Z\left\{\frac{1}{2}[x(n) + x^*(n)]\right\}$

$$= \frac{1}{2}\{Z[x(n)] + Z[x^*(n)]\}$$

$$= \frac{1}{2}[X(z) + X^*(z^*)]$$

z-Transform of Imaginary Part of Sequence

$$\text{Im}[x(n)] \xleftrightarrow{z} \frac{1}{2j}[X(z) - X^*(z^*)]$$

Proof : $\text{Im}[x(n)] = \frac{1}{2j}[x(n) - x^*(n)]$

$$Z\{\text{Im}[x(n)]\} = Z\left\{\frac{1}{2j}[x(n) - x^*(n)]\right\} = \frac{1}{2j}\{Z[x(n) - Z[x^*(n)]\}$$

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

Parseval's Relation

$$\sum_{n=-\infty}^{\infty} x_1(n)\, x_2^*(n) = \frac{1}{2\pi j}\oint_c X_1(v) X_2^*\left(\frac{1}{v^*}\right) v^{-1}\, dv$$

Proof : Inverse z-transform of $X_1(z)$ is, $x_1(n) = \frac{1}{2\pi j}\oint_c X_1(v)\, v^{n-1}\, dv$

$$\therefore \quad \sum_{n=-\infty}^{\infty} x_1(n)\, x_2^*(n) = \sum_{n=-\infty}^{\infty} \frac{1}{2\pi j}\oint_c X_1(v)\, v^{n-1}\, dv\, x_2^*(n)$$

$$= \frac{1}{2\pi j}\oint_c X_1(v)\left\{\sum_{n=-\infty}^{\infty} x_2^*(n)\, v^{n-1}\right\} dv$$

Here $v^{n-1} = v^n \cdot v^{-1} = (v^{-1})^{-n} \cdot v^{-1} = \left(\frac{1}{v}\right)^{-n} \cdot v^{-1}$ then above equation will be,

$$\sum_{n=-\infty}^{\infty} x_1(n)\, x_2^*(n) = \frac{1}{2\pi j}\oint_c X_1(v)\left\{\sum_{n=-\infty}^{\infty} x_2^*(n)\cdot\left(\frac{1}{v}\right)^{-n} \cdot v^{-1}\right\} dv$$

$$= \frac{1}{2\pi j}\oint_c X_1(v)\left[\sum_{n=-\infty}^{\infty} x_2(n)\cdot\left(\frac{1}{v^*}\right)^{-n}\right]^* v^{-1} dv$$

$$= \frac{1}{2\pi j}\oint_c X_1(v)\left[X_2\left(\frac{1}{v^*}\right)\right]^* v^{-1} dv$$

$$= \frac{1}{2\pi j}\oint_c X_1(v) X_2^*\left(\frac{1}{v^*}\right) v^{-1} dv$$

Initial Value Theorem

$$x(0) = \lim_{z \to \infty} X(z)$$

Proof : z-transform of a causal sequence is given as,

$$X(z) = \sum_{n=0}^{\infty} x(n) z^{-n}, \text{ since } x(n) = 0 \text{ for } n < 0$$

$$= x(0) + x(1)z^{-1} + x(2)z^{-2} + x(3)z^{-3} + \cdots$$

$$\therefore \quad \lim_{z \to \infty} X(z) = \lim_{z \to \infty} x(0) + \lim_{z \to \infty} x(1)z^{-1} + \lim_{z \to \infty} x(2)z^{-2} + \cdots$$

$$= x(0) + 0 + 0 + 0 + 0 \cdots$$

$$\therefore \quad x(0) = \lim_{z \to \infty} X(z)$$

PROBLEMS

$$x_1(n) = \delta(n-k)$$

$\delta(n) \xleftrightarrow{z} 1$, ROC : entire z-plane

$x(n-k) \xleftrightarrow{z} z^{-k} X(z)$, By time delay property.

$$Z\{\delta(n-k)\} = z^{-k} Z\{\delta(n)\}$$

$$= z^{-k} \cdot 1 = z^{-k}, \quad \textit{ROC : entire z-plane except } z = 0.$$

$$x_2(n) = \delta(n+k)$$

$$Z\{\delta(n+k)\} = z^{k} Z\{\delta(n)\}$$

$$= z^{k} \cdot 1 = z^{k}, \quad \textit{ROC : entire z-plane except } z = \infty$$

$x_3(n) = u(-n)$

$x(n) \xleftrightarrow{z} X(z)$, ROC : $r_1 < |z| < r_2$

$x(-n) \xleftrightarrow{z} X(z^{-1})$, ROC : $\frac{1}{r_2} < |z| < \frac{1}{r_1}$, By time reversal property

$u(n) \xleftrightarrow{z} \frac{1}{1-z^{-1}}$, ROC : $|z| > 1$. Here $r_1 = 1$

$\therefore u(-n) \xleftrightarrow{z} \frac{1}{1-z}$, ROC : $|z| < 1$, By time reversal property

$x_4(n) = n a^n u(n)$

$$Z\{a^n u(n)\} = \frac{1}{1-az^{-1}}, \quad ROC : |z| > |a|$$

And $Z\{n\, x(n)\} = -z\frac{d}{dz}X(z)$, differentiation in z-domain property

$$\therefore \quad Z\{n \cdot a^n u(n)\} = -z\frac{d}{dz}\frac{1}{1-az^{-1}}, \quad Here\ x(n) = a^n u(n)$$

$$= -z \cdot \frac{(1-az^{-1})\frac{d}{dz}1 - 1 \cdot \frac{d}{dz}(1-az^{-1})}{(1-az^{-1})^2}$$

$$= -z \cdot \frac{0 + a(-1)z^{-2}}{(1-az^{-1})^2}$$

$$= \frac{az^{-1}}{(1-az^{-1})^2}, \quad ROC : |z| > |a|$$

Thus, $\boxed{na^n u(n) \xleftrightarrow{z} \frac{az^{-1}}{(1-az^{-1})^2}, \quad ROC : |z| > |a|}$

$$x(n) = \cos(\omega_0 n)\, u(n)$$

$$= \frac{e^{j\omega_0 n} + e^{-j\omega_0 n}}{2} u(n)$$

$$X(z) = Z\left\{\frac{e^{j\omega_0 n} + e^{-j\omega_0 n}}{2}\right\} u(n)$$

$$= \frac{1}{2} Z\left\{e^{j\omega_0 n} u(n)\right\} + \frac{1}{2} Z\left\{e^{-j\omega_0 n} u(n)\right\}$$

Here use $a^n u(n) \xleftrightarrow{z} \frac{1}{1-az^{-1}}$, ROC $|z|>|a|$ i.e.,

$$X(z) = \frac{1}{2}\cdot\frac{1}{1-e^{j\omega_0 z^{-1}}} + \frac{1}{2}\frac{1}{1-e^{-j\omega_0 z^{-1}}}, \quad ROC : |z|>|e^{j\omega_0}| \text{ and } |z|>|e^{-j\omega_0}|$$

$$X(z) = \frac{1}{2}\left\{\frac{1}{1-e^{j\omega_0 z^{-1}}} + \frac{1}{1-e^{-j\omega_0 z^{-1}}}\right\} \quad ROC : |z|>1$$

$$= \frac{1}{2}\left\{\frac{1-e^{-j\omega_0 z^{-1}} + 1-e^{j\omega_0 z^{-1}}}{(1-e^{j\omega_0 z^{-1}})(1-e^{-j\omega_0 z^{-1}})}\right\}$$

$$= \frac{1}{2}\left\{\frac{2-z^{-1}(e^{j\omega_0} + e^{-j\omega_0})}{1-z^{-1}(e^{j\omega_0} + e^{-j\omega_0}) + z^{-2}}\right\} = \frac{1}{2}\left\{\frac{2-z^{-1}\cdot 2\cos\omega_0}{1-z^{-1}\cdot 2\cos\omega_0 + z^{-2}}\right\}$$

$$= \frac{1-z^{-1}\cos\omega_0}{1-2z^{-1}\cos\omega_0 + z^{-2}}, \quad ROC : |z|>1$$

$$x(n) = \sin\omega_0 n\, u(n)$$

$$= \frac{e^{j\omega_0 n} - e^{-j\omega_0 n})}{2j} u(n)$$

$$X(z) = Z\left\{\frac{e^{j\omega_0 n} - e^{-j\omega_0 n})}{2j}\right\} u(n)$$

$$= \frac{1}{2j}\left[Z\{e^{j\omega_0 n}\, u(n)\} - Z\{e^{-j\omega_0 n}\, u(n)\}\right]$$

$$= \frac{1}{2j}\left[\frac{1}{1-e^{j\omega_0 z^{-1}}} - \frac{1}{1-e^{-j\omega_0 z^{-1}}}\right],$$

$$ROC : |z| > |e^{j\omega_0}| \text{ and } |z| > |e^{-j\omega_0}|$$

$$i.e.\ |z| > 1$$

$$= \frac{1}{2j}\left[\frac{1-e^{-j\omega_0 z^{-1}} - 1 + e^{j\omega_0 z^{-1}}}{(1-e^{j\omega_0 z^{-1}})(1-e^{-j\omega_0 z^{-1}})}\right], \quad ROC : |z| > 1$$

$$= \frac{1}{2j}\left[\frac{(e^{j\omega_0} - e^{-j\omega_0})z^{-1}}{1 - z^{-1}(e^{j\omega_0} + e^{-j\omega_0}) + z^{-2}}\right]$$

$$= \frac{1}{2j}\left[\frac{2j \sin\omega_0\, z^{-1}}{1 - z^{-1} \cdot 2\cos\omega_0 + z^{-2}}\right]$$

$$= \frac{z^{-1}\sin\omega_0}{1 - 2z^{-1}\cos\omega_0 + z^{-2}}, \quad ROC : |z| > 1$$

$x(n) = a^n \cos(\omega_0 n)\, u(n)$

Let $x_1(n) = \cos(\omega_0 n)\, u(n)$, *hence* $X_1(z) = \dfrac{1 - z^{-1}\cos\omega_0}{1 - 2z^{-1}\cos\omega_0 + z^{-2}}$

From above equations, $x(n) = a^n x_1(n)$

$\therefore$ $X(z) = Z\{a^n x_1(n)\}$

$= X_1\left(\dfrac{z}{a}\right)$, *ROC* : $|a|r_1 < |z| < |a|r_2$, *By scaling in z-domain*

Replacing z by $\dfrac{z}{a}$ in $X_1(z)$, $= \dfrac{1 - \left(\dfrac{z}{a}\right)^{-1}\cos\omega_0}{1 - 2\left(\dfrac{z}{a}\right)^{-1}\cos\omega_0 + \left(\dfrac{z}{a}\right)^{-2}}$, ROC : $|z| > 1\,|a|$ i.e. $|z| > |a|$

$x(n) = a^n \sin(\omega_0 n)\, u(n)$

Let $x_1(n) = \sin(\omega_0 n)\, u(n)$, *hence* $X_1(z) = \dfrac{z^{-1}\sin\omega_0}{1 - 2z^{-1}\cos\omega_0 + z^{-2}}$

From above equations, $x(n) = a^n x_1(n)$

$\therefore$ $X(z) = z\{a^n x_1(n)\}$

$= X_1\left(\dfrac{z}{a}\right)$, *ROC* : $|a|r_1 < |z| < |a|r_2$, *By scaling in z-domain*

Replacing z by $\dfrac{z}{a}$ in $X_1(z)$, $= \dfrac{\left(\dfrac{z}{a}\right)^{-1}\sin\omega_0}{1 - 2\left(\dfrac{z}{a}\right)^{-1}\cos\omega_0 + \left(\dfrac{z}{a}\right)^{-2}}$, ROC : $|z| > 1\,|a|$ i.e. $|z| > |a|$

INVERSE Z TRANSFORM

The inverse z-transform can be obtained by,

i) Power series expansion

ii) Partial fraction expansion

iii) Contour integration.

iv) Convolution method

Inverse z-Transform using Power Series Expansion

By definition z-transform of the sequence x(n) is given as,

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$= \cdots + x(-2)z^{2} + x(-1)z + x(0) + x(1)z^{-1} + x(2)z^{-2} + \cdots$$

From above expansion of z-transform, the sequence x(n) can be obtained as,

$$x(n) = \{\cdots, x(-2), x(-1), x(0), x(1), x(2), \cdots\}$$

The power series expansion can be obtained directly or by long division method.

Determine inverse z-transform of the following :

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

$$1 - az^{-1} \overline{)\,1} \quad \leftarrow \text{Negative power of 'z'}$$

$$1 + az^{-1} + a^2 z^{-2} + a^3 z^{-3}$$

$$1 - az^{-1}$$

$$az^{-1}$$

$$az^{-1} - a^2 z^{-2}$$

$$a^2 z^{-2}$$

$$a^2 z^{-2} - a^3 z^{-3}$$

$$a^3 z^{-3}$$

$$a^3 z^{-3} - a^4 z^{-4}$$

$$a^4 z^{-4} \ \ldots$$

Thus we have, $X(z) = \dfrac{1}{1-az^{-1}} = 1 + az^{-1} + a^2 z^{-2} + a^3 z^{-3} + \cdots$

Taking inverse z-transform, $x(n) = \{1, a, a^2, a^3, \cdots\}$

$$= a^n u(n)$$

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

$$-az^{-1}+1 \;\overline{)\; 1}$$

$$-a^{-1}z - a^{-2}z^2 - a^{-3}z^3 - a^{-4}z^4 \leftarrow \textit{Positive powers of 'z'}$$

$$1 - a^{-1}z$$

$$a^{-1}z$$

$$a^{-1}z - a^{-2}z^2$$

$$a^{-2}z^2$$

$$a^{-2}z^2 - a^{-3}z^3$$

$$a^{-3}z^3$$

$$a^{-3}z^3 - a^{-4}z^4$$

$$a^{-4}z^4 \ldots$$

Thus we have, $$X(z) = \frac{1}{1-az^{-1}} = -a^{-1}z - a^{-2}z^2 - a^{-3}z^3 - a^{-4}z^4 \ldots$$

Rearranging above equation, $$= \ldots\ldots - a^{-4}z^4 - a^{-3}z^3 - a^{-2}z^2 - a^{-1}z$$

Taking inverse z-transform, $$x(n) = \{\ldots - a^{-4}, -a^{-3}, -a^{-2}, -a^{-1}\}$$

$$\uparrow$$

$$= -a^n u(-n-1)$$

Inverse z-Transform using Partial Fraction Expansion

Following steps are to be performed for partial fraction expansions :

Step 1 : Arrange the given X(z) as,

$$\frac{X(z)}{z} = \frac{Numerator\ polynomial}{(z-p_1)(z-p_2)\cdots(z-p_N)}$$

Step 2 :
$$\frac{X(z)}{z} = \frac{A_1}{z-p_1}+\frac{A_2}{z-p_2}+\frac{A_3}{z-p_3}+\cdots\frac{A_N}{z-p_N}$$

Where, $A_k = (z-p_k)\cdot\left.\frac{X(z)}{z}\right|_{z=p_k}$, $k = 1, 2, \ldots N$

If $\frac{X(z)}{z}$ has the pole of multiplicity 'n' i.e.,

$$\frac{X(z)}{z} = \frac{Numerator\ polynomial}{(z-p)^n}$$

$$= \frac{A_1}{z-p}+\frac{A_2}{(z-p)^2}+\cdots\frac{A_n}{(z-p)^n}$$

Where $A_1, A_2, \cdots A_n$ are given as,

$$A_k = \frac{1}{(n-k)!}\cdot\frac{d^{n-k}}{dz^{n-k}}\left\{(z-p)^n\cdot\frac{X(z)}{z}\right\}\Bigg|_{z=p}$$

k = 1, 2, 3, ... n

Step 3 : Equation (3.7.1) can be written as,

$$X(z) = \frac{A_1 z}{z-p_1}+\frac{A_2 z}{z-p_2}+\cdots+\frac{A_N z}{z-p_N}$$

$$= \frac{A_1}{1-p_1 z^{-1}}+\frac{A_2}{1-p_2 z^{-1}}+\cdots+\frac{A_N}{1-p_N z^{-1}}$$

Step 4 : All the terms in above step are of the form $\frac{A_k}{1-p_k z^{-1}}$. Depending upon ROC, following standard z-transform pairs must be used.

$p_k^n u(n) \xleftrightarrow{z} \frac{1}{1-p_k z^{-1}}$, *ROC : $|z| > |a|$ i.e. causal sequence*

$-(p_k)^n u(-n-1) \xleftrightarrow{z} \frac{1}{1-p_k z^{-1}}$, *ROC : $|z| < |a|$ i.e. noncausal sequence*

Determine inverse z-transform of $X(z) = \dfrac{1}{1-1.5z^{-1}+0.5z^{-2}}$.

For (i) ROC : $|z| > 1$, *(ii) ROC :* $|z| < 0.5$ *and (iii) ROC :* $0.5 < |z| < 1$

Step 1 : First convert X(z) to positive powers of z. i.e.,

$$X(z) = \frac{z^2}{z^2-1.5z+0.5}$$

$$\frac{X(z)}{z} = \frac{z}{z^2-1.5z+0.5}$$

$$= \frac{z}{(z-1)(z-0.5)}$$

Step 2 : $$\frac{X(z)}{z} = \frac{A_1}{z-1}+\frac{A_2}{z-0.5}$$

$$A_1 = (z-1)\cdot\frac{z}{(z-1)(z-0.5)}\bigg|_{z=1} = \frac{1}{1-0.5} = 2$$

and $$A_2 = (z-0.5)\cdot\frac{z}{(z-1)(z-0.5)}\Big|_{z=0.5} = \frac{0.5}{0.5-1} = -1$$

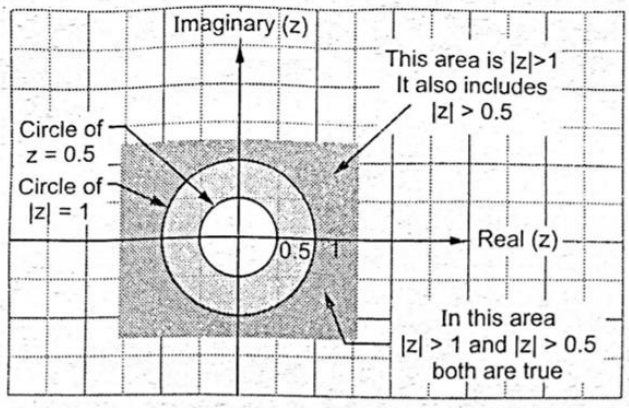
Equation will be, $$\frac{X(z)}{z} = \frac{2}{z-1} - \frac{1}{z-0.5}$$

Step 3 : $$X(z) = \frac{2z}{z-1} - \frac{z}{z-0.5}$$

$$= \frac{2}{1-z^{-1}} - \frac{1}{1-0.5z^{-1}}$$

Step 4 : i) x(n) for ROC of $|z| > 1$

- Here the poles are at $z = 1$ and $z = 0.5$ from equation
- Now ROC of $|z| > 1$ indicates that sequence corresponding to the term $\frac{2}{1-z^{-1}}$ in equation (3.7.7) must be causal.
- Fig. 3.7.1 shows the ROCs of $|z|>1$ and $|z|> 0.5$. Observe that $|z|>1$ includes $|z|> 0.5$.

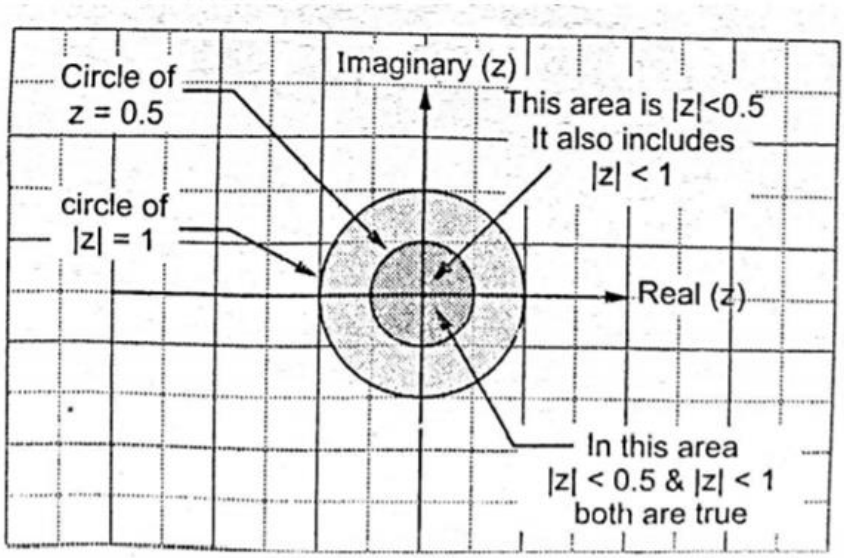


Hence the sequence corresponding to the term $\frac{1}{1-0.5z^{-1}}$ in equation

Therefore from equation (3.7.7) inverse z-transform becomes,

$$x(n) = 2(1)^n u(n) - 1.(0.5)^n u(n)$$
$$= [2-(0.5)^n]u(n)$$

x(n) for ROC : $|z| < 0.5$



$$x(n) = 2[-1^n u(-n-1)] - [-0.5^n u(-n-1)]$$
$$= [-2+0.5^n]u(-n-1)$$

x(n) for ROC : $0.5 < |z| < 1$

- This ROC can be written as $|z| > 0.5$ and $|z| < 1$.
- The sequence corresponding to $\frac{2}{1-z^{-1}}$ in equation

  ROC is $|z| < 1$.
- The sequence corresponding to $\frac{1}{1-0.5z^{-1}}$ in equation

  ROC is $|z| > 0.5$.

Taking inverse z-transform of equation

$$x(n) = 2[-1^n u(-n-1)] - (0.5)^n u(n)$$

$$= -2u(-n-1) - (0.5)^n u(n)$$

Inverse z-Transform using Contour Integration

Cauchy integral theorem is used to calculate inverse z-transform. Following steps are to be followed :

Step 1 : Define the function $X_0(z)$, which is rational and its denominator is expanded into product of poles.

i.e., $$X_0(z) = X(z)z^{n-1}$$

$$= \frac{N(z)}{\prod_{i=1}^{m} (z-p_i)^m}$$

Here 'm' is order of the pole.

Step 2 : i) For simple poles, i.e. m = 1, the residue of $X_0(z)$ at pole p_i is given as,

$$\underset{z=p_i}{\text{Res}}\, X_0(z) = \lim_{z \to p_i} [(z-p_i) X_0(z)]$$

$$= (z-p_i) X_0(z)\big|_{z=p_i}$$

ii) For multiple poles of order m_0, the residue of $X_0(z)$ can be calculated as,

$$\underset{z=p_i}{\text{Res}}\, X_0(z) = \frac{1}{(m-1)!}\left\{\frac{d^{m-1}}{dz^{m-1}}(z-p_i)^m X_0(z)\right\}_{z=p_i}$$

iii) If $X_0(z)$ has simple pole at the origin, i.e. n = 0, then x(0) can be calculated independently.

Step 3 : i) Using residue theorem, calculate x(n) for poles inside the unit circle. i.e.,

$$x(n) = \sum_{i=1}^{N} \underset{z=p_i}{\text{Res}}\, X_0(z)$$

ii) For poles outside the contour of integration,

$$x(n) = -\sum_{i=1}^{N} \underset{z=p_i}{\text{Res}}\, X_0(z) \text{ with } n < 0$$

Determine the inverse z-transform of $X(z) = \dfrac{z^2}{(z-a)^2}$, *ROC :* $|z|>|a|$ *using contour integration (i.e. residue method).*

Step 1 : $X_0(z) = X(z)\, z^{n-1} = \dfrac{z^2}{(z-a)^2} z^{n-1}$

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

Step 2 : Here the pole is at z = a and it has order m = 2. Hence using equation we can calculate residue of $X_0(z)$ at z = a as,

$$\underset{z=a}{\text{Res}}\, X_0(z) = \frac{1}{(2-1)!}\left\{\frac{d^{2-1}}{dz^{2-1}}(z-a)^2 X_0(z)\right\}_{z=a}$$

$$= \frac{d}{dz} z^{n+1}\Big|_{z=a} = (n+1)z^n\Big|_{z=a} = (n+1)\,a^n$$

Step 3 : By equation the sequence x(n) is given as,

$$x(n) = \sum_{i=1} \underset{z=a}{\text{Res}}\, X_0(z)$$

$$= (n+1)\,a^n\, u(n) \quad \text{since ROC : } |z|>|a|$$

Using residue method find the inverse z-transform of $X(z) = \frac{z+1}{(z+0.2)(z-1)}, \quad |z| > 1$

$$x(n) = \frac{1}{2\pi j} \oint_C X(z) z^{n-1} dz$$

$$= \sum \text{ residues of } X(z)z^{n-1} \text{ at poles of } X(z)z^{n-1} \text{ within C}$$

$$= \sum \text{ residues of } \frac{(z+1)z^{n-1}}{(z+0.2)(z-1)} \text{ at poles of same within C}$$

The closed contour C, begin in the ROC $|z| > 1$ encloses the poles at $z = -0.2$, $z = 1$ and, for $n = 0$ the pole is at $z = 0$. Therefore for $n = 0$

$$x(0) = \sum \text{ residues of } \frac{z+1}{z(z+0.2)(z+1)} \text{ at poles } z = 0, z = 1 \text{ and } z = -0.2$$

$$= \cancel{(z)} \frac{z+1}{\cancel{(z)}(z+0.2)(z-1)}\Big|_{z=0} + \cancel{(z+0.2)} \frac{(z+1)}{\cancel{(z+0.2)}(z-1)}\Big|_{z=0.2}$$

$$+ \cancel{(z-1)} \frac{(z+1)}{z(z+0.2)\cancel{(z-1)}}\Big|_{z=1}$$

$$= -5 + \frac{10}{3} + \frac{5}{3} = 0$$

i.e., $x(0) = 0$.

For $n \geq 1$

$$x(n) = \sum \text{ residues of } \frac{(z+1)z^{n-1}}{(z+0.2)(z-1)} \text{ at poles } z = -0.2 \text{ and } z = 1$$

$$= \text{ residue of } \frac{(z+1)z^{n-1}}{(z+0.2)(z-1)} \text{ at } z = -0.2$$

$$+ \text{ residue of } \frac{(z+1)z^{n-1}}{(z+0.2)(z-1)} \text{ at } z = 1$$

$$= \cancel{(z+0.2)} \frac{(z+1)z^{n-1}}{\cancel{(z+0.2)}(z-1)}\Big|_{z=-0.2} + \cancel{(z-1)} \frac{(z+1)z^{n-1}}{(z+0.2)\cancel{(z-1)}}\Big|_{z=1}$$

$$= -\frac{2}{3}(-0.2)^{n-1} + \frac{5}{3}$$

Therefore, $x(n) = -\frac{2}{3}(-0.2)^{n-1}u(n-1) + \frac{5}{3}u(n-1)$.

Convolution Method

In this method the given $X(z)$ is split into $X_1(z)$ and $X_2(z)$ such that $X(z) = X_1(z)X_2(z)$. Next we find $x_1(n)$ and $x_2(n)$ by taking inverse z-transform of $X_1(z)$ and $X_2(z)$ respectively. From convolution property of z-transform we know

$$Z[x_1(n) * x_2(n)] = X_1(z)X_2(z) = X(z)$$

Find the inverse z-transform of

$X(z) = \dfrac{1}{1 - 3z^{-1} + 2z^{-2}}$ using convolution method

$$X(z) = \frac{1}{1 - 3z^{-1} + 2z^{-2}}$$
$$= \frac{1}{(1 - z^{-1})(1 - 2z^{-1})} = X_1(z)X_2(z)$$

where $X_1(z) = \frac{1}{1-z^{-1}}$ and $X_2(z) = \frac{1}{1-2z^{-1}}$. From Table we find $x_1(n) = u(n)$ and $x_2(n) = 2^n u(n)$. We know

$$x(n) = x_1(n) * x_2(n)$$
$$= \sum_{k=-\infty}^{\infty} x_1(k)x_2(n-k)$$
$$= \sum_{k=-\infty}^{\infty} u(k)2^{n-k}u(n-k)$$
$$= \sum_{k=0}^{n} 2^{n-k} = 2^n \sum_{k=0}^{n} 2^{-k}$$
$$= 2^n \left[\frac{1 - \left(\frac{1}{2}\right)^{n+1}}{1 - \frac{1}{2}}\right] u(n)$$
$$= [2^{n+1} - 1]u(n)$$

REFERENCES

1. John G. Proakis & Dimitris G.Manolakis, Digital Signal Processing - Principles, Algorithms & Applications, 4th Edition, Pearson Education / Prentice Hall, 2007.
2. Haykin. S and Van Been. B., Signals and Systems, 2nd Edition, John Wiley & Sons, 2003.
3. Willis J.Tompkins, Biomedical Digital Signal Processing, Prentice Hall of India Pvt. Ltd., 2012.
4. Sanjit K. Mitra, Digital Signal Processing - A Computer Based Approach, Tata McGraw Hill, 2007.
5. Oppenheim, A.V., R.W. Schafer and J.R. Buck, Discrete Time Signal Processing, 8th Indian Reprint, Pearson, 2004.
6. Stephane Mallat, Wavelet Tour of Signal Processing: The Sparse Way, 3rd Edition, Academic Press, 2008.

QUESTIONS

1. Define signal with an example
2. Distinguish between continuous & discrete time signal
3. Define sampling & Quantization
4. Give the merits of digital signal processing compared with analog processing
5. Define Z transform
6. Explain in detail about the classification of discrete time signal with an example
7. Explain in detail about all the applications of digital signal processing
8. Determine whether the following DT signals or periodic or not
   (i) $\cos 2\pi n/5 + \cos 2\pi n/7$

   (ii) $\cos(n/8)\cos(n\pi/8)$

   (iii) $\sin(\pi+0.2n)$

9. Find and sketch the even and odd components of the following
   (i) $X(n) = e^{-(n/4)}$

   (ii) $X(n) = \text{Im}\,[e^{jn\pi/4}]$

10. Find the inverse Z transform of $X(Z) = z(z^2-4z+5)/(z-3)(z-1)(z-2)$ for ROC $2<|z|<3$, $|z|>3$, $|z|<1$ using partial fraction method
11. Determine the impulse response h(n) for the system described by the second order difference equation
    $y(n) = 0.6y(n-1)-0.08y(n-2)+x(n)$

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF BIO AND CHEMICAL ENGINEERING
DEPARTMENT OF BIOMEDICAL ENGINEERING

UNIT – II – Frequency Analysis of signals – SBMA1402

FREQUENCY ANALYSIS OF THE SIGNALS

1.1 Discrete Fourier Transform

the **discrete Fourier transform** (**DFT**) converts a finite sequence of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids, ordered by their frequencies, that has those same sample values. It can be said to convert the sampled function from its original domain (often time or position along a line) to the frequency domain.

The input samples are complex numbers (in practice, usually real numbers), and the output coefficients are complex as well. The frequencies of the output sinusoids are integer multiples of a fundamental frequency, whose corresponding period is the length of the sampling interval. The combination of sinusoids obtained through the DFT is therefore periodic with that same period. The DFT differs from the discrete-time Fourier transform (DTFT) in that its input **and** output sequences are both finite; it is therefore said to be the Fourier analysis of finite-domain (or periodic) discrete-time functions

The DFT is the most important discrete transform, used to perform Fourier analysis in many practical applications.[1] In digital signal processing, the function is any quantity or signal that varies over time, such as the pressure of a sound wave, a radio signal, or daily temperature readings, sampled over a finite time interval. In image processing, the samples can be the values of pixels along a row or column of a raster image.

