

DIGITAL SIGNAL PROCESSING

(20A04502T)

III B .TECH I SEM – R20

By

Mr.M.P.Chennaiah
Associate Professor
Department of ECE
VEMU INSTITUTE OF TECHNOLOGY

UNIT-1

Introduction

Signal Processing

- Humans are the most advanced signal processors
 - speech and pattern recognition, speech synthesis,...
- We encounter many types of signals in various applications
 - Electrical signals: voltage, current, magnetic and electric fields,...
 - Mechanical signals: velocity, force, displacement,...
 - Acoustic signals: sound, vibration,...
 - Other signals: pressure, temperature,...
- Most real-world signals are analog
 - They are continuous in time and amplitude
 - Convert to voltage or currents using sensors and transducers
- Analog circuits process these signals using
 - Resistors, Capacitors, Inductors, Amplifiers,...
- Analog signal processing examples
 - Audio processing in FM radios
 - Video processing in traditional TV sets

Limitations of Analog Signal Processing

- Accuracy limitations due to
 - Component tolerances
 - Undesired nonlinearities
- Limited repeatability due to
 - Tolerances
 - Changes in environmental conditions
 - Temperature
 - Vibration
- Sensitivity to electrical noise
- Limited dynamic range for voltage and currents
- Inflexibility to changes
- Difficulty of implementing certain operations
 - Nonlinear operations
 - Time-varying operations
- Difficulty of storing information

Digital Signal Processing

- Represent signals by a sequence of numbers
 - Sampling or analog-to-digital conversions
- Perform processing on these numbers with a digital processor
 - Digital signal processing
- Reconstruct analog signal from processed numbers
 - Reconstruction or digital-to-analog conversion



- Analog input – analog output
 - Digital recording of music
- Analog input – digital output
 - Touch tone phone dialing
- Digital input – analog output
 - Text to speech
- Digital input – digital output
 - Compression of a file on computer

Pros and Cons of Digital Signal Processing

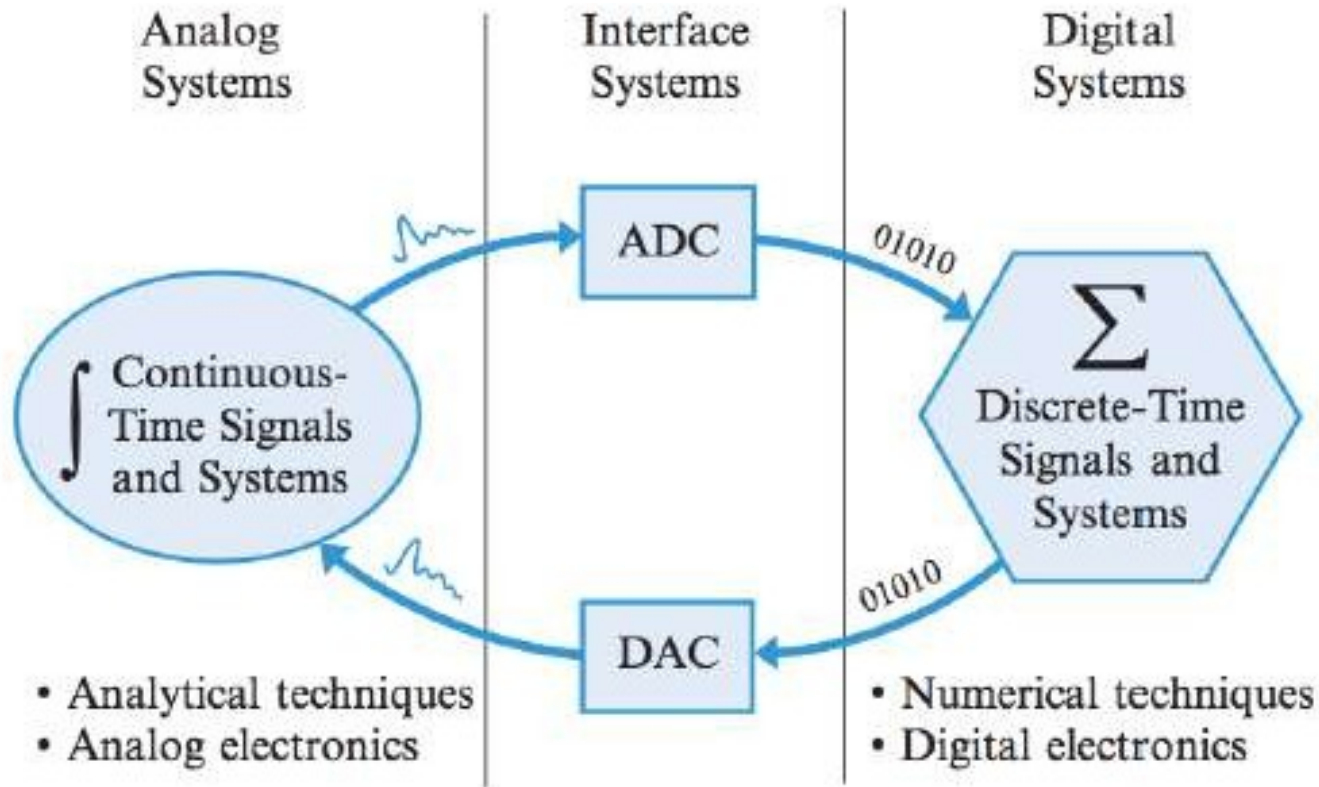
- Pros

- Accuracy can be controlled by choosing word length
- Repeatable
- Sensitivity to electrical noise is minimal
- Dynamic range can be controlled using floating point numbers
- Flexibility can be achieved with software implementations
- Non-linear and time-varying operations are easier to implement
- Digital storage is cheap
- Digital information can be encrypted for security
- Price/performance and reduced time-to-market

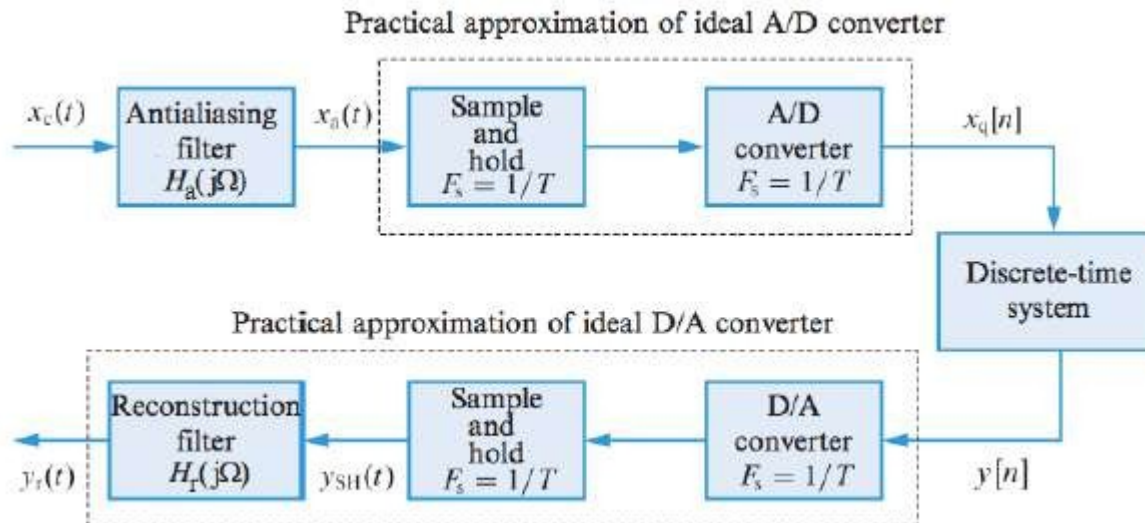
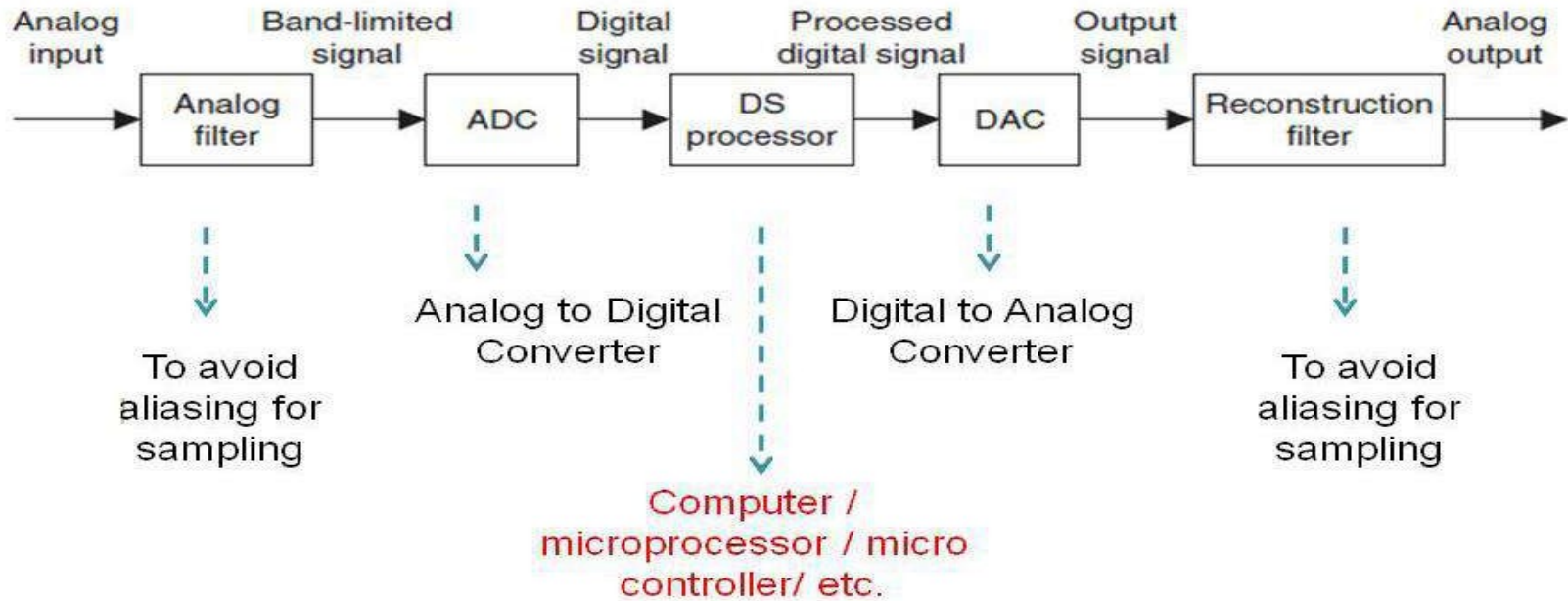
- Cons

- Sampling causes loss of information
- A/D and D/A requires mixed-signal hardware
- Limited speed of processors
- Quantization and round-off errors

Analog, digital, mixed signal processing

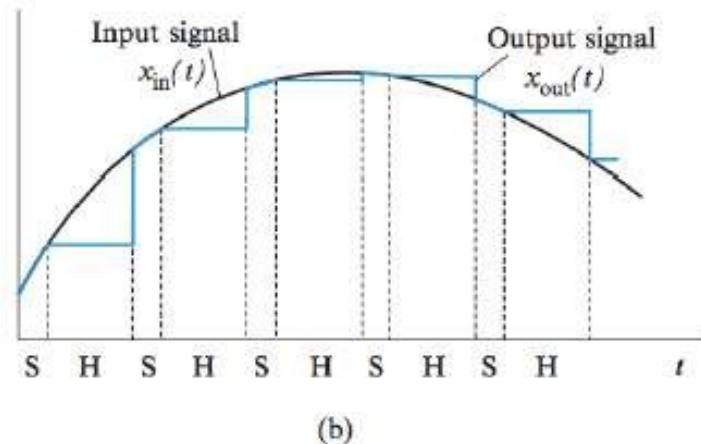
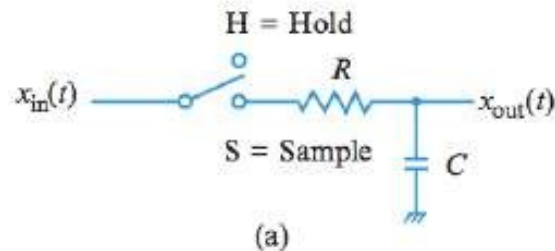


Digital Signal Processing



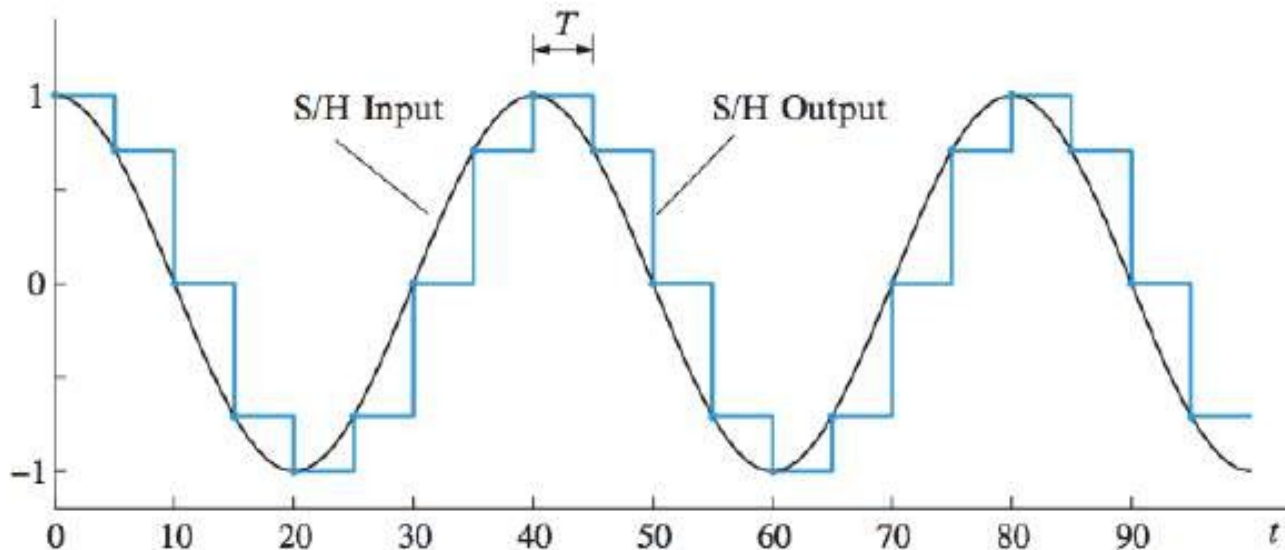
Sampling and reconstruction

- The main function of the *low-pass antialiasing filter* is to band-limit the input signal to the folding frequency without distortion.
- It should be noted that even if the signal is band-limited, there is always wide-band additive noise which will be folded back to create aliasing.
- When an analog voltage is connected directly to an ADC, the conversion process can be adversely affected if the voltage is changing during the conversion time.
- The quality of the conversion process can be improved by using a *sample-and-hold (S/H) circuit*.



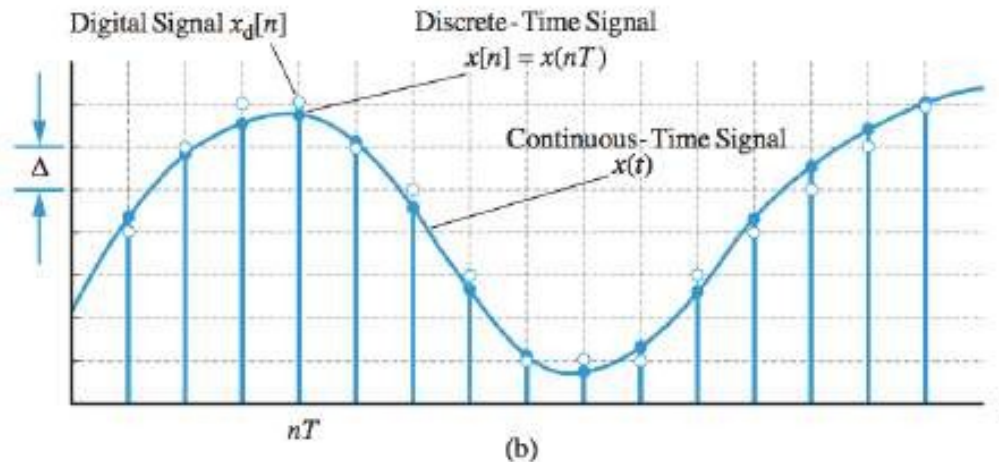
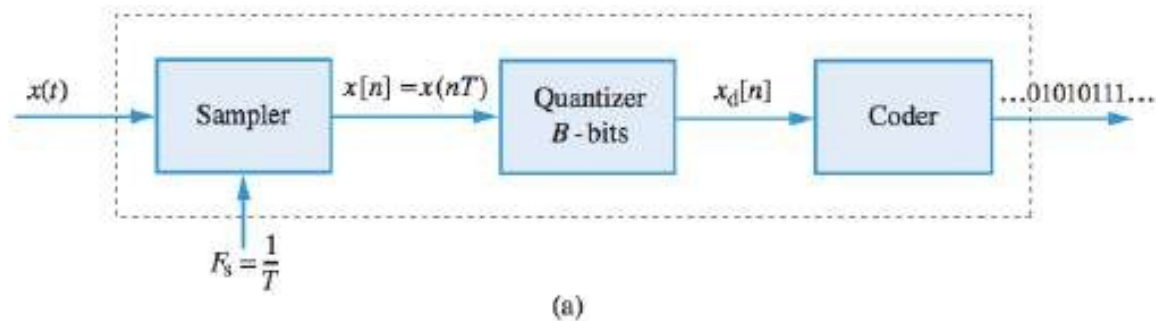
Sample and hold (S/H)circuit

- Since the sampling operation is performed by the S/H circuit, the role of S/H is to sample $x_c(t)$ as instantaneously as possible and to hold the sample value as constant as possible until the next sample.
- Thus, the output of the S/H circuit can be modelled as a staircase waveform where each sample value is held constant until the acquisition of the next sample.