1.2 Computation of DFT

Let, x(n) = Discrete time signal of length L

X(k) = DFT of x(n)

Now, the ***N-point DFT*** of x(n), where N ³ L, is defined as,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; \quad \text{for } k = 0,1,2,\ldots\ldots,N-1$$

Symbolically, the N-point DFT of x(n) can be expressed as,

$$\mathcal{DFT}\{x(n)\}$$

where, $\mathcal{DFT}$ is the operator that represents discrete Fourier transform.

$$\therefore \mathcal{DFT}\{x(n)\} = X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; \quad \text{for } k = 0, 1, 2, \ldots\ldots, N-1$$

Let, x(n) = Discrete time signal

X(k) = N-point DFT of x(n)

The ***inverse DFT*** of the sequence X(k) of length N is defined as,

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi kn}{N}} \quad ; \quad \text{for } n = 0, 1, \ldots, N-1$$

Symbolically the inverse DFT of x(n) can be expressed as,

$$\mathcal{DFT}^{-1}\{X(k)\}$$

where, $\mathcal{DFT}^{-1}$ is the operator that represents inverse DFT.

$$\mathcal{DFT}^{-1}\{X(k)\} = x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi kn}{N}} \quad ; \quad \text{for } n = 0, 1, \ldots, N-1$$

RELATIONSHIP BETWEEN DFT AND Z TRANSFORM

The $\mathbf{z}$-transform of N-point sequence x(n) is given by,

$$\mathbf{z}\{x(n)\} = X(z) = \sum_{n=0}^{N-1} x(n) z^{-n}$$

Let us evaluate X(z) at N equally spaced points on unit circle, i.e., at $z = e^{\frac{j2\pi k}{N}}$

Note : Since, $\left| e^{\frac{j2\pi k}{N}} \right| = 1$ *and* $\angle e^{\frac{j2\pi k}{N}} = \frac{2\pi k}{N}$,

the term, $z = e^{\frac{j2\pi k}{N}}$, *for* $k = 0, 1, 2, 3 \ldots . N-1$

represents N equally spaced points on unit circle in z-plane.

$$\therefore X(z)\Big|_{z = e^{\frac{j2\pi k}{N}}} = \sum_{n=0}^{N-1} x(n) z^{-n}\Big|_{z = e^{\frac{j2\pi k}{N}}} = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}}$$

By the definition of N-point DFT we get,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}}$$

From equations (5.19) and (5.20) we can say that,

$$X(k) = X(z)\Big|_{z = e^{\frac{j2\pi k}{N}}}$$

From equation , we can conclude that the N-point DFT of a finite duration sequence can be obtained from the $\mathbf{z}$-transform of the sequence, by evaluating the $\mathbf{z}$-transform of the sequence at N equally spaced points around the unit circle. Since the evaluation is performed on unit circle the ROC of X(z) should include unit circle.

Compute 4-point DFT and 8-point DFT of causal three sample sequence given by,

$$x(n) = \frac{1}{3} \quad ; \quad 0 \le n \le 2$$
$$= 0 \quad ; \quad \text{else}$$

Show that DFT coefficients are samples of Fourier transform of x(n), (Refer example 4.6 of Chapter 4 for Fourier transform).

Solution

By the definition of N-point DFT, the k^{th} complex coefficient of X(k), for $0 \le k \le N-1$, is given by,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}}$$

a) 4-point DFT (\ N = 4)

$$X(k) = \sum_{n=0}^{4-1} x(n)\, e^{\frac{-j2\pi kn}{4}} = \sum_{n=0}^{2} x(n)\, e^{\frac{-j\pi kn}{2}} = x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi k}{2}} + x(2)\, e^{-j\pi k}$$

$e^{\pm j\theta} = \cos\theta \pm j\sin\theta$

$$= \frac{1}{3} + \frac{1}{3} e^{\frac{-j\pi k}{2}} + \frac{1}{3}\, e^{-j\pi k} = \frac{1}{3}\left[1 + \cos\frac{\pi k}{2} - j\sin\frac{\pi k}{2} + \cos\pi k - j\sin\pi k\right]$$

For 4-point DFT, X(k) has to be evaluated for k = 0, 1, 2, 3.

When k = 0 ; $X(0) = \frac{1}{3}[1 + \cos 0 - j\sin 0 + \cos 0 - j\sin 0]$

$$= \frac{1}{3}(1 + 1 - j0 + 1 - j0) = 1 = 1\angle 0$$

When k = 1 ; $X(1) = \frac{1}{3}\left[1 + \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} + \cos\pi - j\sin\pi\right]$

$$= \frac{1}{3}(1 + 0 - j - 1 - j0) = -j\frac{1}{3} = \frac{1}{3}\angle -\pi/2 = 0.333\angle -0.5\pi$$

When k = 2 ; $X(2) = \frac{1}{3}[1 + \cos\pi - j\sin\pi + \cos 2\pi - j\sin 2\pi]$

$$= \frac{1}{3}(1 - 1 - j0 + 1 - j0) = \frac{1}{3} = 0.333\angle 0$$

When k = 3 ; $X(3) = \frac{1}{3}\left[1 + \cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2} + \cos 3\pi - j\sin 3\pi\right]$

$$= \frac{1}{3}(1 + 0 + j - 1 - j0) = j\frac{1}{3} = \frac{1}{3}\angle \pi/2 = 0.333\angle 0.5\pi$$

\ The 4-point DFT sequence X(k) is given by,

$$X(k) = \{\, 1\angle 0,\quad 0.333\angle -0.5\pi,\quad 0.333\angle 0,\quad 0.333\angle 0.5\pi \,\}$$

$$\therefore \text{ Magnitude Function, } |X(k)| = \{\, 1,\quad 0.333,\quad 0.333,\quad 0.333 \,\}$$

$$\text{Phase Function, } \quad \angle X(k) = \{\, 0,\quad -0.5\pi,\quad 0,\quad 0.5\pi \,\}$$

Phase angles are in radians.

b) 8-point DFT (\ N = 8)

$$X(k) = \sum_{n=0}^{8-1} x(n)\, e^{\frac{-j2\pi kn}{8}} = \sum_{n=0}^{2} x(n)\, e^{\frac{-j\pi kn}{4}} = x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi k}{4}} + x(2)\, e^{\frac{-j\pi k}{2}}$$

$e^{\pm j\theta} = \cos\theta \pm j\sin\theta$

$$= \frac{1}{3} + \frac{1}{3} e^{\frac{-j\pi k}{4}} + \frac{1}{3} e^{\frac{-j\pi k}{2}} = \frac{1}{3}\left[1 + \cos\frac{\pi k}{4} - j\sin\frac{\pi k}{4} + \cos\frac{\pi k}{2} - j\sin\frac{\pi k}{2}\right]$$

For 8-point DFT, X(k) has to be evaluated for k = 0, 1, 2, 3, 4, 5, 6, 7.

When k = 0 ; $X(0) = \frac{1}{3}[1 + \cos 0 - j\sin 0 + \cos 0 - j\sin 0]$

$= \frac{1}{3}(1 + 1 - j0 + 1 - j0) = 1 = 1\angle 0$

When k = 1 ; $X(1) = \frac{1}{3}\left[1 + \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} + \cos\frac{\pi}{2} - j\sin\frac{\pi}{2}\right]$

$= 0.333\,(1 + 0.707 - j0.707 + 0 - j1)$

$= 0.568 - j0.568 = 0.803\angle -0.785 = 0.803\angle -0.25\pi$

$\frac{0.785}{\pi} \times \pi = 0.25\pi$

When k = 2 ; $X(2) = \frac{1}{3}\left[1 + \cos\frac{2\pi}{4} - j\sin\frac{2\pi}{4} + \cos\frac{2\pi}{2} - j\sin\frac{2\pi}{2}\right]$

$= 0.333\,(1 + 0 - j1 - 1 - j0)$

$= -j0.333 = 0.333\angle -\pi/2 = 0.333\angle -0.5\pi$

When k = 3 ; $X(3) = \frac{1}{3}\left[1 + \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} + \cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right]$

$= 0.333\,(1 - 0.707 - j0.707 + 0 + j1)$

$= 0.098 + j0.098 = 0.139\angle 0.785 = 0.139\angle 0.25\pi$

When k = 4 ; $X(4) = \frac{1}{3}\left[1 + \cos\frac{4\pi}{4} - j\sin\frac{4\pi}{4} + \cos\frac{4\pi}{2} - j\sin\frac{4\pi}{2}\right]$

$= 0.333\,(1 - 1 - j0 + 1 - j0) = 0.333 = 0.333\angle 0$

When k = 5 ; $X(5) = \frac{1}{3}\left[1 + \cos\frac{5\pi}{4} - j\sin\frac{5\pi}{4} + \cos\frac{5\pi}{2} - j\sin\frac{5\pi}{2}\right]$

$= 0.333\,(1 - 0.707 + j0.707 + 0 - j1)$

$= 0.098 - j0.098 = 0.139\angle -0.785 = 0.139\angle -0.25\pi$

When k = 6 ; $X(6) = \frac{1}{3}\left[1 + \cos\frac{6\pi}{4} - j\sin\frac{6\pi}{4} + \cos\frac{6\pi}{2} - j\sin\frac{6\pi}{2}\right]$

$= 0.333\,(1 + 0 + j1 - 1 - j0)$

$= j0.333 = 0.333\angle\pi/2 = 0.333\angle 0.5\pi$

When k = 7 ; $X(7) = \frac{1}{3}\left[1 + \cos\frac{7\pi}{4} - j\sin\frac{7\pi}{4} + \cos\frac{7\pi}{2} - j\sin\frac{7\pi}{2}\right]$

$= 0.333\,(1 + 0.707 + j0.707 + 0 + j1)$

$= 0.568 + j0.568 = 0.803\angle 0.785 = 0.803\angle 0.25\pi$

Phase angles are in radians.

\ The 8-point DFT sequence X(k) is given by,

$X(k) = \{1\angle 0,\ 0.803\angle -0.25\pi,\ 0.333\angle -0.5\pi,\ 0.139\angle 0.25\pi,\ 0.333\angle 0,\ 0.139\angle -0.25\pi,$

$0.333\angle 0.5\pi,\ 0.803\angle 0.25\pi\}$

∴ Magnitude Function, $|X(k)| = \{1,\ 0.803,\ 0.333,\ 0.139,\ 0.333,\ 0.139,\ 0.333,\ 0.803\}$

Phase Function, $\angle X(k) = \{0,\ -0.25\pi,\ -0.5\pi,\ 0.25\pi,\ 0,\ -0.25\pi,\ 0.5\pi,\ 0.25\pi\}$

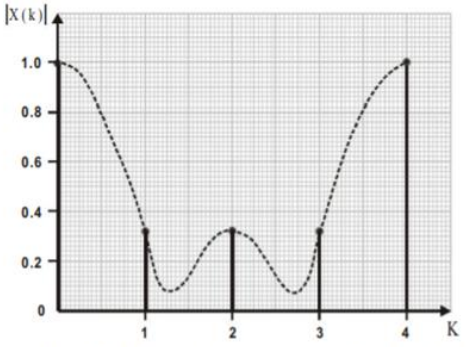


Fig 1 : Magnitude spectrum of X(k) for N=4.

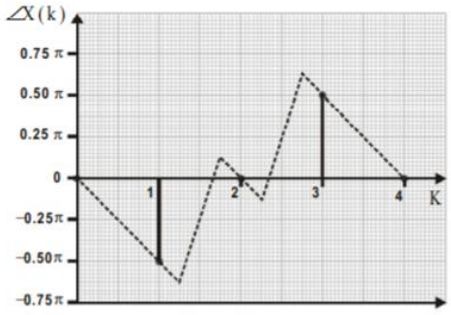


Fig 4 : Phase spectrum of X(k) for N=4.

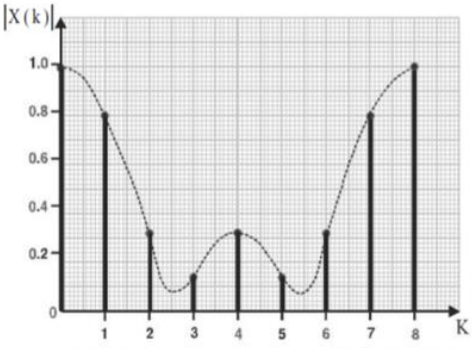


Fig 2 : Magnitude spectrum of X(k) for N=8.

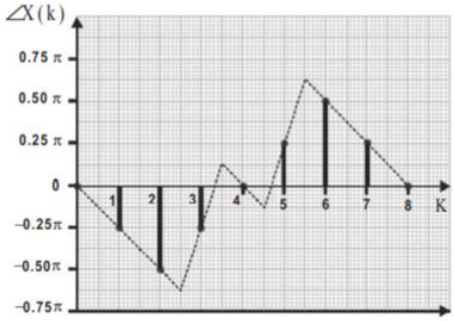


Fig 5 : Phase spectrum of X(k) for N=8.

Compute the DFT of the sequence, x(n) = {0, 1, 2, 1}. Sketch the magnitude and phase spectrum.

Solution

The given signal x(n) is 4-point signal and so, let us compute 4-point DFT.

By the definition of DFT, the 4-point DFT is given by,

$e^{\pm jq} = \cos q \pm j\sin q$

$$X(k) = \sum_{n=0}^{4-1} x(n)\, e^{\frac{-j2\pi kn}{4}} = \sum_{n=0}^{3} x(n)\, e^{\frac{-j\pi kn}{2}}$$

$$= x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi k}{2}} + x(2)\, e^{-j\pi k} + x(3)\, e^{\frac{-j3\pi k}{2}} = 0 + e^{\frac{-j\pi k}{2}} + 2\, e^{-j\pi k} + e^{\frac{-j3\pi k}{2}}$$

$$= \cos\frac{\pi k}{2} - j\sin\frac{\pi k}{2} + 2(\cos \pi k - j\sin \pi k) + \cos\frac{3\pi k}{2} - j\sin\frac{3\pi k}{2}$$

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

$\sin \pi k = 0$ for integer k

When k = 0 ; $X(0) = (\cos 0 + 2\cos 0 + \cos 0) - j(\sin 0 + \sin 0)$

$= (1 + 2 + 1) - j(0 + 0) = 4 = 4\angle 0$

When k = 1; $X(1) = \left(\cos\frac{\pi}{2} + 2\cos\pi + \cos\frac{3\pi}{2}\right) - j\left(\sin\frac{\pi}{2} + \sin\frac{3\pi}{2}\right)$

$= (0 - 2 + 0) - j(1 - 1) = -2 = 2\angle 180^\circ = 2\angle\pi$

When k = 2; $X(2) = (\cos\pi + 2\cos 2\pi + \cos 3\pi) - j(\sin\pi + \sin 3\pi)$

$= (-1 + 2 - 1) - j(0 + 0) = 0$

When k = 3; $X(3) = \left(\cos\frac{3\pi}{2} + 2\cos 3\pi + \cos\frac{9\pi}{2}\right) - j\left(\sin\frac{3\pi}{2} + \sin\frac{9\pi}{2}\right)$

$= (0 - 2 + 0) - j(-1 + 1) = -2 = 2\angle 180^\circ = 2\angle\pi$

$\therefore X(k) = \{4\angle 0,\ 2\angle p,\ 0,\ 2\angle p\}$

Magnitude Spectrum, $|X(k)| = \{4,\ 2,\ 0,\ 2\}$

Phase Spectrum, $\angle X(k) = \{0,\ p,\ 0,\ p\}$

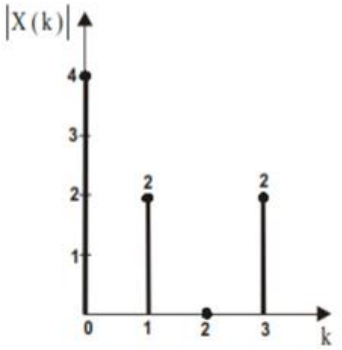


Fig 1 : Magnitude Spectrum.

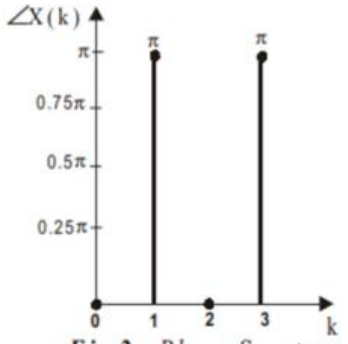


Fig 2 : Phase Spectrum.

Fast Fourier Transform (FFT)

The **F**ast **F**ourier **T**ransform (**FFT**) is a method (or algorithm) for computing the discrete Fourier transform (DFT) with reduced number of calculations. The computational efficiency is achieved if we adopt a divide and conquer approach. This approach is based on the decomposition of an N-point DFT into successively smaller DFTs. This basic approach leads to a family of an efficient computational algorithms known collectively as FFT algorithms.

Radix-r FFT

In an N-point sequence, if N can be expressed as $N = r^m$, then the sequence can be decimated into r-point sequences. For each r-point sequence, r-point DFT can be computed. From the results of r-point DFT, the r^2-point DFTs are computed. From the results of r^2-point DFTs, the r^3-point DFTs are computed and so on, until we get r^m point DFT. This FFT algorithm is called radix-**r** FFT. In computing N-point DFT by this method the number of stages of computation will be **m** times.

Radix-2 FFT

For radix-2 FFT, the value of N should be such that, $N = 2^m$, so that the N-point sequence is decimated into 2-point sequences and the 2-point DFT for each decimated sequence is computed. From the results of 2-point DFTs, the 4-point DFTs can be computed. From the results of 4-point DFTs, the 8-point DFTs can be computed and so on, until we get N-point DFT.

COMPARISON BETWEEN DFT AND FFT

Number of points N	Direct Computation		Radix-2 FFT	
	Complex additions $N(N-1)$	Complex Multiplications N^2	Complex additions $N\log_2 N$	Complex Multiplications $(N/2)\log_2 N$
$4 (=2^2)$	12	16	$4 \times \log_2 2^2 = 4 \times 2 = 8$	$\frac{4}{2} \times \log_2 2^2 = \frac{4}{2} \times 2 = 4$
$8 (=2^3)$	56	64	$8 \times \log_2 2^3 = 8 \times 3 = 24$	$\frac{8}{2} \times \log_2 2^3 = \frac{8}{2} \times 3 = 12$
$16 (=2^4)$	240	256	$16 \times \log_2 2^4 = 16 \times 4 = 64$	$\frac{16}{2} \times \log_2 2^4 = \frac{16}{2} \times 4 = 32$
$32 (=2^5)$	992	1,024	$32 \times \log_2 2^5 = 32 \times 5 = 160$	$\frac{32}{2} \times \log_2 2^5 = \frac{32}{2} \times 5 = 80$
$64 (=2^6)$	4,032	4,096	$64 \times \log_2 2^6 = 64 \times 6 = 384$	$\frac{64}{2} \times \log_2 2^6 = \frac{64}{2} \times 6 = 192$
$128 (=2^7)$	16,256	16,384	$128 \times \log_2 2^7 = 128 \times 7 = 896$	$\frac{128}{2} \times \log_2 2^7 = \frac{128}{2} \times 7 = 448$

Phase or Twiddle Factor

By the definition of DFT, the N-point DFT is given by,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} \quad ; \text{for } k = 0, 1, 2, \ldots\ldots, N-1$$

To simplify the notation it is desirable to define the complex valued phase factor W_N (also called as twiddle factor) which is an N^{th} root of unity as,

$$W_N = e^{\frac{-j2\pi}{N}}$$

Here, W represents a complex number 1Ð–2p. Hence the phase or argument of W is –2p. Therefore, when a number is multiplied by W, only its phase changes by –2p but magnitude remains same.

$$\therefore W = e^{-j2\pi}$$

The phase value –2p of W can be multiplied by any integer and it is represented as prefix in W. For example multiplying –2p by k can be represented as W^k.

$$\therefore e^{-j2\pi \times k} \Rightarrow W^k$$

The phase value –2p of W can be divided by any integer and it is represented as suffix in W. For example dividing –2p by N can be represented as W_N.

$$\therefore e^{-j2\pi \div N} = e^{-j2\pi \times \frac{1}{N}} \Rightarrow W_N$$

$$\therefore e^{\frac{j2\pi nk}{N}} = \left(e^{-j2\pi}\right)^{\frac{nk}{N}} = W_N^{nk}$$

Using equation (5.25) the equation (5.24) can be written as,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn} \quad ; \text{for } k = 0, 1, 2, \ldots\ldots, N-1$$

The equation (5.26) is the definition of N-point DFT using phase factor, and this equation is popularly used in FFT.

8-Point DFT Using Radix-2 DIT FFT

The input sequence is 8-point sequence. Therefore, $N = 8 = 2^3 = r^m$. Here, $r = 2$ and $m = 3$.

Therefore, the computation of 8-point DFT using radix-2 FFT, involves three stages of computation. The given 8-point sequence is decimated to 2-point sequences. For each 2-point sequence, the 2-point DFT is computed. From the results of 2-point DFT, the 4-point DFT can be computed. From the results of 4-point DFT, the 8-point DFT can be computed.

Let the given sequence be x(0), x(1), x(2), x(3), x(4),x(5), x(6), x(7), which consists of 8 samples. The 8-samples should be decimated into sequences of 2-samples. Before decimation they are arranged in bit reversed order, as shown in table 5.3.

The x(n) in bit reversed order is decimated into 4 numbers of 2-point sequences as shown below.

Sequence-1 : {x(0), x(4)}

Sequence-2 : {x(2), x(6)}

Sequence-3 : {x(1), x(5)}

Sequence-4 : {x(3), x(7)}

Normal order		Bit reversed order	
x(0)	x(000)	x(0)	x(000)
x(1)	x(001)	x(4)	x(100)
x(2)	x(010)	x(2)	x(010)
x(3)	x(011)	x(6)	x(110)
x(4)	x(100)	x(1)	x(001)
x(5)	x(101)	x(5)	x(101)
x(6)	x(110)	x(3)	x(011)
x(7)	x(111)	x(7)	x(111)

Using the decimated sequences as input the 8-point DFT is computed.The fig shows the three stages of computation of an 8-point DFT.

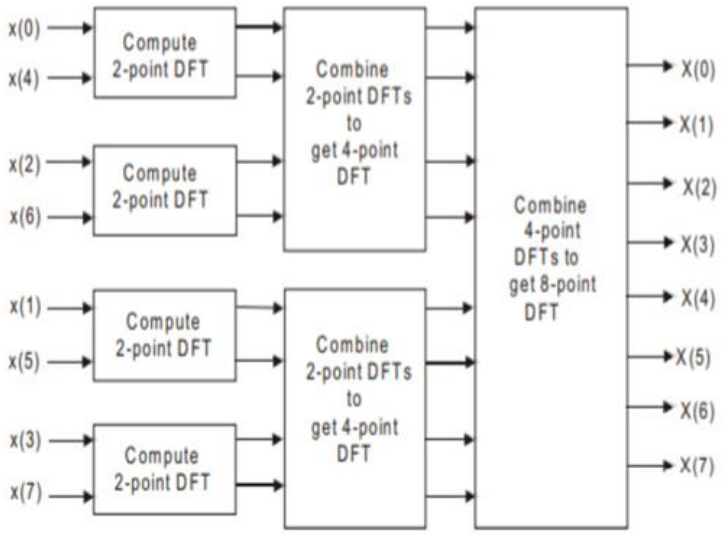


Fig Three stages of computations in 8-point DFT.

Flow Graph for 8-Point DFT using Radix-2 DIT FFT

If we observe the basic computation performed at every stage of radix-2 DIT FFT in previous section, we can arrive at the following conclusion.

1. In each computation two complex numbers "a" and "b" are considered.
2. The complex number "b" is multiplied by a phase factor "W_N^k".
3. The product "bW_N^k" is added to complex number "a" to form new complex number "A".
4. The product "bW_N^k" is subtracted from complex number "a" to form new complex number "B".

The signal flow graph is also called ***butterfly diagram*** since it resembles a butterfly. In radix-2 FFT, N/2 butterflies per stage are required to represent the computational process. The butterfly diagram used to compute the 8-point DFT via radix-2 DIT FFT can be arrived as shown below, using the computations shown in previous section.

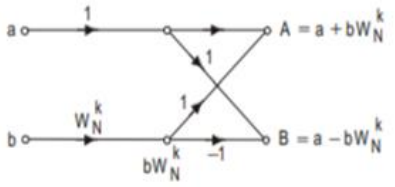


Fig *Basic butterfly or flow graph of DIT radix-2 FFT.*

Flow Graph For 8-point DFT using Radix-2 DIF FFT

If we observe the basic computation performed at every stage of radix-2 DIF FFT in previous section, we can arrive at the following conclusion.

1. In each computation two complex numbers "a" and "b" are considered.
2. The sum of the two complex numbers is computed which forms a new complex number "A".
3. Then subtract complex number "b" from "a" to get the term "a-b". The difference term "a-b" is multiplied with the phase factor or twiddle factor "W_N^k" to form a new complex number "B".

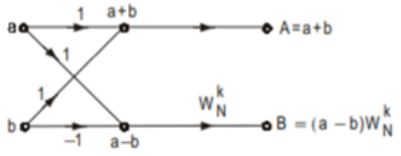


Fig *: Basic butterfly or flow graph of DIF radix-2 FFT.*

The signal flow graph is also called ***butterfly diagram*** since it resembles a butterfly. In radix-2 FFT, N/2 butterflies per stage are required to represent the computational process. The butterfly diagram used to compute the 8-point DFT via radix-2 DIF FFT can be arrived as shown below, using the computations shown in previous section.

Comparison of DIT and DIF Radix-2 FFT

Differences in DIT and DIF

- In DIT the time domain sequence is decimated, whereas in DIF the frequency domain sequence is decimated.
- In DIT the input should be in bit-reversed order and the output will be in normal order. For DIF the reverse is true, i.e., input is normal order, while output is bit reversed.
- Considering the butterfly diagram, in DIT the complex multiplication takes place before the add-subtract operation, whereas in DIF the complex multiplication takes place after the add-subtract operation.

Similarities in DIT and DIF

- For both the algorithms the value of N should be such that, $N = 2^m$, and there will be m stages of butterfly computations, with N/2 butterfly per stage.
- Both algorithms involve same number of operations. The total number of complex additions are $N\log_2 N$ and total number of complex multiplications are $(N/2)\log_2 N$.
- Both algorithms require bit reversal at some place during computation.

Computation of Inverse DFT Using FFT

Let, x(n) and X(k) be N-point DFT pair.

Now by the definition of inverse DFT,

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi nk}{N}} \quad ; \text{for } n = 0, 1, 2, \ldots, N-1$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} X(k)\left(e^{\frac{-j2\pi nk}{N}}\right)^* = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\left(W_N^{nk}\right)^* = \frac{1}{N}\left[\sum_{k=0}^{N-1} X(k)\left(W_N^{nk}\right)^*\right]$$

In equation the expression inside the bracket is similar to that of DFT computation of a sequence, with following differences.

1. The summation index is k instead of n.
2. The input sequence is X(k) instead of x(n).
3. The phase factors are conjugate of the phase factor used for DFT.

Hence, in order to compute inverse DFT of X(k), the FFT algorithm can be used by taking the conjugate of phase factors. Also from equation it is observed that the output of FFT computation should be divided by N to get x(n).

The following procedure can be followed to compute inverse DFT using FFT algorithm.

1. Take N-point frequency domain sequence X(k) as input sequence.
2. Compute FFT by using conjugate of phase factors.
3. Divide the output sequence obtained in FFT computation by N, to get the sequence x(n).

Thus a single FFT algorithm can be used for evaluation of both DFT and inverse DFT.

An 8-point sequence is given by x(n) = {2, 1, 2, 1, 1, 2, 1, 2}. Compute 8-point DFT of x(n) by **a)** radix-2 DIT-FFT and **b)** radix-2 DIF-FFT. Also sketch the magnitude and phase spectrum.

Solution

a) 8-point DFT by Radix-2 DIT-FFT

The given sequence is first arranged in the bit reversed order.

The sequence x(n) in normal order	The sequence x(n) in bit reversed order
x(0) = 2	x(0) = 2
x(1) = 1	x(4) = 1
x(2) = 2	x(2) = 2
x(3) = 1	x(6) = 1
x(4) = 1	x(1) = 1
x(5) = 2	x(5) = 2
x(6) = 1	x(3) = 1
x(7) = 2	x(7) = 2

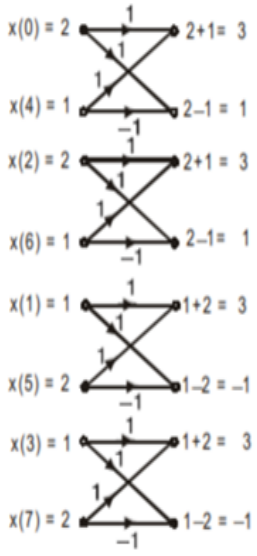


Fig 1 : Butterfly diagram for first stage of radix-2 DIT FFT.

The 8-point DFT by radix-2 FFT involve 3 stages of computation with 4-butterfly computations in each stage. The sequence rearranged in the bit reversed order forms the input to the first stage. For other stages of computation the output of previous stage will be the input for current stage.

First stage computation

The input sequence of first stage computation = { 2, 1, 2, 1, 1, 2, 1, 2}

The butterfly computations of first stage are shown in fig 1.

The phase factor involved in first stage of computation is W_2^0. Since, $W_2^0 = 1$, it is not considered for computation.

Second stage computation

The input sequence to second stage computation = { 3, 1, 3, 1, 3, –1, 3, –1 }

The phase factors involved in second stage computation are W_4^0 and W_4^1.

The butterfly computations of second stage are shown in fig 2.

$$W_4^0 = e^{-j2\pi \times \frac{0}{4}} = e^0 = 1$$

$$W_4^1 = e^{-j2\pi \times \frac{1}{4}} = e^{-j \times \frac{\pi}{2}}$$

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

$$= -j$$

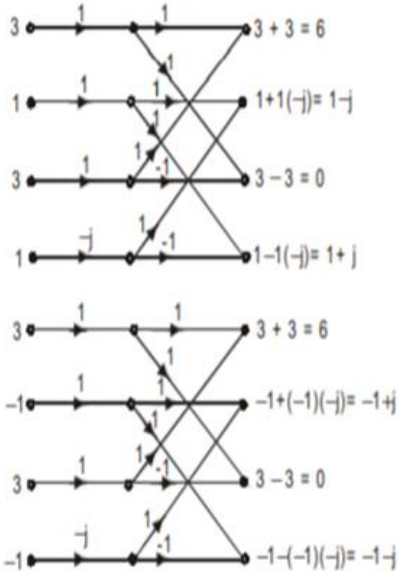


Fig 2 : Butterfly diagram for second stage of radix-2 DIT FFT.

The output sequence of second stage of computation = {6, 1–j, 0, 1+j, 6, –1+j, 0, –1–j}

Third stage computation

The input sequence to third stage computation = {6, 1−j, 0, 1+j, 6, −1+j, 0, −1−j}

The phase factors involved in third stage computation are W_8^0, W_8^1, W_8^2 and W_8^3.

The butterfly computations of third stage are shown in fig 3.

$$W_8^0 = e^{-j2\pi \times \frac{0}{8}} = e^0 = 1$$

$$W_8^1 = e^{-j2\pi \times \frac{1}{8}} = e^{-j \times \frac{\pi}{4}} = \cos\left(\frac{-\pi}{4}\right) + j\sin\left(\frac{-\pi}{4}\right) = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$

$$W_8^2 = e^{-j2\pi \times \frac{2}{8}} = e^{-j \times \frac{\pi}{2}} = \cos\left(\frac{-\pi}{2}\right) + j\sin\left(\frac{-\pi}{2}\right) = -j$$

$$W_8^3 = e^{-j2\pi \times \frac{3}{8}} = e^{-j \times \frac{3\pi}{4}} = \cos\left(\frac{-3\pi}{4}\right) + j\sin\left(\frac{-3\pi}{4}\right) = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$

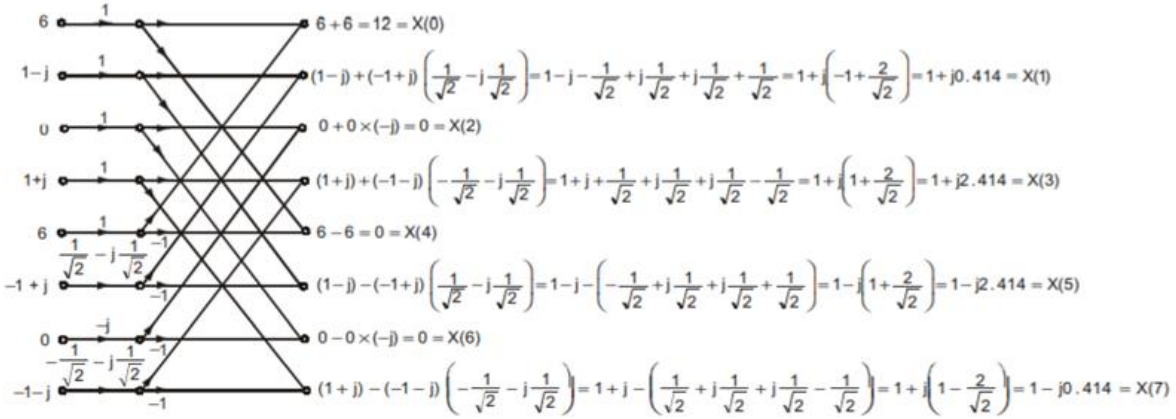


Fig 3 : *Butterfly diagram for third stage of radix-2 DIT FFT of X(k) .*

The output sequence of third stage of computation = {12, 1+j0.414, 0, 1+j2.414, 0, 1−j2.414, 0, 1−j0.414}

The output sequence of third stage of computation is the 8-point DFT of the given sequence in normal order.

$$\therefore DFT\{x(n)\} = X(k) = \{12,\ 1+j0.414,\ 0,\ 1+j2.414,\ 0,\ 1-j2.414,\ 0,\ 1-j0.414\}$$

b) 8-point DFT by Radix-2 DIF-FFT

For 8-point DFT by radix-2 FFT we require 3-stages of computation with 4-butterfly computation in each stage. The given sequence is the input to first stage. For other stages of computations, the output of previous stage will be the input for current stage.

First stage computation

The input sequence for first stage of computation = {2, 1, 2, 1, 1, 2, 1, 2}

The phase factors involved in first stage computation are W_8^0, W_8^1, W_8^2 and W_8^3.

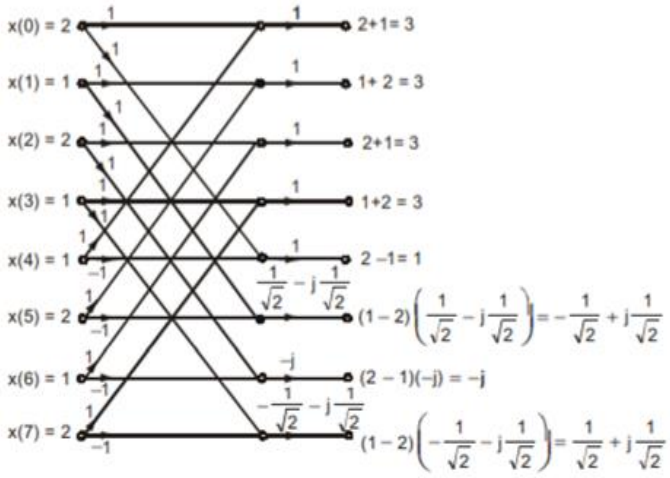
The butterfly computations of first stage are shown in fig 4.

$$W_8^0 = e^{-j2\pi \times \frac{0}{8}} = 1$$

$$W_8^1 = e^{-j2\pi \times \frac{1}{8}} = e^{-j\frac{\pi}{4}} = \cos\left(-\frac{\pi}{4}\right) + j\sin\left(-\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$

$$W_8^2 = e^{-j2\pi \times \frac{2}{8}} = e^{-j\frac{\pi}{2}} = \cos\left(-\frac{\pi}{2}\right) + j\sin\left(-\frac{\pi}{2}\right) = -j$$

$$W_8^3 = e^{-j2\pi \times \frac{3}{8}} = e^{-j\frac{3\pi}{4}} = \cos\left(-\frac{3\pi}{4}\right) + j\sin\left(-\frac{3\pi}{4}\right) = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$



The output sequence of first stage of computation $= \left\{3,\ 3,\ 3,\ 3,\ 1,\ -\frac{1}{\sqrt{2}}+j\frac{1}{\sqrt{2}},\ -j,\ \frac{1}{\sqrt{2}}+j\frac{1}{\sqrt{2}}\right\}$

Second stage computation

The input sequence for second stage of computation $= \left\{3,\ 3,\ 3,\ 3,\ 1,\ -\frac{1}{\sqrt{2}}+j\frac{1}{\sqrt{2}},\ -j,\ \frac{1}{\sqrt{2}}+j\frac{1}{\sqrt{2}}\right\}$

The phase factors involved in second stage computation are W_4^0 and W_4^1.
The butterfly computations of second stage are shown in fig 5.

$$W_4^0 = e^{-j2\pi \times \frac{0}{4}} = 1$$

$$W_4^1 = e^{-j2\pi \times \frac{1}{4}} = e^{-j \times \frac{\pi}{2}}$$

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

$$= -j$$

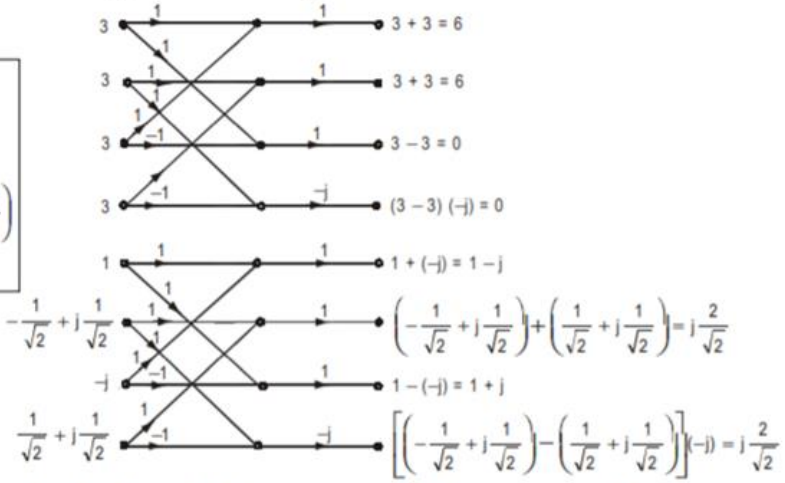


Fig 5 : Butterfly diagram for second stage of radix-2 DIF FFT.

The output sequence of second stage of computation $= \left\{6,\ 6,\ 0,\ 0,\ 1-j,\ j\frac{2}{\sqrt{2}},\ 1+j,\ j\frac{2}{\sqrt{2}}\right\}$

Third stage computation

The input sequence to third stage of computation $= \left\{6,\ 6,\ 0,\ 0,\ 1-j,\ j\frac{2}{\sqrt{2}},\ 1+j,\ j\frac{2}{\sqrt{2}}\right\}$

The butterfly computations of third stage are shown in fig 6.

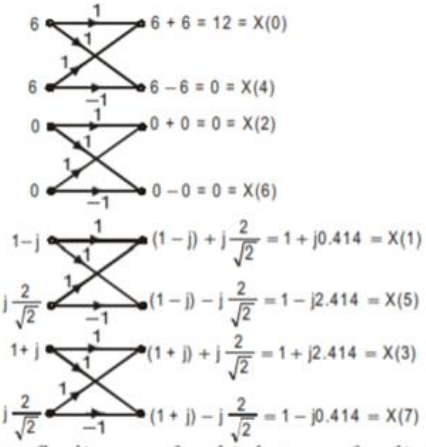


Fig 6 : Butterfly diagram for third stage of radix-2 DIF FFT.

The phase factor involved in third stage of computation is W_2^0. Since, $W_2^0 = 1$, it is not considered for computation.

The output sequence of third stage of computation = { 12, 0, 0, 0, 1+j0.414, 1–j2.414, 1+j2.414, 1–j0.414 }

The output sequence of third stage of computation is the 8-point DFT of the given sequence in bit reversed order.

In DIF-FFT algorithm the input to first stage is in normal order and the output of third stage will be in the bit reversed order. Hence the actual result is obtained by arranging the output sequence of third stage in normal order as shown below.

The sequence X(k) in bit reversed order	The sequence X(k) in normal order
X(0) = 12	X(0) = 12
X(4) = 0	X(1) = 1+ j0.414
X(2) = 0	X(2) = 0
X(6) = 0	X(3) = 1+ j2.414
X(1) = 1+ j0.414	X(4) = 0
X(5) = 1– j2.414	X(5) = 1– j2.414
X(3) = 1+ j2.414	X(6) = 0
X(7) = 1– j0.414	X(7) = 1– j0.414

$\therefore DFT\{x(n)\} = X(k) = \{12,\ 1+j0.414,\ 0,\ 1+j2.414,\ 0,\ 1-j2.414,\ 0,\ 1-j0.414\}$

Magnitude and phase specturm

Each element of the sequence X(k) is a complex number and they are expressed in rectangular coordinates. If they are converted to polar coordinates then the magnitude and phase of each element can be obtained.

Note : The rectangular to polar conversion can be obtained by using R ® P conversion in calculator.

$$X(k) = \{12,\ 1+j0.414,\ 0,\ 1+j2.414,\ 0,\ 1-j2.414,\ 0,\ 1-j0.414\}$$

$$= \{12\angle 0^\circ,\ 1.08\angle 22^\circ,\ 0\angle 0^\circ,\ 2.61\angle 67^\circ,\ 0\angle 0^\circ,\ 2.61\angle -67^\circ,\ 0\angle 0^\circ,\ 1.08\angle -22^\circ\}$$

$$= \left\{\begin{matrix} 12\angle 0,\ 1.08\angle 22^\circ\times\frac{\pi}{180^\circ},\ 0\angle 0,\ 2.61\angle 67^\circ\times\frac{\pi}{180^\circ},\ 0\angle 0, \\ 2.61\angle -67^\circ\times\frac{\pi}{180^\circ},\ 0\angle 0,\ 1.08\angle -22^\circ\times\frac{\pi}{180^\circ} \end{matrix}\right\}$$

$$= \{12\angle 0,\ 1.08\angle 0.12\pi,\ 0\angle 0,\ 2.61\angle 0.37\pi,\ 0\angle 0,\ 2.61\angle -0.37\pi,\ 0\angle 0,\ 1.08\angle -0.12\pi\}$$

$$\therefore |X(k)| = \{12,\ 1.08,\ 0,\ 2.61,\ 0,\ 2.61,\ 0,\ 1.08\}$$

$$\angle X(k) = \{0,\ 0.12\pi,\ 0,\ 0.37\pi,\ 0,\ -0.37\pi,\ 0,\ -0.12\pi\}$$

The magnitude specturm is the plot of the magnitude of each sample of X(k) as a function of k as shown in fig 7. The phase spectrum is the plot of phase of X(k) as a function of k as shown in fig 8.

When N-point DFT is performed on a sequence x(n) then the DFT sequence X(k) will have a periodicity of N. Hence in this example the magnitude and phase specturm will have a periodicity of 8 as shown in fig 7 and fig 8.

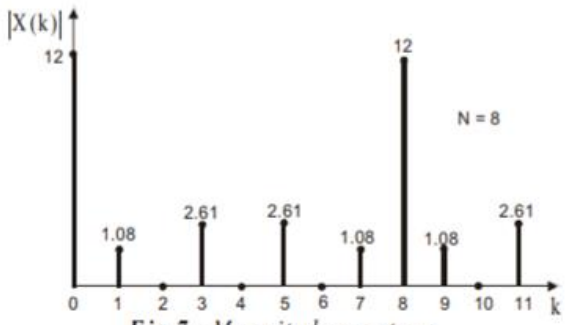


Fig 7 : Magnitude spectrum.

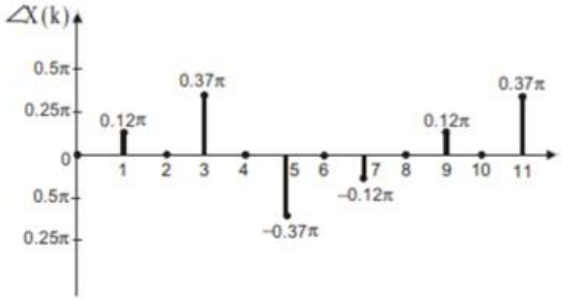


Fig 8: Phase spectrum.

To determine inverse DFT of Y(k)

The 8-point inverse DFT of Y(k) can be computed using radix-2 DIT FFT by taking conjugate of the phase factors and then dividing the output sequence of FFT by 8.

The 8-point inverse DFT of Y(k) using radix-2 DIT FFT involves three stages of computations with 4-butterflies in each stage. The sequence Y(k) is arranged in bit reversed order as shown in the following table.

The sequence arranged in bit reversed order forms the input sequence to first stage computation.

Y(k) Normal order	Y(k) Bit reversed order
$Y(0) = 0$	$Y(0) = 0$
$Y(1) = -2 + j4.828$	$Y(4) = 16$
$Y(2) = 0$	$Y(2) = 0$
$Y(3) = -2 + j0.828$	$Y(6) = 0$
$Y(4) = 16$	$Y(1) = -2 + j4.828$
$Y(5) = -2 - j0.828$	$Y(5) = -2 - j0.828$
$Y(6) = 0$	$Y(3) = -2 + j0.828$
$Y(7) = -2 - j4.828$	$Y(7) = -2 - j4.828$

First stage computation

$$\text{Input sequence of first stage} = \left\{ \begin{matrix} 0, & 16, & 0, & 0, & -2+j4.828, & -2+j0.828, \\ -2+j0.828, & & -2-j4.828 & & & \end{matrix} \right\}$$

The butterfly computations of first stage are shown in fig 7.

The phase factor involved in first stage of computation is $\left(W_2^0\right)^*$.

Since, $\left(W_2^0\right)^* = e^{j2\pi \times \frac{0}{4}} = e^0 = 1$,

it is not considered for computation.

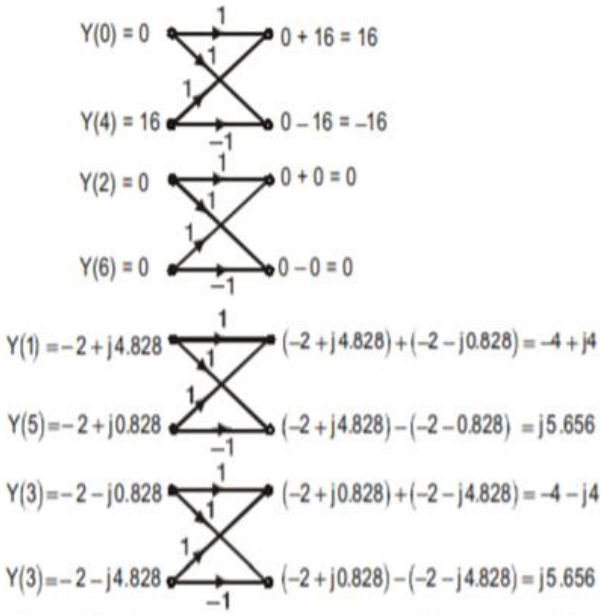


Fig 7 : Butterfly diagram for first stage of inverse DFT of Y(k).

$$\text{Output sequence of first stage} = \{16, -16, 0, 0, -4+j4, j5.656, -4-j4, j5.656\}$$

Second stage computation

Input sequence of second stage $= \{16, -16, 0, 0, -4+j4, j5.656, -4-j4, j5.656\}$

The butterfly computation of second stage is shown in fig 8.

The phase factors involved are $(W_4^0)^*$ and $(W_4^1)^*$.

$$(W_4^0)^* = e^{j2\pi \times \frac{0}{4}} = e^0 = 1$$

$$(W_4^1)^* = e^{j2\pi \times \frac{1}{4}} = e^{j \times \frac{\pi}{2}} = \cos\left(\frac{\pi}{2}\right) + j\sin\left(\frac{\pi}{2}\right) = j$$

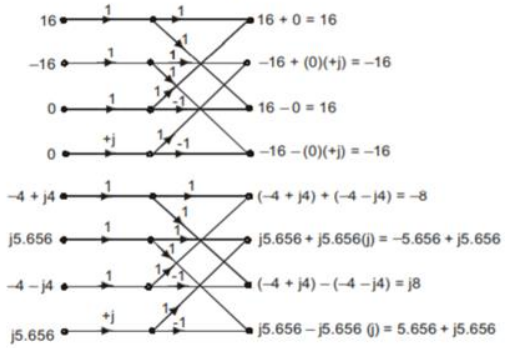


Fig 8: Butterfly diagram for second stage of inverse DFT of Y(k)

Output sequence of second stage computation $= \{16, -16, 16, -16, -8, -5.656+j5.656, j8, 5.656+j5.656\}$

Third stage computation

Input sequence of third stage computation $= \{16, -16, 16, -16, -8, -5.656+j5.656, j8, 5.656+j5.656\}$

The butterfly computation of third stage is shown in fig 9.

The phase factors involved are $(W_8^0)^*$, $(W_8^1)^*$, $(W_8^2)^*$ and $(W_8^3)^*$.

$$(W_8^0)^* = e^{j2\pi \times \frac{0}{8}} = 1$$

$$(W_8^1)^* = e^{j2\pi \times \frac{1}{8}} = e^{j \times \frac{\pi}{4}} = \cos\left(\frac{\pi}{4}\right) + j\sin\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}} = 0.707 + j0.707$$

$$(W_8^2)^* = e^{j2\pi \times \frac{2}{8}} = e^{j \times \frac{\pi}{2}} = \cos\left(\frac{\pi}{2}\right) + j\sin\left(\frac{\pi}{2}\right) = j$$

$$(W_8^3)^* = e^{j2\pi \times \frac{3}{8}} = e^{j \times \frac{3\pi}{4}} = \cos\left(\frac{3\pi}{4}\right) + j\sin\left(\frac{3\pi}{4}\right) = -\frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}} = -0.707 + j0.707$$

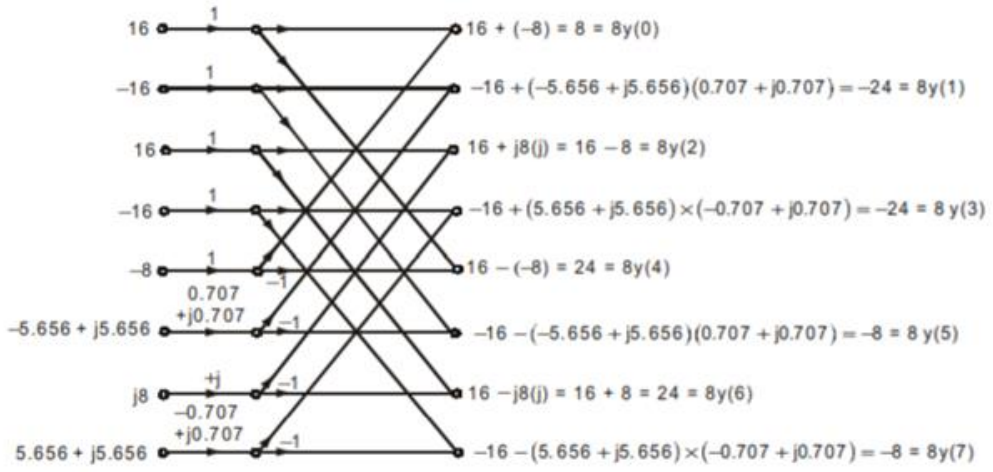


Fig 9 : Butterfly diagram for third stage of inverse DFT of Y(k).

Output sequence of third stage computation = { 8, −24, 8, −24, 24, −8, 24, −8 }

The sequence y(n) is obtained by dividing each sample of output sequence of third stage by 8.

\ The response of the LTI system, y(n) = { 1, −3, 1, −3, 3, −1, 3, −1 }

LINEAR CONVOLUTION

Linear convolution is a very powerful technique used for the analysis of LTI systems. In the last subsection we have seen that how the sequence $x(n)$ can be expressed as sum of weighted impulses. It is given by equation

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

If $x(n)$ is applied as an input to the discrete time system, then response $y(n)$ of the system is given as,

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

Putting for $x(n)$ in above equation from equation (4.3.7),

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

The above equation we have written on the basis of scaling property. It states that if $y(n) = T[a\,x(n)]$, then $y(n) = a\,T[x(n)]$ for a = constant. The above equation can be written in compact form with the help of '$\sum$' sign. i.e.,

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

The response of the system to unit sample sequence $\delta(n)$ is given as,

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

Here $h(n)$ is called unit sample response or impulse response of the system. If the discrete time system is shift invariant, then above equation can be written as,

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

Here 'k' is some shift in samples. The above equation indicates that; if the excitation of the shift invariant system is delayed, then its response is also delayed by the same amount. Putting for $T[\delta(n-k)] = h(n-k)$ in equation we get,

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

This equation gives the response of linear shift invariant (LTI) system or LTI system to an input $x(n)$. The behaviour of the LTI system is completely characterized by the unit sample response $h(n)$. The above equation is basically linear convolution of $x(n)$ and $h(n)$. This linear convolution gives $y(n)$. Thus,

Convolution sum : $y(n) = x(n) * h(n)$

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

Convolve the following two sequences $x(n)$ and $h(n)$ to get $y(n)$ $x(n) = \{1, 1, 1, 1\}$, $h(n) = \{2, 2\}$

Solution : Here upward arrow $(\uparrow)$ is not shown in $x(n)$ as well as $h(n)$ means, the first sample in the sequence is 0^{th} sample. Thus the sample values are :

$x(k=0) = 1$

$x(k=1) = 1$

$x(k=2) = 1$

$x(k=3) = 1$

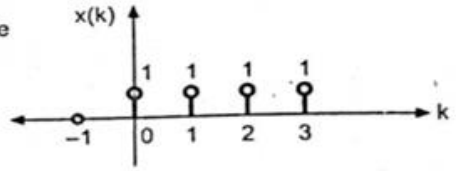
and $h(k=0) = 2$

$h(k=1) = 2$

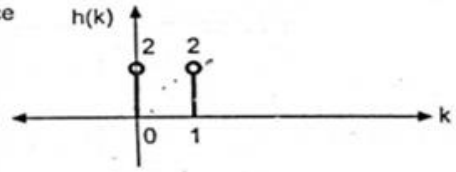
The convolution of $x(n)$ and $h(n)$ is given by equation

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

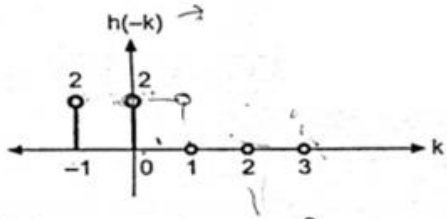
(a). Sequence x(k)

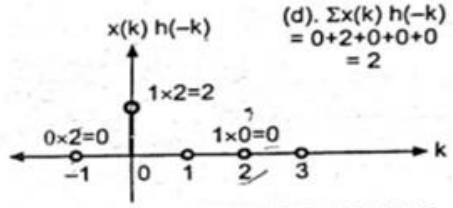


(b). Sequence h(k)



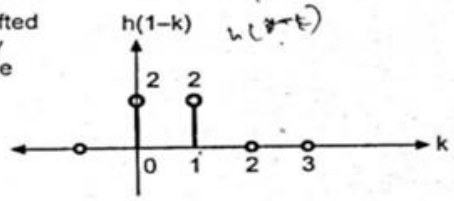
(c). Folded sequence h(–k)

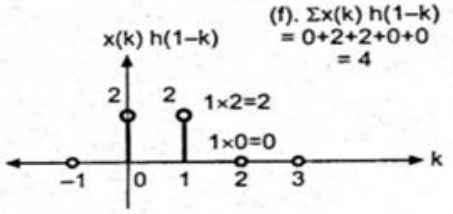




(d). $\Sigma x(k)\, h(-k) = 0+2+0+0+0 = 2$

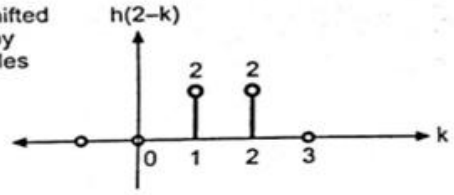
(e). h(–k) shifted to right by one sample

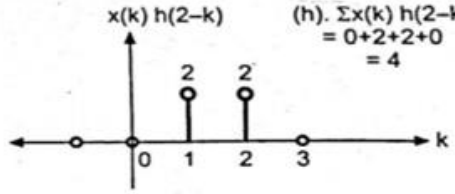




(f). $\Sigma x(k)\, h(1-k) = 0+2+2+0+0 = 4$

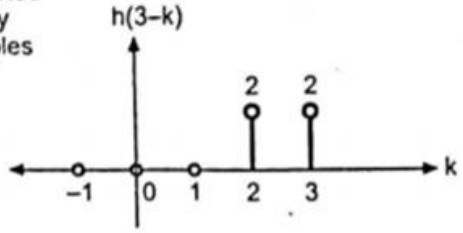
(g). h(–k) shifted to right by two samples

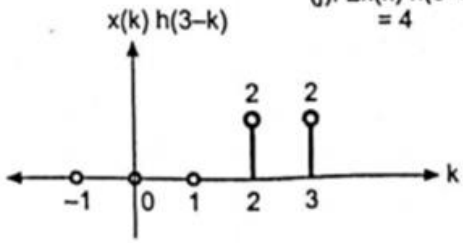




(h). $\Sigma x(k)\, h(2-k) = 0+2+2+0 = 4$

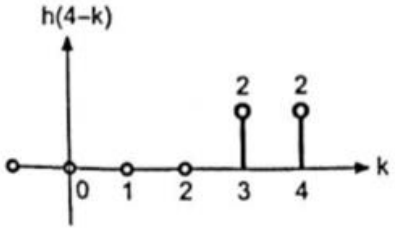
(i). h(–k) shifted to right by three samples

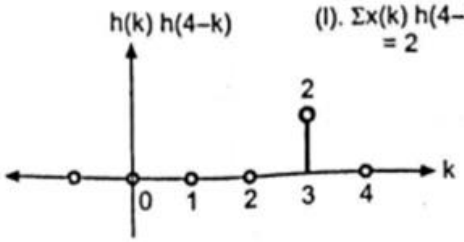




(j). $\Sigma x(k)\, h(3-k) = 4$

(k). h(–k) shifted to right by four samples





(l). $\Sigma x(k)\, h(4-k) = 2$

CIRCULAR CONVOLUTION
CONCENTRIC CIRCLE METHOD

Given two sequences $x_1(n)$ and $x_2(n)$, the circular convolution of these two sequences $x_3(n) = x_1(n) \textcircled{N} x_2(n)$ can be found by using the following steps.

1. Graph N samples of $x_1(n)$ as equally spaced points around an outer circle in counterclockwise direction.
2. Start at the same point as $x_1(n)$ graph N samples of $x_2(n)$ as equally spaced points around an inner circle in clockwise direction.
3. Multiply corresponding samples on the two circles and sum the products to produce output.
4. Rotate the inner circle one sample at a time in counterclockwise direction and go to step 3 to obtain the next value of output.
5. Repeat step No.4 until the inner circle first sample lines up with the first sample of the exterior circle once again.

MATRIX MULTIPLICATION METHOD

In this method, the circular convolution of two sequences $x_1(n)$ and $x_2(n)$ can be obtained by representing the sequences in matrix form as shown below

$$\begin{bmatrix} x_2(0) & x_2(N-1) & x_2(N-2) & \dots & x_2(2) & x_2(1) \\ x_2(1) & x_2(0) & x_2(N-1) & \dots & x_2(3) & x_2(2) \\ x_2(2) & x_2(1) & x_2(0) & \dots & x_2(4) & x_2(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_2(N-2) & x_2(N-3) & x_2(N-4) & \dots & x_2(0) & x_2(N-1) \\ x_2(N-1) & x_2(N-2) & x_2(N-3) & \dots & x_2(1) & x_2(0) \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ \vdots \\ x_2(N-2) \\ x_1(N-1) \end{bmatrix} = \begin{bmatrix} x_3(0) \\ x_3(1) \\ x_3(2) \\ \vdots \\ x_3(N-2) \\ x_3(N-1) \end{bmatrix}$$

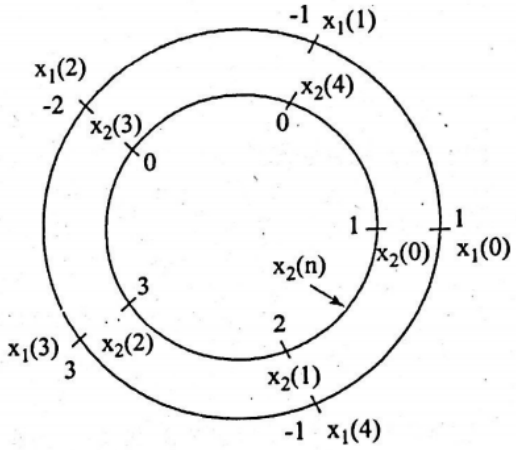
The sequence $x_2(n)$ is repeated via circular shift of samples and represented in $N \times N$ matrix form. The sequence $x_1(n)$ is represented as column matrix. The multiplication of these two matrices gives the sequences $x_3(n)$.

Find the circular convolution of two finite duration sequence $x_1(n) = \{1,-1,-2,3,-1\}$, $x_2(n) = \{1,2,3\}$

Solution To find circular convolution, both sequences must be of same length. Therefore we append two zeros to the sequence $x_2(n)$ and use concentric circle method to find circular convolution.

We have

$$x_1(n) = \{1, -1, -2, 3, -1\}$$
$$x_2(n) = \{1, 2, 3, 0, 0\}$$

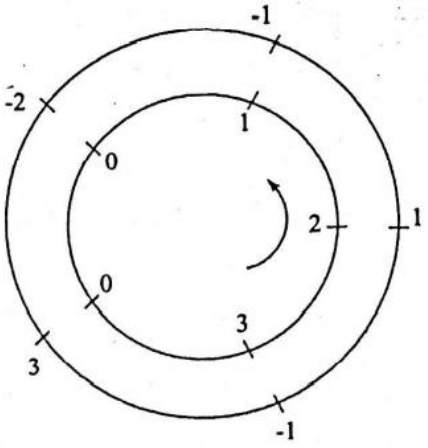


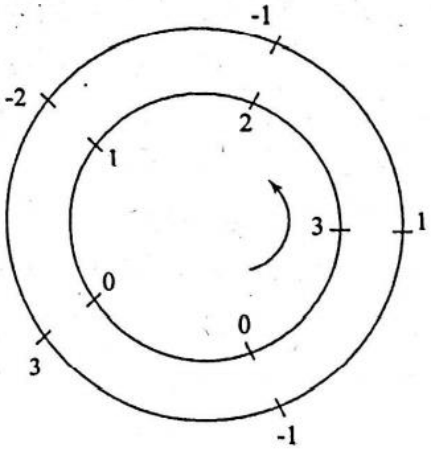
Graph all the points of $x_1(n)$ on the outer circle in the counterclockwise direction. Starting at same point as $x_1(n)$ graph all points of $x_2(n)$ on the inner circle in clockwise direction.