- Note that the S/H system is linear but time-varying.

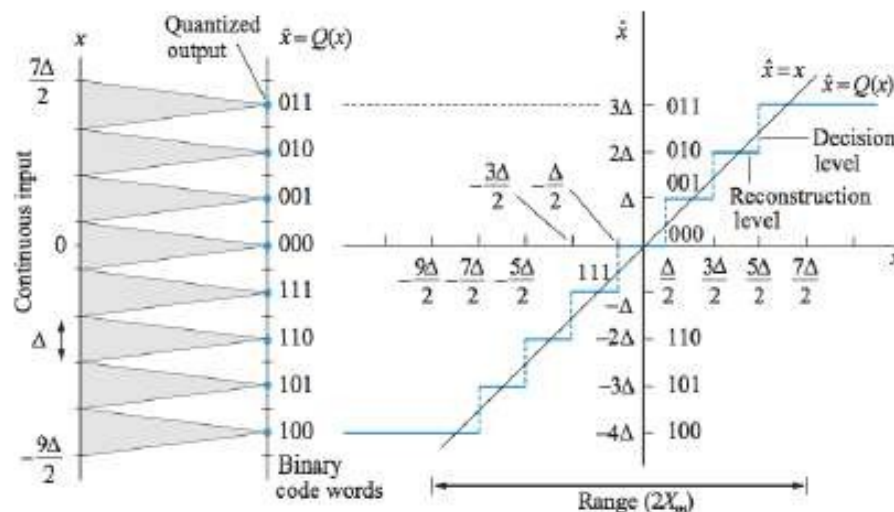
A/D converter



- *Quantization* converts a continuous-amplitude signal $x(t)$ into a discrete-amplitude signal $x_d[n]$.
- In theory, we are dealing with discrete-time signals; in practice, we are dealing with digital signals.

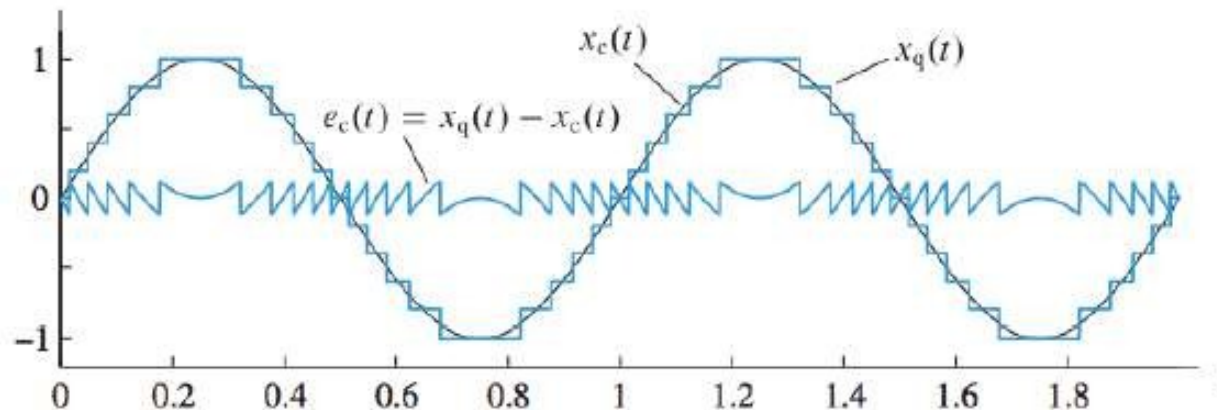
A/D converter

- The major difference between ideal and practical conversion is that an ADC generates sample values that are known with finite precision.
- The ADC is the device in which both quantization and binary coding of the sampled signal take place.
- A B -bit quantizer can represent 2^B different numbers.
- If the input amplitude range is divided into K quantization intervals of equal width Δ (*quantization step*) and the output levels are uniformly spaced, the resulting quantizer is called *uniform*.



Quantization noise

- The two major types of error introduced by an ADC are aliasing error and *quantization error*.



- Since quantization is a nonlinear operation, analysis of quantization error is done using statistical techniques.
- If there is a large number of small quantization intervals, the signal $x_c(t)$ can be assumed to be approximately linear between quantization levels. In this case:

$$e_c(t) \triangleq x_q(t) - x_c(t) = \frac{\Delta}{2\tau}t, \quad -\tau \leq t \leq \tau$$

- Then the mean squared quantization error power is

$$P_Q = \frac{1}{2\tau} \int_{-\tau}^{\tau} |e_c(t)|^2 dt = \frac{\Delta^2}{12}$$

D/A conversion

- A band-limited signal can be reconstructed from a sequence of samples using the ideal DAC described by

$$x_r(t) = \sum_n x[n] g_{BL}(t - nT) = \sum_n x[n] \text{sinc}(t/T - n)$$

- A system that implements the above formula, for an arbitrary function $g_r(t)$, is known as a *practical digital-to-analog converter (DAC)*.
- The function $g_r(t)$ is also known as the *characteristic pulse* of a DAC. At each sample time $t = nT$, the converter generates a pulse $g_r(t - nT)$ scaled by $x[n]$.
- In particular, the *switch-and-hold* DAC performs the following operation

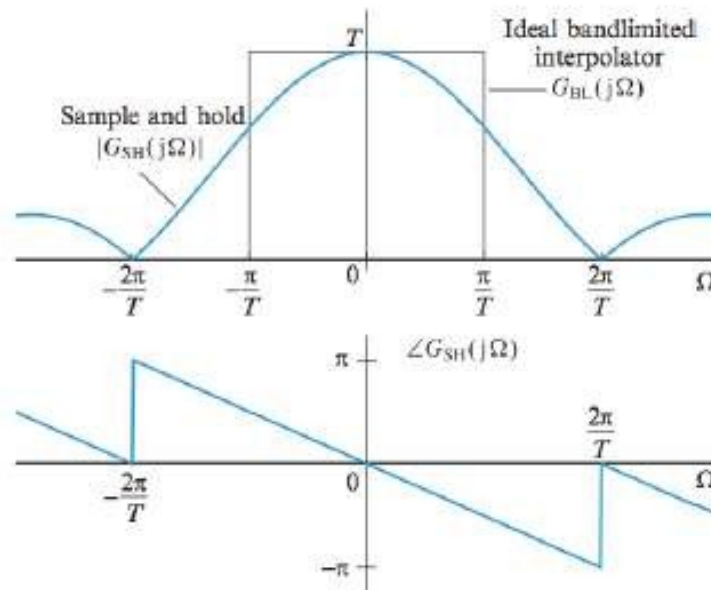
$$x_{SH}(t) = \sum_n x_q[n] g_{SH}(t - nT)$$

where

$$g_{SH}(t) = \begin{cases} 1, & 0 \leq t \leq T \\ 0, & \text{otherwise} \end{cases} \iff G_{SH}(j\Omega) = \frac{2 \sin(\Omega T/2)}{\Omega} e^{-j\Omega T/2}$$

D/A conversion

- The S/H circuit cannot completely eliminate the spectral replicas introduced by the sampling process.
- Moreover, it introduces amplitude distortion in the Nyquist band $|F_s| < F_s/2$.



- To compensate for the effects of the S/H circuit, we can use an analog post-filter $H_r(j\Omega)$ so that $G_{SH}(j\Omega) H_r(j\Omega) = G_{BL}(j\Omega)$:

$$H_r(j\Omega) = \begin{cases} \frac{\Omega T/2}{\sin(\Omega T/2)} e^{j\Omega T/2}, & |\Omega| < \pi/T \\ 0, & \text{otherwise} \end{cases}$$

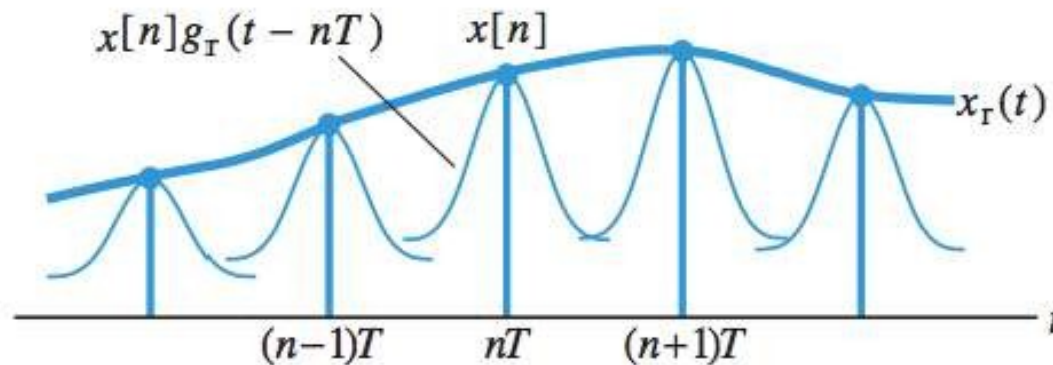
Reconstruction

- A general formula that describes a broad class of reconstruction processes is given by

$$x_r(t) = \sum_n x[n]g_r(t - nT)$$

where $g_r(t)$ is an interpolating reconstruction function.

- The process of fitting a continuous function to a set of samples is known as an *interpolation*.



- Thus, if the interpolation function has duration greater than or equal to T , the addition of the overlapping copies “fills the gaps” between samples.

Reconstruction

- In the Fourier domain, the interpolation formula becomes

$$X_r(j\Omega) = \sum_n x[n] G_r(j\Omega) e^{-j\Omega T} = G_r(j\Omega) \underbrace{\sum_n x[n] e^{-j\Omega T}}_{X(e^{j\Omega T})}$$

- Consequently, we obtain

$$X_r(j\Omega) = G_r(j\Omega) X(e^{j\Omega T})$$

- Specifically, if we choose $g_r(t)$ so that

$$G_r(j\Omega) \triangleq G_{BL}(j\Omega) = \begin{cases} T, & |\Omega| \leq \Omega_s/2 \\ 0, & |\Omega| > \Omega_s/2 \end{cases}$$

then $X_r(j\Omega) = X_c(j\Omega)$ and, therefore, $x_r(t) = x_c(t)$.

Reconstruction

- Evaluating the inverse Fourier transform of $G_{BL}(j\Omega)$, we obtain

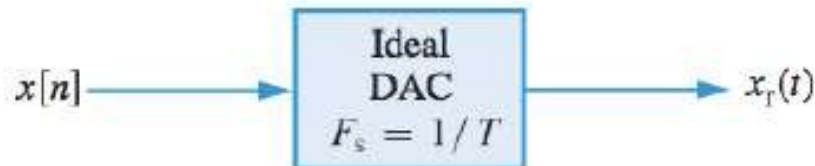
$$g_r(t) \triangleq g_{BL}(t) = \frac{\sin(\pi t/T)}{\pi t/T} = \text{sinc}(t/T)$$

- In this case we obtain:

The ideal interpolation formula

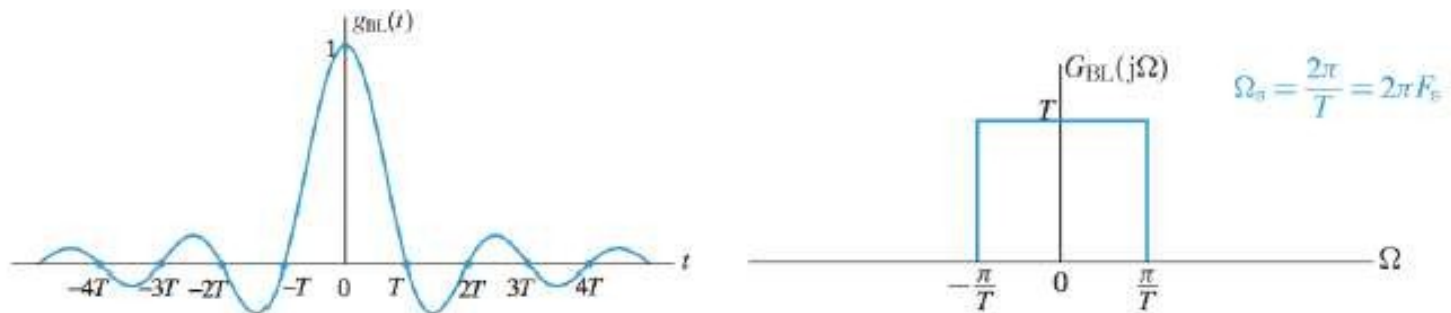
$$x_c(t) = \sum_n x[n] \frac{\sin[\pi(t - nT)/T]}{\pi(t - nT)/T}$$

- The system used to implement the ideal interpolation is known as an *ideal DAC*.

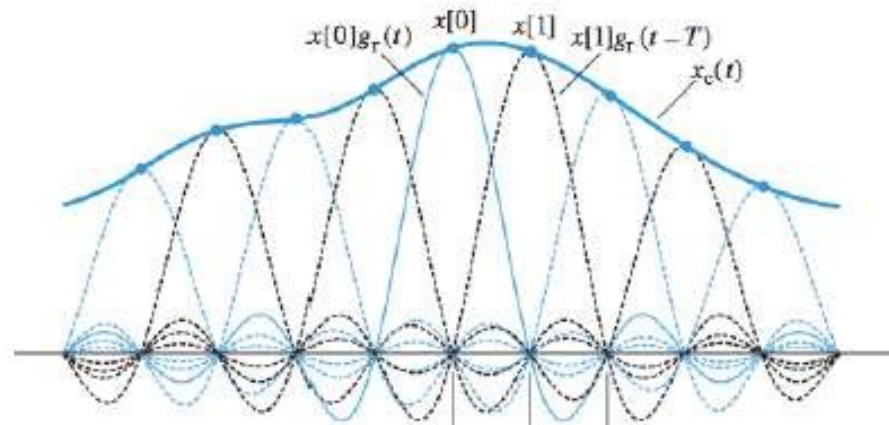


Reconstruction

- To understand the meaning and implications of the ideal interpolation we look more closely at the sinc function $g_{BL}(t)$.



- We note that $g_{BL}(t) = 0$ at all $t = nT$, except at $t = 0$ where $g_{BL}(t) = 1$. Thus, it is always true that $x_r(nT) = x_c(nT)$ regardless of whether aliasing occurred during sampling.



Signals

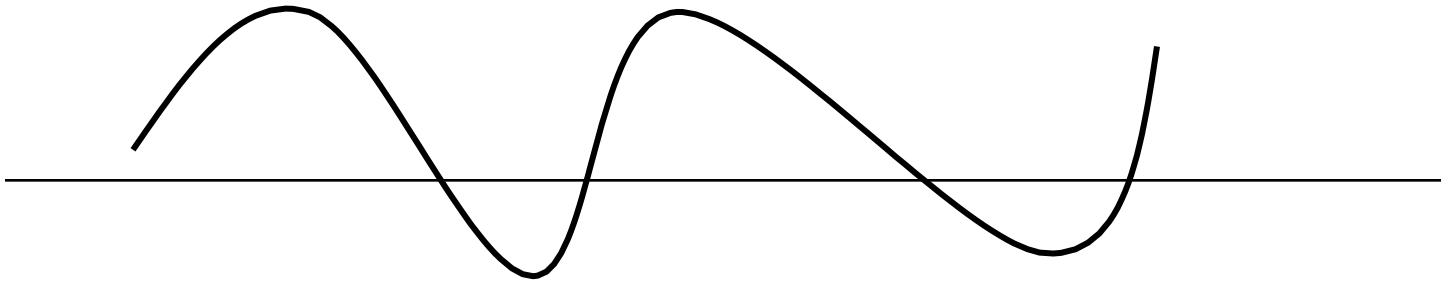
- *Continuous-time* signals are functions of a real argument $x(t)$ where t can take any real value
 $x(t)$ may be 0 for a given range of values of t
- *Discrete-time signals* are functions of an argument that takes values from a discrete set
 $x[n]$ where $n \in \{\dots-3,-2,-1,0,1,2,3\dots\}$
Integer index n instead of time t for discrete-time systems
- \mathbf{x} may be an array of values (multi channel signal)
- Values for \mathbf{x} may be real or complex

Discrete-time Signals and Systems

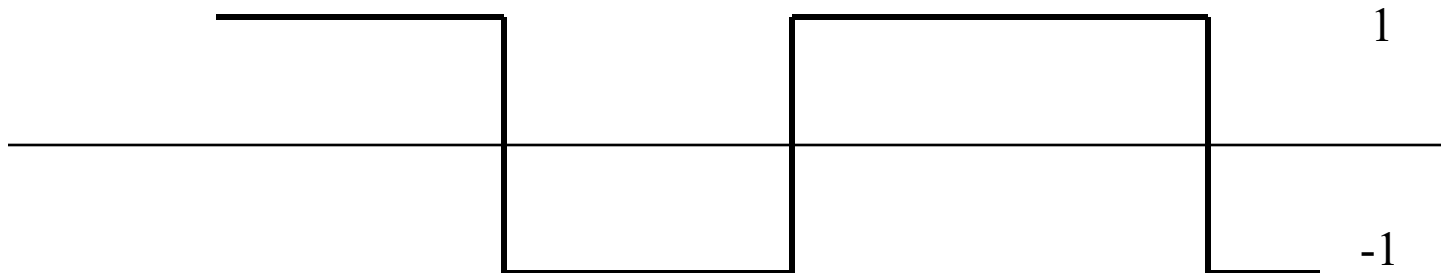
- *Continuous-time signals* are defined over a continuum of times and thus are represented by a continuous independent variable.
- *Discrete-time signals* are defined at discrete times and thus the independent variable has discrete values.
- *Analog signals* are those for which both time and amplitude are continuous.
- *Digital signals* are those for which both time and amplitude are discrete.