Multiply corresponding samples on the circle and add to obtain

$$y(0) = 1(1) + 0(-1) + 0(-2) + 3(3) + 2(-1)$$
$$= 8$$

Rotate the inner circle in counterclockwise direction by one sample, multiply the corresponding samples to obtain $y(1)$.

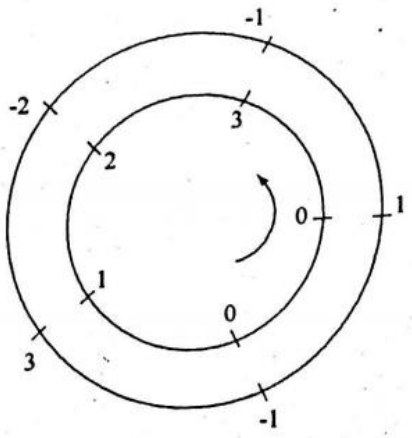


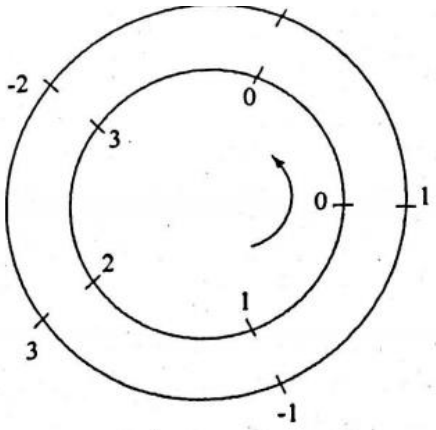


$y(1) = 1(2) + (-1)1 + (-2)0 + 3(0) + 3(-1);$
$= -2$

$y(2) = 3(1) + 2(-1) + 1(-2) + 0(3) + -1(0)$
$= -1$

Obtain remaining samples by repeating above procedure until the inner circle first sample lines up with the first sample of the exterior circle.





$y(3) = (0)1 + 3(-1) + 2(-2) + 1(3) + (-1)(0);$ $y(4) = 0(1) + 0(-1) + 3(-2) + 3(2) + 1(-1)$

$= -4$ $= -1$

$$y(n) = \{8, -2, -1, -4, -1\}$$

Matrix Method

Given

$$x_1(n) = \{1, -1, -2, 3, -1\}$$
$$x_2(n) = \{1, 2, 3, \}$$

By adding two zeros to the sequence $x_2(n)$, we bring the length of the sequence $x_2(n)$ to 5.

Now

$$x_2(n) = \{1, 2, 3, 0, 0\}$$

The matrix form can be written by substituting $N = 5$ in Eq. (3.55).

$$\begin{bmatrix} x_2(0) & x_2(4) & x_2(3) & x_2(2) & x_2(1) \\ x_2(1) & x_2(0) & x_2(4) & x_2(3) & x_2(2) \\ x_2(2) & x_2(1) & x_2(0) & x_2(4) & x_2(3) \\ x_2(3) & x_2(2) & x_2(1) & x_2(0) & x_2(4) \\ x_2(4) & x_2(3) & x_2(2) & x_2(1) & x_2(0) \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ x_1(3) \\ x_1(4) \end{bmatrix} = \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \end{bmatrix}$$

Represent the sequence $x_2(n)$ in $N \times N$ matrix form and $x_1(n)$ in column matrix form and multiply to get $y(n)$.

$$\begin{matrix} x_2(n) \end{matrix} \quad \begin{matrix} x_1(n) \end{matrix} \quad \begin{matrix} y(n) \end{matrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 3 & 2 \\ 2 & 1 & 0 & 0 & 3 \\ 3 & 2 & 1 & 0 & 0 \\ 0 & 3 & 2 & 1 & 0 \\ 0 & 0 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -2 \\ 3 \\ -1 \end{bmatrix} = \begin{bmatrix} 8 \\ -2 \\ -1 \\ -4 \\ -1 \end{bmatrix}$$

$$y(n) = \{8, -2, -1, -4, -1\}$$

REFERENCES

1. John G. Proakis & Dimitris G.Manolakis, Digital Signal Processing - Principles, Algorithms & Applications, 4th Edition, Pearson Education / Prentice Hall, 2007.
2. Haykin. S and Van Been. B., Signals and Systems, 2nd Edition, John Wiley & Sons, 2003.
3. Willis J.Tompkins, Biomedical Digital Signal Processing, Prentice Hall of India Pvt. Ltd., 2012.
4. Sanjit K. Mitra, Digital Signal Processing - A Computer Based Approach, Tata McGraw Hill, 2007.
5. Oppenheim, A.V., R.W. Schafer and J.R. Buck, Discrete Time Signal Processing, 8th Indian Reprint, Pearson, 2004.
6. Stephane Mallat, Wavelet Tour of Signal Processing: The Sparse Way, 3rd Edition, Academic Press, 2008.

QUESTIONS

1. Give the comparison of DFT & FFT
2. Find the IDFT of a sequence Y(k) = {1, 0, 1, 0}
3. Compare DIT & DIF
4. Define convolution
5. For 128 point DFT how many complex multiplications are required
6. Draw the butterfly diagram for four point radix 2 DIT FFT
7. Determine the 8 point DFT of a sequence
8. x(n) = {1, 1, 1, 1, 1, 1, 0, 0}
9. Find IDFT of a sequence X(K) = {5, 0, 1-j, 0, 1, 0, 1+j, 0}
10. Find the DFT of a sequence x(n) = {1, 2, 3, 4, 4, , 2, 1} using DIT algorithm
11. Find the IDFT of a sequence X(K) = {10, -2+j2, -2, -2-j2} using DIT & Find the DFT of a sequence x(n) = {1, 0, 0, 1} using DIF algorithm
12. Find the linear convolution of the given sequences
13. x(n) = {1, 1, 1, 1}, h(n) = {2, 2}
14. Find the discrete convolution of two finite duration sequences x(n) = {1, 2, -1, 1}, h(n) = {1, 0, 1, 1}.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF BIO AND CHEMICAL ENGINEERING
DEPARTMENT OF BIOMEDICAL ENGINEERING

UNIT – III – Digital Filter Designing – SBMA1402

FIR FILTER DESIGNING

DIGITAL FILTERS

In signal processing, a **digital filter** is a system that performs mathematical operations on a sampled, discrete-time signal to reduce or enhance certain aspects of that signal. This is in contrast to the other major type of electronic filter, the analog filter, which is an electronic circuit operating on continuous-time analog signals.

A digital filter system usually consists of an analog-to-digital converter to sample the input signal, followed by a microprocessor and some peripheral components such as memory to store data and filter coefficients etc. Finally a digital-to-analog converter to complete the output stage. Program Instructions (software) running on the microprocessor implement the digital filter by performing the necessary mathematical operations on the numbers received from the ADC. In some high performance applications, an FPGA or ASIC is used instead of a general purpose microprocessor, or a specialized DSP with specific paralleled architecture for expediting operations such as filtering.

Digital filters may be more expensive than an equivalent analog filter due to their increased complexity, but they make practical many designs that are impractical or impossible as analog filters. When used in the context of real-time analog systems, digital filters sometimes have problematic latency (the difference in time between the input and the response) due to the associated analog-to-digital and digital-to-analog conversions and anti-aliasing filters, or due to other delays in their implementation.

A digital filter is characterized by its transfer function, or equivalently, its difference equation. Mathematical analysis of the transfer function can describe how it will respond to any input. As such, designing a filter consists of developing specifications appropriate to the problem (for example, a second-order low pass filter with a specific cut-off frequency), and then producing a transfer function which meets the specifications.

The transfer function for a linear, time-invariant, digital filter can be expressed as a transfer function in the *Z*-domain; if it is causal, then it has the form:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

where the order of the filter is the greater of *N* or *M*. See *Z*-transform's LCCD equation for further discussion of this transfer function.

This is the form for a recursive filter with both the inputs (Numerator) and outputs (Denominator), which typically leads to an IIR infinite impulse response behaviour, but if the

denominator is made equal to unity i.e. no feedback, then this becomes an FIR or finite impulse response filter.

ANALYSIS TECHNIQUES

A variety of mathematical techniques may be employed to analyze the behaviour of a given digital filter. Many of these analysis techniques may also be employed in designs, and often form the basis of a filter specification.

Typically, one characterizes filters by calculating how they will respond to a simple input such as an impulse. One can then extend this information to compute the filter's response to more complex signals.

IMPULSE RESPONSE

The impulse response, often denoted $h[k]$ or h_k, is a measurement of how a filter will respond to the Kronecker delta function. For example, given a difference equation, one would set $x_0 = 1$ and $x_k = 0$ for $k \neq 0$ and evaluate. The impulse response is a characterization of the filter's behaviour. Digital filters are typically considered in two categories: infinite impulse response (IIR) and finite impulse response (FIR). In the case of linear time-invariant FIR filters, the impulse response is exactly equal to the sequence of filter coefficients:

$$y_n = \sum_{k=0}^{N} h_k x_{n-k}$$

IIR filters on the other hand are recursive, with the output depending on both current and previous inputs as well as previous outputs. The general form of an IIR filter is thus:

$$\sum_{m=0}^{M} a_m y_{n-m} = \sum_{k=0}^{N} b_k x_{n-k}$$

Plotting the impulse response will reveal how a filter will respond to a sudden, momentary disturbance.

DIFFERENCE EQUATION

In discrete-time systems, the digital filter is often implemented by converting the transfer function to a linear constant-coefficient difference equation (LCCD) via the Z-transform. The discrete frequency-domain transfer function is written as the ratio of two polynomials. For example:

$$H(z) = \frac{(z+1)^2}{(z-\frac{1}{2})(z+\frac{3}{4})}$$

This is expanded:

$$H(z) = \frac{z^2 + 2z + 1}{z^2 + \frac{1}{4}z - \frac{3}{8}}$$

and to make the corresponding filter causal, the numerator and denominator are divided by the highest order of z

$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 + \frac{1}{4}z^{-1} - \frac{3}{8}z^{-2}} = \frac{Y(z)}{X(z)}$$

The coefficients of the denominator, a_k, are the 'feed-backward' coefficients and the coefficients of the numerator are the 'feed-forward' coefficients, b_k. The resultant linear difference equation is:

$$y[n] = -\sum_{k=1}^{M} a_k y[n-k] + \sum_{k=0}^{N} b_k x[n-k]$$

or, for the example above:

$$\frac{Y(z)}{X(z)} = \frac{1 + 2z^{-1} + z^{-2}}{1 + \frac{1}{4}z^{-1} - \frac{3}{8}z^{-2}}$$

rearranging terms:

$$\Rightarrow (1 + \frac{1}{4}z^{-1} - \frac{3}{8}z^{-2})Y(z) = (1 + 2z^{-1} + z^{-2})X(z)$$

then by taking the inverse *z*-transform:

$$\Rightarrow y[n] + \frac{1}{4}y[n-1] - \frac{3}{8}y[n-2] = x[n] + 2x[n-1] + x[n-2]$$

and finally, by solving for $y[n]$:

$$y[n] = -\frac{1}{4}y[n-1] + \frac{3}{8}y[n-2] + x[n] + 2x[n-1] + x[n-2]$$

This equation shows how to compute the next output sample, $y[n]$, in terms of the past outputs, $y[n-p]$, the present input, $x[n]$, and the past inputs, $x[n-p]$. Applying the filter to an input

in this form is equivalent to a Direct Form I or II realization, depending on the exact order of evaluation.

DESIGN OF DIGITAL FILTER FROM ANALOG FILTER

The most common technique used for designing IIR digital filters known as indirect method, involves first designing an analog prototype filter and then transforming the prototype to a digital filter. For the given specifications of a digital filter, the derivation of the digital filter transfer function requires three steps.

1. Map the desired digital filter specifications into those for an equivalent analog filter.
2. Derive the analog transfer function for the analog prototype.
3. Transform the transfer function of the analog prototype into an equivalent digital filter transfer function.

COMPARISON

Analog Filter	Digital Filter
1. Analog filter processes analog inputs and generates analog outputs.	1. A digital filter processes and generates digital data.
2. Analog filters are constructed from active or passive electronic components.	2. A digital filter consists of elements like adder, multiplier and delay unit.
3. Analog filter is described by a differential equation.	3. Digital filter is described by a difference equation.
4. The frequency response of an analog filter can be modified by changing the components.	4. The frequency response can be changed by changing the filter coefficients.

MERITS AND DEMERITS OF DIGITAL FILTER

Advantages

1. Unlike analog filter, the digital filter performance is not influenced by component ageing, temperature and power supply variations.
2. A digital filter is highly immune to noise and possesses considerable parameter stability.
3. Digital filters afford a wide variety of shapes for the amplitude and phase responses.
4. There are no problems of input or output impedance matching with digital filters.
5. Digital filters can be operated over a wide range of frequencies.
6. The coefficients of digital filter can be programmed and altered any time to obtain the desired characteristics.
7. Multiple filtering is possible only in digital filter.

Disadvantage

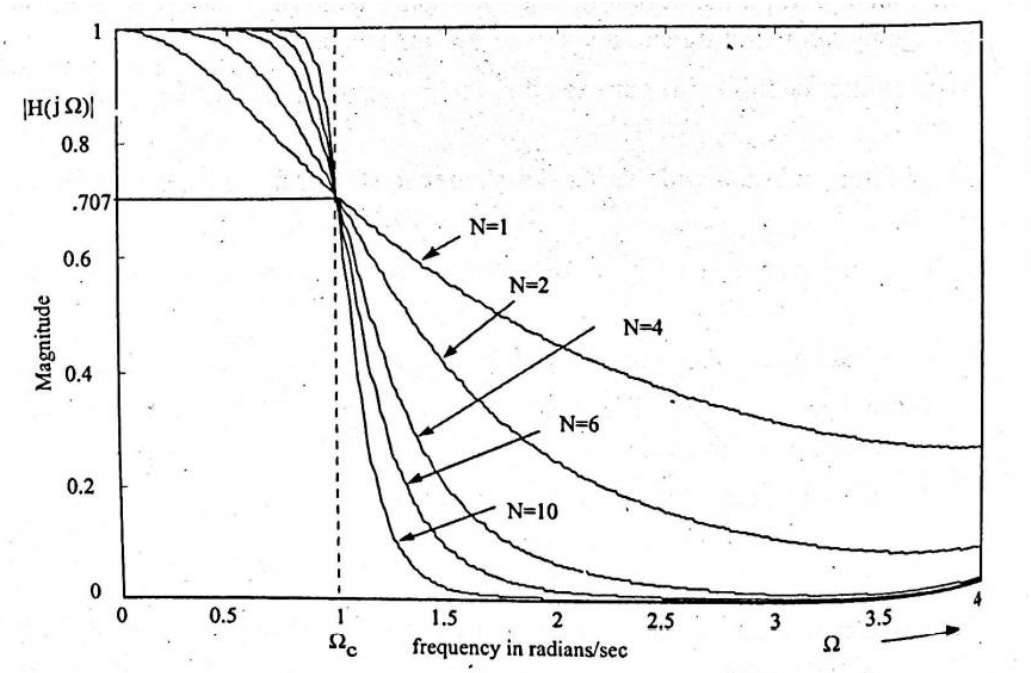
1. The quantization error arises due to finite word length in the representation of signals and parameters.

ANALOG LOW PASS BUTTERWORTH FILTER

The magnitude function of the Butterworth lowpass filter is given by

$$|H(j\Omega)| = \frac{1}{\left[1 + (\Omega/\Omega_c)^{2N}\right]^{1/2}} \quad N = 1, 2, 3, \ldots$$

where N is the order of the filter and Ω_c is the cutoff frequency. As shown in Fig. 5.5 the function is monotonically decreasing, where the maximum response is unity at $\Omega = 0$. The ideal response is shown by the dash line. It can be seen that the magnitude response approaches the ideal lowpass characteristics as the order N increases. For values $\Omega < \Omega_c$; $|H(j\Omega)| \approx 1$, for values $\Omega > \Omega_c$, the value of $|H(j\Omega)|$ decreases rapidly. At $\Omega = \Omega_c$, the curves pass through 0.707, which corresponds to - 3 dB point.



LIST OF BUTTERWORTH POLYNOMIALS

N	Denominator of $H(s)$
1	$s+1$
2	$s^2+\sqrt{2}s+1$
3	$(s+1)(s^2+s+1)$
4	$(s^2+0.76537s+1)(s^2+1.8477s+1)$
5	$(s+1)(s^2+0.61803s+1)(s^2+1.61803s+1)$
6	$(s^2+1.931855s+1)(s^2+\sqrt{2}s+1)(s^2+0.51764s+1)$
7	$(s+1)(s^2+1.80194s+1)(s^2+1.247s+1)(s^2+0.445s+1)$

STEPS TO DESIGN ANALOG LOW PASS BUTTERWORTH FILTER

1. From the given specifications find the order of the filter N.
2. Round off it to the next higher integer.
3. Find the transfer function $H(s)$ for $\Omega_c = 1$ rad/sec for the value of N.
4. Calculate the value of cutoff frequency Ω_c.
5. Find the transfer function $H_a(s)$ for the above value of Ω_c by substituting $s \to \frac{s}{\Omega_c}$ in $H(s)$.

ORDER FORMULA

$$N \geq \frac{\log \sqrt{\frac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}}}{\log \frac{\Omega_s}{\Omega_p}}$$

$$\geq \frac{\log \left(\frac{\lambda}{\varepsilon}\right)}{\log \frac{\Omega_s}{\Omega_p}}$$

$$\varepsilon = (10^{0.1\alpha_p} - 1)^{0.5}$$

$$\lambda = (10^{0.1\alpha_s} - 1)^{0.5}$$

$$A = \frac{\lambda}{\varepsilon} = \left(\frac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}\right)^{0.5}$$

$$k = \frac{\Omega_p}{\Omega_s}$$

$$N \geq \frac{\log A}{\log(1/k)}$$

$$\Omega_c = \frac{\Omega_p}{(10^{0.1\alpha_p} - 1)^{1/2N}} = \frac{\Omega_s}{(10^{0.1\alpha_s} - 1)^{1/2N}}$$

Example Given the specification $\alpha_p = 1\,\text{dB}$; $\alpha_s = 30\,\text{dB}$; $\Omega_p = 200\,\text{rad/sec}$; $\Omega_s = 600\,\text{rad/sec}$. Determine the order of the filter.

$$A = \frac{\lambda}{\varepsilon} = \left(\frac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}\right)^{0.5}$$

$$= \left(\frac{10^3 - 1}{10^{0.1} - 1}\right)^{0.5} = 62.115$$

$$k = \frac{\Omega_p}{\Omega_s} = \frac{200}{600} = \frac{1}{3}$$

$$N \geq \frac{\log A}{\log 1/k}$$

$$\geq \frac{\log 62.115}{\log 3} = 3.758$$

Rounding off N to the next higher integer we get $N = 4$.

Example Determine the order and the poles of lowpass Butterworth filter that has a 3 dB attenuation at 500 Hz and an attenuation of 40 dB at 1000 Hz.

Solution

Given data $\alpha_p = 3\,\text{dB}$; $\alpha_s = 40\,\text{dB}$; $\Omega_p = 2 \times \pi \times 500 = 1000\pi$ rad/sec.
$\Omega_s = 2 \times \pi \times 1000 = 2000\pi$ rad/sec.

$$N \geq \frac{\log \sqrt{\dfrac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}}}{\log \dfrac{\Omega_s}{\Omega_p}}$$

$$\geq \frac{\log \sqrt{\dfrac{10^4 - 1}{10^{0.3} - 1}}}{\log \dfrac{2000\pi}{1000\pi}} = 6.6$$

Rounding 'N' to nearest higher value we get $N = 7$.

The poles of Butterworth filter are given by

$$s_k = \Omega_c e^{j\phi_k} = 1000\pi e^{j\phi_k} \quad k = 1, 2, \ldots 7$$

where $\phi_k = \dfrac{\pi}{2} + \dfrac{(2k-1)\pi}{2N} \quad k = 1, 2, \ldots 7.$

Example Design an analog Butterworth filter that has a -2 dB passband attenuation at a frequency of 20 rad/sec and atleast -10 dB stopband attenuation at 30 rad/sec.

Solution

Given $\alpha_p = 2\,\text{dB}; \quad \Omega_p = 20\,\text{rad/sec}$

$\alpha_s = 10\,\text{dB}; \quad \Omega_s = 30\,\text{rad/sec}$

$$N \geq \frac{\log \sqrt{\dfrac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}}}{\log \dfrac{\Omega_s}{\Omega_p}}$$

$$\geq \frac{\log \sqrt{\dfrac{10 - 1}{10^{0.2} - 1}}}{\log \dfrac{30}{20}}$$

$$\geq 3.37$$

Rounding off N to the next highest integer we get

$$N = 4$$

The normalized lowpass Butterworth filter for $N = 4$ can be found from table as

$$H(s) = \frac{1}{(s^2 + 0.76537s + 1)(s^2 + 1.8477s + 1)}$$

From Eq. (5.31) we have

$$\Omega_c = \frac{\Omega_p}{(10^{0.1\alpha_p} - 1)^{1/2N}} = \frac{20}{(10^{0.2} - 1)^{1/8}} = 21.3868$$

The transfer function for $\Omega_c = 21.3868$ can be obtained by substituting

$$s \rightarrow \frac{s}{21.3868} \text{ in } H(s)$$

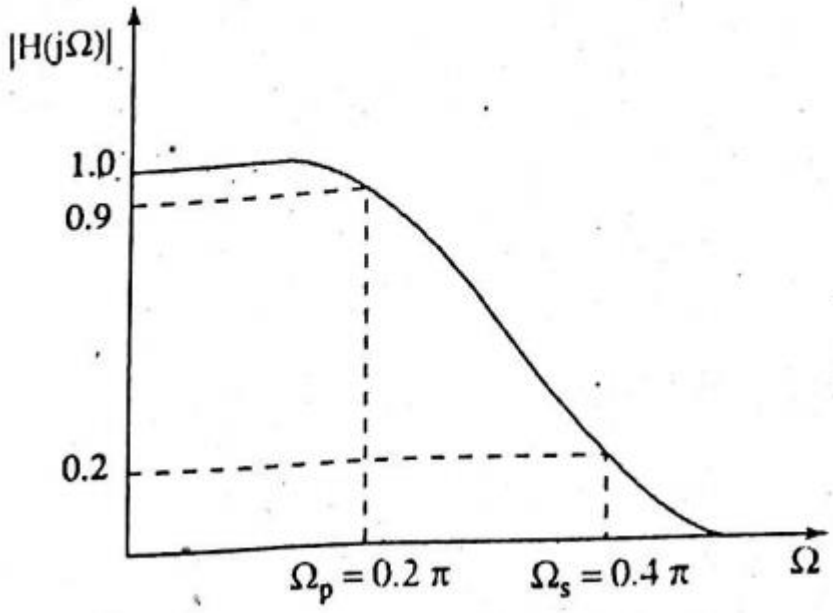
$$\text{i.e., } H(s) = \frac{1}{\left(\frac{s}{21.3868}\right)^2 + 0.76537\left(\frac{s}{21.3868}\right) + 1} \times \frac{1}{\left(\frac{s}{21.3868}\right)^2 + 1.8477\left(\frac{s}{21.3868}\right) + 1}$$

$$= \frac{0.20921 \times 10^6}{(s^2 + 16.3686s + 457.394)(s^2 + 39.5176s + 457.394)}$$

Example For the given specifications design an analog Butterworth filter. $0.9 \leq |H(j\Omega)| \leq 1$ for $0 \leq \Omega \leq 0.2\pi$. $|H(j\Omega)| \leq 0.2$ for $0.4\pi \leq \Omega \leq \pi$.

Solution

From the data we find $\Omega_p = 0.2\pi$; $\Omega_s = 0.4\pi$; $\frac{1}{\sqrt{1+\epsilon^2}} = 0.9$ and $\frac{1}{\sqrt{1+\lambda^2}} = 0.2$ from which we obtain



$$\varepsilon = 0.484 \text{ and } \lambda = 4.898$$

$$N \geq \frac{\log\left(\frac{\lambda}{\varepsilon}\right)}{\log\frac{\Omega_s}{\Omega_p}} = \frac{\log\frac{4.898}{0.484}}{\log\left(\frac{0.4\pi}{0.2\pi}\right)} = 3.34$$

i.e., $N = 4$

From the table 5.1, for $N = 4$, the transfer function of normalised Butterworth filter is

$$H(s) = \frac{1}{(s^2 + 0.76537s + 1)(s^2 + 1.8477s + 1)}$$

we know $\Omega_c = \frac{\Omega_p}{(10^{0.1\alpha_p} - 1)^{1/2N}} = \frac{\Omega_p}{\varepsilon^{1/N}} = \frac{0.2\pi}{(0.484)^{1/4}} = 0.24\pi.$

$H(s)$ for $\Omega_c = 0.24\pi$ can be obtained by substituting $s \rightarrow \frac{s}{0.24\pi}$ in $H(s)$ i.e.,

$$H(s) = \frac{1}{\left\{\left(\frac{s}{0.24\pi}\right)^2 + 0.76537\left(\frac{s}{0.24\pi}\right) + 1\right\}} \times \frac{1}{\left(\frac{s}{0.24\pi}\right)^2 + 1.8477\left(\frac{s}{0.24\pi}\right) + 1}$$

$$= \frac{0.323}{(s^2 + 0.577s + 0.0576\pi^2)(s^2 + 1.393s + 0.0576\pi^2)}$$

COKPARISON BETWEEN BUTTERWORTH FILTER AND CHEBYSHEV FILTER

1. The magnitude response of Butterworth filter decreases monotonically as the frequency Ω increases from 0 to ∞, whereas the magnitude response of the Chebyshev filter exhibits ripples in the passband or stopband according to the type.
2. The transition band is more in Butterworth filter when compared to Chebyshev filter.
3. The poles of the Butterworth filter lie on a circle, whereas the poles of the Chebyshev filter lie on an ellipse.
4. For the same specifications, the number of poles in Butterworth are more when compared to the Chebyshev filter i.e., the order of the Chebyshev filter is less than that of Butterworth. This is a great advantage because less number of discrete components will be necessary to construct the filter.

STEPS TO DESIGN ANALOG LOW PASS CHEBYSHEV FILTER

1. From the given specifications find the order of the filter N.
2. Round off it to the next higher integer.
3. Using the following formulas find the values of a and b, which are minor and major axis of the ellipse respectively.

$$a = \Omega_p \frac{[\mu^{1/N} - \mu^{-1/N}]}{2}; \quad b = \Omega_p \left[\frac{\mu^{1/N} + \mu^{-1/N}}{2}\right]$$

where

$$\mu = \varepsilon^{-1} + \sqrt{\varepsilon^{-2} + 1}$$

$$\varepsilon = \sqrt{10^{0.1\alpha_p} - 1}$$

Ω_p = Passband frequency

α_p = Maximum allowable attenuation in the passband

(∵ For normalized Chebyshev filter $\Omega p = 1$ rad/sec)

4. Calculate the poles of Chebyshev filter which lie on an ellipse by using the formula.

$$s_k = a\cos\phi_k + jb\sin\phi_k \quad k = 1, 2, \ldots, N$$

$$\text{where } \phi_k = \frac{\pi}{2} + \left(\frac{2k-1}{2N}\right)\pi \quad k = 1, 2, \ldots, N$$

5. Find the denominator polynomial of the transfer function using the above poles.

6. The numerator of the transfer function depends on the value of N.

   (a) For N odd substitute $s = 0$ in the denominator polynomial and find the value. This value is equal to the numerator of the transfer function.
   (∵ For N odd the magnitude response $|H(j\Omega)|$ starts at 1.)

   (b) For N even substitute $s = 0$ in the denominator polynomial and divide the result by $\sqrt{1+\varepsilon^2}$. This value is equal to the numerator.

Example Given the specifications $\alpha_p = 3\,\text{dB}$; $\alpha_s = 16\,\text{dB}$; $f_p = 1\,\text{KHz}$ and $f_s = 2\,\text{KHz}$. Determine the order of the filter using Chebyshev approximation. Find $H(s)$. □

Solution

From the given data we can find

$$\Omega_p = 2\pi \times 1000\,\text{Hz} = 2000\,\pi\,\text{rad/sec}$$
$$\Omega_s = 2\pi \times 2000\,\text{Hz} = 4000\,\pi\,\text{rad/sec}$$

and $\alpha_p = 3\,\text{dB}$; $\alpha_s = 16\,\text{dB}$.

Step 1:

$$N \geq \frac{\cosh^{-1}\sqrt{\dfrac{10^{0.1\alpha_s}-1}{10^{0.1\alpha_p}-1}}}{\cosh^{-1}\dfrac{\Omega_s}{\Omega_p}} = \cosh^{-1}\frac{\sqrt{\dfrac{10^{1.6}-1}{10^{0.3}-1}}}{\cosh^{-1}\dfrac{4000\pi}{2000\pi}}$$
$$= 1.91$$

Step 2: Rounding N to next higher value we get $N = 2$.

For N even, the oscillatory curve starts from $\dfrac{1}{\sqrt{1+\varepsilon^2}}$.

Step 3: The values of minor axis and major axis can be found as below

$$\varepsilon = (10^{0.1\alpha_p} - 1)^{0.5} = (10^{0.3} - 1)^{0.5} = 1$$
$$\mu = \varepsilon^{-1} + \sqrt{1+\varepsilon^{-2}} = 2.414$$
$$a = \Omega_p \frac{[\mu^{1/N} - \mu^{-1/N}]}{2} = 2000\pi \frac{[(2.414)^{1/2} - (2.414)^{-1/2}]}{2} = 910\pi$$
$$b = \Omega_p \frac{[\mu^{1/N} + \mu^{-1/N}]}{2} = 2000\pi \frac{[(2.414)^{1/2} + (2.414)^{-1/2}]}{2} = 2197\pi$$

Step 4: The poles are given by

$$s_k = a\cos\phi_k + jb\sin\phi_k, \quad k = 1, 2$$

$$\phi_k = \frac{\pi}{2} + \frac{(2k-1)\pi}{2N} \quad k = 1, 2$$

$$\phi_1 = \frac{\pi}{2} + \frac{\pi}{4} = 135°$$

$$\phi_2 = \frac{\pi}{2} + \frac{3\pi}{4} = 225°$$

$$s_1 = a\cos\phi_1 + j\, b\sin\phi_1 = -643.46\pi + j1554\pi$$

$$s_2 = a\cos\phi_2 + j\, b\sin\phi_2 = -643.46\pi - j1554\pi$$

Step 5: The denominator of $H(s) = (s + 643.46\pi)^2 + (1554\pi)^2$

Step 6: The numerator of $H(s) = \dfrac{(643.46\pi)^2 + (1554\pi)^2}{\sqrt{1+\varepsilon^2}} = (1414.38)^2\pi^2$

The transfer function $H(s) = \dfrac{(1414.38)^2\pi^2}{s^2 + 1287\pi s + (1682)^2\pi^2}$.

Example Obtain an analog Chebyshev filter transfer function that satisfies the constraints $\frac{1}{\sqrt{2}} \leq |H(j\Omega)| \leq 1; \quad 0 \leq \Omega \leq 2$

□

$$|H(j\Omega)| < 0.1; \quad \Omega \geq 4$$

Solution

Step 1: From the given data we can find that

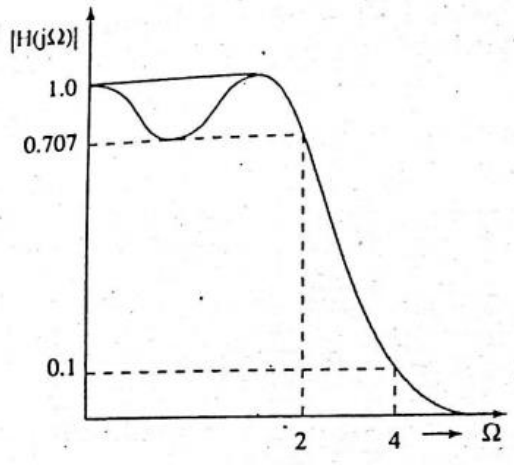
$$\frac{1}{\sqrt{1+\varepsilon^2}} = \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{1+\lambda^2}} = 0.1,$$

$\Omega_p = 2$ and $\Omega_s = 4$, from which we can obtain $\varepsilon = 1$ and $\lambda = 9.95$.

We know

$$N \geq \frac{\cosh^{-1}\frac{\lambda}{\varepsilon}}{\cosh^{-1}\frac{\Omega_s}{\Omega_p}} = \frac{\cosh^{-1} 9.95}{\cosh^{-1} 2} = 2.269$$

Step 2: Rounding N to next higher value we get $N = 3$. For N odd, the oscillatory curve starts from unity as shown in Fig.



Step 3: Finding the values of a and b

$$\mu = \varepsilon^{-1} + \sqrt{1+\varepsilon^{-2}} = 2.414$$

$$a = \Omega_p \left[\frac{\mu^{1/N} - \mu^{-1/N}}{2}\right] = 2\left[\frac{(2.414)^{1/3} - (2.414)^{-1/3}}{2}\right]$$

$$= 0.596$$

$$b = \Omega_p \left[\frac{\mu^{1/N} + \mu^{-1/N}}{2}\right] = 2\left[\frac{(2.414)^{1/3} + (2.414)^{-1/3}}{2}\right]$$

$$= 2.087$$

Step 4: To calculate the poles of Chebyshev filter

$$\phi_k = \frac{\pi}{2} + \frac{(2k-1)\pi}{2N} \quad k = 1, 2, 3$$

$$\phi_1 = 120°,\ \phi_2 = 180°,\ \phi_3 = 240°$$

We know $s_k = a\cos\phi_k + jb\sin\phi_k \quad k = 1, 2, 3$ from which we get

$$s_1 = a\cos\phi_1 + jb\sin\phi_1 = 0.596\cos 120° + j2.087\sin 120° = -0.298 + j1.807$$

$$s_2 = a\cos\phi_2 + jb\sin\phi_2 = 0.596\cos 180° + j2.087\sin 180° = -0.596$$

$$s = a\cos\phi_3 + jb\sin\phi_3 = 0.596\cos 240° + j2.087\sin 240° = -0.298 - j1.807$$

Step 5: The denominator polynomial is given by

$$(s+0.596)\{(s+0.298) - j1.807\}\{(s+0.298) + j1.807\}$$

$$\therefore (s+0.596)[(s+0.298)^2 + (1.807)^2]$$

$$= (s+0.596)(s^2 + 0.596s + 3.354)$$

Step 6: The numerator of $H(s)$ can be obtained by substituting $s = 0$ (for N odd) in the denominator.

Therefore the numerator of $H(s) = 2$

The transfer function of Chebyshev filter for the given specifications is given by

$$H(s) = \frac{2}{(s + 0.596)(s^2 + 0.596s + 3.354)}$$

FIR FILTERS & ITS DESIGNING

In signal processing, a **finite impulse response (FIR)** filter is a filter whose impulse response (or response to any finite length input) is of *finite* duration, because it settles to zero in finite time. This is in contrast to infinite impulse response (IIR) filters, which may have internal feedback and may continue to respond indefinitely (usually decaying).

The impulse response (that is, the output in response to a Kronecker delta input) of an Nth-order discrete-time FIR filter lasts exactly $N + 1$ samples (from first nonzero element through last nonzero element) before it then settles to zero.

FIR filters can be discrete-time or continuous-time, and digital or analog.

For a causal discrete-time FIR filter of order *N*, each value of the output sequence is a weighted sum of the most recent input values**:**

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \\ &= \sum_{i=0}^{N} b_i \cdot x[n-i], \end{aligned}$$

where**:**

- $x[n]$ is the input signal,
- $y[n]$ is the output signal,
- N is the filter order; an Nth-order filter has $(N+1)$ terms on the right-hand side
- b_i is the value of the impulse response at the *i*'th instant for $0 \le i \le N$ of an Nth-order FIR filter. If the filter is a direct form FIR filter then b_i is also a coefficient of the filter .

This computation is also known as discrete convolution.

The $x[n-i]$ in these terms are commonly referred to as *taps*, based on the structure of a tapped delay line that in many implementations or block diagrams provides the delayed inputs to the multiplication operations. One may speak of a *5th order/6-tap filter*, for instance.

The impulse response of the filter as defined is nonzero over a finite duration. Including zeros, the impulse response is the infinite sequence:

1. FIR filters are always stable.
2. FIR filters with exactly linear phase can easily be designed.
3. FIR filters can be realized in both recursive and non-recursive structures.
4. FIR filters are free of limit cycle oscillations, when implemented on a finite-word length digital system.
5. Excellent design methods are available for various kinds of FIR filters.

The disadvantages of FIR filter are

1. The implementation of narrow transition band FIR filters are very costly, as it requires considerably more arithmetic operations and hardware components such as multipliers, adders and delay elements.
2. Memory requirement and execution time are very high.

LINEAR PHASE FILR FILTER

Table Summary of characteristics of linear phase FIR filters

Type	Frequency response $H(e^{j\omega})$	Magnitude response $\lvert\overline{H}(e^{j\omega})\rvert$	Phase response $\angle H(e^{j\omega})$	Applications
Symmetrical impulse response N odd	$e^{-j\omega(N-1)/2}\left[\sum_{n=0}^{\frac{N-1}{2}} a(n)\cos\omega n\right]$ $a(0) = h\left(\frac{N-1}{2}\right)$ $a(n) = 2h\left[\left(\frac{N-1}{2}\right) - n\right]$	$\left\lvert\sum_{n=0}^{\frac{N-1}{2}} a(n)\cos\omega n\right\rvert$	$-\alpha\omega + \theta$ where $\theta = 0$ for $\overline{H}(e^{j\omega}) > 0$ $\theta = \pi$ for $\overline{H}(e^{j\omega}) < 0$	lowpass, highpass, bandpass, bandstop,
Symmetrical impulse response, N even	$e^{-j\omega(N-1)/2}\left[\sum_{n=1}^{\frac{N}{2}} b(n)\cos\left(n-\frac{1}{2}\right)\omega\right]$ $b(n) = 2h\left(\frac{N}{2} - n\right)$	$\left\lvert\sum_{n=1}^{\frac{N}{2}} b(n)\cos\left(n-\frac{1}{2}\right)\omega\right\rvert$	$-\alpha\omega + \theta$ where $\theta = 0$ for $\overline{H}(e^{j\omega}) > 0$ $\theta = \pi$ for $\overline{H}(e^{j\omega}) < 0$	lowpass, bandpass

(Continuous)

Table *Continuous*

Type	Frequency response $H(e^{j\omega})$	Magnitude response $\lvert\overline{H}(e^{j\omega})\rvert$	Phase response $\angle H(e^{j\omega})$	Applications
Antisymmetrical impulse response N odd	$e^{j\pi/2}e^{-j\omega(N-1)/2}\sum_{n=1}^{\frac{N-1}{2}} c(n)\sin\omega n$ $c(n) = 2h\left(\frac{N-1}{2} - n\right)$	$\left\lvert\sum_{n=1}^{\frac{N-1}{2}} c(n)\sin\omega n\right\rvert$	$-\alpha\omega + \frac{\pi}{2} + \theta$ where $\theta = 0$ for $\overline{H}(e^{j\omega}) > 0$ $\theta = \pi$ for $\overline{H}(e^{j\omega}) < 0$	differentiator, Hilbert-transformer
Antisymmetrical impulse response N even	$e^{j\pi/2}e^{-j\omega(N-1)/2}\sum_{n=1}^{\frac{N}{2}} d(n)\sin\omega\left(n-\frac{1}{2}\right)$ $d(n) = 2h\left(\frac{N}{2} - n\right)$	$\left\lvert\sum_{n=1}^{\frac{N}{2}} d(n)\sin\omega\left(n-\frac{1}{2}\right)\right\rvert$	$-\alpha\omega + \frac{\pi}{2} + \theta$ where $\theta = 0$ for $\overline{H}(e^{j\omega}) > 0$ $\theta = \pi$ for $\overline{H}(e^{j\omega}) < 0$	differentiator, Hilbert-transformer

The Fourier Series Method of Designing FIR Filters

The frequency response $H(e^{j\omega})$ of a system is periodic in 2π. From Fourier series analysis we know that any periodic function can be expressed as a linear combination of complex exponentials. Therefore, the desired frequency response of an FIR filter can be represented by the Fourier series

$$H_d(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_d(n)e^{-j\omega n}$$

where the Fourier coefficients $h_d(n)$ are the desired impulse response sequence of the filter

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega$$

The z-transform of the sequence is given by

$$H(z) = \sum_{n=-\infty}^{\infty} h_d(n)z^{-n}$$

The Eq. (6.47) represents a non-causal digital filter of infinite duration. To get an FIR filter transfer function, the series can be truncated by assigning

$$h(n) = h_d(n) \text{ for } |n| \leq \frac{N-1}{2}$$
$$= 0 \quad \text{otherwise}$$

Then

$$H(z) = \sum_{h=-\left(\frac{N-1}{2}\right)}^{\frac{N-1}{2}} h(n)z^{-n}$$

$$= h\left(\frac{N-1}{2}\right) z^{-(N-1)/2} + \ldots + h(1)z^{-1} + h(0) + h(-1)z + h(-2)z^2 + \ldots$$

$$+ h\left[-\left(\frac{N-1}{2}\right)\right] z^{(N-1)/2}$$

$$= h(0) + \sum_{n=1}^{\frac{N-1}{2}} [h(n)z^{-n} + h(-n)z^n]$$

For a symmetrical impulse response having symmetry at $n = 0$

$$h(-n) = h(n)$$

Therefore, Eq. (6.50) can be written as

$$H(z) = h(0) + \sum_{n=1}^{\frac{N-1}{2}} h(n)[z^n + z^{-n}]$$

The above transfer function is not physically realizable. Realizability can be brought by multiplying Eq. (6.52) by $z^{-(N-1)/2}$ where $\frac{N-1}{2}$ is delay in samples.

$$H'(z) = z^{-(N-1)/2} H(z)$$

$$= z^{-(N-1)/2} \left[h(0) + \sum_{n=1}^{\frac{N-1}{2}} h(n)(z^n + z^{-n})\right]$$

Example Design an ideal lowpass filter with a frequency response

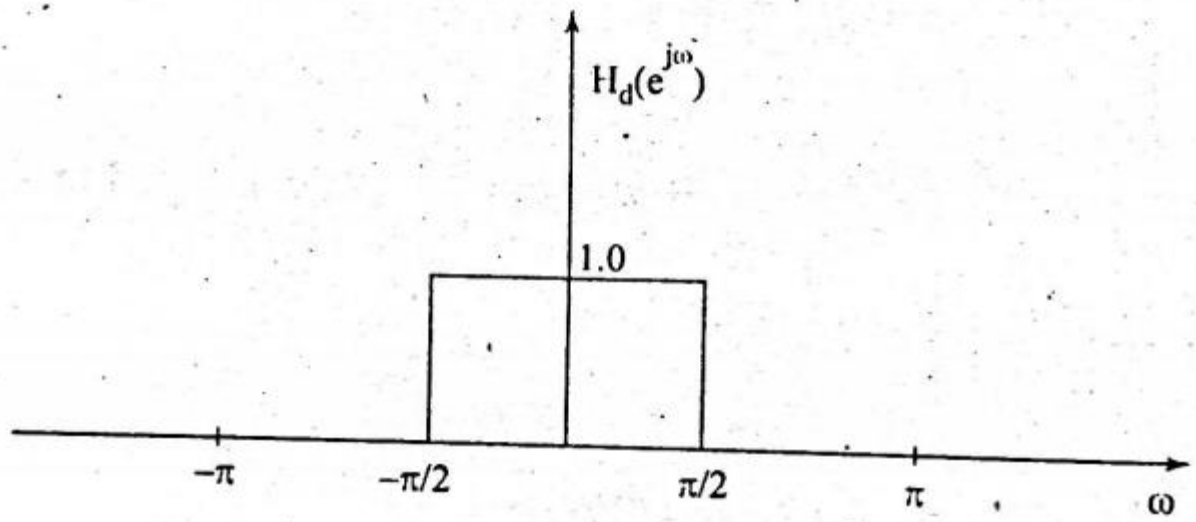
$$H_d(e^{j\omega}) = 1 \text{ for } -\frac{\pi}{2} \le \omega \le \frac{\pi}{2}$$
$$= 0 \text{ for } \frac{\pi}{2} \le |\omega| \le \pi$$

Find the values of $h(n)$ for $N = 11$. Find $H(z)$. Plot the magnitude response.

Solution

The frequency response of lowpass filter with $\omega_c = \frac{\pi}{2}$ is shown in Fig. Given

$$H_d(e^{j\omega}) = 1 \text{ for } \frac{-\pi}{2} \le \omega \le \frac{\pi}{2}$$
$$= 0 \text{ for } \frac{\pi}{2} \le |\omega| \le \pi$$



From the frequency response we can find that $\alpha = 0$. Therefore, we get a non-causal filter coefficients symmetrical about $n = 0$, i.e., $h_d(n) = h_d(-n)$. The filter coefficients can be obtained by using the formula given in table 6.2 for zero phase frequency response (or) we can proceed as follows.

We know

$$
\begin{aligned}
h_d(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega \\
&= \frac{1}{2\pi}\int_{\frac{-\pi}{2}}^{\frac{\pi}{2}} e^{j\omega n}d\omega \\
&= \frac{1}{2\pi jn}e^{j\omega n}\Big|_{-\pi/2}^{\pi/2} \\
&= \frac{1}{\pi n(2j)}\left[e^{j\pi n/2} - e^{-j\pi n/2}\right] \\
&= \frac{\sin\frac{\pi}{2}n}{\pi n} \qquad -\infty \le n \le \infty
\end{aligned}
$$

Truncating $h_d(n)$ to 11 samples, we have

$$
\begin{aligned}
h(n) &= \frac{\sin\frac{\pi}{2}n}{\pi n} \quad \text{for} \quad |n| \le 5 \\
&= 0 \quad \text{otherwise}
\end{aligned}
$$

For $n = 0$ Eq. (6.56) becomes indeterminate. So

$$
\begin{aligned}
h(0) &= \lim_{n\to 0}\frac{\sin\frac{\pi}{2}n}{\pi n} = \frac{1}{2}\lim_{n\to 0}\frac{\sin\frac{\pi}{2}n}{\frac{\pi n}{2}} \\
&= \frac{1}{2}
\end{aligned}
\qquad \boxed{\therefore \lim_{\theta\to 0}\frac{\sin\theta}{\theta} = 1}
$$

$$h(0) = h_d(0) = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2} d\omega = \frac{1}{2\pi}\omega\Big|_{-\pi/2}^{\pi/2} = \frac{\pi}{2\pi} = \frac{1}{2}.$$

For $n = 1$

$$h(1) = h(-1) = \frac{\sin\frac{\pi}{2}}{\pi} = \frac{1}{\pi} = 0.3183.$$

Similarly

$$h(2) = h(-2) = \frac{\sin\pi}{2\pi} = 0$$

$$h(3) = h(-3) = \frac{\sin\frac{3\pi}{2}}{3\pi} = -\frac{1}{3\pi} = -0.106$$

$$h(4) = h(-4) = \frac{\sin\frac{4\pi}{2}}{4\pi} = 0$$

$$h(5) = h(-5) = \frac{\sin\frac{5\pi}{2}}{5\pi} = \frac{1}{5\pi} = 0.06366.$$

The transfer function of the filter is given by

$$H(z) = h(0) + \sum_{n=1}^{\frac{N-1}{2}} \left[h(n)\left(z^n + z^{-n}\right)\right]$$

$$= 0.5 + \sum_{n=1}^{5} h(n)\left(z^n + z^{-n}\right)$$

$$= 0.5 + 0.3183\left(z^1 + z^{-1}\right) - 0.106\left(z^3 + z^{-3}\right) + 0.06366\left(z^5 + z^{-5}\right)$$

The transfer function of the realizable filter is

$$H'(z) = z^{-(N-1)/2}H(z)$$

$$= z^{-5}\left[0.5 + 0.3183\left(z + z^{-1}\right) - 0.106\left(z^3 + z^{-3}\right) + 0.06366\left(z^5 + z^{-5}\right)\right]$$

$$= 0.06366 - 0.106z^{-2} + 0.3183z^{-4} + 0.5z^{-5} + 0.3183z^{-6}$$

$$- 0.106z^{-8} + 0.06366z^{-10}$$

From the above Eq. (6.57) the filter coefficients of causal filter are given by

$$h(0) = h(10) = 0.06366; \quad h(1) = h(9) = 0; \quad h(2) = h(8) = -0.106$$

$$h(3) = h(7) = 0; \quad h(4) = h(6) = 0.3183; \quad h(5) = 0.5$$

The frequency response is given by

$$\overline{H}(e^{j\omega}) = \sum_{n=0}^{5} a(n)\cos\omega n \quad \text{where}$$

$$a(0) = h\left(\frac{N-1}{2}\right) = h(5) = 0.5$$

$$a(n) = 2h\left(\frac{N-1}{2} - n\right)$$

$$a(1) = 2h(5-1) = 2h(4) = 0.6366$$

$$a(2) = 2h(5-2) = 2h(3) = 0$$

$$a(3) = 2h(5-3) = 2h(2) = -0.212$$

$$a(4) = 2h(5-4) = 2h(1) = 0$$

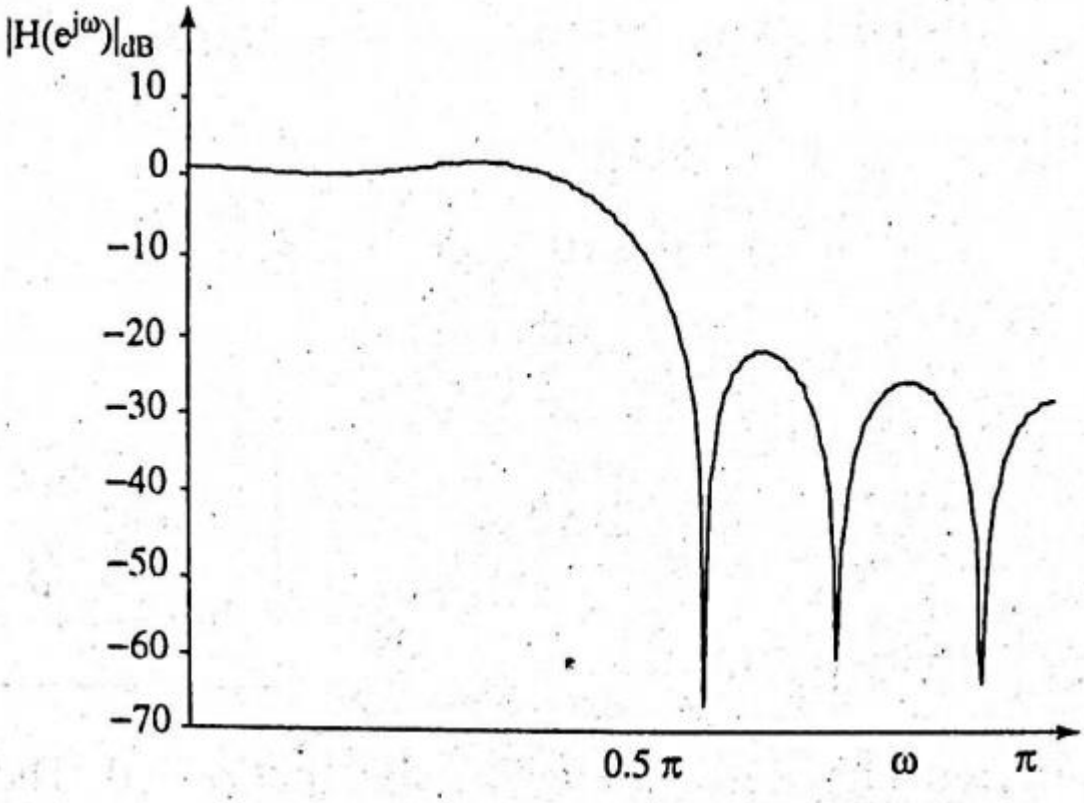
$$a(5) = 2h(5-5) = 2h(0) = 0.127$$

$$\overline{H}(e^{j\omega}) = 0.5 + 0.6366\cos\omega - 0.212\cos 3\omega + 0.127\cos 5\omega$$

The magnitude in dB is calculated by varying ω from 0 to π and tabulated below. The magnitude $|H(e^{j\omega})|_{dB} = 20 \log |H(e^{j\omega})|$.

ω(in degrees)	0	10	20	30	40	50	60	70	80
$\left\|H(e^{j\omega})\right\|_{dB}$	0.4	0.21	-0.26	-0.517	-0.21	0.42	0.77	0.21	-1.79

90	100	110	120	130	140	150	160	170	180
-6	-14.56	-31.89	-20.6	-26	-32	-24.7	-30.55	-32	-26



If an FIR filter is non-causal, the range of nonzero values in its impulse response can start before $n = 0$, with the defining formula appropriately generalized.

An FIR filter has a number of useful properties which sometimes make it preferable to an infinite impulse response (IIR) filter. FIR filters:

Require no feedback. This means that any rounding errors are not compounded by summed iterations. The same relative error occurs in each calculation. This also makes implementation simpler.

Are inherently stable, since the output is a sum of a finite number of finite multiples of the input values, so can be no greater than $\sum |b_i|$ times the largest value appearing in the input.

Can easily be designed to be linear phase by making the coefficient sequence symmetric. This property is sometimes desired for phase-sensitive applications, for example data communications, seismology, crossover filters, and mastering.

The main disadvantage of FIR filters is that considerably more computation power in a general purpose processor is required compared to an IIR filter with similar sharpness or selectivity, especially when low frequency (relative to the sample rate) cutoffs are needed. However many digital signal processors provide specialized hardware features to make FIR filters approximately as efficient as IIR for many applications.

An FIR filter is designed by finding the coefficients and filter order that meet certain specifications, which can be in the time-domain (e.g. a matched filter) and/or the frequency domain (most common). Matched filters perform a cross-correlation between the input signal and a known pulse-shape. The FIR convolution is a cross-correlation between the input signal and a time-reversed copy of the impulse-response. Therefore, the matched-filter's impulse response is "designed" by sampling the known pulse-shape and using those samples in reverse order as the coefficients of the filter.[1]

When a particular frequency response is desired, several different design methods are common:

1. Window design method
2. Frequency Sampling method
3. Weighted least squares design
4. Parks-McClellan method (also known as the Equiripple, Optimal, or Minimax method). The Remez exchange algorithm is commonly used to find an optimal equiripple set of coefficients. Here the user specifies a desired frequency response, a weighting function for errors from this response, and a filter order *N*. The algorithm then finds the set of $(N+1)$ coefficients that minimize the maximum deviation from the ideal. Intuitively, this finds the filter that is as close as you can get to the desired response given that you can use only $(N+1)$ coefficients. This method is particularly easy in practice since at least one text[2] includes a program that takes the desired filter and *N*, and returns the optimum coefficients.

5. Equiripple FIR filters can be designed using the FFT algorithms as well.[3] The algorithm is iterative in nature. You simply compute the DFT of an initial filter design that you have using the FFT algorithm (if you don't have an initial estimate you can start with h[n]=delta[n]). In the Fourier domain or FFT domain you correct the frequency response according to your desired specs and compute the inverse FFT. In time-domain you retain only N of the coefficients (force the other coefficients to zero). Compute the FFT once again. Correct the frequency response according to specs.

Software packages like MATLAB, GNU Octave, Scilab, and SciPy provide convenient ways to apply these different methods.

In the window design method, one first designs an ideal IIR filter and then truncates the infinite impulse response by multiplying it with a finite length window function. The result is a finite impulse response filter whose frequency response is modified from that of the IIR filter. Multiplying the infinite impulse by the window function in the time domain results in the frequency response of the IIR being convolved with the Fourier transform (or DTFT) of the window function. If the window's main lobe is narrow, the composite frequency response remains close to that of the ideal IIR filter.

The ideal response is usually rectangular, and the corresponding IIR is a sinc function. The result of the frequency domain convolution is that the edges of the rectangle are tapered, and ripples appear in the passband and stopband. Working backward, one can specify the slope (or width) of the tapered region (*transition band*) and the height of the ripples, and thereby derive the frequency domain parameters of an appropriate window function. Continuing backward to an impulse response can be done by iterating a filter design program to find the minimum filter order. Another method is to restrict the solution set to the parametric family of Kaiser windows, which provides closed form relationships between the time-domain and frequency domain parameters. In general, that method will not achieve the minimum possible filter order, but it is particularly convenient for automated applications that require dynamic, on-the-fly, filter design.

The window design method is also advantageous for creating efficient half-band filters, because the corresponding sinc function is zero at every other sample point (except the center one). The product with the window function does not alter the zeros, so almost half of the coefficients of the final impulse response are zero. An appropriate implementation of the FIR calculations can exploit that property to double the filter's efficiency.

A moving average filter is a very simple FIR filter. It is sometimes called a boxcar filter, especially when followed by decimation. The filter coefficients, $b_0, \ldots, b_N$, are found via the following equation:

$$b_i = \frac{1}{N+1}$$

To provide a more specific example, we select the filter order:

$$N = 2$$

The impulse response of the resulting filter is:

$$h[n] = \frac{1}{3}\delta[n] + \frac{1}{3}\delta[n-1] + \frac{1}{3}\delta[n-2]$$

The Fig. (a) on the right shows the block diagram of a 2nd-order moving-average filter discussed below. The transfer function is:

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

Fig. (b) on the right shows the corresponding pole–zero diagram. Zero frequency (DC) corresponds to (1,0), positive frequencies advancing counterclockwise around the circle to the Nyquist frequency at (-1,0). Two poles are located at the origin, and two zeros are located at $z_1 = -\frac{1}{2} + j\frac{\sqrt{3}}{2}$, $z_2 = -\frac{1}{2} - j\frac{\sqrt{3}}{2}$.

The frequency response, in terms of normalized frequency ω, is:

$$H\left(e^{j\omega}\right) = \frac{1}{3} + \frac{1}{3}e^{-j\omega} + \frac{1}{3}e^{-j2\omega}.$$

Fig. (c) on the right shows the magnitude and phase components of $H\left(e^{j\omega}\right)$. But plots like these can also be generated by doing a discrete Fourier transform (DFT) of the impulse response.[note 2] And because of symmetry, filter design or viewing software often displays only the [0,π] region. The magnitude plot indicates that the moving-average filter passes low frequencies with a gain near 1 and attenuates high frequencies, and is thus a crude low-pass filter. The phase plot is linear except for discontinuities at the two frequencies where the magnitude goes to zero. The size of the discontinuities is π, representing a sign reversal. They do not affect the property of linear phase. That fact is illustrated in Fig. (d).

The frequency response of an ideal low pass filter is shown in the image below. The frequency axis is normalised with respect to the sampling frequency. The cut-off, or transition frequency (f_t) is always between 0 and 0.5, as 0.5 represents the Nyquist frequency. As you would expect from a low pass filter, all frequencies below f_t are passed, where-as all those above are stopped.

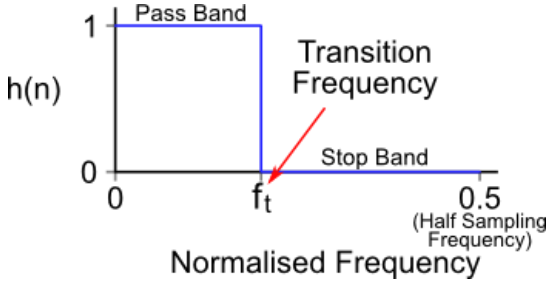
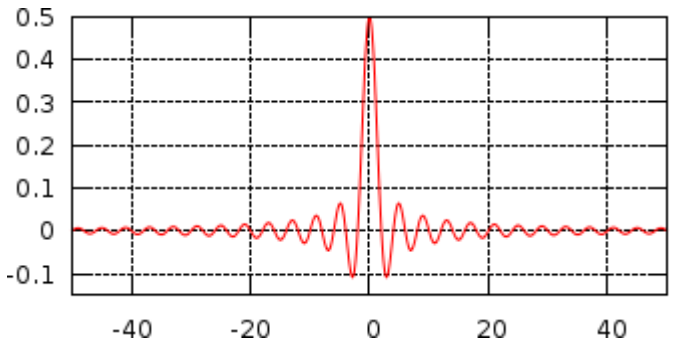


Fig. 1: Transition Frequency

The impulse response of this ideal low pass filter is shown below, it is a sinc function. The equation is shown next to the plot. If we could create a filter with this impulse response we would have an ideal low pass filter like that shown above. A set of Gnuplot commands are also given for recreating this graph.



$$h(x) = 2f_t\ sinc(2\pi f_t x)$$
$$= 2f_t \frac{sin\,(2\pi f_t x)}{2\pi f_t x}$$
$$= \frac{sin\,(2\pi f_t x)}{\pi x}$$

Fig. 2: Impulse response of ideal LPF (ft=0.25)

Below is a plot of impulse responses for different values of f_t.

"

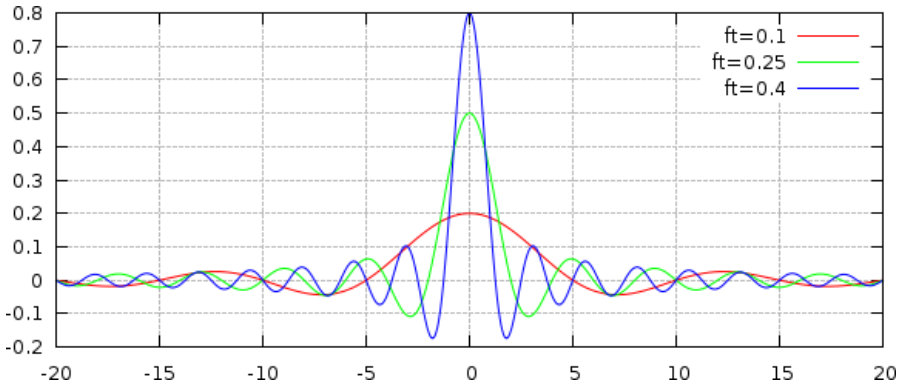


Fig. 3: Sinc function for different values of normalised transition frequency

Unfortunately it is not as easy as that. Given the non-recursive filter structure like that shown below, there are two problems with creating this ideal impulse response.

- First, the sinc function is infinite in the x direction, the ripples keep on going in both directions. However, the FIR filter only allows us to create finite impulse responses, the number of filter taps must be finite.
- Second, the impulse response is non-casual, this means an implementation would require samples from the future.