Analog vs. Digital

- The amplitude of an analog signal can take any real or complex value at each time/sample

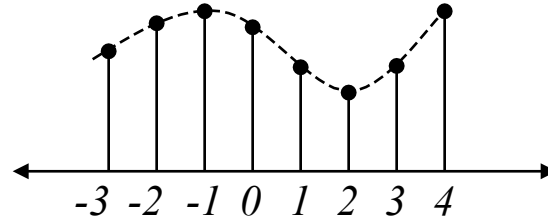


- The amplitude of a digital signal takes values from a discrete set



Periodic (Uniform) Sampling

- Sampling is a continuous to discrete-time conversion



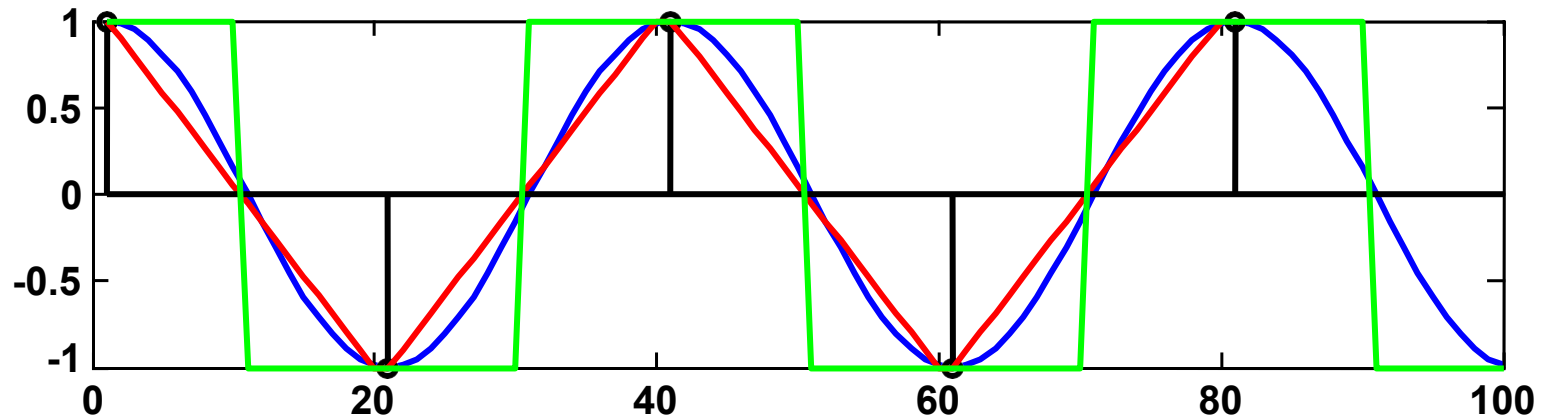
- Most common sampling is periodic

$$x[n] = x_c(nT) \quad -\infty < n < \infty$$

- T is the sampling period in second
- $f_s = 1/T$ is the sampling frequency in Hz
- Sampling frequency in radian-per-second $\Omega_s = 2\pi f_s$ rad/sec
- Use $[\cdot]$ for discrete-time and (\cdot) for continuous time signals
- This is the ideal case not the practical but close enough
 - In practice it is implement with an analog-to-digital converters
 - We get digital signals that are quantized in amplitude and time

Periodic Sampling

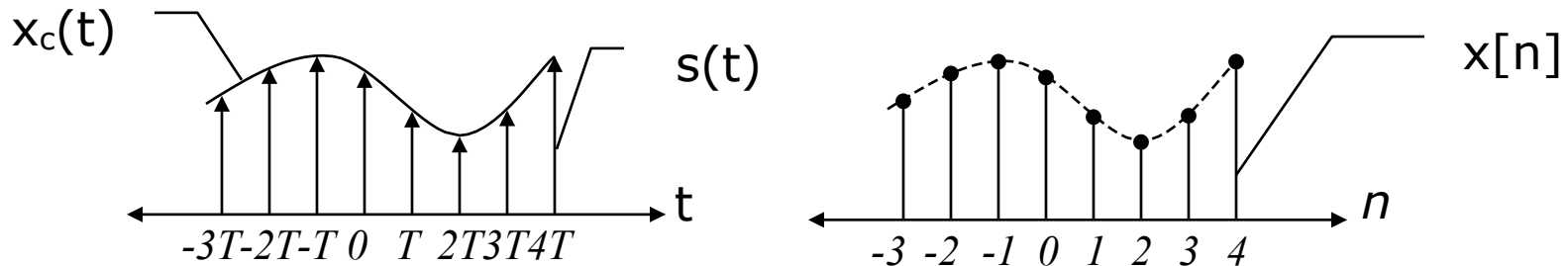
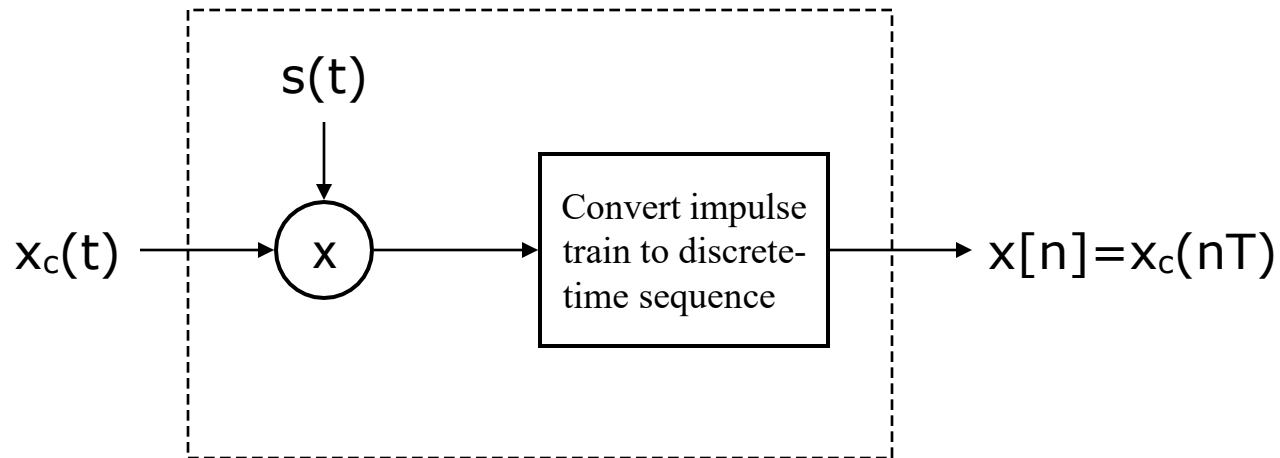
- Sampling is, in general, not reversible
- Given a sampled signal one could fit infinite continuous signals through the samples



- Fundamental issue in digital signal processing
 - If we lose information during sampling we cannot recover it
- *Under certain conditions an analog signal can be sampled without loss so that it can be reconstructed perfectly*

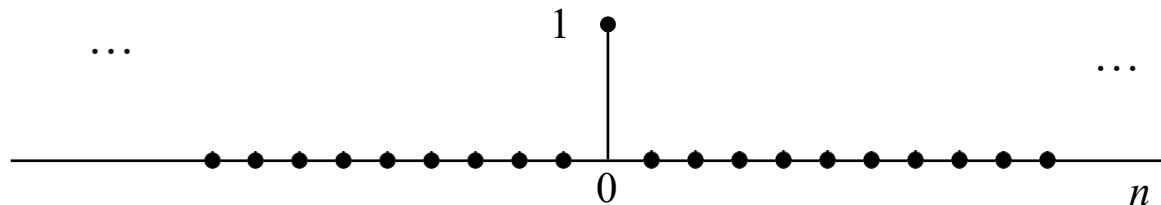
Representation of Sampling

- Mathematically convenient to represent in two stages
 - Impulse train modulator
 - Conversion of impulse train to a sequence



Unit Sample Sequence

$$\begin{aligned}\delta[n] &= 0, \quad n \neq 0 \\ &= 1, \quad n = 0.\end{aligned}$$



The unit sample sequence plays the same role for discrete-time sequences and systems that the unit impulse (Dirac delta function) does for continuous-time signals and systems.

Impulse Function

The *impulse function*, also known as Dirac's delta function, is used to represent quantities that are highly localized in space. Examples include point optical sources and electrical charges.

The impulse function can be visualized as a narrow spike having infinite height and zero width, such that its area is equal to unity.

Definition of Impulse Function

The impulse function may be defined from its basic properties.

$$\delta(x - x_0) = 0, \quad x \neq x_0$$

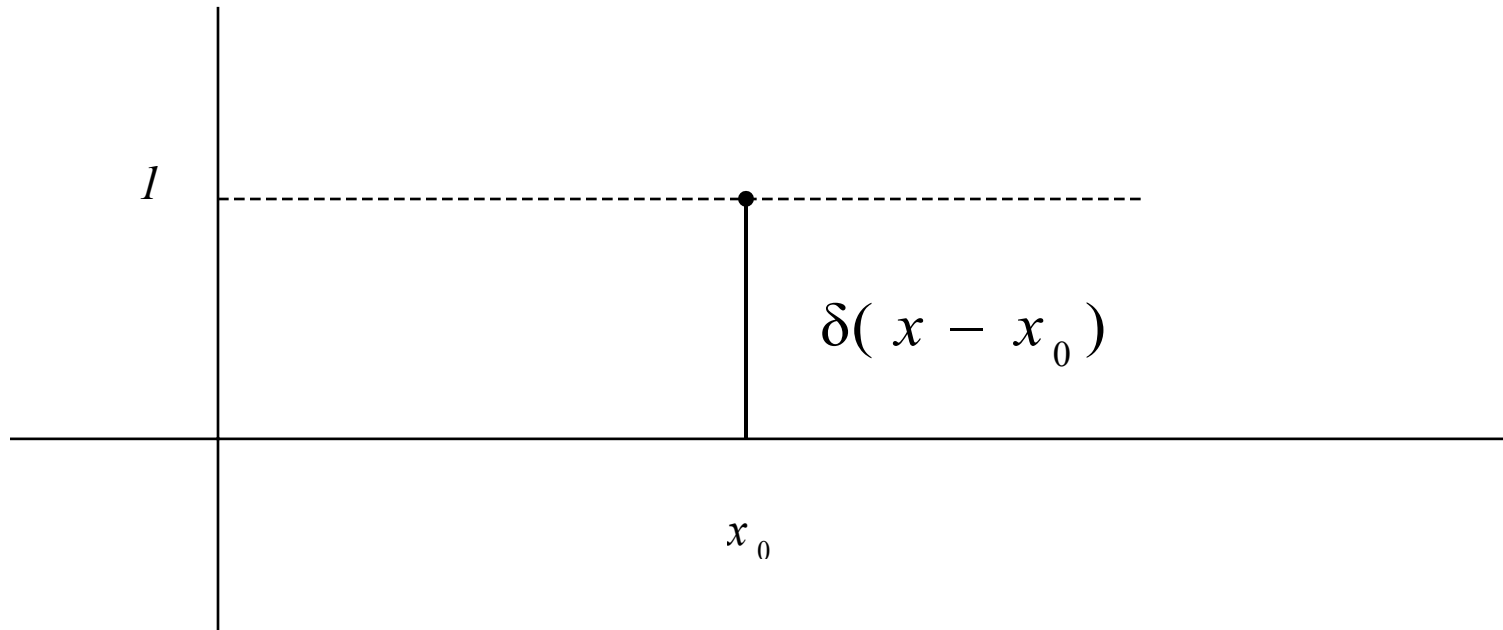
$$\int_{x_1}^{x_2} f(x) \delta(x - x_0) dx = f(x_0), \quad x_1 < x_0 < x_2$$

Where $f(x)$ is any complex-valued function of x . If $f(x)$ is discontinuous at the point x_0 , the value of $f(x_0)$ is taken as the average of the limiting values as x approaches x_0 from above and below.

This property is called the *sifting* property.

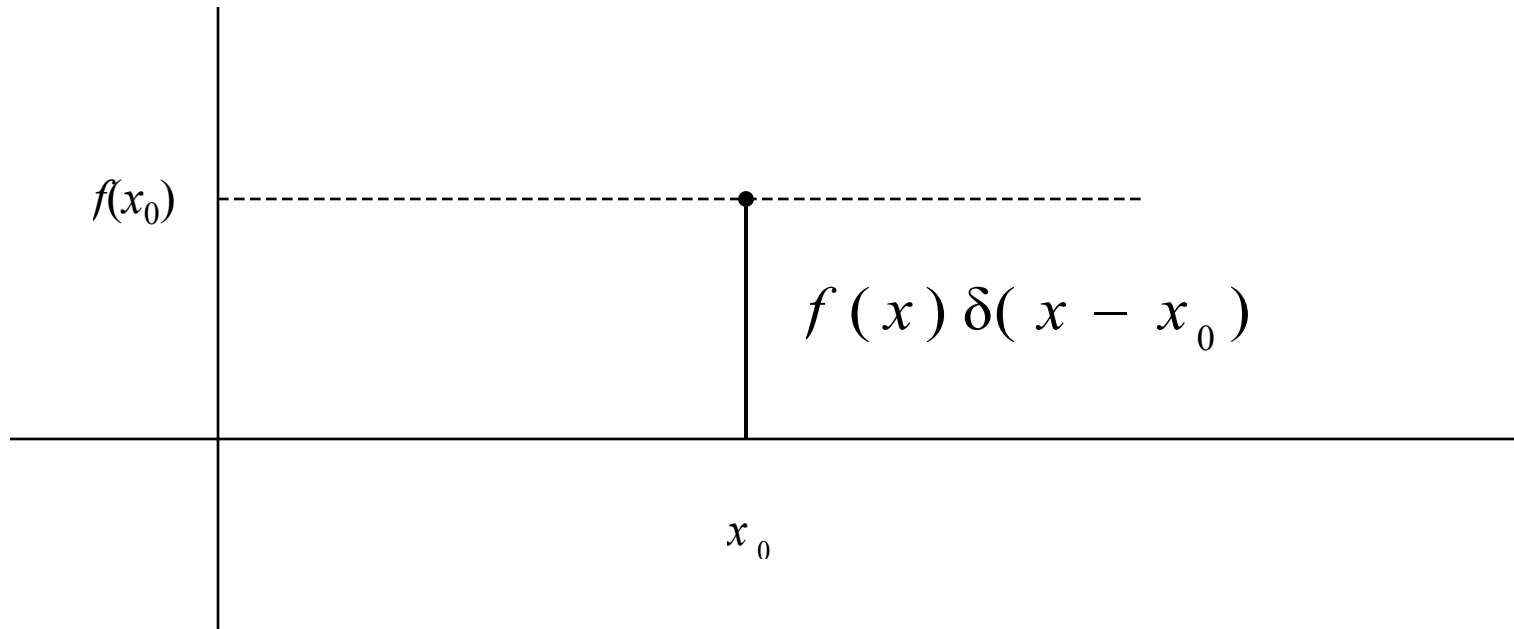
Graphical Representation

On graphs we will represent $\delta(x-x_0)$ as a spike of unit height located at the point x_0 .



Sampling Operation

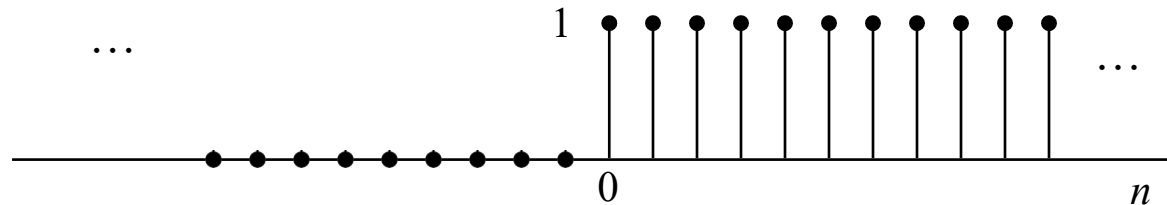
The delta function *samples* the function $f(x)$.



The function $f(x) \delta(x-x_0)$ is graphed as a spike of height $f(x_0)$ located at the point x_0 .

Unit Step Sequence

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



$$u[n] = \delta[n] + \delta[n-1] + \delta[n-2] + \dots$$

$$u[n] = \sum_{k=0}^{\infty} \delta[n-k]$$

Conversely, the impulse sequence can be expressed as the first backward difference of the unit step sequence:

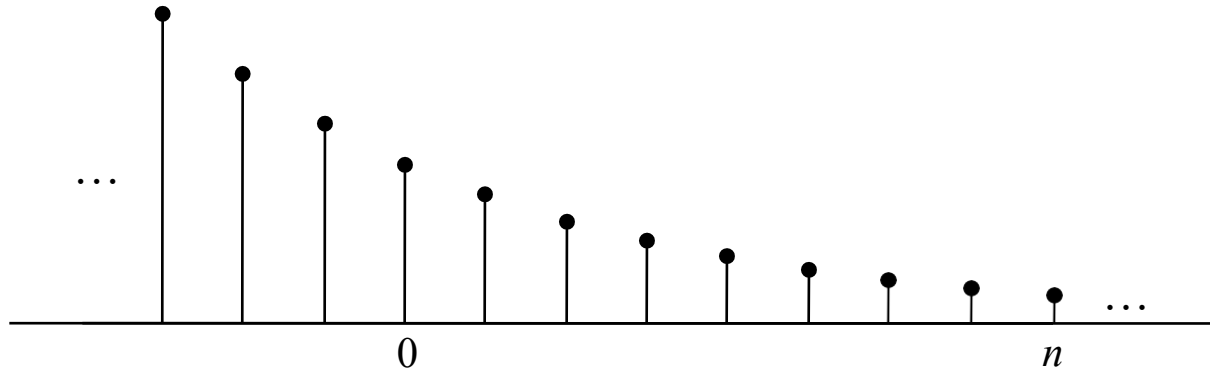
or

$$u[n] = \sum_{k=-\infty}^n \delta[k]$$

$$\delta[n] = u[n] - u[n-1]$$

Exponential Sequence

$$x[n] = A \alpha^n$$



If we want an exponential sequence that is zero for $n < 0$, we can write this as:

$$x[n] = A \alpha^n u[n]$$

Geometric Series

A one-sided exponential sequence of the form

α^n , for $n \geq 0$ and α an arbitrary constant

is called a *geometric series*. The series converges for $|\alpha| < 1$, and its sum converges to

$$\sum_{n=0}^{\infty} \alpha^n \rightarrow \frac{1}{1 - \alpha}$$

The sum of a finite number N of terms is

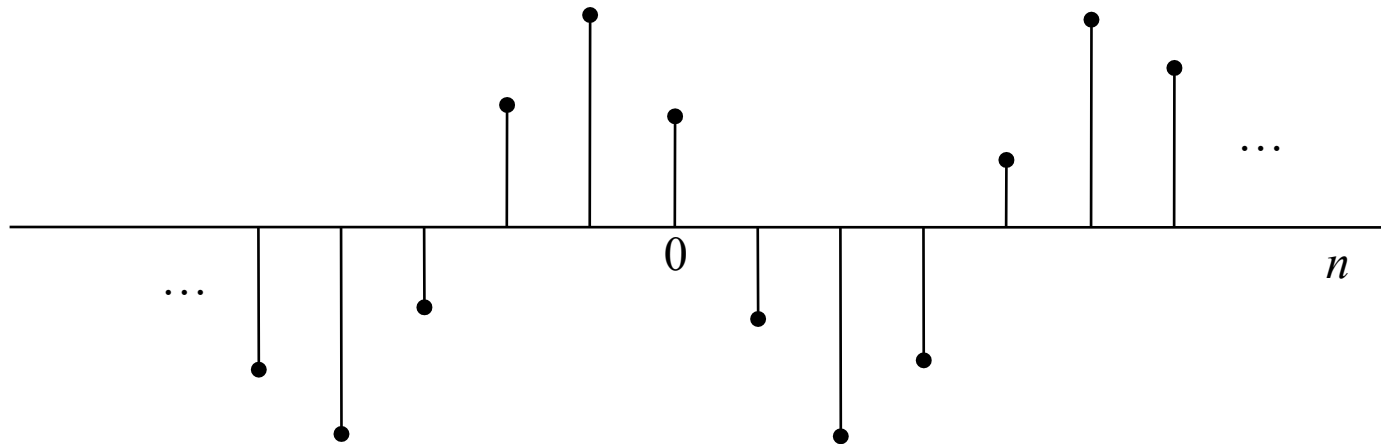
$$\sum_{n=0}^N \alpha^n \rightarrow \frac{1 - \alpha^{N+1}}{1 - \alpha}$$

A general form can also be written:

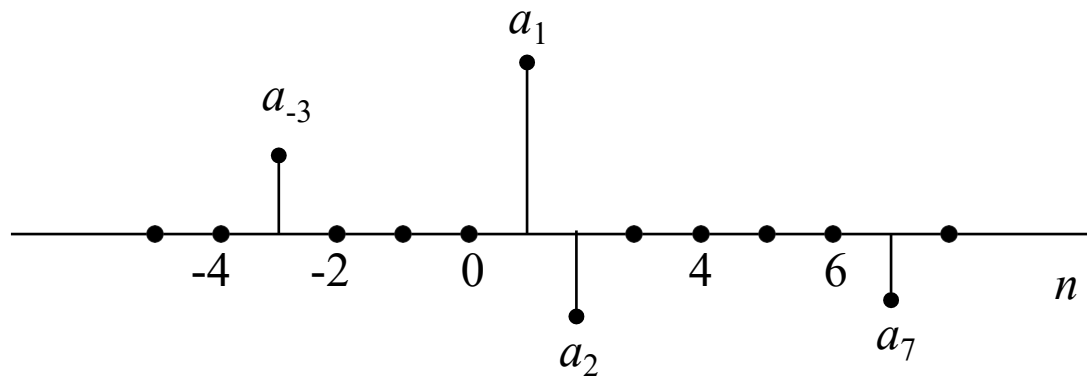
$$\sum_{n=N_1}^{N_2} \alpha^n \rightarrow \frac{\alpha^{N_1} - \alpha^{N_2+1}}{1 - \alpha}$$

Sinusoidal Sequence

$$x[n] = A \cos(\omega_o n + \phi)$$



Sequence as a sum of scaled, delayed impulses



$$p[n] = a_{-3} \delta[n + 3] + a_1 \delta[n - 1] - a_2 \delta[n - 2] - a_7 \delta[n - 7]$$

Sequence Operations

- The product and sum of two sequences are defined as the sample-by-sample product and sum, respectively.
- Multiplication of a sequence by a number is defined as multiplication of each sample value by this number.
- A sequence $y[n]$ is said to be a delayed or shifted version of a sequence $x[n]$ if

$$y[n] = x[n - n_d]$$

where n_d is an integer.

- Combination of Basic Sequences

Ex 1

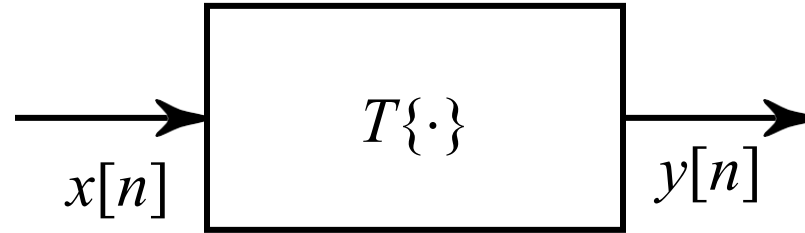
$$x[n] = K\alpha^n, \quad n \geq 0,$$
$$= 0, \quad n < 0,$$

or

$$x[n] = K\alpha^n u[n].$$

Systems

Systems

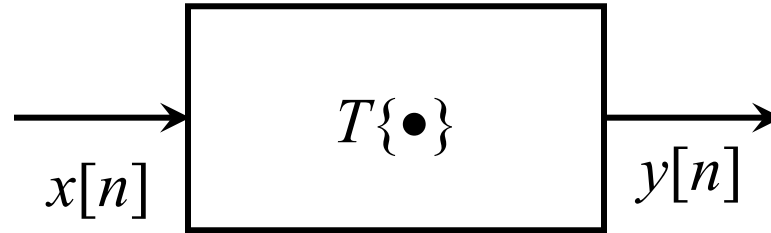


A discrete-time system is a transformation that maps an input sequence $x[n]$ into an output sequence $y[n]$.

System Characteristics

1. Linear vs. non-linear
2. Causal vs. non-causal
3. Time invariant

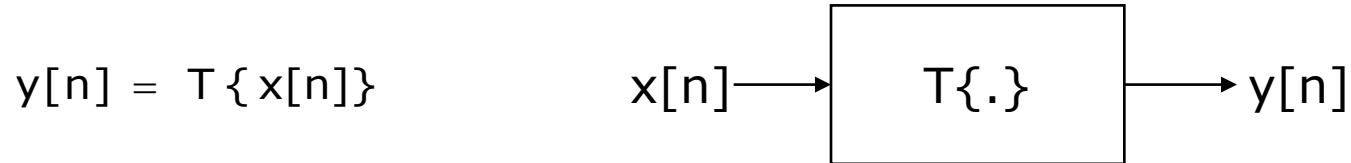
System Characteristics



1. Linear vs. non-linear
2. Time invariant vs. time variant
3. Causal vs. non-causal
4. Stable vs. unstable
5. Memoryless vs. state-sensitive
6. Invertible vs. non-invertible

Discrete-Time Systems

- Discrete-Time Sequence is a mathematical operation that maps a given input sequence $x[n]$ into an output sequence $y[n]$



- Example Discrete-Time Systems

- Moving (Running) Average

$$y[n] = x[n] + x[n - 1] + x[n - 2] + x[n - 3]$$

- Maximum

$$y[n] = \max \{x[n], x[n - 1], x[n - 2]\}$$

- Ideal Delay System

$$y[n] = x[n - n_o]$$

Linearity

A **linear** system is one that obeys the principle of superposition,

$$T \{ a_1 x [n] + a_2 x [n] \} = a_1 y_1 [n] + a_2 y_2 [n]$$

where the output of a linear combination of inputs is the same linear combination applied to the individual outputs. This result means that a complicated system can be decomposed into a linear combination of elementary functions whose transformation is known, and then taking the same linear combination of the results. Linearity also implies that the behavior of the system is independent of the magnitude of the input.

Linear Systems

- Linear System: A system is linear if and only if

$$T\{x_1[n] + x_2[n]\} = T\{x_1[n]\} + T\{x_2[n]\} \quad (\text{additivity})$$

and

$$T\{ax[n]\} = aT\{x[n]\} \quad (\text{scaling})$$

- Examples

- Ideal Delay System

$$y[n] = x[n - n_o]$$

$$T\{x_1[n] + x_2[n]\} = x_1[n - n_o] + x_2[n - n_o]$$

$$T\{x_2[n]\} + T\{x_1[n]\} = x_1[n - n_o] + x_2[n - n_o]$$

$$T\{ax[n]\} = ax_1[n - n_o]$$

$$aT\{x[n]\} = ax_1[n - n_o]$$

Time (Shift) Invariance

A system is said to be **shift invariant** if the only effect caused by a shift in the position of the input is an equal shift in the position of the output, that is

$$T \{ x [n - n_0] \} = y [n - n_0]$$

The magnitude and shape of the output are unchanged, only the location of the output is changed.

Time-Invariant Systems

- Time-Invariant (shift-invariant) Systems
 - A time shift at the input causes corresponding time-shift at output

$$y[n] = T\{x[n]\} \Rightarrow y[n - n_o] = T\{x[n - n_o]\}$$

- Example
 - Square

	Delay the input	the output is	$y_1[n] = (x[n - n_o])^2$
$y[n] = (x[n])^2$	Delay the output	gives	$y[n - n_o] = (x[n - n_o])^2$

- Counter Example
 - Compressor System

	Delay the input	the output is	$y_1[n] = x[Mn - n_o]$
$y[n] = x[Mn]$	Delay the output	gives	$y[n - n_o] = x[M(n - n_o)]$

Impulse Response

When the input to a system is a single impulse, the output is called the **impulse response**. Let $h[n]$ be the impulse response, given by

$$T \{ \delta [n] \} = h[n]$$

A general sequence $f[x]$ can be represented as a linear combination of impulses, since

$$f(x) = f(x) * \delta(x) = \int_{-\infty}^{\infty} f(u) \delta(x - u) du$$

$$f[n] = f[n] * \delta[n] = \sum_{k=-\infty}^{\infty} f[k] \delta[n - k]$$

Linear Shift-Invariant Systems

Suppose that $T\{\}$ is a linear, shift-invariant system with $h[n]$ as its impulse response.

Then, using the principle of superposition,

$$T\{s[n]\} = T\left\{\sum_{k=-\infty}^{\infty} s[k]\delta[n-k]\right\} = \sum_{k=-\infty}^{\infty} s[k]T\{\delta[n-k]\}$$

and finally after invoking shift-invariance

$$T\{s[n]\} = \sum_{k=-\infty}^{\infty} s[k]T\{\delta[n-k]\} = \sum_{k=-\infty}^{\infty} s[k]h[n-k]$$

$$T\{s[n]\} = s[n] * h[n]$$

This very important result says that the output of any linear, shift-invariant system is given by the convolution of the input with the impulse response of the system.

Causality

A system is *causal* if, for every choice of n_0 , the output sequence at the index $n = n_0$ depends only on the input sequence values for $n \leq 0$.

All physical time-based systems are causal because they are unable to look into the future and anticipate a signal value that will occur later.

Causal System

- Causality

- A system is causal if its output is a function of only the current and previous samples

- Examples

- Backward Difference

$$y[n] = x[n] - x[n - 1]$$

- Counter Example

- Forward Difference

$$y[n] = x[n + 1] + x[n]$$

Stability

A system is stable in the bounded-input, bounded-output (BIBO) sense if and only if every bounded input produces a bounded output sequence.

The input $x[n]$ is bounded if there exists a fixed positive finite value B_x such that

$$|x[n]| \leq B_x < \infty \quad \text{for all } n$$

Stability requires that for any possible input sequence there exist a fixed positive value B_y such that

$$|y[n]| \leq B_y < \infty$$

Stable System

- Stability (in the sense of bounded-input bounded-output BIBO)
 - A system is stable if and only if every bounded input produces a bounded output

$$|x[n]| \leq B_x < \infty \Rightarrow |y[n]| \leq B_y < \infty$$

- Example
 - Square

$$y[n] = (x[n])^2$$

if input is bounded by $|x[n]| \leq B_x < \infty$

output is bounded by $|y[n]| \leq B_x^2 < \infty$

- Counter Example
 - Log

$$y[n] = \log_{10}(|x[n]|)$$

even if input is bounded by $|x[n]| \leq B_x < \infty$

output not bounded for $x[n] = 0 \Rightarrow y[0] = \log_{10}(|x[n]|) = -\infty$

Memory (State)

A system is referred to as *memoryless* if the output $y[n]$ at every value of n depends only on the input $x[n]$ at the same value of n .

If the system has no memory, it is called a *static* system. Otherwise it is a *dynamic* system.

Memoryless System

- Memoryless System
 - A system is memoryless if the output $y[n]$ at every value of n depends only on the input $x[n]$ at the same value of n
- Example Memoryless Systems
 - Square
$$y[n] = (x[n])^2$$
 - Sign
$$y[n] = \text{sign } \{x[n]\}$$
- Counter Example
 - Ideal Delay System
$$y[n] = x[n - n_o]$$

Invertible System

A system is *invertible* if for each output sequence we can find a unique input sequence. If two or more input sequences produce the same output sequence, the system is not invertible.

Passive and Lossless Systems

A system is said to be *passive* if, for every finite energy input sequence $x[n]$, the output sequence has at most the same energy:

$$\sum_{n=-\infty}^{\infty} |y[n]|^2 \leq \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty$$

If the above inequality is satisfied with an equal sign for every input signal, the system is said to be *lossless*.

Examples of Systems

Ideal Delay System $y[n] = x[n - n_d]$

Moving Average System
$$y[n] = \frac{1}{M_2 + M_1 + 1} \sum_{k=-M_1}^{M_2} x[n - k]$$

Memoryless non-linear System $y[n] = x[n]^2$

Accumulator System
$$y[n] = \sum_{k=-\infty}^n x[k]$$

Compressor System $y[n] = x[Mn]$ where M is a positive integer

Backward Difference System $y[n] = x[n] - x[n - 1]$

Impulse Response of LTI Systems

Find the impulse response by computing the response to $\delta[n]$.