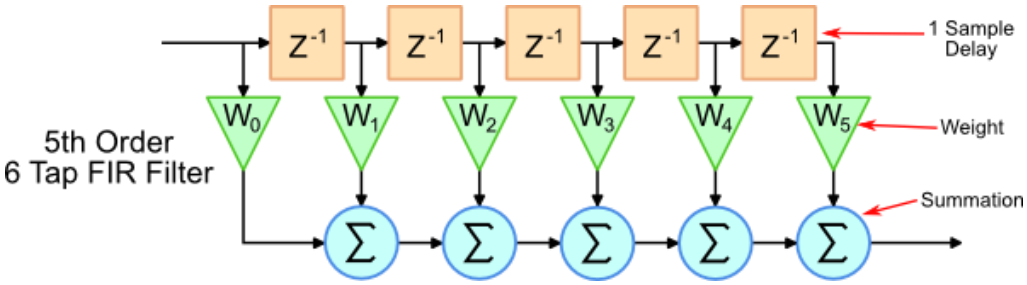


Fig. 4: Non-Recursive, Finite Impulse Response Filter

Luckily, the solutions to these issues are quite simple. First, a window is applied to the sinc function such that only a portion of the impulse response is actually used. Secondly, the impulse response is shifted such that the filter only operates on available samples (those from the past). These techniques are demonstrated in the the following example.

3.6 LOW PASS FILTER EXAMPLE

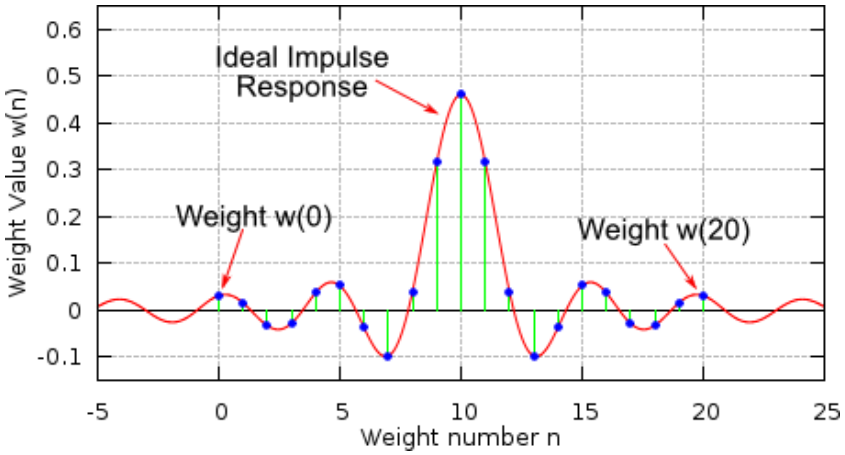


Fig. 4: Filter weights M=20, ft=0.23

Consider the filter with the properties given below.

- **Filter Type:** Low Pass
- **Sampling Frequency:** 2000 Hz
- **Cut off Frequency:** 460 Hz
- **Filter Length (# weights):** 21

The plot to the right shows the filter weights that have been calculated using the equations below.

- ***M*** **-** This is the filter order, it is always equal to the number of taps minus 1

- f_t - This is the normalised transition frequency.

There are 21 weights that fit on the ideal impulse response curve. It can be seen how the impulse response is effectively cropped by only using 21 weights.

Note: When calculating weight values with an odd number of weights, a divide by zero will occur at $n=M/2$. Therefore, based on l'Hôpital's rule, the value of $2f_t$ is used.

$$M = \text{filter length} - 1 = 20$$

$$f_t = \frac{\text{Cut off frequency}}{\text{Sampling Frequency}} = \frac{460}{2000} = 0.23$$

$$w_{lpf}(n) = \begin{cases} \frac{sin\left[2\pi f_t\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} & n \neq \frac{M}{2} \\ 2f_t & n = \frac{M}{2} \end{cases}$$

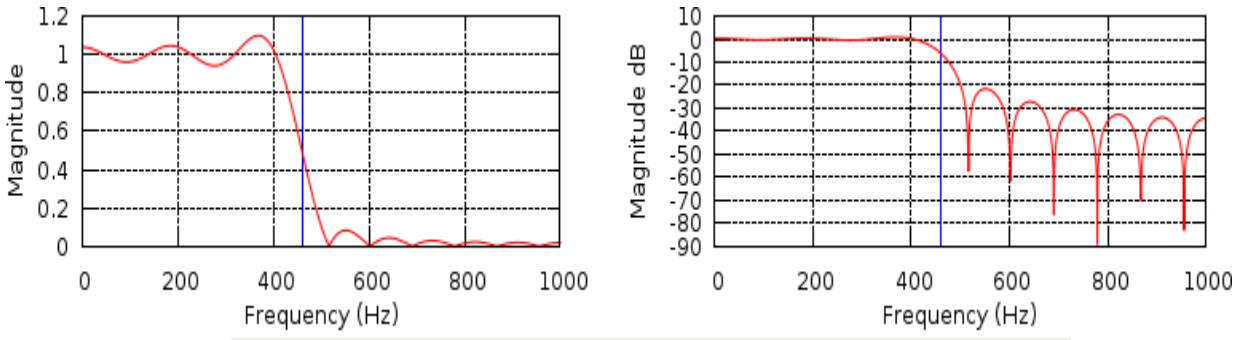


Fig. 5: The large amount of ripple visible on the non-dB plot is due to the rather crude approach of truncating the infinite ideal impulse response. The approach that has just been used is called applying a **Rectangular Window**. The next section describes different window types that can decrease the ripple and improve the attenuation of the stop band.

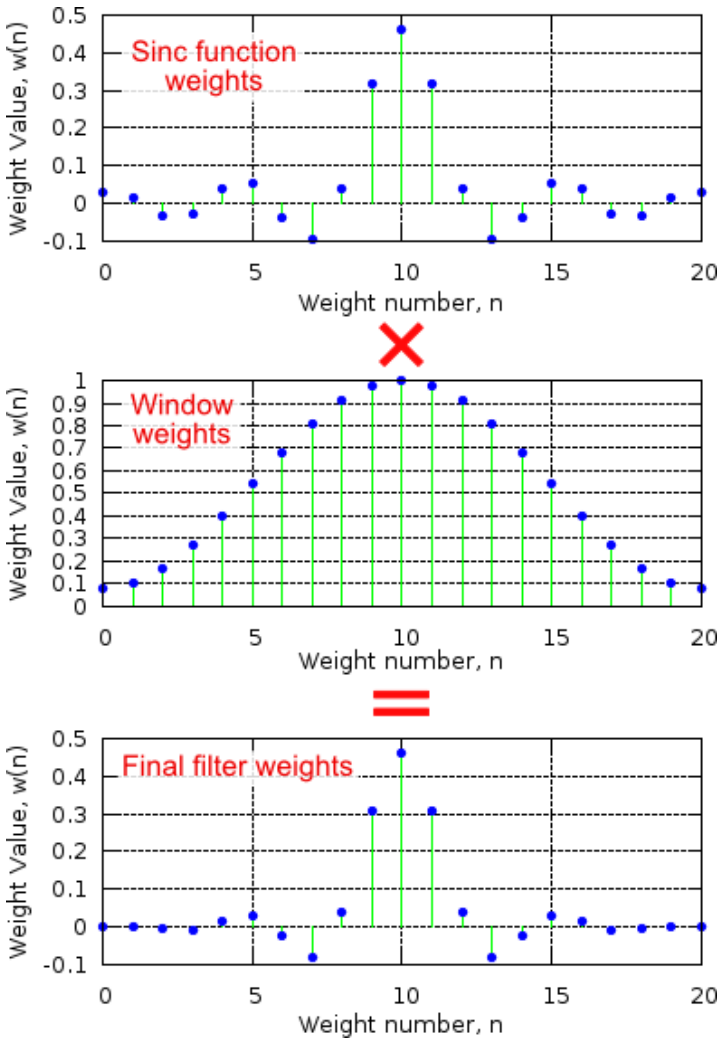


Fig. 6: Windowing

Applying a window to the sinc function weights provides extra control over the characteristics of the filter. The image to the right illustrates the process.

First, the normal sinc weights are calculated as described above. Then the window weights are calculated, in this case a Hamming Window has been used, the equation is below. The two sets of weights are multiplied together to create the final set of filter weights.

$$w(n) = 0.54 - 0.46\ cos\left(\frac{2\pi n}{M}\right)$$

3.7 Hamming Windowing Equation

Once again, *M* is the order of the filter, which is equal to the filter length - 1.

The plots below show the effect on the filter's frequency response before applying the Hamming Window (green) and after (red). The trick is to select the window type and filter length that will give a filter with the correct rate of roll-off and level of attenuation in the stop band.

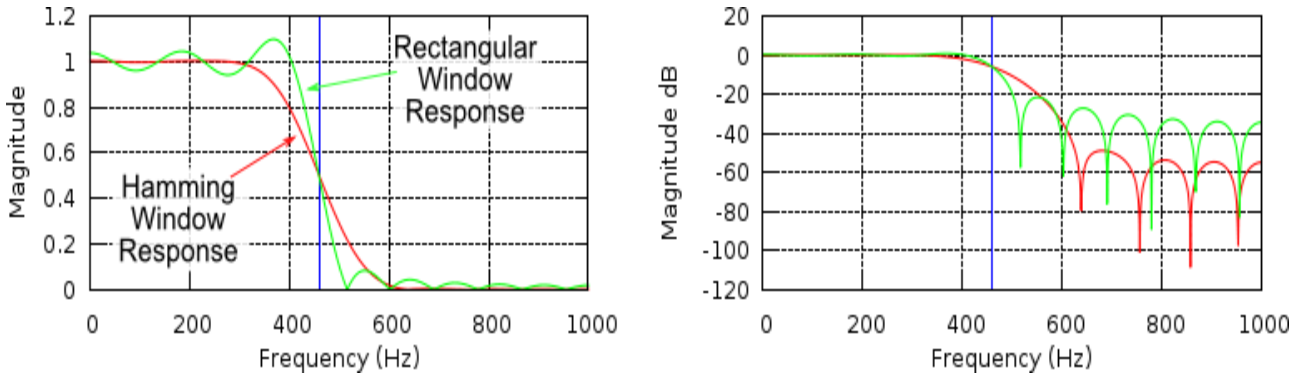


Fig. 7: Different Windows

The table below gives the equations for different window types.

3.8 Window Type Weight Equation

Rectangular $w(n) = 1$

Bartlett $w(n) = 1 - \frac{2\left|n - \frac{M}{2}\right|}{M}$

Hanning $w(n) = 0.5 - 0.5\,cos\left(\frac{2\pi n}{M}\right)$

Hamming $w(n) = 0.54 - 0.46\,cos\left(\frac{2\pi n}{M}\right)$

Blackman $$w(n) = 0.42 - 0.5\ cos\left(\frac{2\pi n}{M}\right) + 0.08\ cos\left(\frac{4\pi n}{M}\right)$$

The image below shows the effect of different windows on the frequency response of a 28th Order (29 weights) low pass filter, with a cut-off frequency of 5000Hz and sampling frequency of 44100Hz.

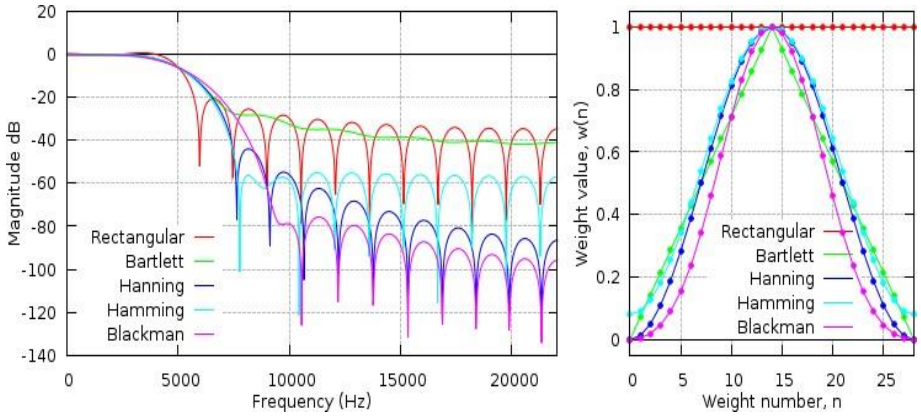


Fig. 8: Frequency Response and Weight Values of different windows types

3.9 HIGH PASS, BAND PASS AND BAND STOP FILTERS

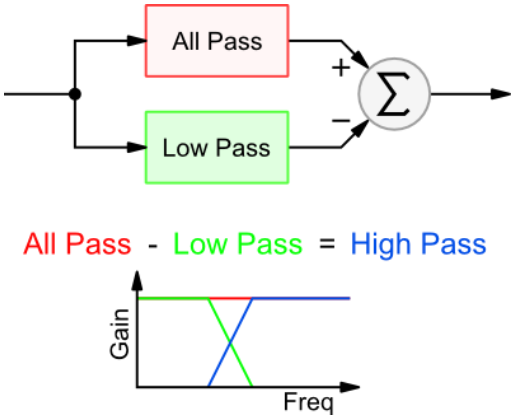


Fig. 9: The high pass filter is made up from a low pass and an all pass filter. The image to the right demonstrates how this works. If you take an all pass filter and subtract the output of the low pass, you are left with a high pass filter.

The all pass filter is of the same order as the low pass filter. All the weight values are 0.0 apart from the centre weight which has a value 1.0. **Note:** This places the constraint that when creating a high pass filter in this way, the order must be even (an odd number of taps).

The equation for calculating the weights (before windowing) is shown below. Comparing this equation with the low pass filter it is easy to see the subtraction and the all pass filter's single 1.0 weight applied in the case of *n=M/2*. Windows are applied in exactly the same way as with the low pass filter.

$$w_{hpf}(n) = \begin{cases} -\frac{sin\left[2\pi f_t\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} & n \neq \frac{M}{2} \\ 1-2f_t & n = \frac{M}{2} \end{cases}$$

Below is a Low Pass and High Pass filter frequency response with the same transition frequency.

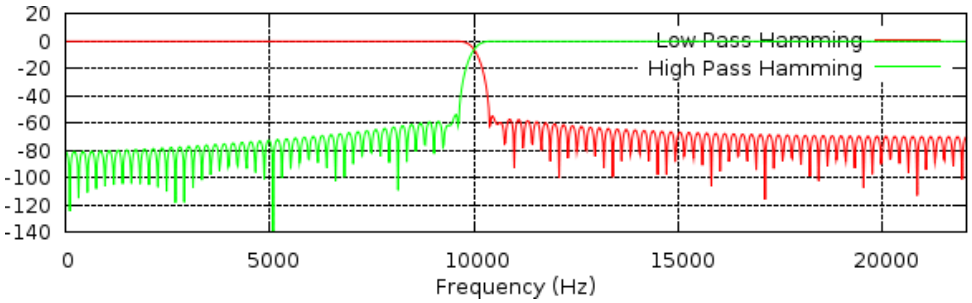
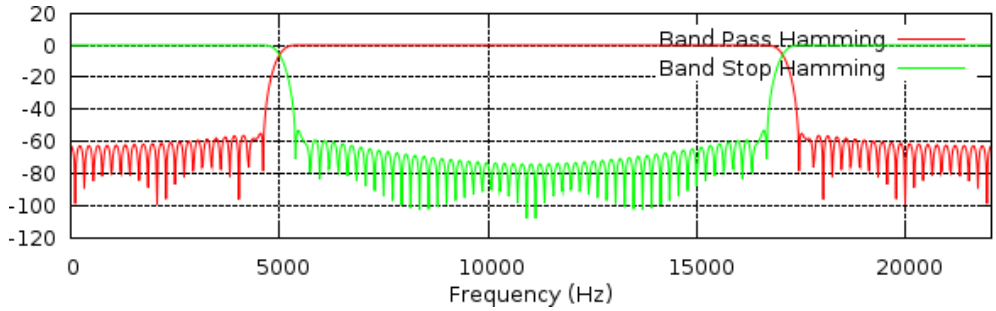


Fig. 10: Low pass and high pass hamming

The band stop and band pass are achieved in a similar way. The equations for calculating the weights are shown below. For both band pass and band stop, the filter order needs to be even (an odd filter length). Once again, windows are applied across the weights as before.

$$w_{bp}(n) = \begin{cases} \frac{sin\left[2\pi f_{t2}\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} - \frac{sin\left[2\pi f_{t1}\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} & n \neq \frac{M}{2} \\ 2(f_{t2} - f_{t1}) & n = \frac{M}{2} \end{cases}$$

$$w_{bs}(n) = \begin{cases} \frac{sin\left[2\pi f_{t1}\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} - \frac{sin\left[2\pi f_{t2}\left(n-\frac{M}{2}\right)\right]}{\pi\left(n-\frac{M}{2}\right)} & n \neq \frac{M}{2} \\ 1 - 2(f_{t2} - f_{t1}) & n = \frac{M}{2} \end{cases}$$



REFERENCES

1. John G. Proakis & Dimitris G.Manolakis, Digital Signal Processing - Principles, Algorithms & Applications, 4th Edition, Pearson Education / Prentice Hall, 2007.
2. Haykin. S and Van Been. B., Signals and Systems, 2nd Edition, John Wiley & Sons, 2003.
3. Willis J.Tompkins, Biomedical Digital Signal Processing, Prentice Hall of India Pvt. Ltd., 2012.
4. Sanjit K. Mitra, Digital Signal Processing - A Computer Based Approach, Tata McGraw Hill, 2007.
5. Oppenheim, A.V., R.W. Schafer and J.R. Buck, Discrete Time Signal Processing, 8th Indian Reprint, Pearson, 2004.
6. Stephane Mallat, Wavelet Tour of Signal Processing: The Sparse Way, 3rd Edition, Academic Press, 2008.

QUESTIONS

1. What are the merits of FIR filters
2. What are the demerits of FIR filter
3. Distinguish between FIR & IIR filter
4. What are the different types of filters based on frequency response
5. Give any two properties of Butterworth low pass filter
6. Give the equation for order and cutoff frequency of Butterworth filter
7. Write the magnitude response of the Chebyshev filter
8. Give the Butterworth polynomials for the order 1 & 4
9. Write the transfer function of Butterworth filter
10. Explain the fourier series method of designing FIR filter in detail
11. Design an ideal low pass filter with a frequency response

$$Hd(e^{j\omega}) = 1 \text{ for } -\pi/2 \le \omega \le \pi/2$$
$$0 \text{ for } \pi/2 \le |\omega| \le \pi$$

Find the values of h(n) for N = 11.

12. Design an ideal low pass filter with a frequency response

$$Hd(e^{j\omega}) = 1 \text{ for } -\pi/2 \le \omega \le \pi/2$$
$$0 \text{ for } \pi/2 \le |\omega| \le \pi$$

Find the values of h(n) for N = 11 using Hamming window

13. Design an ideal high pass filter with a frequency response

$$Hd(e^{j\omega}) = 1 \text{ for } -\pi/4 \le \omega \le \pi$$
$$0 \text{ for } |\omega| \le \pi/4$$

Find the values of h(n) for N = 11. Using Rectangular window.

14. Obtain an analog Chebyshev filter transfer function that satisfies the constraints

$$1/\sqrt{2} \le |H(j\Omega)| \le 1; \quad 0 \le \Omega \le 2$$
$$|H(j\Omega)| \le 0.1; \quad \Omega \ge 4$$

15. For the given specifications design an analog Butterworth filter

$$0.9 \le |H(j\Omega)| \le 1; \quad 0 \le \Omega \le 0.2\pi$$
$$|H(j\Omega)| \le 0.2; \quad 0.4\pi \le \Omega \le \pi$$

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF BIO AND CHEMICAL ENGINEERING
DEPARTMENT OF BIOMEDICAL ENGINEERING

UNIT – IV – Biosignal Detection and Compression – SBMA1402

Detection of events and waves

Derivative based operators in QRS detection

QRS complex has the largest slope (rate of change of voltage) in a cardiac cycle ventricles. As the rate of change is given by the derivative operator, the operation would be the most logical starting point in an attempt to develop an algorithm to detect the QRS complex. The derivative operator enhances the QRS, although the resulting wave does not bear any resemblance to a typical QRS complex. The slow P and T waves have been suppressed by the derivative operators, while the output is the highest at the QRS. However, given the noisy nature of the results of the derivative-based operators, it is also evident that significant smoothing will be required before further processing can take

$$y_0(n) = |x(n) - x(n-2)|.$$

place. Derivative-based algorithm for QRS detection progresses as follows: the smoothed three-point first derivative yo (n) of the given signal x(n) is approximated as

The second derivative is approximated as

$$y_1(n) = |x(n) - 2x(n-2) + x(n-4)|.$$

The two results are weighted and combined to obtain

$$y_2(n) = 1.3y_0(n) + 1.1y_1(n).$$

The result y2(n) is scanned with a threshold of 1.0. Whenever the threshold is crossed, the subsequent eight samples are also tested against the same threshold. If at least six of the eight points pass the threshold test, the segment of eight samples is taken to be a part of a QRS complex. The procedure results in a pulse with its width proportional to that of the QRS complex; however, the method is sensitive to noise.

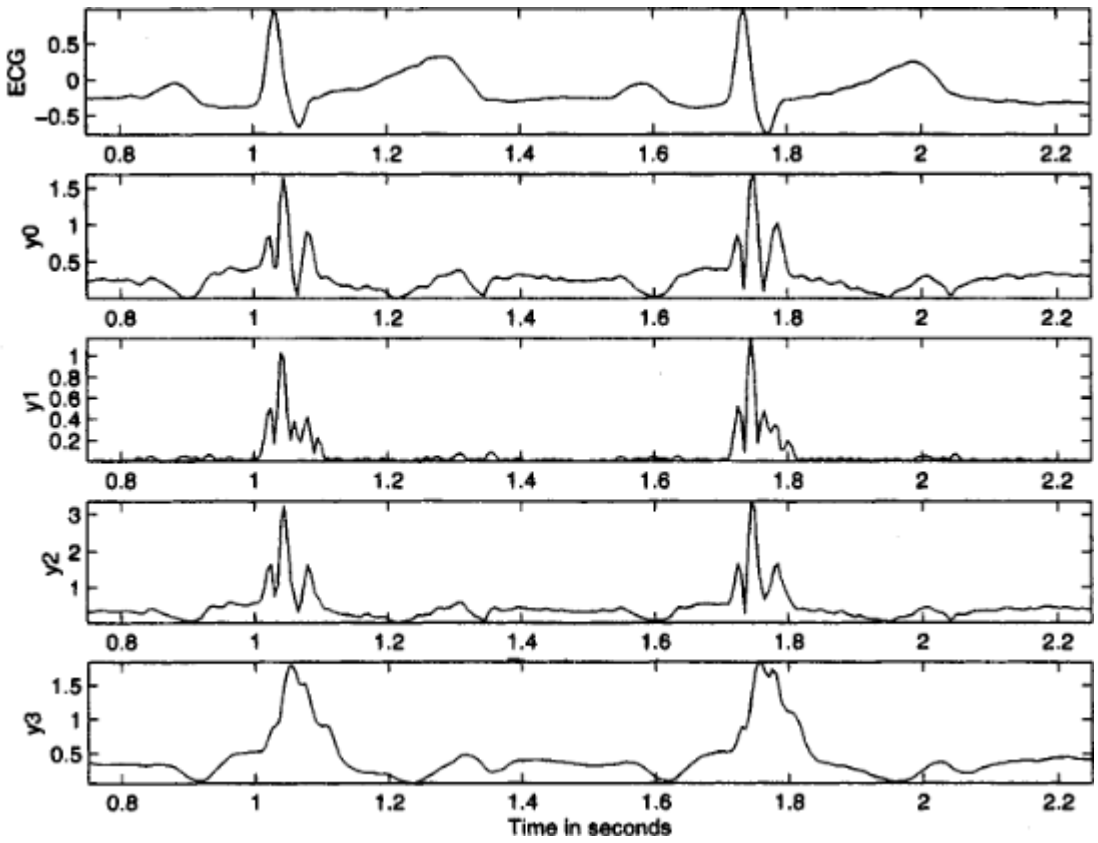


Fig.1. From top to bottom: two cycles of a filtered version of the ECG signal, output yo(n)of the first-derivative-based operator, output yl(n) of the second-derivative-based operator and the result y3(n) of passing y2(n) through the 8-point MA filter

Pan Tompkins algorithm

Pan and Tompkins proposed a real-time QRS detection al- gorithm based on analysis of the slope, amplitude, and width of QRS complexes. The algorithm includes a series of filters and methods that perform lowpass, high- pass, derivative, squaring, integration, adaptive thresholding, and search procedures.

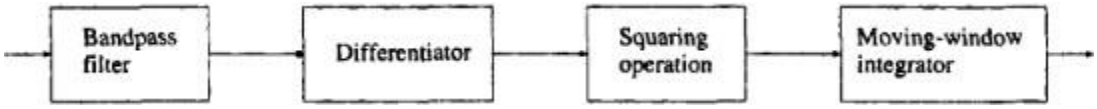


Fig.2 Block diagram of the Pan-Tompkins algorithm for QRS detection

Low pass filter: The recursive lowpass filter used in the Pan-Tompkins algorithm has integer coefficients to reduce computational complexity, with the transfer function

defined as

The output y(n) is related to the input x(n) as

$$H(z) = \frac{1}{32}\,\frac{(1-z^{-6})^2}{(1-z^{-1})^2}\,.$$

The output y(n) is related to the input x(n) as

$$y(n) = 2y(n-1) - y(n-2) + \frac{1}{32}\,[x(n) - 2x(n-6) + x(n-12)].$$

With the sampling rate being 200 Hz, the filter has a rather low cutoff frequency of fc =

11 Hz, and introduces a delay of 5 samples or 25 ms.The filter provides an attenuation greater than

35 dB at 60 Hz,and effectively suppresses power-line interference, if present.

Highpass filter: The highpass filter used in the algorithm is implemented as an allpass

filter minus a lowpass filter. The lowpass component has the transfer function the input - output relationship is

$$H_{lp}(z) = \frac{(1-z^{-32})}{(1-z^{-1})}\,;$$

the input - output relationship is

$$y(n) = y(n-1) + x(n) - x(n-32).$$

The transfer function Hhp(z) of the highpass filter is specified as

$$H_{hp}(z) = z^{-16} - \frac{1}{32}\,H_{lp}(z)$$

Equivalently, the output p(n) of the highpass filter is given by the difference equation

$$p(n) = x(n-16) - \frac{1}{32}\,[y(n-1) + x(n) - x(n-32)].$$

The high pass filter has a cutoff frequency of 5 Hz and introduces a delay of 80 ms.

Derivative operator: The derivative operation used by Pan and Tompkins is specified as

$$y(n) = \frac{1}{8}\,[2x(n) + x(n-1) - x(n-3) - 2x(n-4)],$$

and approximates the ideal operator up to 30 Hz. The derivative procedure suppresses the low-frequency components of the P and T waves, and provides a large gain to the high-frequency components arising from the high slopes of the QRS complex.

Squaring: The squaring operation makes the result positive and emphasizes large differences resulting from QRS complexes; the small differences arising from P and T waves are suppresses. The high- frequency components in the signal related to the QRS complex are further enhanced.

Integration: As observed in the previous subsection, the output of a derivative- based operation will exhibit multiple peaks within the duration of a single QRS complex. The Pan-Tompkins algorithm performs smoothing of the output of the preceding operations through a moving-window integration filter as

$$y(n) = \frac{1}{N}\,[x(n-(N-1)) + x(n-(N-2)) + \cdots + x(n)]$$

The choice of the window width N is to be made with the following considerations: too large a value will result in the outputs due to the QRS and T waves being merged, whereas too small a value could yield several peaks for a single QRS. A window width of N = 30 was found to be suitable for fb = 200 Hz.

Adaptive thresholding: The thresholding procedure in the Pan-Tompkins algo- rithm adapts to changes in the ECG signal by computing running estimates of signal and noise peaks. A peak is said to be detected whenever the final output changes direction within a specified interval. In the following discussion, SPKI represents the peak level that the algorithm has learned to be that corresponding to QRS peaks, and NPKI represents the peak level related to non- QRS events (noise, EMG,etc.).

THRESHOLD I1 and THRESHOLD I2 are two thresholds used to categorize peaks detected

as signal (QRS) or noise. Every new peak detected is categorized as a signal peak or a noise peak. If a peak exceeds THRESHOLD I1 during the first step of analysis, it is classified as a QRS (signal) peak. If the searchback technique the peak.should be above THRESHOLDI2 to be called a QRS.The peak levels and thresholds are updated after each peak is detected and classified as if PEAKI is a signal peak; if PEAKI
is a noise peak;

SPKI = 0.125 PEAKI +
0.875 SPKI NPKI = 0.125
PEAKI + 0.875 NPKI
THRESHOLD I1 = NPKI +
0.25(SPKI - NPKI); THRESHOLD
I2 = 0.5 THRESHOLD II.
The updating formula for SPKI is
changed to SPKI = 0.25 PEAKI +
0.75 SPKI
If a QRS is detected in the searchback procedure using THRESHOLD 12.

Searchback procedure: The Pan-Tompkins algorithm maintains two RR-interval averages: RR AVERAGE1 is the average of the eight most-recent beats, and RR AVERAGE2 is the average of the eight most-recent beats having RR intervals within the range specified by

RR LOW LIMIT = 0.92 x RR
AVERAGE2 and RR HIGH LIMIT
= 1.16 x RR AVERAGE2.
Whenever a QRS is not detected for a certain interval specified as

RR MISSED LIMIT = 1.06 x RR AVERAGE2, the QRS is taken to be the peak between the established thresh- olds applied in the searchback procedure.

1. **Correlation analysis**

ACF and CCF in rhythm analysis

Cross Correlation is defined as

$$x \cdot y = \langle x, y \rangle = \sum_{n=0}^{N-1} x(n)\, y(n),$$

where the signals $x(n)$ and $y(n)$ have N samples each. The dot product represents the projection of one signal on to the other, with each signal being viewed as an N-dimensional vector. The dot product may be normalized by the geometric mean of the energies of the two signals to obtain a correlation coefficient as

$$\gamma_{xy} = \frac{\sum_{n=0}^{N-1} x(n)\, y(n)}{\left[\sum_{n=0}^{N-1} x^2(n)\ \sum_{n=0}^{N-1} y^2(n)\right]^{1/2}}.$$

The means of the signals may be subtracted out, if desired, as in Equation

In the case of two continuous-time signals $x(t)$ and $y(t)$, the projection of one signal on to the other is defined as

$$\theta_{xy} = \int_{-\infty}^{\infty} x(t)\, y(t)\, dt.$$

When a shift or time delay may be present in the occurrence of the epoch of interest in the two signals being compared, it becomes necessary to introduce a time-shift parameter to compute the projection for every possible position of overlap. The shift parameter facilitates searching one signal for the occurrence of an event matching that in the other signal at any time instant within the available duration of the signals. The CCF between two signals for a shift or delay of τ seconds or k samples may be obtained as

$$\theta_{xy}(\tau) = \int_{-\infty}^{\infty} x(t)\, y(t+\tau)\, dt,$$

$y(n)$ with the most-recent N samples being available in each signal at the time instant n. If $\mathbf{x}(n)$ and $\mathbf{y}(n)$ are sample observations of random processes, their CCF is defined as

$$\Theta_{xy} = E[\mathbf{x}(n)\,\mathbf{y}^T(n)],$$

The outer product, which is an $N \times N$ matrix, provides the cross-terms that include all possible delays (shifts) within the duration of the given signals.

All of the equations given above may be modified to obtain the ACF by replacing the second signal y with the first signal x. The signal x is then compared with itself.