Systems whose impulse responses have only a finite number of nonzero samples are called *finite-duration impulse response* (FIR) systems.

Systems whose impulse responses are of infinite duration are called *infinite-duration impulse response* (IIR) systems.

If $h[n] = 0$ for $n < 0$, the system is *causal*.

Impulse Response for Examples

Find the impulse response by computing the response to $\delta[n]$

Ideal Delay System $y[n] = \delta[n - n_d]$ **FIR**

Moving Average System $y[n] = \begin{cases} \frac{1}{M_2 + M_1 + 1}, & -M_1 \leq n \leq M_2 \\ 0, & \text{otherwise} \end{cases}$ **FIR**

Accumulator System $y[n] = \sum_{k=-\infty}^n \delta[k] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}$ **IIR**

$$y[n] = u[n]$$

Backward Difference System $y[n] = \delta[n] - \delta[n - 1]$ **FIR**

Stability Condition for LTI Systems

An LTI system is BIBO stable if and only if its impulse response is absolutely summable, that is

$$S = \sum_{k=-\infty}^{\infty} |h[k]| < \infty$$

Stable and Causal LTI Systems

- An LTI system is (BIBO) stable if and only if
 - Impulse response is absolute summable

$$\sum_{k=-\infty}^{\infty} |h[k]| < \infty$$

- Let's write the output of the system as

$$|y[n]| = \left| \sum_{k=-\infty}^{\infty} h[k]x[n-k] \right| \leq \sum_{k=-\infty}^{\infty} |h[k]| |x[n-k]|$$

- If the input is bounded

$$|x[n]| \leq B_x$$

- Then the output is bounded by

$$|y[n]| \leq B_x \sum_{k=-\infty}^{\infty} |h[k]|$$

- The output is bounded if the absolute sum is finite

- An LTI system is causal if and only if

$$h[k] = 0 \quad \text{for } k < 0$$

Difference Equations

An important subclass of LTI systems are defined by an N th-order linear constant-coefficient difference equation:

$$\sum_{k=0}^N a_k y[n-k] = \sum_{m=0}^M b_m x[n-m]$$

Often the leading coefficient $a_0 = 1$. Then the output $y[n]$ can be computed recursively from

$$y[n] = - \sum_{k=1}^N a_k y[n-k] + \sum_{m=0}^M b_m x[n-m]$$

A causal LTI system of this form can be simulated in MATLAB using the function **filter**

y = filter(a,b,x) ;

Accumulator Example

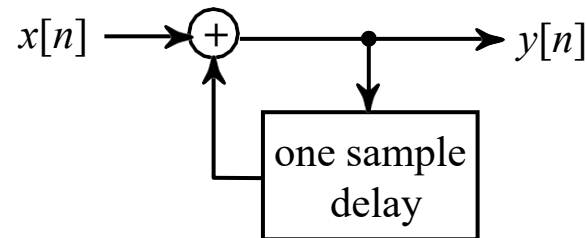
Accumulator System $y[n] = \sum_{k=-\infty}^n x[k]$

$$y[n] = \sum_{k=-\infty}^n x[k] = x[n] + \sum_{k=-\infty}^{n-1} x[k] = x[n] + y[n-1];$$

$$b_0 = 1$$

$$a_0 = 1$$

$$a_1 = -1$$



positive feedback system

Total Solution Calculation

$$\sum_{k=0}^N a_k y[n-k] = \sum_{m=0}^M b_m x[n-m]$$

The output sequence $y[n]$ consists of a homogeneous solution $y_h[n]$ and a particular solution $y_p[n]$.

$$y[n] = y_h[n] + y_p[n]$$

where the homogenous solution $y_h[n]$ is obtained from the homogeneous equation:

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

Homogeneous Solution

Given the homogeneous equation:

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

Assume that the homogeneous solution is of the form

$$y_h[n] = \lambda^n$$

then

$$y_h[n] = \sum_{k=0}^N a_k \lambda^{n-k} = \lambda^{n-N} (a_0 \lambda^N + a_1 \lambda^{N-1} + \dots + a_N) = 0$$

defines an N th order characteristic polynomial with roots $\lambda_1, \lambda_2 \dots \lambda_N$

The general solution is then a sequence $y_h[n]$

$$y_h[n] = \sum_1^N A_m \lambda_m^n$$

(if the roots are all distinct)

The coefficients A_m may be found from the initial conditions.

Particular Solution

The particular solution is required to satisfy the difference equation for a specific input signal $x[n]$, $n \geq 0$.

$$\sum_{k=0}^N a_k y[n-k] = \sum_{m=0}^M b_m x[n-m]$$

To find the particular solution we assume for the solution $y_p[n]$ a form that depends on the form of the specific input signal $x[n]$.

$$y[n] = y_h[n] + y_p[n]$$

General Form of Particular Solution

Input Signal $x[n]$	Particular Solution $y_p[n]$
A (constant)	K
AM^n	KM^n
An^M	$K_0n^M + K_1n^{M-1} + \dots + K_M$
$A^n n^M$	$A^n (K_0n^M + K_1n^{M-1} + \dots + K_M)$
$\begin{cases} A \cos(\omega_o n) \\ A \sin(\omega_o n) \end{cases}$	$K_1 \cos(\omega_o n) + K_2 \sin(\omega_o n)$

Example (66

Determine the *homogeneous* solution for

$$y[n] + y[n - 1] - 6y[n - 2] = 0$$

Substitute $y_h[n] = \lambda^n$

$$\begin{aligned}\lambda^n + \lambda^{n-1} - 6\lambda^{n-2} &= \lambda^{n-2}(\lambda^2 + \lambda - 6) = 0 \\ &= \lambda^{n-2}(\lambda + 3)(\lambda - 2) = 0\end{aligned}$$

Homogeneous solution is then

$$y_h[n] = A_1 \lambda_1^n + A_2 \lambda_2^n = A_1 (-3)^n + A_2 (2)^n$$

Example (67

Determine the *particular* solution for

$$y[n] + y[n - 1] - 6 y[n - 2] = x[n]$$

with $x[n] = 8u[n]$ and $y[-1] = 1$ and $y[-2] = -1$

The particular solution has the form $y_p[n] = \beta$

$$\beta + \beta - 6\beta = 8$$

which is satisfied by $\beta = -2$

Example (68

Determine the *total* solution for

$$y[n] + y[n-1] - 6y[n-2] = x[n]$$

with $x[n] = 8u[n]$ and $y[-1] = 1$ and $y[-2] = -1$

The total solution has the form

$$y[n] = y_h[n] + y_p[n] = A_1(-3)^n + A_2(2)^n - 2$$

then

$$y[-1] = -\frac{1}{3}A_1 + \frac{1}{2}A_2 - 2 = 1 \quad A_1 = -1.8$$

$$y[-3] = \frac{1}{9}A_1 + \frac{1}{4}A_2 - 2 = -1 \quad A_2 = 4.8$$

$$y[n] = -1.8(-3)^n + 4.8(2)^n - 2$$

Initial-Rest Conditions

The output for a given input is not uniquely specified.
Auxiliary information or conditions are required.

Linearity, time invariance, and causality of the system will depend on the auxiliary conditions. If an additional condition is that the system is initially at rest (called *initial-rest conditions*), then the system will be linear, time invariant, and causal.

Zero-input, Zero-state Response

An alternate approach for determining the total solution $y[n]$ of a difference equation is by computing its *zero-input* response $y_{zi}[n]$, and *zero-state* response $y_{zs}[n]$. Then the total solution is given by $y[n] = y_{zi}[n] + y_{zs}[n]$.

The *zero-input* response is obtained by setting the input $x[n] = 0$ and satisfying the initial conditions. The *zero-state* response is obtained by applying the specified input with all initial conditions set to zero.

Example (1/2)

$$y[n] + y[n-1] - 6y[n-2] = 0$$

$$y[-1] = 1$$

$$y[-2] = -1$$

$$y_h[n] = A_1 \lambda_1^n + A_2 \lambda_2^n = A_1 (-3)^n + A_2 (2)^n$$

Zero-input response:

$$y_{zi}[0] = A_1 + A_2 = -y[-1] + 6y[-2] = -1 - 6 = -7$$

$$A_1 = -\frac{27}{5} = -5.4$$

$$y_{zi}[1] = A_1(-3) + A_2(2) = -y[0] + 6y[-1] = 7 + 6 = 13$$

$$A_2 = -\frac{8}{5} = -1.6$$

Zero-state response: $y[n] + y[n-1] - 6y[n-2] = x[n]$

with $x[n] = 8u[n]$

$$y_{zs}[0] = A_1 + A_2 - 2 = x[0] = 8$$

$$A_1 = \frac{18}{5} = 3.6$$

$$y_{zs}[1] = A_1(-3) + A_2(2) - 2 = x[1] - y[0] = 8 - 8 = 0$$

$$A_2 = \frac{32}{5} = 6.4^{71}$$

Mitra Example (2/2)

Zero-input response:

$$y_{zi}[n] = -5.4(-3)^n - 1.6(2)^n$$

Zero-state response:

$$y_{zs}[n] = 3.6(-3)^n + 6.4(2)^n - 2$$

Total solution is

$$y[n] = y_z[n] + y_{zs}[n]$$

$$y[n] = 1.8(-3)^n + 4.8(2)^n - 2$$

This is the same as before

Impulse Response

The impulse response $h[n]$ of a causal system is the output observed with input $x[n] = \delta[n]$.

For such a system, $x[n] = 0$ for $n > 0$, so the particular solution is zero, $y_p[n] = 0$. Thus the impulse response can be generated from the homogeneous solution by determining the coefficients A_m to satisfy the zero initial conditions (for a causal system).

Example

Determine the *impulse response* for

$$y[n] + y[n - 1] - 6y[n - 2] = x[n]$$

The impulse response is obtained from the homogenous solution:

$$h[n] = A_1(-3)^n + A_2(2)^n$$

For $n=0$

$$y[0] + y[-1] - 6y[-2] = x[0]$$

$$h[0] = \delta[0] = 1$$

$$A_1 + A_2 = 1$$

For $n=1$

$$y[1] + y[0] - 6y[-1] = x[1]$$

$$(-3A_1 + 2A_2) + (A_1 + A_2) = 0$$

$$h[1] + h[0] = \delta[1] = 0$$

$$A_1 = \frac{3}{5}, A_2 = \frac{2}{5}$$

$$h[n] = \frac{3}{5}(-3)^n + \frac{2}{5}(2)^n$$

$$n \geq 0$$

DSP Applications

- *Image Processing*
 - Pattern recognition
 - Robotic vision
 - Image enhancement
 - Facsimile
 - Satellite weather map
 - Animation
- *Instrumentation/Control*
 - Spectrum analysis
 - Position and rate control
 - Noise reduction
 - Data compression
- *Speech/audio*
 - Speech recognition/synthesis
 - Text to speech
 - Digital audio
 - equalization
- *Military*
 - Secure communication
 - Radar processing
 - Sonar processing
 - Missile guidance
- *Telecommunications*
 - Echo cancellation
 - Adaptive equalization
 - ADPCM transcoders
 - Spread spectrum
 - Video conferencing
 - Data communication
- *Biomedical*
 - Patient monitoring
 - Scanners
 - EEG brain mappers
 - ECG analysis
 - X-ray storage/enhancement

Some Applications of DSP

- Noise removal from image.

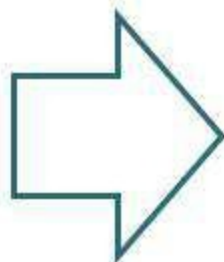
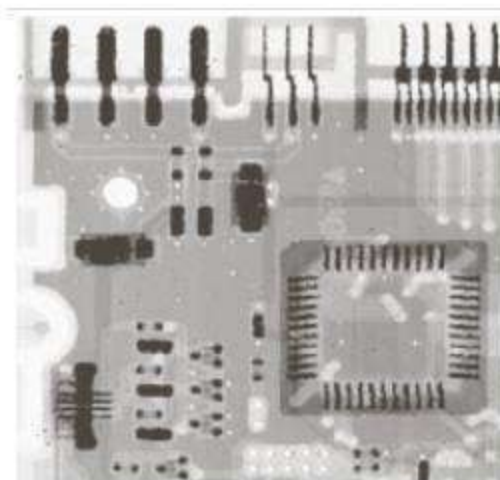
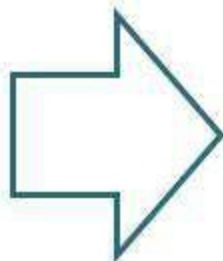
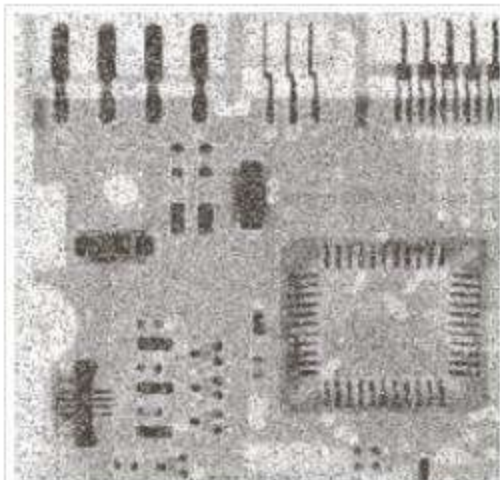
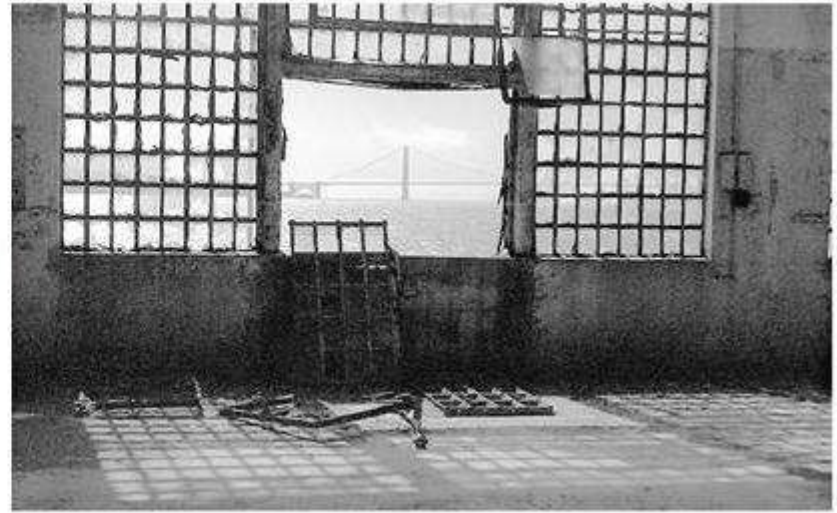


Image enhancement



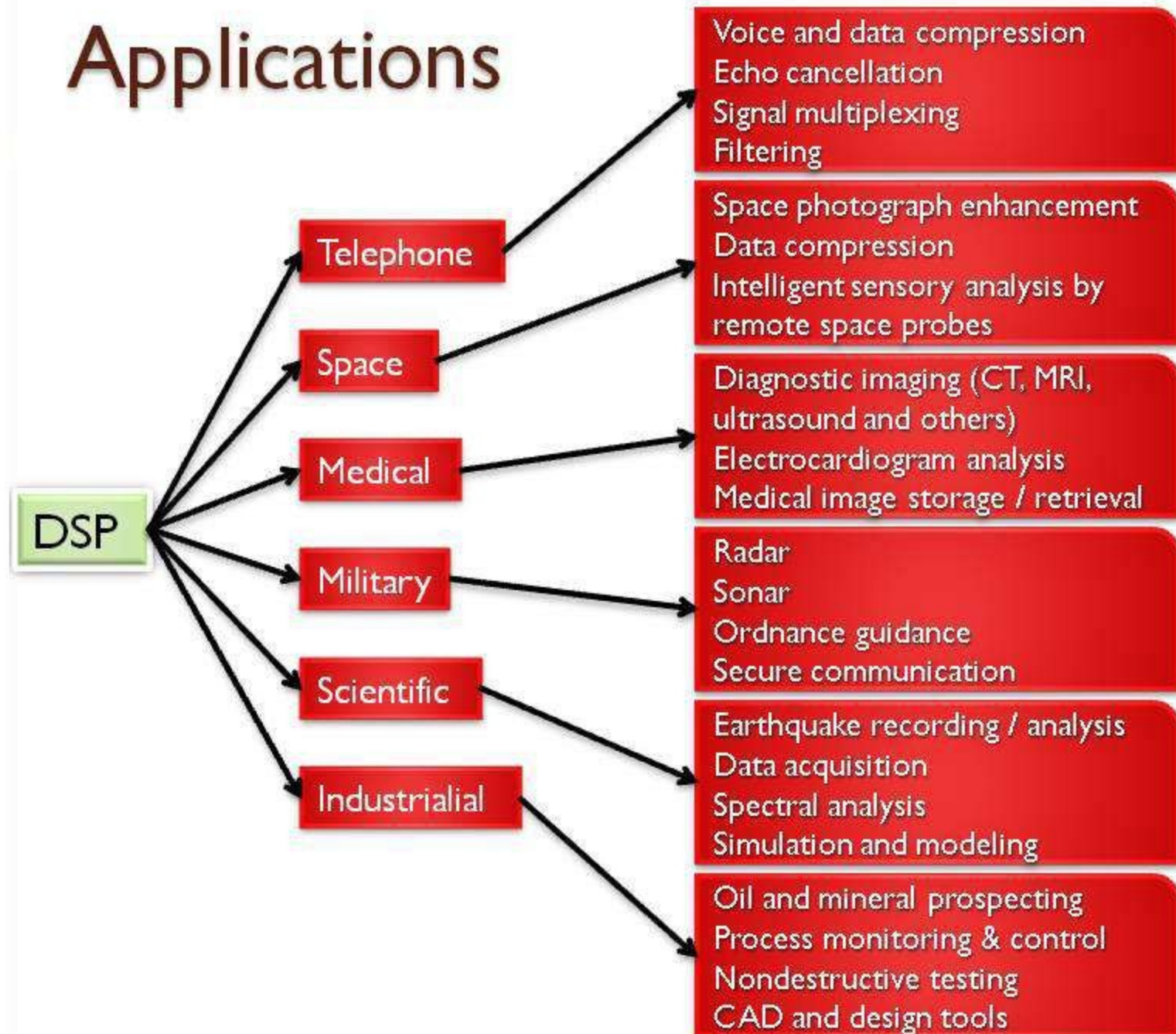
More Examples of Applications

- Sound Recording Applications
 - Compressors and Limiters
 - Expanders and Noise Gates
 - Equalizers and Filters
 - Noise Reduction Systems
 - Delay and Reverberation Systems
 - Special Effect Circuits
- Speech Processing
 - Speech Recognition
 - Speech Communication
- Telephone Dialing Applications
- FM Stereo Applications
- Electronic Music Synthesis
 - Subtractive Synthesis
 - Additive Synthesis
- Echo Cancellation in Telephone Networks
- Interference Cancellation in Wireless Telecommunication

Reasons for Using DSP

- Signals and data of many types are increasingly stored in digital computers, and transmitted in digital form from place to place. In many cases it makes sense to process them digitally as well.
- Digital processing is inherently stable and reliable. It also offers certain technical possibilities not available with analog methods.
- Rapid advances in IC design and manufacture are producing ever more powerful DSP devices at decreasing cost.
- Flexibility in reconfiguring
- Better control of accuracy requirement
- Easily transportable and possible offline processing
- Cheaper hardware in some case
- In many case DSP is used to process a number of signals simultaneously. This may be done by *interlacing* samples (technique known as TDM) obtained from the various signal channels.
- Limitation in speed & Requirement in larger bandwidth

Applications



System Analysis

- Three domains

- Time domain: impulse response, convolution sum

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

- Frequency domain: frequency response

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega})$$

- z-transform: system function

$$Y(z) = X(z)H(z)$$

- LTI system is completely characterized by ...

Frequency Response

- Relationship btw Fourier transforms of input and output

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega})$$

- In polar form

- Magnitude → magnitude response, gain, distortion

$$|Y(e^{j\omega})| = |X(e^{j\omega})| \cdot |H(e^{j\omega})|$$

- Phase → phase response, phase shift, distortion

$$\angle Y(e^{j\omega}) = \angle X(e^{j\omega}) + \angle H(e^{j\omega})$$

Ideal lowpass filter-example

■ Frequency response

$$H(e^{j\omega}) = \begin{cases} 1, & |\omega| < \omega_c, \\ 0, & \omega_c < |\omega| < \pi \end{cases}$$

□ Frequency selective filter

■ Impulse response

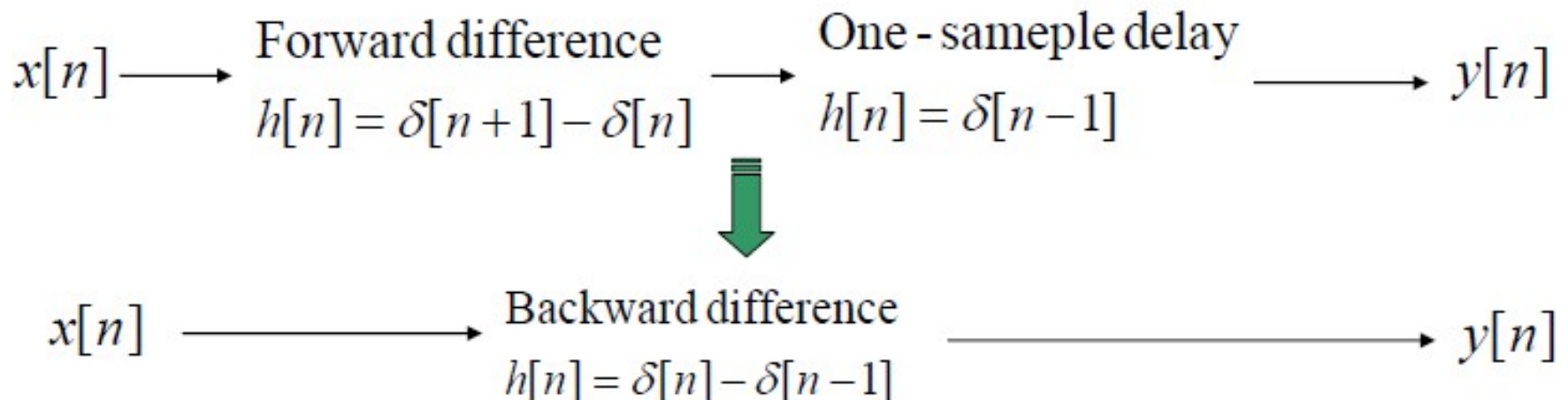
$$h_{lp}[n] = \frac{\sin \omega_c n}{\pi n}, \quad -\infty < n < \infty$$

- Noncausal, cannot be implemented! $h[n] \stackrel{?}{=} 0, \quad n < 0$
- How to make a noncausal system causal?

Non causal to causal

■ Cascading systems

- Ideal delay $h[n] = \delta[n - n_d]$



- In general, any noncausal FIR system can be made cause by cascading it with a sufficiently long delay!
- But ideal lowpass filter is an IIR system!

Phase distortion and delay

■ Ideal delay system

$$h_{id}[n] = \delta[n - n_d]$$

Delay distortion

$$H_{id}(e^{j\omega}) = e^{-j\omega n_d}$$

$$|H_{id}(e^{j\omega})| = 1$$

$$\angle H_{id}(e^{j\omega}) = -\omega n_d, |\omega| < \pi$$

Linear phase distortion

■ Ideal lowpass filter with linear phase

$$H_{lp}(e^{j\omega}) = \begin{cases} e^{-j\omega n_d}, & |\omega| < \omega_c, \\ 0, & \omega_c < |\omega| < \pi \end{cases}$$

Ideal lowpass filter is
always noncausal!

$$h_{lp}[n] = \frac{\sin \omega_c (n - n_d)}{\pi (n - n_d)}, \quad -\infty < n < \infty$$

Phase delay

- Recall that the phase response $\angle H(e^{j\omega})$ gives the phase shift experienced by each sinusoidal component of the input signal.
- In the case when $x[n] = A_x \cos(\omega n + \phi_x)$, we have

$$\begin{aligned} y[n] &= A_x |H(e^{j\omega})| \cos(\omega n + \phi_x + \angle H(e^{j\omega})) = \\ &= A_x |H(e^{j\omega})| \cos \left(\omega \left[n + \frac{\phi_x}{\omega} + \frac{\angle H(e^{j\omega})}{\omega} \right] \right) \end{aligned}$$

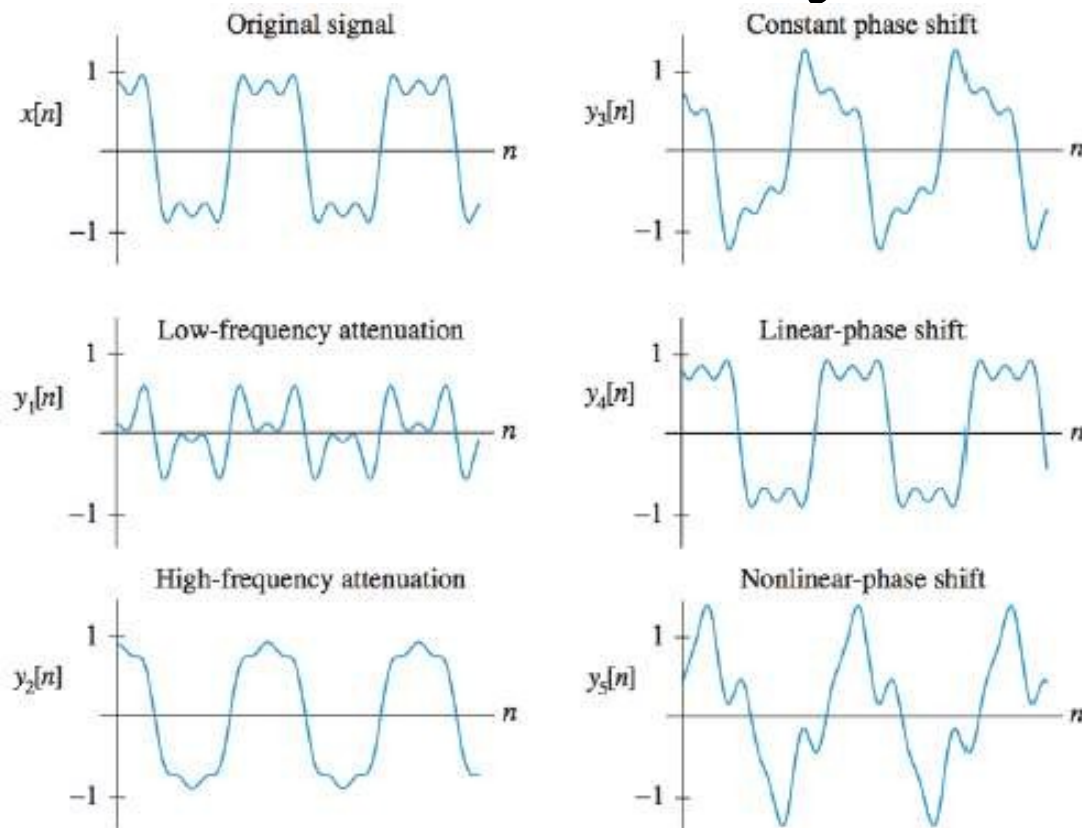
where the quantity $\frac{\angle H(e^{j\omega})}{\omega}$ shows the time shift (in number of sampling intervals) experienced by each sinusoidal component of the input signal.

- Therefore, sometimes it is more meaningful to use the *phase delay* defined by

$$\tau_{pd} \triangleq -\frac{\angle H(e^{j\omega})}{\omega}$$

- Nonlinear phase responses may lead to severe shape alterations.

Phase delay



Note that the constant phase shift in $y_3[n]$ causes distortion because each frequency component is delayed by a different amount. In contrast, the linear-phase shift in $y_4[n]$ does not cause any distortion because it results in a constant phase delay.

Group delay

- A measure of the linearity of the phase
- Concerning the phase distortion on a **narrowband** signal

$$x[n] = s[n] \cos(\omega_0 n)$$



- For this input with spectrum only around ω_0 , phase effect can be approximated around ω_0 as the **linear approximation** (though in reality maybe nonlinear)

$$\angle H(e^{j\omega}) \approx -\omega n_d - \phi_0$$

and the output is approximately

$$y[n] \approx |H(e^{j\omega_0})| s[\underline{n - n_d}] \cos(\omega_0 (\underline{n - n_d}) - \phi_0)$$

- Group delay $\tau = \text{grd}[H(e^{j\omega})] = -\frac{d}{d\omega} \{ \arg[\underline{H(e^{j\omega})}] \}$

Group delay

- A convenient way to check the linearity of phase response is to use the *group delay*, defined as the negative of the slope of the phase as follows

$$\tau_{gd} \triangleq -\frac{d\Psi(\omega)}{d\omega}$$

- The derivative in this definition requires the use of the unwrapped phase response $\Psi(\omega)$.
- Phase responses which are linear in frequency correspond to constant phase delay and constant group delay; both delays are identical, and each may be interpreted as time delay.
- Note that both the linear-phase response $\angle H(e^{j\omega}) = -\omega n_d$ and the *generalized linear phase* response $\angle H(e^{j\omega}) = \theta_0 - \omega n_d$ have a constant group delay.
- The name group delay comes because $\tau_{gd}(\omega_c)$ shows the delay of the “bundle” of frequency components about ω_c .

The Z-Transform

Z-Transform Definition

- Counterpart of the Laplace transform for discrete-time signals
- Generalization of the Fourier Transform
 - Fourier Transform does not exist for all signals

The z-transform of a sequence $x[n]$ is defined as

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

The inverse z-transform is given by

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

This expression denotes a closed contour integral taken counterclockwise about the origin and in the region of convergence (ROC). It incorporates the *Cauchy integral theorem* from the theory of complex variables. This result is not given directly in Oppenheim, but may be found in Proakis.

Relationship to Fourier Transform

The z-transform of a sequence $x[n]$ is defined as

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

The Fourier transform of a sequence $x[n]$ is defined as

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

For $z = e^{j\omega}$ the z-transform reduces to the Fourier transform

This domain is a circle of unit radius in the complex plane, that is $|z| = 1$.

Convergence of the z-Transform

- DTFT does not always converge

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

- Infinite sum not always finite if $x[n]$ not absolutely summable
- Example: $x[n] = a^n u[n]$ for $|a| > 1$ does not have a DTFT

- Complex variable z can be written as $r e^{j\omega}$ so the z-transform

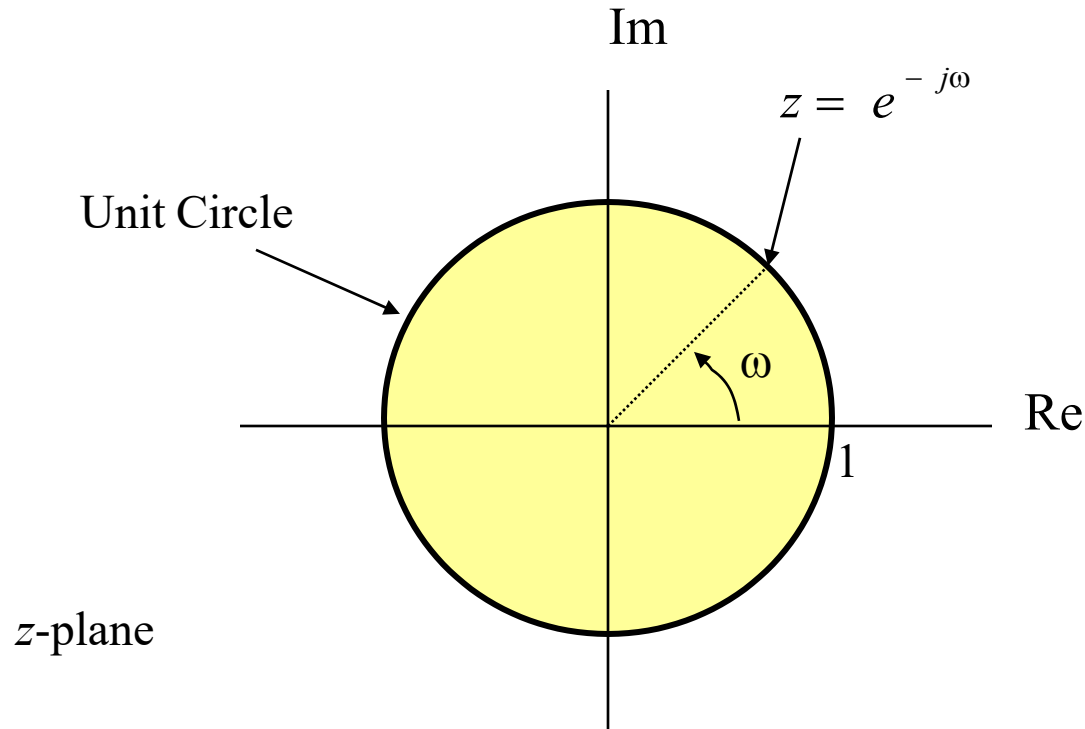
$$X(re^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n] (re^{j\omega})^{-n} = \sum_{n=-\infty}^{\infty} x[n] r^{-n} e^{-j\omega n}$$

- DTFT of $x[n]$ multiplied with exponential sequence r^{-n}
 - For certain choices of r the sum may be made finite

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

Unit Circle in Complex Z-Plane

- The z-transform is a function of the complex z variable
- Convenient to describe on the complex z-plane
- If we plot $z=e^{j\omega}$ for $\omega=0$ to 2π we get the unit circle