The ACF displays peaks at intervals corresponding to the period (and integral multiples thereof) of any periodic or repetitive pattern present in the signal. This property facilitates the detection of rhythms in signals such as the EEG: The presence of the α rhythm would be indicated by a peak in the neighborhood of 0.1 s. The ACF of most signals decays and reaches negligible values after delays of a few milliseconds, except for periodic signals of infinite or indefinite duration for which the ACF will also exhibit periodic peaks. The ACF will also exhibit multiple peaks when the same event repeats itself at regular or irregular intervals. One may need to compute the ACF only up to certain delay limits depending upon the expected characteristics of the signal being analyzed.

The CCF displays peaks at the period of any periodic pattern present in *both* of the signals being analyzed. The CCF may, therefore, be used to detect rhythms present in common between between two signals, for example, between two channels of the EEG.

1. Murmur detection

Cardiac mechanical activity is appraised by auscultation and processing of heart sound recordings (known as *phonocardiographic signals*—PCG), which is an inexpensive and noninvasive procedure. The importance of classic auscultation has decreased due to its inherent restrictions: the performance of human ear with its physical limitations, the subjectivity of the examiner, difficult skills that take years to acquire and refine, etc. Anyway, the PCG has preserved its importance in pediatric cardiology, cardiology, and internal diseases, evaluating congenital cardiac defects,and primary home health care, where an intelligent stethoscope with decision support abilities would be valuable. Mostly, heart sounds consist of two regularly repeated thuds, known as S1 and S2, each appearing one after the other, for every heart beat. The time interval between S1 and S2 is the systole, while the S2 and next S1 gap corresponds to the diastole. S1 implies the closing of the

tricuspid and mitral valves immediately preceding the systole, while S2 corresponds to the closing of the aortic and

laminar and therefore silent; but when the blood flow becomes turbulent it causes vibration of surrounding tissue and hence the blood flow is noisy and perceivable, originating the murmur, which according to the instant they appear are sorted into

systolic or diastolic. Murmurs are some of the basic signs of pathological changes to be identified, but they overlap with the cardiac beat and can not be easily separated by the human ear. The automatic detection of murmurs strongly depends on the appropriate features (data representation), which mostly are related to timing, morphology, and spectral properties of heart sounds. Although cardiac murmurs are nonstationary signals and exhibit sudden frequency changes and transients, t is common to assume linearity of the feature sets extracted from heart sounds (time and spectral features, frequency representation with time resolution, and parametric modeling. o capture nonstationary transients and fast changes of PCG, the time–frequency features are widely used in heart sound analysis.

or

$$\theta_{xy}(k) = \sum_n x(n)\, y(n+k).$$

As in the case of the multiplicative homomorphic system, our goal is to convert the convolution operation to addition. From the convolution property of the Fourier transform, we know that

$$Y(\omega) = X(\omega)\, H(\omega).$$

Thus, application of the Fourier transform converts convolution to multiplication. Now, it is readily seen that the multiplicative homomorphic system may be applied to convert the multiplication to addition. Taking the complex logarithm of $Y(\omega)$, we have

$$\log[Y(\omega)] = \log[X(\omega)] + \log[H(\omega)]; \qquad X(\omega) \neq 0,\ H(\omega) \neq 0\ \forall\omega.$$

[Note: $\log_e[X(\omega)] = \hat{X}(\omega) = \log_e[|X(\omega)|\angle X(\omega)] = \log_e[|X(\omega)|] + j\angle X(\omega)$, where $|X(\omega)|$ and $\angle X(\omega)$ are the magnitude and phase spectra of $x(t)$.]

A linear filter may now be used to separate the transformed components of x and h, with the assumption as before that they are separable in the transformed space. A series of the inverses of the transformations applied initially will take us back to the original domain.

While the discussion here has been in terms of application of the Fourier transform, the general formulation of the homomorphic filter by Oppenheim and Schafer [174] is in terms of the z-transform. However, the Fourier transform is equivalent to the z-transform evaluated on the unit circle in the z-plane, and the Fourier transform is more commonly used in signal processing than the z-transform.

Figure · gives a block diagram of the steps involved in a homomorphic filter for convolved signals. The path formed by the first three blocks (in the top row) transforms the convolution operation at the input to addition. The third block with the inverse Fourier transform is used to move back to a pseudo time domain. The last three blocks (in the bottom row) perform the reverse transformation, converting addition to convolution. The filter in between deals with (transformed) signals that

Derivation of the transfer function of the matched filter

In order to derive the transfer function, $H(\omega)$, of the matched filter let the signal $x(t)$ be the input to the matched filter. The Fourier transform of $x(t)$ is

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

The output of the matched filter, $y(t)$, is given by the inverse Fourier transform of $Y(\omega) = X(\omega)H(\omega)$, as follows:

$$\begin{aligned} y(t) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)\ H(\omega)\ \exp(+j\omega t)\ d\omega \\ &= \int^{\infty} X(f)\ H(f)\ \exp(+j\,2\pi\, ft)\ df. \end{aligned}$$

In the second expression of the equation given above, the frequency variable has been changed from ω in radians per second to f in Hz.

Consider the presence of white noise at the input, with the PSD

$$S_{\eta i}(f) = \frac{P_{\eta i}}{2},$$

where $P_{\eta i}$ is the average noise power at the input. Then, the noise PSD at the output is

$$S_{\eta o}(f) = \frac{P_{\eta i}}{2}\ |H(f)|^2.$$

The average output noise power is

$$P_{\eta o} = \frac{P_{\eta i}}{2} \int_{-\infty}^{\infty} |H(f)|^2\, df.$$

The RMS value of the noise in the absence of any signal is $\sqrt{P_{\eta o}}$.

Letting $t = t_0$ the magnitude of the instantaneous output signal at $t = t_0$ is

$$M_y = |y(t_0)| = \left| \int_{-\infty}^{\infty} X(f)\, H(f)\, \exp(+j\, 2\pi\, f t_0)\, df \right|.$$

Thus, the SNR at the output is $\frac{M_y}{\sqrt{P_{\eta o}}}$.

To derive the optimal transfer function of the matched filter, we could maximize the SNR, which is equivalent to maximizing the expression

$$\frac{M_y^2}{P_{\eta o}} = \frac{\text{instantaneous peak power of signal}}{\text{noise mean power}},$$

which represents peak-power SNR [212].

For a given signal $x(t)$, the total energy is a constant, given by

$$E_x = \int_{-\infty}^{\infty} x^2(t)\, dt = \int_{-\infty}^{\infty} |X(f)|^2\, df.$$

Let us consider the following ratio:

$$\frac{M_y^2}{E_x\, P_{\eta o}} = \frac{\left| \int_{-\infty}^{\infty} H(f)\, X(f)\, \exp(+j\, 2\pi\, f t_0)\, df \right|^2}{\frac{P_{\eta i}}{2} \int_{-\infty}^{\infty} |H(f)|^2\, df \int_{-\infty}^{\infty} |X(f)|^2\, df}.$$

The quantity E_x is a constant for a given input signal; hence, maximizing the expression in Equation is equivalent to maximizing the expression in Equation .

In order to determine the condition for maximization of the expression in Equation recall Schwarz's inequality for two arbitrary complex functions $A(f)$ and $B(f)$:

$$\left| \int_{-\infty}^{\infty} A(f)\, B(f)\, df \right|^2 \leq \left[\int_{-\infty}^{\infty} |A(f)|^2\, df \right] \left[\int_{-\infty}^{\infty} |B(f)|^2\, df \right]$$

For any two real functions $a(t)$ and $b(t)$, the corresponding inequality is

$$\left[\int_{-\infty}^{\infty} a(t)\, b(t)\, dt \right]^2 \leq \left[\int_{-\infty}^{\infty} a^2(t)\, dt \right] \left[\int_{-\infty}^{\infty} b^2(t)\, dt \right].$$

For any two vectors **a** and **b**, Schwarz's inequality states that

$$|\mathbf{a}.\mathbf{b}| \leq |\mathbf{a}|\, |\mathbf{b}|,$$

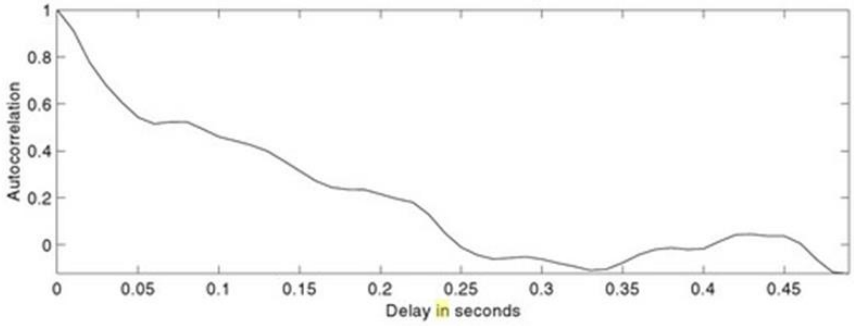
and

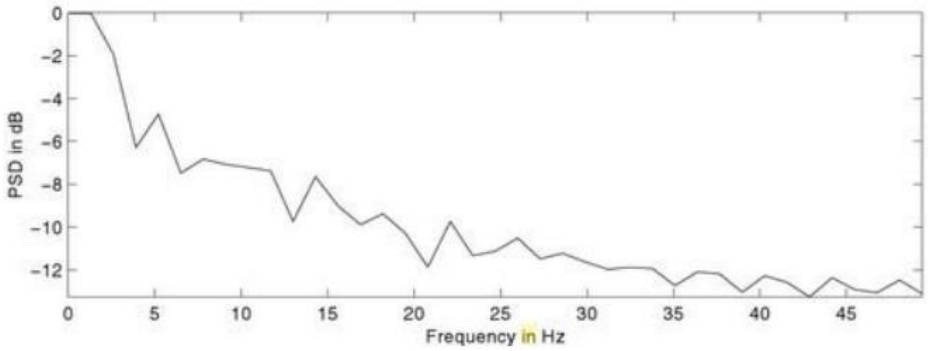
$$|\mathbf{a} + \mathbf{b}| \leq |\mathbf{a}| + |\mathbf{b}|.$$

6. Wavelet detection – Spike and wave detection

Problem: *Propose a method to detect spike-and-wave complexes in an EEG signal. You may assume that a sample segment of a spike-and-wave complex is available.*

Solution: A spike-and-wave complex is a well-defined event in an EEG signal. The complex is composed of a sharp spike followed by a wave with a frequency of about 3 Hz; the wave may contain a half period or a full period of an almost-





sinusoidal pattern. One may, therefore, extract an epoch of a spike-and-wave complex from an EEG channel and use it for template matching with the same formula

7. Extraction of vocal tract response and other applications

Problem: *Design a homomorphic filter to extract the basic wavelet corresponding to the vocal-tract response from a voiced speech signal.*

Solution: . voiced speech is generated by excitation of the vocal tract, while it is held in a particular form, with a glottal waveform that may be approximated as a series of pulses. The voiced speech signal may, therefore, be expressed in discrete-time terms as $y(n) = x(n) * h(n)$, where $y(n)$ is the speech signal, $x(n)$ is the glottal waveform (excitation sequence), and $h(n)$ is the impulse response of the vocal tract (basic wavelet). The $*$ symbol represents convolution, with the assumption that the vocal-tract filter may be approximated by an LSI filter. We may, therefore, use the homomorphic filter for convolved signals as introduced in the preceding section to separate $h(n)$ and $x(n)$.

The glottal excitation sequence may be further expressed as $x(n) = p(n) * g(n)$, where $p(n)$ is a train of ideal impulses (Dirac delta functions) and $g(n)$ is a smoothing function, to indicate that the physical vocal-cord system cannot produce ideal impulses but rather pulses of finite duration and slope

ECG DATA REDUCTION

1. Direct data compression Techniques:

A data reduction algorithm seeks to minimize the number of code bits stored by reducing the redundancy present in the original signal. We obtain the reduction ratio by dividing the number of bits of the original signal by the number saved in the compressed signal. A data reduction algorithm must also represent the data with acceptable fidelity.

In biomedical data reduction, we usually determine the clinical acceptability of the reconstructed signal through visual inspection. We may also measure the residual, that is, the difference between the reconstructed signal and the original signal. Such a numerical measure is the percent root-mean-square difference, PRD, given by

$$\text{PRD} = \left\{ \frac{\sum_{i=1}^{n} [x_{org}(i) - x_{rec}(i)]^2}{\sum_{i=1}^{n} [x_{org}(i)]^2} \right\}^{\frac{1}{2}} \times 100\ \%$$

where n is the number of samples and xorg and xrec are samples of the original and reconstructed data sequences.

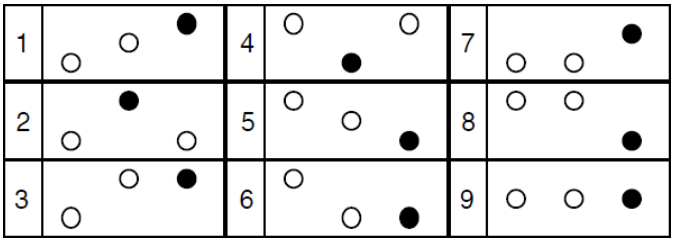
Turning Point

The original motivation for the turning point (TP) algorithm was to reduce the sampling frequency of an ECG signal from 200 to 100 samples/s .The algorithm developed from the observation that, except for QRS complexes with large amplitudes and slopes, a sampling rate of 100 samples/s is adequate.TP is based on the concept that ECG signals are normally oversampled at four or five times faster than the highest frequency present. For example, an ECG used in monitoring may have a bandwidth of 50 Hz and be sampled at 200 sps in order to easily visualize the higher- frequency attributes of the QRS complex. Sampling theory tells us that we can sample such a signal at 100 sps. TP provides away to reduce the effective sampling rate by half to 100 sps by selectively saving important signal points (i.e., the peaks and valleys or turning points).

The algorithm processes three data points at a time. It stores the first sample point and assigns it as the reference point X0. The next two consecutive points become X1 and X2. The algorithm retains either X1 or X2, depending on which point preserves the turning point (i.e., slope change) of the original signal. Fig. shows all the possible configurations of three consecutive sample points. In each frame, the solid point preserves the slope of the original three points. The algorithm saves this point and makes it the reference point X0 for the next iteration. It then samples the next two points, assigns them to X1 and X2, and repeats the process.

We use a simple mathematical criterion to determine the saved point. First consider a sign(x) operation

$$sign(x) = \begin{cases} 0 & x = 0 \\ +1 & x > 0 \\ -1 & x < 0 \end{cases}$$



Pattern	$s_1 = sign(X_1 - X_0)$	$s_2 = sign(X_2 - X_1)$	NOT(s_1) OR ($s_1 + s_2$)	Saved sample
1	+1	+1	1	X_2
2	+1	–1	0	X_1
3	+1	0	1	X_2
4	–1	+1	0	X_1
5	–1	–1	1	X_2
6	–1	0	1	X_2
7	0	+1	1	X_2
8	0	–1	1	X_2
9	0	0	1	X_2

Fig. 3. Turning point Algorithm

We then obtain s1 = sign(X1 – X0) and s2 = sign(X2 – X1), where (X1 – X0) and (X2 – X1) are the slopes of the two pairs of consecutive points. If a slope is zero,this operator produces a zero result. For positive or negative slopes, it yields +1 or–1 respectively. A turning point occurs only when a slope changes from positive to negative or vice versa. We use the logical Boolean operators, NOT and OR, as implemented in the C language to make the final judgment of when a turning point occurs. In the C language, NOT(c) = 1 if c = 0; otherwise NOT(c) = 0. Also logical OR means that (a OR b) = 0 only if a and b are both 0. Thus, we retain X1 only if {NOT(s1) OR (s1 + s2)} is zero, and save X2 otherwise. In this expression, (s1 + s2) is the arithmetic sum of the signs produced by the sign function. The final effect of this processingis a Boolean decision whether to save X1 or X2. Point X1 is saved only when the slope changes from positive to negative or vice versa. This computation could be easily done arithmetically, but the Boolean operation is computationally much faster.

The TP algorithm is simple and fast, producing a fixed reduction ratio of 2:1. After selectively discarding exactly half the sampled data, we can restore the original resolution by interpolating between pairs of saved data points. A second application of the algorithm to the already reduced data increases the reduction ratio to 4:1. Using data acquired at a 200-sps rate, this produces compressed data with a 50-sps effective sampling rate. If the bandwidth of the acquired ECG is 50 Hz, this approach violates sampling theory since the effective sampling rate is less than twice the highest frequency present in the signal. The

resulting reconstructed signal typically has a widened QRS complex and sharp edges that reduce its clinical acceptability. Another disadvantage of this algorithm is that the saved points do not represent equally spaced time intervals. This introduces short term time distortion. However, this localized distortion is not visible when the reconstructed signal is viewed on the standard clinical monitors and paper recorders.

AZTEC

Originally developed to preprocess ECGs for rhythm analysis, the AZTEC (Amplitude Zone Time Epoch Coding) data reduction algorithm decomposes raw ECG sample points into plateaus and slopes.It provides a sequence of line segments that form a piecewise-linear approximation to the ECG. The algorithm consists of two parts—line detection and line processing. Figure(a) shows the line detection operation which makes use of zero-order interpolation (ZOI) to produce horizontal lines. Two variables Vmx and Vmn always reflect the highest and lowest elevations of the current line. Variable LineLen keeps track of the number of samples examined. We store a plateau if either the difference between Vmxi and Vmni is greater than a predetermined threshold Vth or if LineLen is greater than 50. The stored values are the length (LineLen – 1) and the average amplitude of the plateau (Vmx + Vmn)/2.

Figure (b) shows the line processing algorithm which either produces a

plateau or a slope depending on the value of the variable LineMode. We initialize LineMode to

_PLATEAU in order to begin by producing a plateau. The production

of an AZTEC slope begins when the number of samples needed to form a plateau is less than three. Setting LineMode to _SLOPE indicates that we have entered slope production mode. We then determine the direction or sign of the current slope by subtracting the previous line amplitude V1 from the current amplitude Vsi. We also reset the length of the slope Tsi. The variable Vsi records the current line amplitude so that any change in the direction of the slope can be tracked. Note that Vmxi and Vmni are always updated to the latest sample before line detection begins. This forces ZOI to begin from the value of the latest sample.

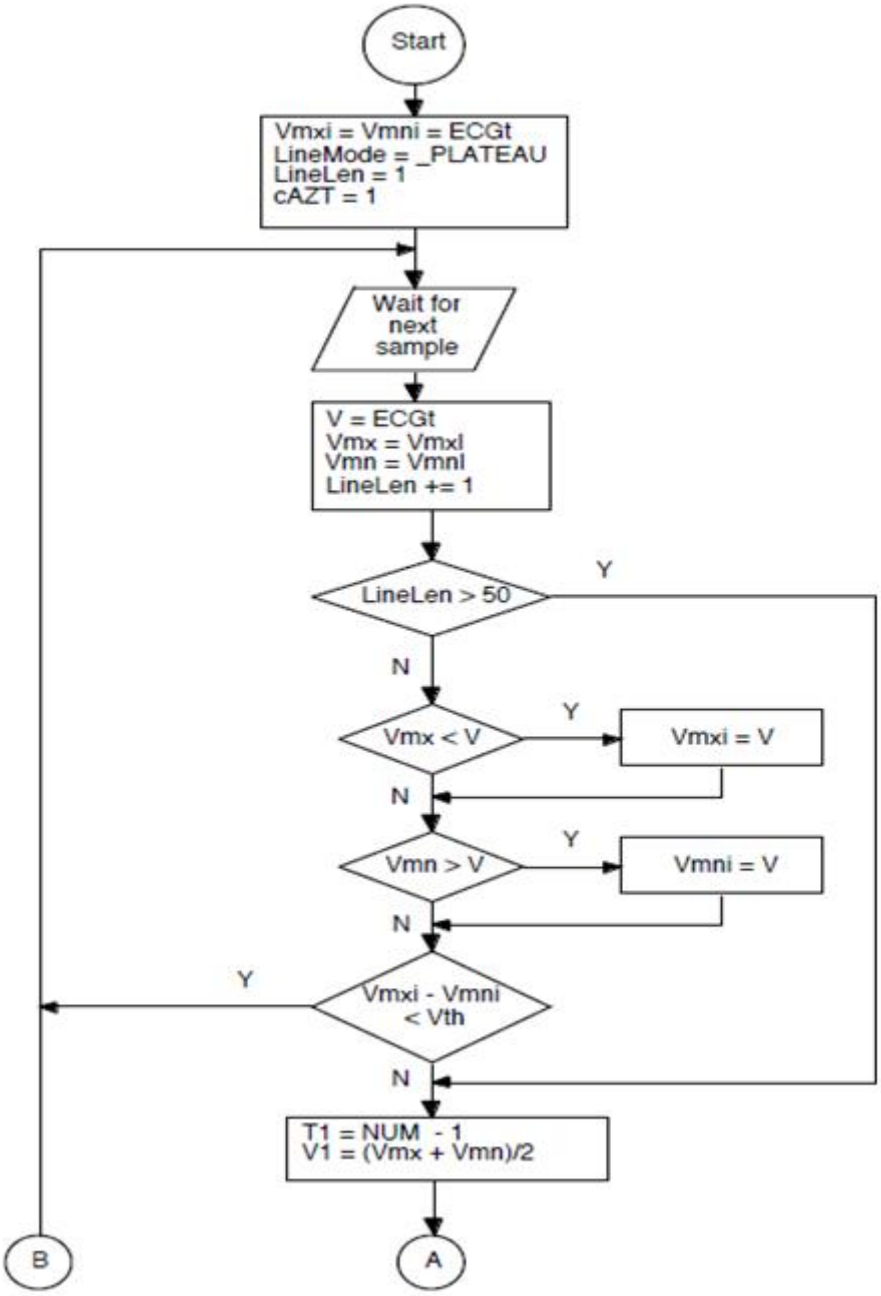

Start
Vmxi = Vmni = ECGt
LineMode = _PLATEAU
LineLen = 1
cAZT = 1
Wait for next sample
V = ECGt
Vmx = Vmxl
Vmn = Vmnl
LineLen += 1
LineLen > 50
Y
N
Vmx < V
Y
Vmxi = V
N
Vmn > V
Y
Vmni = V
N
Vmxi - Vmni < Vth
Y
N
T1 = NUM - 1
V1 = (Vmx + Vmn)/2
B
A

Fig. 4(a) Flowchart for the line detection operation of the AZTEC algorithm

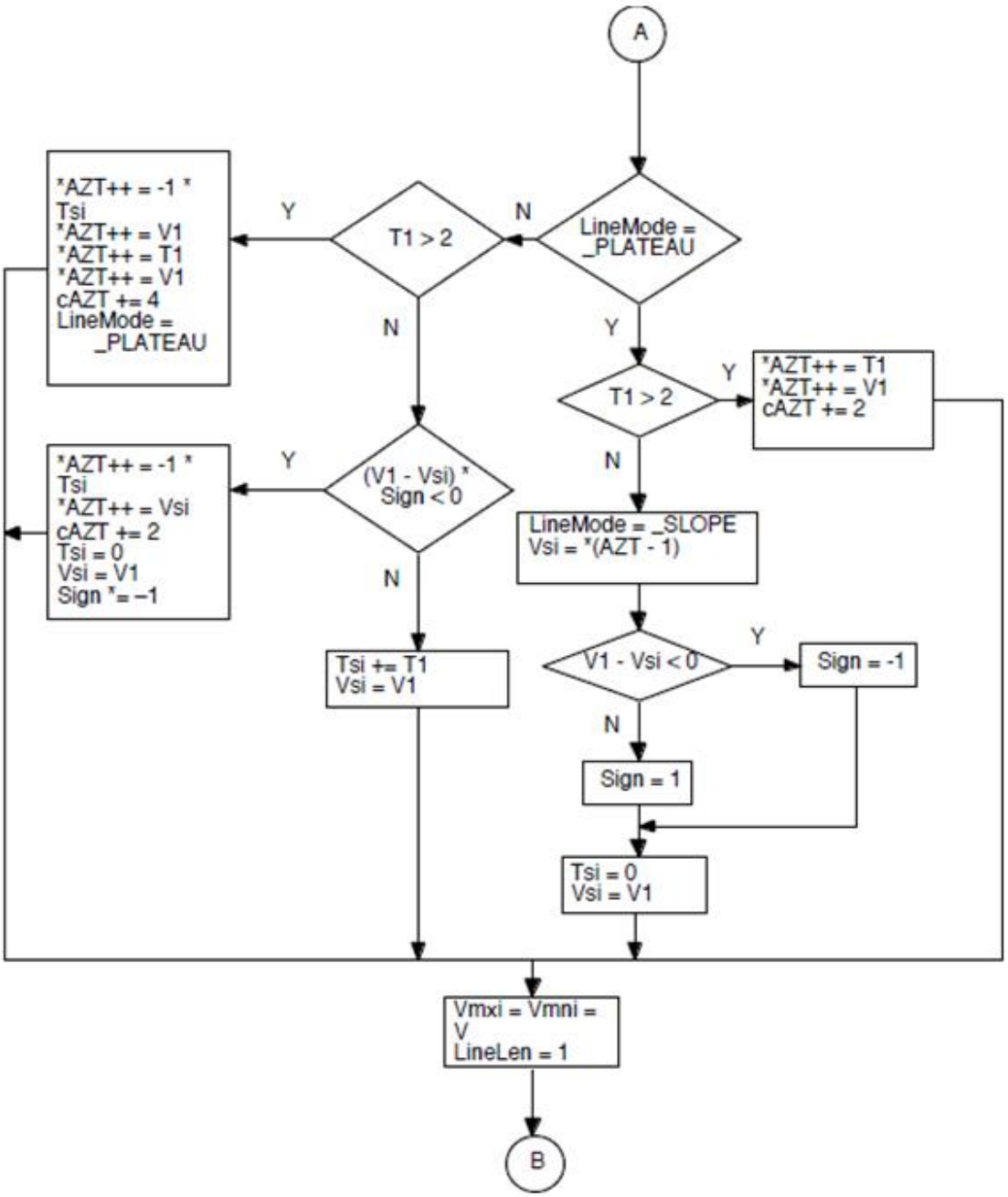


Fig. 4(b) Flowchart of the line processing operation of the AZTEC algorithm

When we reenter line processing with LineMode equal to _SLOPE, we either save or update the slope. The slope is saved either when a plateau of more than three samples can be formed or when a change in direction is detected. If we detect a new plateau of more than three samples, we store the current slope and the new plateau. For the slope, the stored values are its length Tsi and its final elevation V1. Note that Tsi is multiplied by –1 to differentiate a slope from a plateau (i.e., the minus sign serves as a flag to indicate a

slope). We also store the length and the amplitude of the new plateau, then reset all parameters and return to plateau production.

If a change in direction is detected in the slope, we first save the parameters for the current slope and then reset sign, Vsi, Tsi, Vmxi, and Vmni to produce a new AZTEC slope. Now the algorithm returns to line detection but remains in slope production mode. When there is no new plateau or change of direction, we simply update the slope's parameters, Tsi and Vsi, and return to line detection with LineMode remaining set to _SLOPE. AZTEC does not produce a constant data reduction ratio. The ratio is frequently as great as 10 or more, depending on the nature of the signal and the value of the empirically determined threshold.

Cortes Algorithm

The CORTES (Coordinate Reduction Time Encoding System) algorithm is a hybrid of the TP and AZTEC algorithms. It attempts to exploit the strengths of each while sidestepping the weaknesses.

CORTES uses AZTEC to discard clinically insignificant data in

the isoelectric region with a high reduction ratio and applies the TP algorithm to the clinically significant high-frequency regions (QRS complexes). It executes the AZTEC and TP algorithms in parallel on the incoming ECG data. Whenever an AZTEC line is produced, the CORTES algorithm decides, based on the length of the line, whether the AZTEC data or the TP data are to be saved. If the line is longer than an empirically determined threshold, it saves the AZTEC

line. Otherwise it saves the TP data points. Since TP is used to encode the QRS complexes, only AZTEC plateaus, not slopes, are implemented. The CORTES algorithm reconstructs the signal by expanding the AZTEC plateaus and interpolating between each pair of the TP data points. It then applies parabolic smoothing to the AZTEC portions to reduce discontinuities.

FAN Algorithm

Originally used for ECG telemetry, the Fan algorithm draws lines between pairs of starting and ending points so that all intermediate samples are within some specified error

tolerance. We start by accepting the first sample X0 as the nonredundant permanent point. It functions as the origin and is also called the originating point.We then take the second sample X1 and draw two slopes {U1, L1}. U1 passes

through the point (X0, X1 + ε), and L1 passes through the point (X0, X1 – ε). If the third sample X2 falls within the area bounded by the two slopes, we generate two new slopes {U2, L2} that pass through points (X0, X2 + ε) and (X0, X2 – ε). We compare the two pai rs of slopes and retain the most converging (restrictive) slopes (i.e., {U1, L2} in our example). Next we assign the value of X2 to X1 and read the next sample into X2. As a result, X2 always holds the most recent sample and X1 holds the sample immediately preceding X2. We repeat the process by comparing X2 to the values of the most convergent slopes. If it falls outside this area, we save

the length of the line T and its final amplitude X1 which then becomes the new originating point X0, and the process begins anew. The sketch of the slopes drawn from the originating sample to future samples forms a set of radial lines similar to a fan, giving this algorithm its name. When adapting the Fan algorithm to C-language implementation, we create the variables, XU1, XL1, XU2, and XL2, to determine the bounds of X2.

$$X_{L2} = \frac{X_{L1} - X_0}{T} + X_{L1}$$

$$X_{U2} = \frac{X_{U1} - X_0}{T} + X_{U1}$$

where $T = t_T - t_0$.