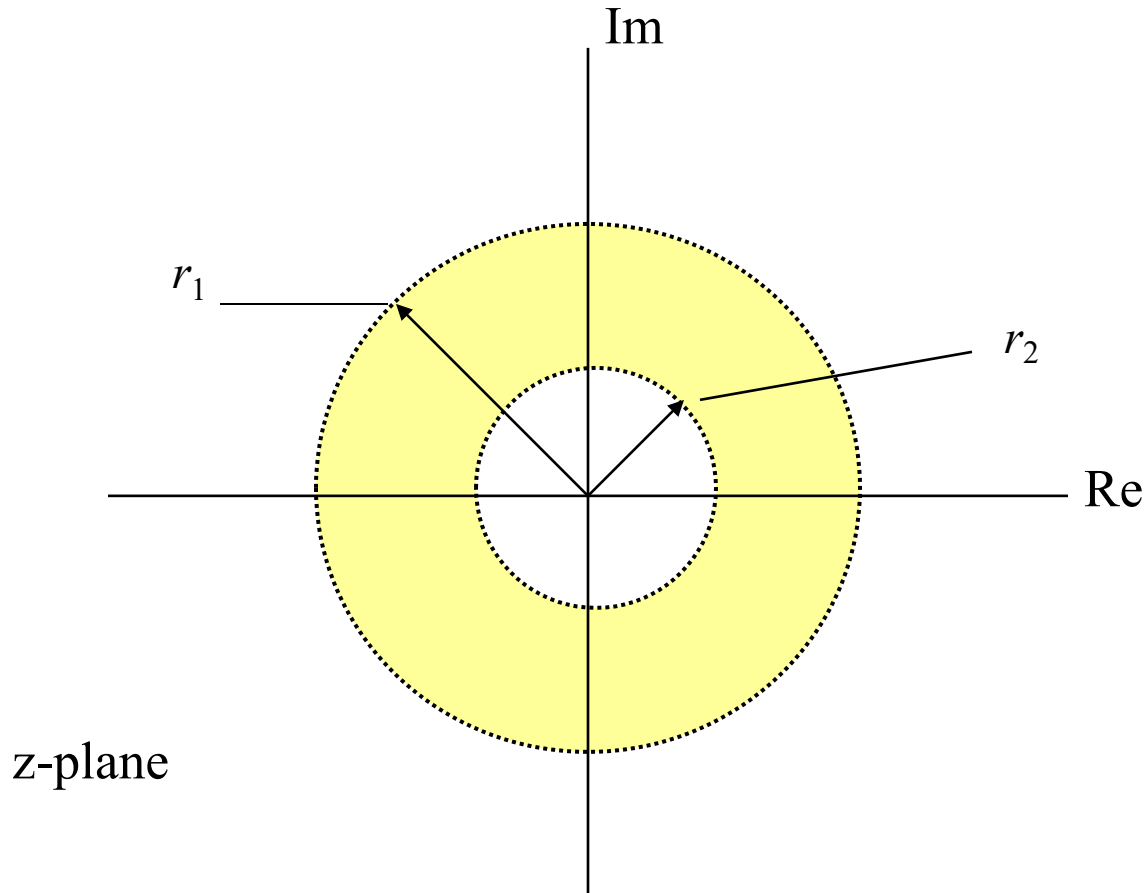
Region of Convergence (ROC)

For any given sequence, the set of value of z for which the z -transform converges is called the *region of convergence*, ROC. The criterion for convergence is that the z -transform be absolutely summable:

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

If some value of z , say, $z = z_1$, is in the ROC, then all values of z on the circle defined by $|z| = |z_1|$ will also be in the ROC. So, the ROC will consist of a ring in the z -plane centered about the origin. Its outer boundary will be a circle (*or the ROC may extend outward to infinity*), and its inner boundary will be a circle (*or it may extend inward to include the origin*).

Region of Convergence



The region of convergence (ROC) as a ring in the z-plane. For specific cases, the inner boundary can extend inward to the origin, and the ROC becomes a disk. In other cases, the outer boundary can extend outward to infinity.

Laurent Series

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

A power series of this form is a *Laurent series*. A Laurent series, and therefore a z-transform, represents an analytic function at every point within the region of convergence. This means that the z-transform and all its derivatives must be continuous functions of z within the region of convergence.

$$X(z) = \frac{P(z)}{Q(z)}$$

Among the most useful z-transforms are those for which $X(z)$ is a rational function inside the region of convergence, for example where $P(z)$ and $Q(z)$ are polynomials. The values of z for which $X(z)$ are zero are the *zeros* of $X(z)$ and the values for which $X(z)$ is infinite are the *poles* of $X(z)$.

Properties of the ROC

1. The ROC is a ring or disk in the z -plane centered at the origin

$$0 \leq r_R < |z| < r_L \leq \infty$$

2. The Fourier transform of $x[n]$ converges absolutely if and only if the ROC of the z -transform of $x[n]$ includes the unit circle.

3. The ROC cannot contain any poles.

4. If $x[n]$ is a *finite-duration sequence*, then the ROC is the entire z -plane except possibly $z=0$ or $z=\infty$.

5. If $x[n]$ is a *right-handed sequence* (zero for $n < N_1 < \infty$), the ROC extends outward from the outermost (largest magnitude) finite pole in $X(z)$ to (and possibly including infinity).

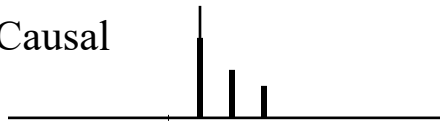
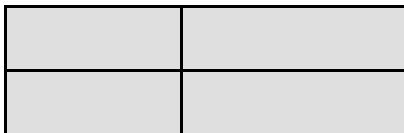
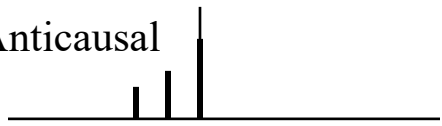
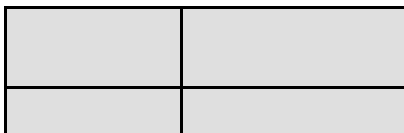
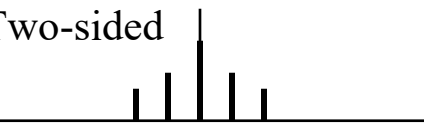
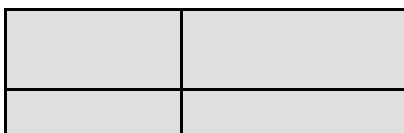
6. If $x[n]$ is a *left-handed sequence* (zero for $n > N_2 > -\infty$), the ROC extends inward from the innermost (smallest magnitude) nonzero pole in $X(z)$ to (and possibly including) zero.

7. A *two-sided sequence* is an infinite-duration sequence that is neither right-sided or left-sided. If $x[n]$ is a two-sided sequence, the ROC will consist of a ring in the z -plane, bounded on the interior and exterior by a pole and , consistent with property 3, not containing any poles.

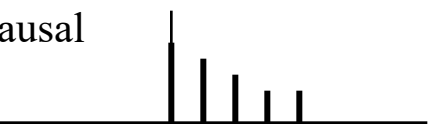
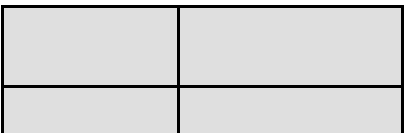
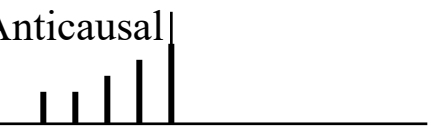
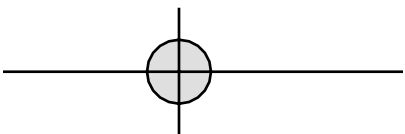
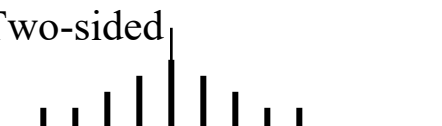
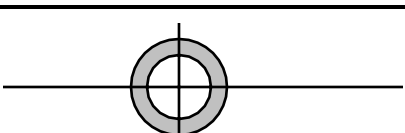
8. The ROC must be a connected region.

Properties of ROC Shown Graphically

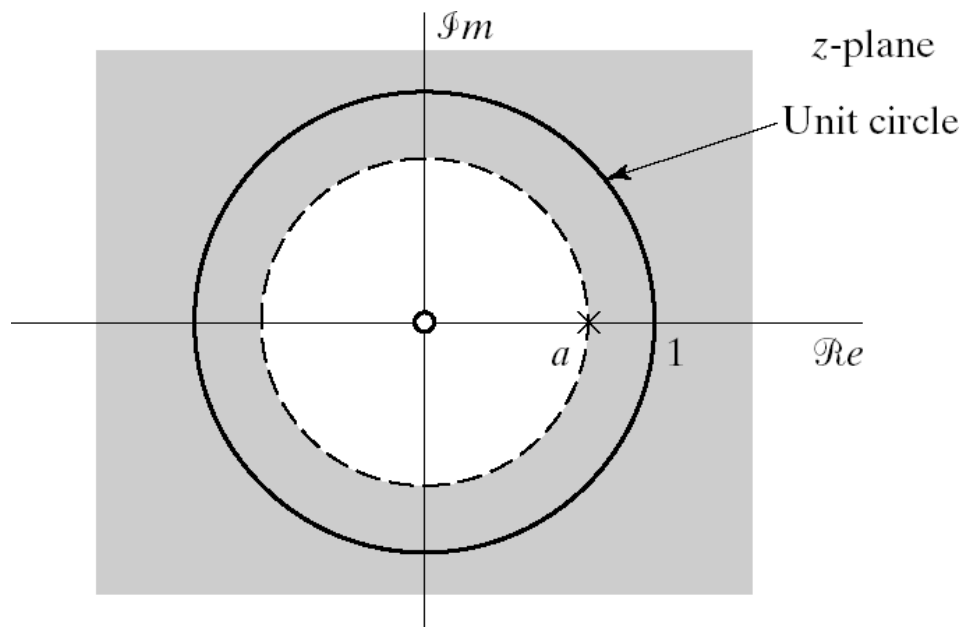
Finite-Duration Signals

<p>Causal</p> 	<p>Entire z-plane Except $z = 0$</p> 
<p>Anticausal</p> 	<p>Entire z-plane Except $z = \text{infinity}$</p> 
<p>Two-sided</p> 	<p>Entire z-plane Except $z = 0$ and $z = \infty$</p> 

Infinite-Duration Signals

<p>Causal</p> 	<p>$z > r_2$</p> 
<p>Anticausal</p> 	<p>$z < r_1$</p> 
<p>Two-sided</p> 	<p>$r_2 < z < r_1$</p> 

Example: Right-Sided Sequence



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

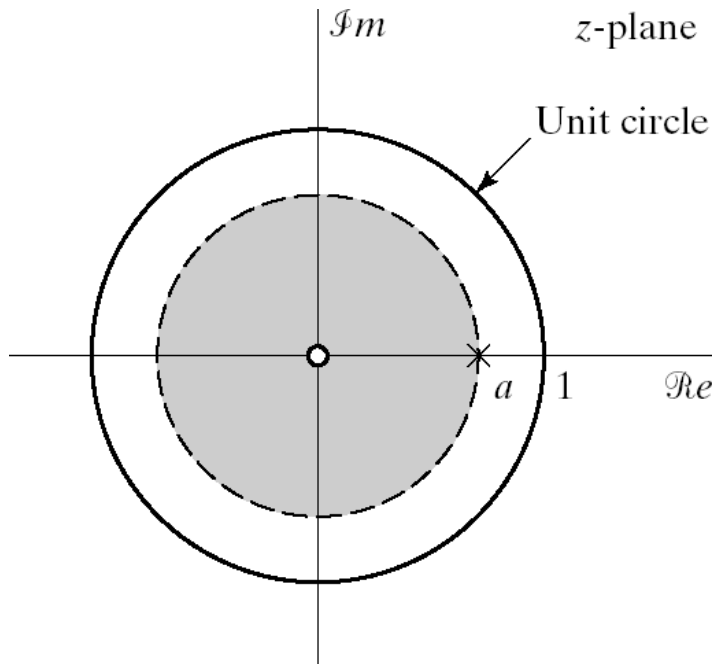
$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$$

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

$$\text{ROC} \quad |az^{-1}| < 1$$

$$\text{or} \quad |z| > |a|$$

Example: Left-Sided Sequence



$$x[n] = -a^n u[-n-1]$$

$$\text{nonzero for } n \leq -1$$

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

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

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

$$\text{ROC} \quad |a^{-1}z| < 1$$

$$\text{or} \quad |z| < |a|$$

Example: Sum of Two Exponential Sequences

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

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

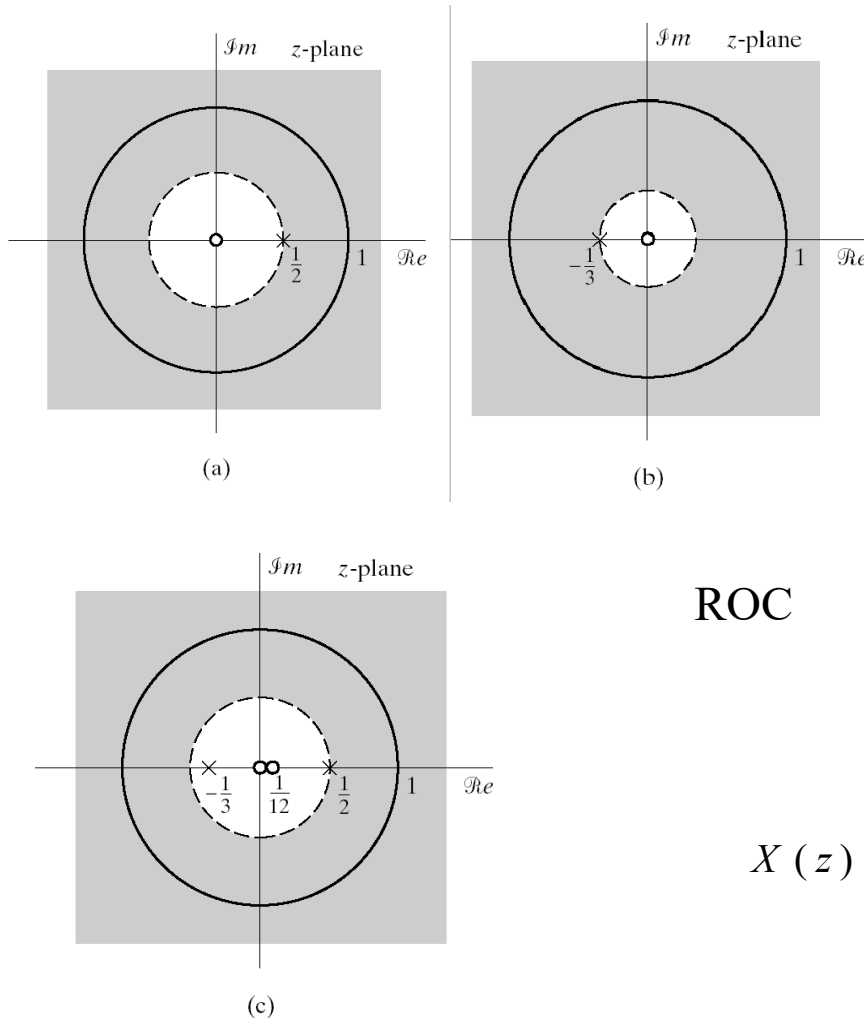
$$X(z) = \frac{1}{1 - \frac{1}{2} z^{-1}} + \frac{1}{1 + \frac{1}{3} z^{-1}}$$

$$\text{ROC} \quad \left|\frac{1}{2} z\right| < 1 \quad \text{and} \quad \left|-\frac{1}{3} z\right| < 1$$

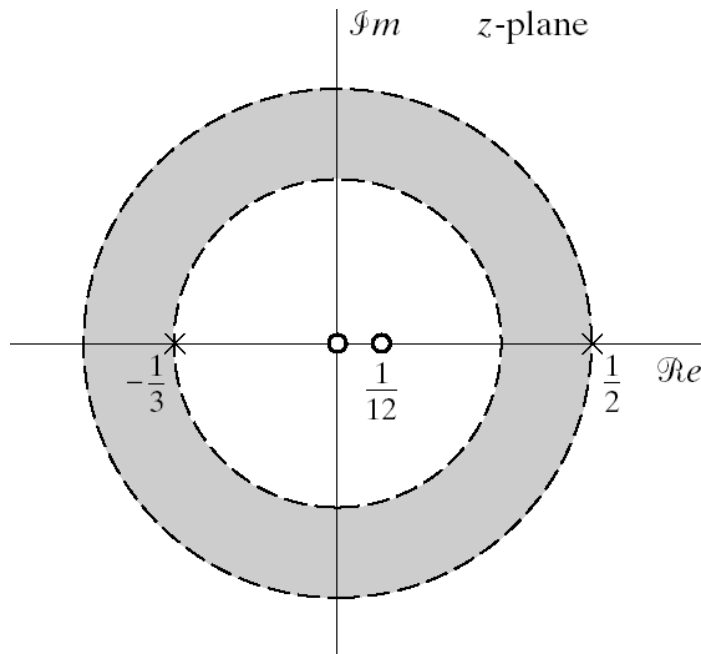
$$|z| > \frac{1}{2}$$

$$X(z) = \frac{2 + \frac{1}{3} z^{-1} - \frac{1}{2} z^{-1}}{\left(1 - \frac{1}{2} z^{-1}\right)\left(1 + \frac{1}{3} z^{-1}\right)} = \frac{2z\left(z - \frac{1}{12}\right)}{\left(z - \frac{1}{2}\right)\left(z + \frac{1}{3}\right)}$$