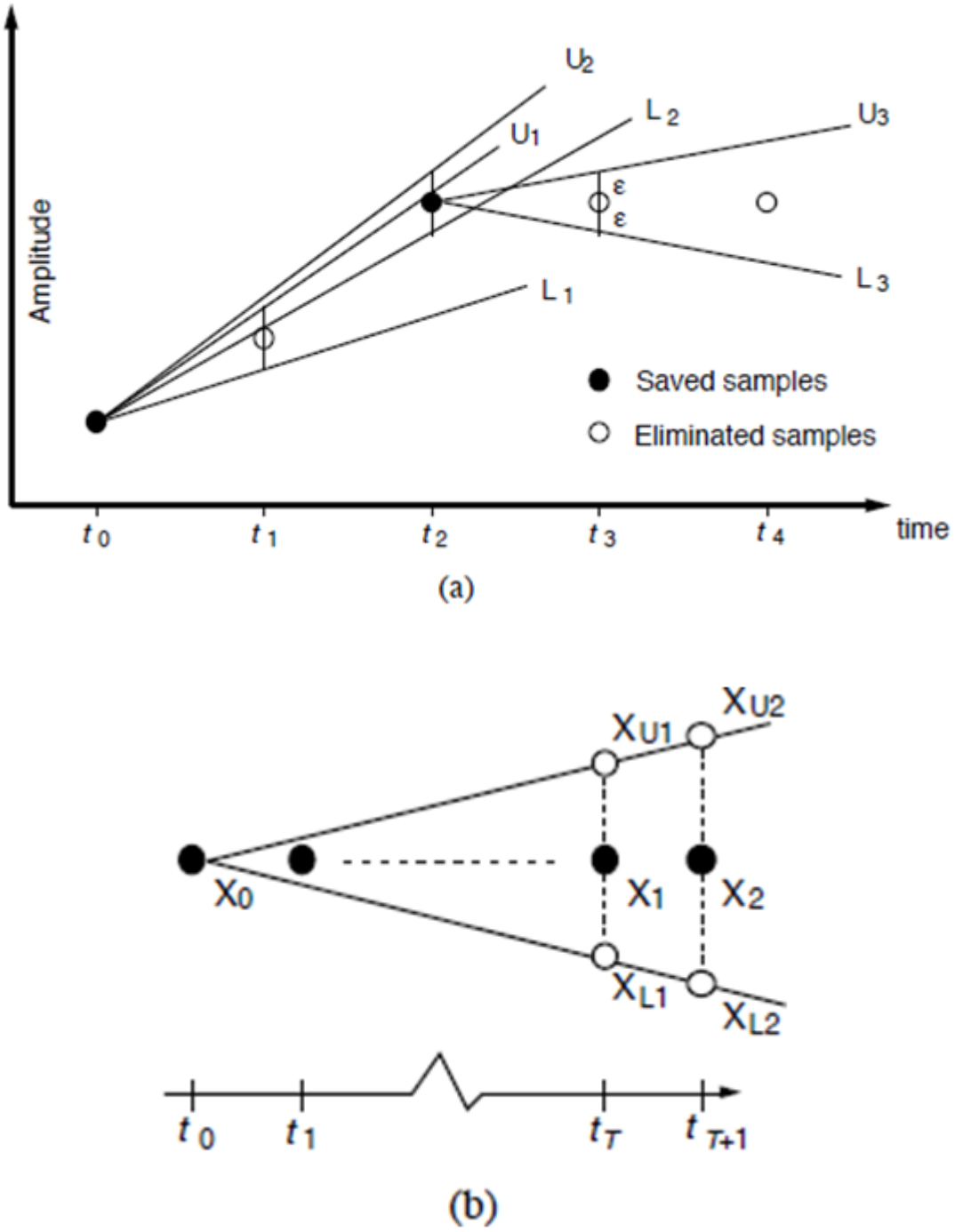


Fig. 5 a) Upper and lower slopes (U and L) are drawn within error threshold ε around sample points taken at t1, t2, … (b) Extrapolation of XU2 and XL2 from XU1, XL1, and X0.

We reconstruct the compressed data by expanding the lines into discrete points. The Fan algorithm guarantees that the error between the line joining any two permanent sample points and any actual

redundant) sample along the line is less than or equal to the magnitude of the preset error tolerance. The algorithm's reduction ratio depends on the error tolerance. When compared to the TP and AZTEC algorithms, the Fan algorithm produces better signal fidelity for the same reduction ratio

2. Transformation Compression Techniques:

Karhunen-Loeve Transform

The Karhunen-Loeve Transform (KLT) (also known as Hotelling Transform and Eigenvector Transform) is closely related to the Principal Component Analysis (PCA) and widely used in data analysis in many fields

Let be the eigenvector corresponding to the kth eigenvalue of the covariance matrix i.e.,

$$\mathbf{\Sigma}_x \phi_k = \lambda_k \phi_k \qquad (k = 1, \cdots, N)$$

$$\begin{bmatrix} \cdots & \cdots & \cdots \\ \cdots & \sigma_{ij} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} \\ \phi_k \\ \\ \end{bmatrix} = \lambda_k \begin{bmatrix} \\ \phi_k \\ \\ \end{bmatrix} \qquad (k = 1, \cdots, N)$$

or in matrix form:

As the covariance matrixis Hermitian (symmetric ifis real), its eigenvector'sare orthogonal:

$$\langle \phi_i, \phi_j \rangle = \phi_i^T \phi_j^* = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

Other data compression Techniques:

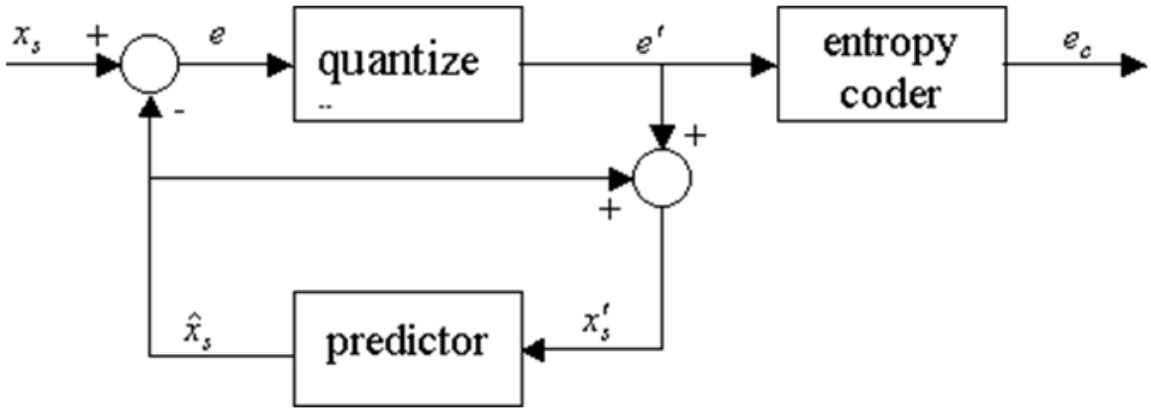
DPCM

Differential **p**ulse **c**ode **m**odulation (DPCM) is a procedure of converting an analog into a digital signal in which an analog signal is sampled and then the difference between the actual sample value and its

predicted value (predicted value is based on previous sample or samples) is quantized and then encoded forming a digital value. DPCM code words represent differences between samples unlike PCM where code words represented a sample value. Basic concept of

DPCM - coding a difference, is based on the fact that most source signals show significant correlation between successive samples so encoding uses redundancy in sample values which implies lower bit rate. Realization of basic concept (described above) is based on a technique in which we have to predict current sample value based upon previous samples (or sample) and we have to encode the difference between actual value of sample and predicted value (the difference between samples can be interpreted as prediction error). Because it's necessary to predict sample value DPCM is form of predictive coding.

DPCM compression depends on the prediction technique, well-conducted prediction techniques lead to good compression rates, in other cases DPCM could mean expansion comparing to regular PCM encoding.



$$L = 2^n$$

Fig.6 DPCM encoder (transmitter)

Huffman coding

Huffman coding exploits the fact that discrete amplitudes of quantized signal do not occur with equal probability (Huffman, 1952). It assigns variable-length code words to a given quantized data sequence according to their frequency of occurrence. Data that occur frequently are assigned shorter code words.

Static Huffman coding

As an example, assume that we wish to transmit the set of 28 data points

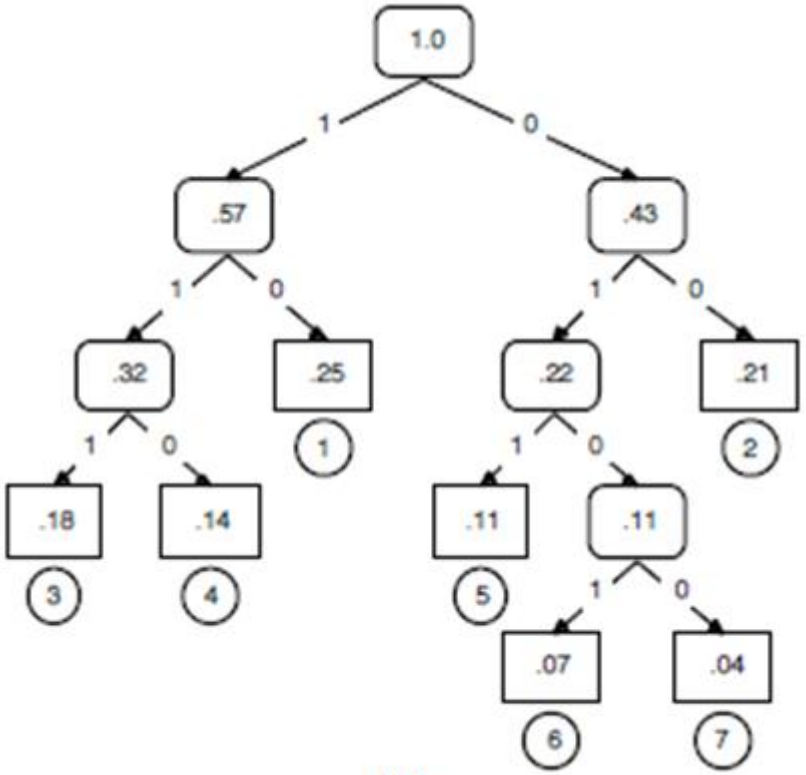
{1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 7}

The set consists of seven distinct quantized levels, or symbols. For each symbol, Si, we calculate its probability of occurrence Pi by dividing its frequency of occurrence by 28, the total number of data points. Consequently, the construction of a Huffman code for this set begins with seven nodes, one associated with each Pi. At each step we sort the Pi list in descending order, breaking the ties arbitrarily.

The two nodes with smallest probability, Pi and Pj, are merged into a new node with probability Pi + Pj. This process continues until the probability list contains a single value

S_i	Lists of P_i						
1	.25	.25	.25	.32	.43	.57	1.0
2	.21	.21	.22	.25	.32	.43	
3	.18	.18	.21	.22	.25		
4	.14	.14	.18	.21			
5	.11	.11	.14				
6	.07	.11					
7	.04						

(a)



(b)

Symbols, S_i	3-bit binary code	Probability of occurrence, P_i	Huffman code
1	001	0.25	10
2	010	0.21	00
3	011	0.18	111
4	100	0.14	110
5	101	0.11	011
6	110	0.07	0101
7	111	0.04	0100

(c)

Fig. 7 Illustration of Huffman coding. (a) At each step, Pi are sorted in descending order

and the two lowest Pi are merged. (b) Merging operation depicted in a binary tree. (c) Summary of Huffman coding for the data set.

The process of merging nodes produces a binary tree as in Figure (b). When we merge two nodes with probability Pi + Pj, we create a parent node with two children represented by Pi and Pj. The root of the tree has probability 1.0. We obtain the Huffman code of the symbols by traversing down the tree, assigning 1 to the left child and 0 to the right child. The resulting code words have the prefix property (i.e., no code word is a proper prefix of any other code word). This property ensures that a coded message is uniquely decodable without the need for lookahead.

Figure (c) summarizes the results and shows the Huffman codes for

the seven symbols. We enter these code word mappings into a translation table and use the table to pad the appropriate code word into the output bit stream in the reduction process. The reduction ratio of Huffman coding depends on the distribution of the source symbols. In our example, the original data

requires three bits to represent the seven quantized levels. After Huffman coding, we can calculate the expected code word length in our example, resulting in an expected reduction ratio of 3:2.65.

$$E[l] = \sum_{i=1}^{7} l_i P_i$$

The reconstruction process begins at the root of the tree. If bit 1 is received, we traverse

down the left branch, otherwise the right branch. We continue traversing until we reach a node with no child. We then output the symbol corresponding to this node and begin traversal from the root again.

The reconstruction process of Huffman coding perfectly recovers the original data. Therefore it is a lossless algorithm. However, a transmission error of a single bit may result in more than one decoding error. This propagation of transmission error is a consequence of all algorithms that produce variable- length code words.

REFERENCES

1. John G. Proakis & Dimitris G.Manolakis, Digital Signal Processing - Principles, Algorithms & Applications, 4th Edition, Pearson Education / Prentice Hall, 2007.

2. Haykin. S and Van Been. B., Signals and Systems, 2nd Edition, John Wiley & Sons, 2003.

3. Willis J.Tompkins, Biomedical Digital Signal Processing, Prentice Hall of India Pvt. Ltd., 2012.

4. Sanjit K. Mitra, Digital Signal Processing - A Computer Based Approach, Tata McGraw Hill, 2007.

5. Oppenheim, A.V., R.W. Schafer and J.R. Buck, Discrete Time Signal Processing, 8th Indian Reprint, Pearson, 2004.

6. Stephane Mallat, Wavelet Tour of Signal Processing: The Sparse Way, 3rd Edition, Academic Press, 2008.

QUESTIONS

1. Develop signal processing techniques to facilitate detection of QRS complex.
2. Describe about the data reduction algorithms
3. Write notes on FAN algorithm.
4. Write notes on AZTEC algorithm.
5. Write short notes on Turning point algorithm.
6. Write notes on Huffman coding
7. Write notes on DPCM.
8. Propose an algorithm to detect QRS complexes in an ongoing ECG signal.
9. Describe the function of matched filter in detecting spike and wave complexes of EEG signal.

10. With neat block diagram explain the working of homomorphic filtering for multiplied signals.
11. Design a homomorphic filter to separate two signals that have been combined through convolution operation.
12. Explain the filtering technique used to study the presence of EEG rhythms in multiple channel.
13. Explain the filtering technique used to extract the vocal tract response.
14. Explain in detail about the turning point algorithm.
15. Write notes on
    (i) CORTES algorithm
    (ii) AZTEC algorithm

Explain in detail about the concepts of the Huffman coding

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF BIO AND CHEMICAL ENGINEERING
DEPARTMENT OF BIOMEDICAL ENGINEERING

UNIT – V – Biosignal Analysis Using Wavelets – SBMA1402

NEED FOR WAVELETS

The Fourier transform is an useful tool to analyze the frequency components of the signal. However, if we take the Fourier transform over the whole time axis, we cannot tell at what instant a particular frequency rises. Short-time Fourier transform (STFT) uses a sliding window to find spectrogram, which gives the information of both time and frequency. But still another problem exists: The length of window limits the resolution in frequency. Wavelet transform seems to be a solution to the problem above. Wavelet transforms are based on small wavelets with limited duration. The translated-version wavelets locate where we concern. Whereas the scaled-version wavelets allow us to analyze the signal in different scale.

The first literature that relates to the wavelet transform is Haar wavelet. It was proposed by the mathematician Alfrd Haar in 1909. However, the concept of the wavelet did not exist at that time. Until 1981, the concept was proposed by the geophysicist Jean Morlet. Afterward, Morlet and the physicist Alex Grossman invented the term wavelet in 1984. Before 1985, Haar wavelet was the only orthogonal wavelet people know. A lot of researchers even thought that there was no orthogonal wavelet except Haar wavelet. Fortunately, the mathematician Yves Meyer constructed the second orthogonal wavelet called Meyer wavelet in 1985. As more and more scholars joined in this field, the 1st international conference was held in France in 1987.

In 1988, Stephane Mallat and Meyer proposed the concept of multiresolution. In the same year, Ingrid Daubechies found a systematical method to construct the compact support orthogonal wavelet. In 1989, Mallat proposed the fast wavelet transform. With the appearance of this fast algorithm, the wavelet transform had numerous applications in the signal processing field.

$$Wf(s,u) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\psi^*(\frac{t-u}{s})dt.$$

Wavelet transform overcomes the previous problem. The wavelet function is designed to strike a balance between time domain (finite length) and frequency domain (finite bandwidth). As we dilate and translate the mother wavelet, we can see very low frequency components at large s while very high frequency component can be located precisely at small s.

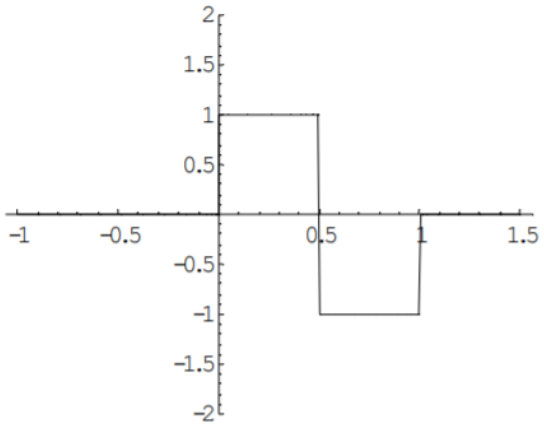
Fourier transform based spectral analysis is the dominant analytical tool for frequency domain analysis. However, Fourier transform cannot provide any information of the spectrum changes with respect to time. Fourier transform assumes the signal is stationary, but PD signal is always non-stationary. To overcome this deficiency, a modified method-short time Fourier transform allows to represent the signal in both time and frequency domain through time windowing function [29]. The window length determines a constant time and frequency resolution. Thus, a shorter time windowing is used in order to capture the transient behavior of a signal; we sacrifice the frequency resolution. The nature of the real PD signals is nonperiodic and transient as shown in Fig 18, Fig 19; such signals cannot easily be analyzed by conventional transforms. So, an alternative mathematical tool- wavelet transform must be selected to extract the relevant time-amplitude information from a signal. In the meantime, we can improve the signal to noise ratio based on prior knowledge of the signal characteristics.

TYPES OF WAVELETS

HAAR WAVELETS

Now define the desired wavelet $\psi(x)$

$$\equiv \begin{cases} 1 & \text{if } 0 \le x \le 1/2 \\ -1 & \text{if } 1/2 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$$



DISCRETE WAVELET TRANSFORM

One drawback of the CWT is that the representation of the signal is often redundant, since a and b are continuous over R (the real number). The original signal can be completely reconstructed by a sample version of $W_f(b,a)$. Typically, we sample $W_f(b,a)$ in dyadic grid, i.e.,

$$a = 2^{-m} \qquad \text{and} \qquad b = n2^{-m}$$

$$DWT_\psi f(m,n) = \int_{-\infty}^{\infty} f(t)\psi^*_{m,n}(t)\,dt$$

where $\psi_{m,n}(t) = 2^{-m}\psi(2^m t - n)$ is the dilated and translated version of the mother wavelet $\psi(t)$.

The family of dilated mother wavelets of selected a and b constitute an orthonormal basis of $L^2(R)$. In addition, we sample $W_f(b,a)$ in dyadic grid, this wavelet transform is also called dyadic-orthonormal wavelet transform. Due to the orthonormal properties, there is no information redundancy in the discrete wavelet transform. In addition, with this choice of a and b, there exists the multiresolution analysis (MRA) algorithm, which decompose a signal into scales with different time and frequency resolution. MRA is designed to give good time resolution and poor frequency resolution at high frequencies and good frequency resolution and poor time resolution at low frequencies.

The differences between different mother wavelet functions (e.g. Haar, Daubechies, Coiflets, Symlet, Biorthogonal and etc.) consist in how these scaling signals and the wavelets are defined. The choice of wavelet determines the final waveform shape; likewise, for Fourier transform, the decomposed waveforms are always sinusoid. To have a unique reconstructed signal from wavelet transform, we need to select the orthogonal wavelets to perform the transforms.

DAUBECHIES WAVELETS

Based on these equation, Daubechies [9], designed a type of wavelet for a given vanishing moment p and find the minimum size discrete filter. The conclusion is that if we want the wavelet function with p vanishing moments, the minimum filter size is $2p$. The derivation first starts from 5.17, rewrite as

$$H_\phi(e^{j\omega}) = \sqrt{2}(\frac{1+e^{-j\omega}}{2})^p R(e^{j\omega}).$$

The absolute-square of this function is

$$\begin{aligned} |H_\phi(e^{j\omega})|^2 &= H_\phi(e^{j\omega})H_\phi^*(e^{j\omega}) \\ &= 2(\frac{1+e^{-j\omega}}{2}\frac{1+e^{j\omega}}{2})^p R(e^{j\omega})R^*(e^{j\omega}) \\ &= 2(\frac{2+e^{j\omega}+e^{-j\omega}}{4})^{2p}|R(e^{j\omega})|^2 \\ &= 2(\cos\frac{\omega}{2})^{2p}P(\sin^2\frac{\omega}{2}). \end{aligned}$$

DECOMPOSITION AND RECONSTRUCTION OF SIGNALS USING WAVELETS

The wavelet decomposition results in levels of approximated and detailed coefficients. The algorithm of wavelet signal decomposition is illustrated in Fig 22. Reconstruction of the signal from the wavelet transform and post processing, the algorithm is shown in Fig 23. This multi-resolution analysis enables us to analyze the signal in different frequency bands; therefore, we could observe any transient in time domain as well as in frequency domain.

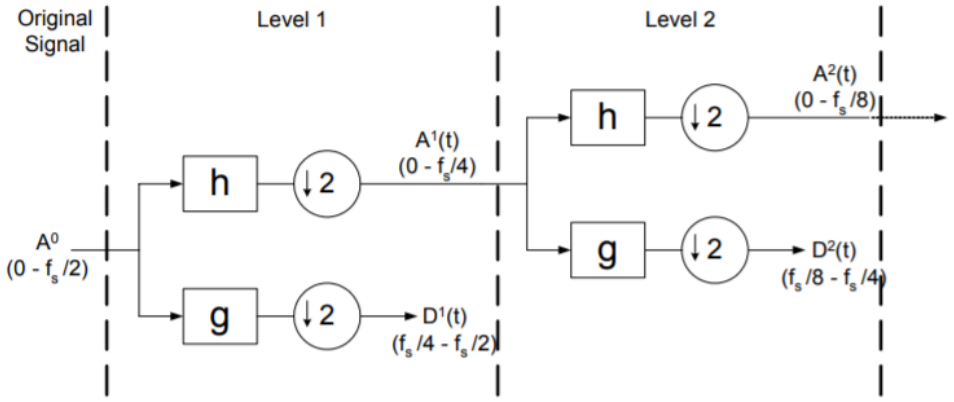


Fig . Multi-resolution wavelet decomposition. h = low-pass decomposition filter; g = high-pass decomposition filter; (↓2) = down-sampling operation. $A^1(t)$, $A^2(t)$ are the approximated coefficient of the original signal at levels 1, 2 etc. $D^1(t)$, $D^2(t)$ are the detailed coefficient at levels 1,2.

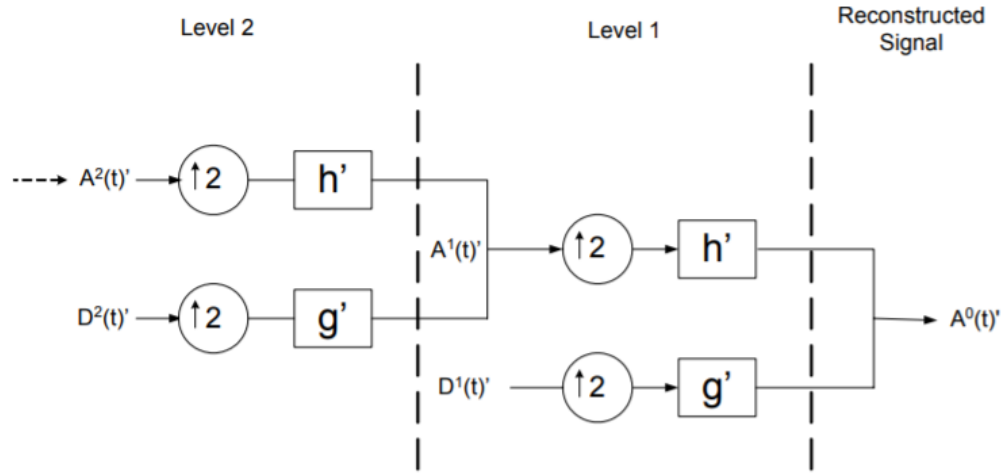


Fig Multi-resolution wavelet reconstruction. h' = low pass reconstruction filter; g' = high-pass reconstruction filter; (↑2) = up-sampling operation. $A^1(t)'$, $A^2(t)'$ are the processed or non-processed approximated coefficient of the original signal at levels 1, 2 etc. $D^1(t)'$, $D^2(t)'$ are the processed or non-processed detailed coefficient at levels 1,2.

The relation between the low-pass and high-pass filter and the scalar function $\psi(t)$ and the wavelet $\phi(t)$ can be states as following:

$$\phi(t) = \sum_k h[k]\phi[2t-k]$$

$$\psi(t) = \sum_k g[k]\phi[2t-k]$$

The relation between the low-pass filter and high-pass filter is not independent to each other, they are related by:

$$g[L-1-n] = (-1)^n \cdot h[n]$$

where g[n] is the high-pass, h[n] is the low-pass filter, L is the filter length (total number of points). Filters satisfying this condition are commonly used in signal processing, and they are known as the Quadrature Mirror Filters (QMF). The two filtering and down-sampling operation can be expressed by:

$$A^i[k] = \sum_n A^{i-1}(t) \cdot h[2k-n]$$

$$D^i[k] = \sum_n A^{i-1}(t) \cdot g[2k-n]$$

The reconstruction in this case is very easy since the halfband filters form the orthonormal bases. The above procedure is followed in reverse order for the reconstruction. The signals at every level are upsampled by two, passed through the synthesis filters g'[n], and h'[n] (highpass and lowpass, respectively), and then added. The interesting point here is that the analysis and synthesis filters are identical to each other, except for a time reversal. Therefore, the reconstruction formula becomes (for each layer)

DENOISING USING WAVELETS

Apply wavelet transform to the noisy signal to produce the noisy wavelet coefficients to the level which we can properly distinguish the PD occurrence.

Select appropriate threshold limit at each level and threshold method (hard or soft thresholding) to best remove the noises.

Inverse wavelet transform of the thresholded wavelet coefficients to obtain a denoised signal.

WAVELET SELECTION

To best characterize the PD spikes in a noisy signal, such as Fig 18, and Fig 20, we should select our "mother wavelet" carefully to better approximate and capture the transient spikes of the original signal. "Mother wavelet" will not only determine how well we estimate the original signal in terms of the shape of the PD spikes, but also, it will affect the frequency spectrum of the denoised signal. The choice of mother wavelet can be based on eyeball inspection of the PD spikes [29], or it can be selected based on correlation γ (21) between the signal of interest and the wavelet-denoised signal [38], or based on the cumulative energy (22) over some interval where PD spikes occur.

$$\gamma = \frac{\sum (X - \overline{X})(Y - \overline{Y})}{\sqrt{\sum (X - \overline{X})^2 (Y - \overline{Y})^2}}$$

where $\overline{X}$ and $\overline{Y}$ are the mean value of set X and Y, respectively.

$$E = \sum X^2$$

where E is the energy and X is the signal vector.

We choose to select the mother wavelet based on the last two methods: correlation between two signals and cumulative energy over some interval of PD spike occurrence. We found that the two methods give us a very similar outcome.

THRESHOLD LIMITS

Many methods for setting the threshold have been proposed. The most time-consuming way is to set the threshold limit on a case-by-case basis. The limit is selected such that satisfactory noise removal is achieved. For a Gaussian noise [39]; if we apply orthogonal wavelet transform to the noise signal, the transformed signal will preserve the Gaussian nature of the noise, which the histogram of the noise will be a symmetrical bell-shaped curve about its mean value. From theory, four times the standard deviation would cover 99.99% of the noise. Therefore, we could set the threshold be 4.5 times of the standard deviation of the wavelet-transformed signal to remove the Gaussian noise in the signal.

We have found that for the fiber optic signals, we could simply apply the standard deviation methods, since the signal is mostly white noises (see Fig 18 and **Appendix B**), however for the PZT signals, we should set the threshold case-by-case to best denoise the signals.

Two rules are generally used for thresholding the wavelet coefficients (soft/hard thresholding). Hard thresholding sets zeros for all wavelet coefficients whose absolute value is less than the specified threshold limit. It has shown that hard thresholding provides an improved signal to noise ratio [29]. In this study, we adopt the hard thresholding method.

LEVEL OF DECOMPOSITION

From the previous section, we have known that the wavelet transform is constituted by different levels. The maximum level to apply the wavelet transform depends on how many data points contain in a data set, since there is a down-sampling by 2 operation from one level to the next one. In our experience, one factor that affects the number of level we can reach to achieve the satisfactory noise removal results is the signal-to-noise ratio (SNR) in the original signal. Generally, the measured signals from the PZT sensors have higher SNR than that of the measured signals from fiber optic sensors. So to process the PZT data, we need more level of wavelet transform (e.g. 12) to remove most of its noise. For the fiber optic sensor data, we could only go up to 4 or 5 level otherwise we would remove much of the PD signal, therefore the PD spikes wouldn't be captured.

TYPICAL MEDICAL APPLICATIONS

An ECG is not smooth with many sudden transitions. If we analyze the spectrum, noises and desired signal transitions cannot be separated using ordinary filters. Here wavelet transform can also remove the noise and preserve the features of ECG.

There are still another applications which are not noted. In short, the wavelet transform is a powerful tool solving the problems beyond the scope of Fourier transform.

wavelets are applied to the analysis of two types of medical signals, namely infant breathing signals and ultrasound images. One-dimensional wavelets are used to quantify amplitude modulation of infant breathing that occurs during quiet sleep. Two-dimensional wavelets are used to develop enhancement techniques in the wavelet domain, tailored to ultrasound images. The development of wavelets is described from the backgrounds of mathematics, signal processing and sub-band coding. One-dimensional wavelets are defined for both continuous and discrete cases. Two-dimensional wavelets are developed for both separable and non-separable classes. The construction of both one- and two-dimensional wavelets is described, and examples of wavelets that are used in subsequent analyses are presented. The analysis of breathing signals provides information for understanding the physiology of breathing. The wavelet domain is shown to isolate frequency characteristics of breathing signals, as well as indicating the temporal position of those characteristics in the signal. Some of the characteristics that were measured were not distinguished in the original signals. The extent of constant frequency components due to amplitude modulation of the principal breathing rate is quantified. Breathing in infants is of particular interest, as it may provide insight into the cause of Sudden Infant Death Syndrome (SIDS), or cot death. Studies of infants who later succumbed to SIDS, or were at high risk for SIDS, and infants at low risk for SIDS were carried out. The infants who later succumbed to SIDS and those at high risk for SIDS showed different characteristics in the wavelet domain compared to infants at low risk for SIDS. As well as implying that there may be a difference in the physiologies of infants at high and low risk for SIDS, this result confirms that wavelets can be used to analyse breathing signals and produce meaningful results. Ultrasound images typically contain artefacts and low contrast between features of interest and often exhibit a noisy background. The aim of any enhancement procedure is to reduce the contributions from noise and artefacts and increase the contrast between the features of interest and the background. A method to objectively measure improvements in image quality is described, although this attempt achieved mixed results. Separable and non-separable two-dimensional wavelets are used with enhancement functions to improve the image quality. Enhancement schemes, including noise reduction and contrast enhancement, are developed. A spatially varying contrast enhancement function is also developed. Noise reduction, in combination with the spatially varying contrast enhancement function, produces an image that reduces the existing artefacts in the image and increases the contrast for the features of interest. Wavelets are shown to be a useful tool for the analysis of infant breathing signals and for improvement of image quality in ultrasound images.

REFERENCES

1. John G. Proakis & Dimitris G.Manolakis, Digital Signal Processing - Principles, Algorithms & Applications, 4th Edition, Pearson Education / Prentice Hall, 2007.
2. Haykin. S and Van Been. B., Signals and Systems, 2nd Edition, John Wiley & Sons, 2003.
3. Willis J.Tompkins, Biomedical Digital Signal Processing, Prentice Hall of India Pvt. Ltd., 2012.
4. Sanjit K. Mitra, Digital Signal Processing - A Computer Based Approach, Tata McGraw Hill, 2007.
5. Oppenheim, A.V., R.W. Schafer and J.R. Buck, Discrete Time Signal Processing, 8th Indian Reprint, Pearson, 2004.
6. Stephane Mallat, Wavelet Tour of Signal Processing: The Sparse Way, 3rd Edition, Academic Press, 2008.

QUESTIONS

1. Define wavelet
2. What is the need for wavelet
3. Give the types of wavelets
4. Write the equation of Haar wavelet
5. Define decomposition
6. Define reconstruction
7. Mention some medical application of wavelets
8. Explain in detail about the decomposition and reconstruction procedure of wavelets
9. Write about the denoising concepts of wavelets in detail
10. Classify wavelets and explain its classification with necessary waveforms