Poles at $\frac{1}{2}$ and $-\frac{1}{3}$, zeros at 0 and $\frac{1}{12}$



Example: Two-Sided Sequence



$$x[n] = \left(-\frac{1}{3} \right)^n u[n] - \left(\frac{1}{2} \right)^n u[-n-1]$$

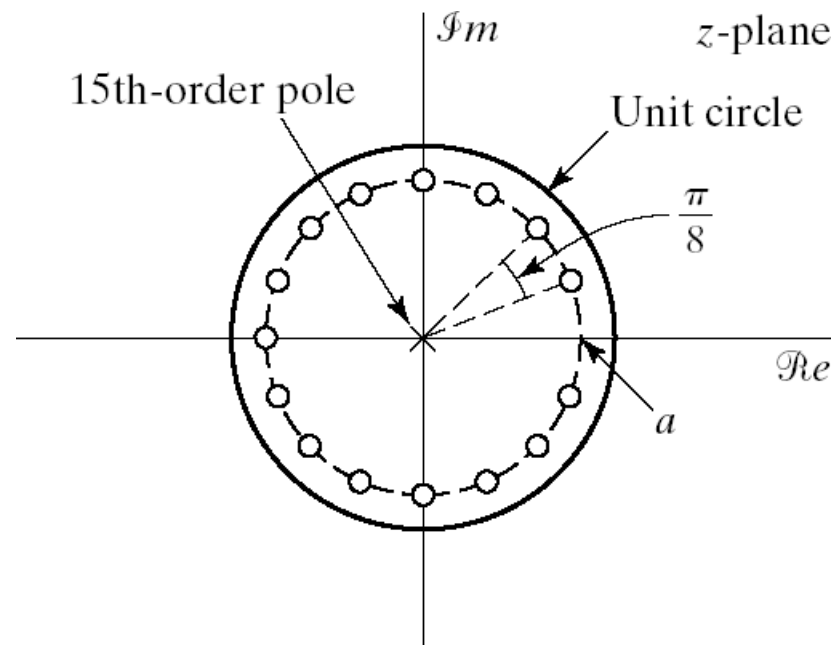
$$\left(-\frac{1}{3} \right)^n u[n] \Rightarrow \frac{1}{1 + \frac{1}{3} z^{-1}} \quad |z| > \frac{1}{3}$$

$$-\left(\frac{1}{2} \right)^n u[-n-1] \Rightarrow \frac{1}{1 - \frac{1}{2} z^{-1}} \quad |z| < \frac{1}{2}$$

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

$$\text{ROC} \quad \frac{1}{3} < |z| < \frac{1}{2}$$

Example: Finite Length Sequence



Pole-zero plot for $N = 16$

$$x[n] = \begin{cases} a^n & 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases}$$

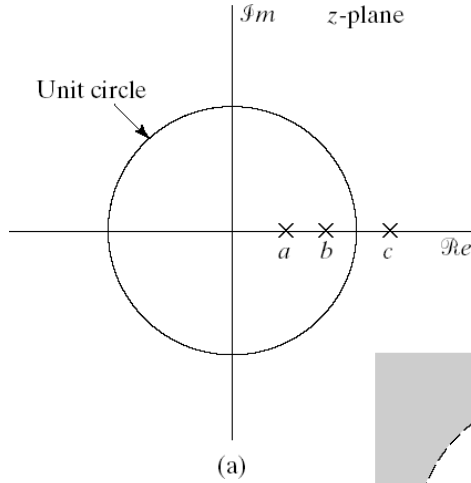
$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$$

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

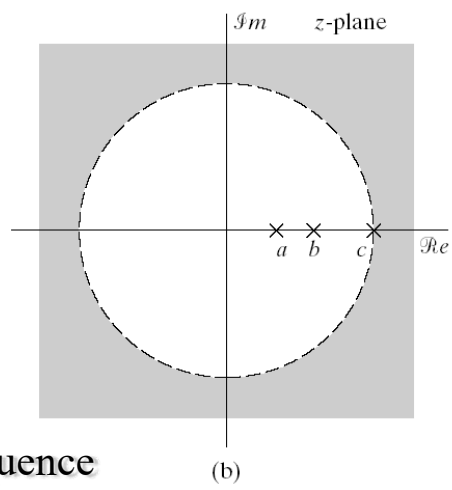
The sum is finite, so

$$\text{ROC} \quad |a| < \infty \quad \text{and} \quad z \neq 0$$

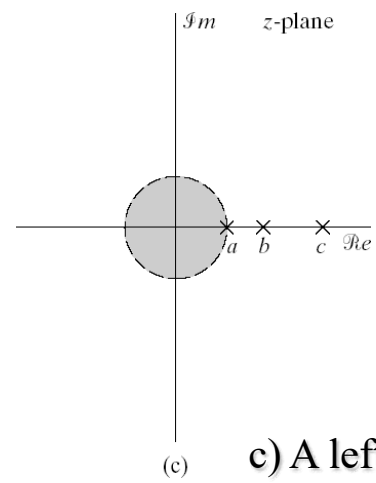
Z-transforms with the same pole-zero locations illustrating the different possibilities for the ROC. Each ROC corresponds to a different sequence.



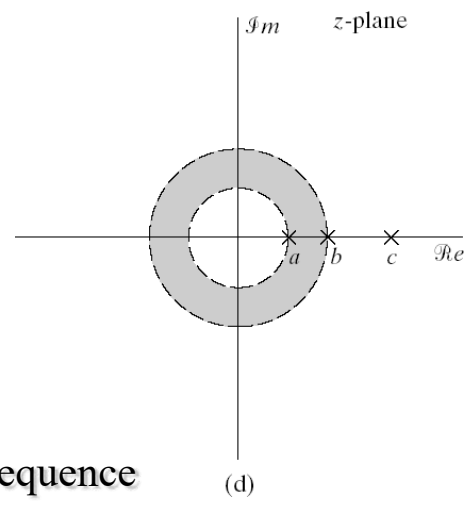
b) A right-sided sequence



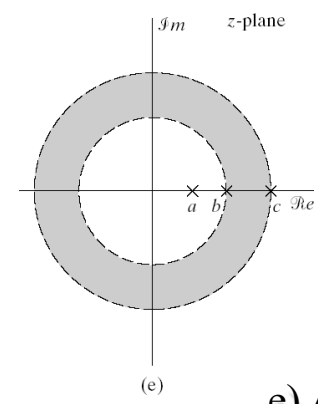
c) A left-sided sequence



d) A two-sided sequence



e) A two-sided sequence



Common Z-Transform Pairs

Sequence	Transform	ROC
$\delta[n]$	1	All z
$u[n]$	$\frac{1}{1 - z^{-1}}$	$ z > 1$
$-u[-n-1]$	$\frac{1}{1 - z^{-1}}$	$ z < 1$
$\delta[n-m]$	z^{-m}	*
$a^n u[n]$	$\frac{1}{1 - az^{-1}}$	$ z > a $
$-a^n u[-n-1]$	$\frac{1}{1 - az^{-1}}$	$ z < a $
$na^n u[n]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z > a $
$-na^n u[-n-1]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z < a $

Sequence	Transform	ROC
$[\cos \omega_0 n]u[n]$	$\frac{1 - [\cos \omega_0]z^{-1}}{1 - [2 \cos \omega_0]z^{-1} + z^{-2}}$	$ z > 1$
$[\sin \omega_0 n]u[n]$	$\frac{[\sin \omega_0]z^{-1}}{1 - [2 \cos \omega_0]z^{-1} + z^{-2}}$	$ z > 1$
$[\cos \omega_0 n]u[n]$	$\frac{1 - [r \cos \omega_0]z^{-1}}{1 - [2r \cos \omega_0]z^{-1} + r^2 z^{-2}}$	$ z > r$
$[\sin \omega_0 n]u[n]$	$\frac{[r \sin \omega_0]z^{-1}}{1 - [2r \cos \omega_0]z^{-1} + r^2 z^{-2}}$	$ z > r$
$a^n, 0 \leq n \leq N-1$ 0, otherwise	$\frac{1 - a^N z^{-N}}{1 - az^{-1}}$	$ z > 0$

*All z except 0 (if $m > 0$) or ∞ (if $m < 0$)

Z-Transform Properties (1/2)

Linearity

$$ax_1[n] + bx_2[n] \xleftrightarrow{Z} aX_1(z) + bX_2(z) \quad \text{ROC contains } R_{x_1} \cap R_{x_2}$$

Time Shifting

$$x[n - n_0] \xleftrightarrow{Z} z^{-n_0} X(z) \quad \text{ROC} = R_x \text{ (except for possible addition or deletion of } 0 \text{ or } \infty)$$

Multiplication by an Exponential Sequence

$$z_0^n x[n] \xleftrightarrow{Z} X\left(\frac{z}{z_0}\right) \quad \text{ROC} = |z_0| R_x$$

Differentiation of $X(z)$

$$nx[n] \xleftrightarrow{Z} -z \frac{dX(z)}{dz} \quad \text{ROC} = R_x$$

Z-Transform Properties (2/2)

Conjugation of a Complex Sequence

$$x^*[n] \xleftrightarrow{Z} X^*(z^*) \quad \text{ROC} = R_x$$

Time Reversal

$$x^*[-n] \xleftrightarrow{Z} X^*\left(\frac{1}{z^*}\right) \quad \text{ROC} = \frac{1}{R_x}$$

Convolution of Sequences

$$x_1[n] * x_2[n] \xleftrightarrow{Z} X_1(z) X_2(z) \quad \text{ROC contains } R_{x_1} \cap R_{x_2}$$

Initial-Value Theorem

$$x[0] = \lim_{z \rightarrow \infty} X(z) \quad \text{provided that } x[n] \text{ is zero for } n < 0, \text{ i.e. that } x[n] \text{ is causal.}$$

Inverse z-Transform

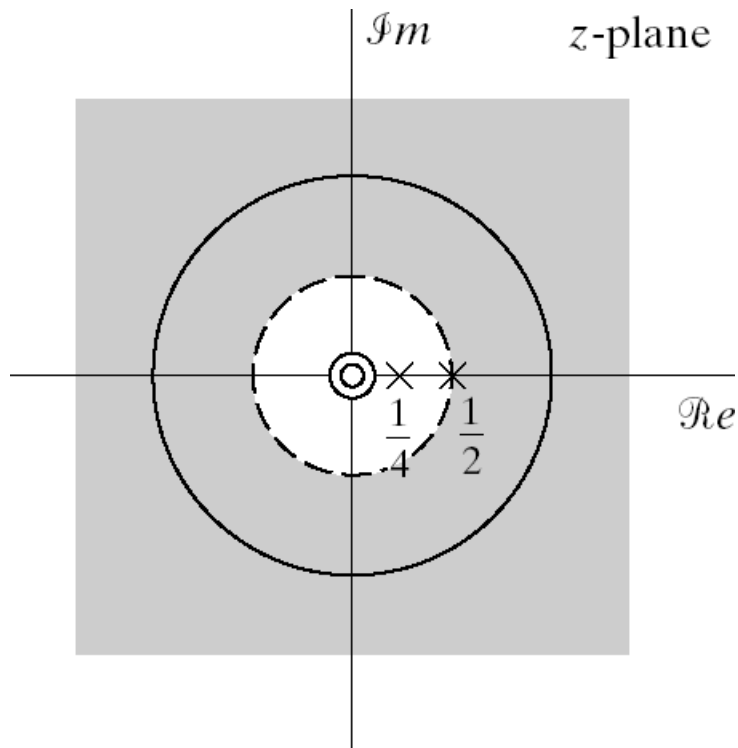
Method of inspection – recognize certain transform pairs.

Partial Fraction Expansion

$$X(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{\sum_{k=0}^N a_k z^{-k}} \quad \text{Factor to} \quad X(z) = \frac{b_0 \prod_{k=1}^M (1 - c_k z^{-1})}{a_0 \prod_{k=1}^N (1 - d_k z^{-1})}$$

$$X(z) = \sum_{k=0}^N \frac{A_k}{1 - d_k z^{-1}} \quad \text{where} \quad A_k = \left((1 - d_k z^{-1}) X(z) \right) \Big|_{z=d_k}$$

Example: Second-Order Z-Transform



$$X(z) = \frac{1}{\left(1 - \frac{1}{4}z^{-1}\right)\left(1 - \frac{1}{2}z^{-1}\right)} \quad |z| > \frac{1}{2}$$

$$X(z) = \frac{A_1}{1 - \frac{1}{4}z^{-1}} + \frac{A_2}{1 - \frac{1}{2}z^{-1}}$$

$$A_1 = \frac{1}{1 - \frac{1}{2}\left(\frac{1}{4}\right)^{-1}} = -1$$

$$A_2 = \frac{1}{1 - \frac{1}{4}\left(\frac{1}{2}\right)^{-1}} = 2$$

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

Partial Fraction Expansion

If $M > N$

$$X(z) = \sum_{r=0}^{M-N} B_r z^{-r} + \sum_{k=1}^N \frac{A_k}{1 - d_k z^{-1}}$$

B_r can be obtained by long division of numerator by denominator, stopping when the remainder is of lower degree than the denominator.

If $M > N$ and $X(z)$ has multiple-order poles, specifically a pole of order s at $z=d_i$

$$X(z) = \sum_{r=0}^{M-N} B_r z^{-r} + \sum_{k=1, k \neq i}^N \frac{A_k}{1 - d_k z^{-1}} + \sum_{m=1}^s \frac{C_m}{(1 - d_i z^{-1})^m}$$

$$C_m = \frac{1}{(s-m)! (-d_i)^{s-m}} \left\{ \frac{d^{s-m}}{dw^{s-m}} \left[\left(\frac{d}{1 - d_i w} \right)^s X\left(\frac{d}{w}\right) \right] \right\}_{w=d_i^{-1}}$$

Example

$$X(z) = \frac{(1 + z^{-1})^2}{(1 - \frac{1}{2}z^{-1})(1 - z^{-1})} = \frac{1 + 2z^{-1} + z^{-2}}{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}} \quad |z| > 1$$

$$X(z) = B_0 + \frac{A_1}{1 - \frac{1}{2}z^{-1}} + \frac{A_2}{1 - z^{-1}}$$

$$X(z) = B_0 + \frac{A_1}{1 - \frac{1}{2}z^{-1}} + \frac{A_2}{1 - z^{-1}}$$

$$X(z) = 2 + \frac{-9}{1 - \frac{1}{2}z^{-1}} + \frac{8}{1 - z^{-1}}$$

$$B_0 = \frac{z^{-2} + \square}{\frac{1}{2}z^{-2} + \square} = 2$$

$$A_1 = \frac{(1 + (\frac{1}{2})^{-1})^2}{1 - (\frac{1}{2})^{-1}} = \frac{(3)^2}{-1} = -9$$

$$A_2 = \frac{(1 + 1)^2}{1 - \frac{1}{2}(1)^{-1}} = 8$$

$$x[n] = 2\delta[n] - \left(\frac{1}{2}\right)^n u[n] + 8u[n]$$

Power Series Expansion

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

Note that

$$\delta[n - m] \xleftrightarrow{Z} z^{-m}$$

Example :

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

$$x[n] = 2\delta[n + 2] - \frac{1}{2}\delta[n + 1] - \delta[n] + \frac{1}{2}\delta[n - 1]$$

Example

$$X(z) = \log(1 + az^{-1}) \quad |z| > |a|$$

Expand in power series:

$$\log(1 + az^{-1}) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1} a^n z^{-n}}{n}$$

$$x[n] = \begin{cases} (-1)^{n+1} \frac{a^n}{n}, & n \geq 1 \\ 0, & n \leq 0 \end{cases}$$

Contour Integration

Cauchy integral theorem

$$\frac{1}{2\pi j} \oint_C z^{-k} dz = \begin{cases} 1, & k = 1 \\ 0, & k \neq 1 \end{cases}$$

C is a counterclockwise contour that encircles the origin.

Then one can show that

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

$$x[n] = \sum \left[\text{residues of } X(z)z^{n-1} \text{ at the poles inside } C \right]$$

Residue Calculation

If $X(z)$ is a rational function of z , we can write

$$X(z)z^{n-1} = \frac{\psi(z)}{(z - d_0)^s}$$

Then one can show that

$$\text{Res} \left[X(z)z^{n-1} \text{ at } z = d_0 \right] = \frac{1}{(s-1)!} \psi(d_0)$$

Quick Review of LTI Systems

- LTI Systems are uniquely determined by their impulse response

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

- We can write the input-output relation also in the z-domain

$$Y(z) = H(z)X(z)$$

- Or we can define an LTI system with its frequency response

$$Y(e^{j\omega}) = H(e^{j\omega})X(e^{j\omega})$$

- $H(e^{j\omega})$ defines magnitude and phase change at each frequency
- We can define a magnitude response

$$|Y(e^{j\omega})| = |H(e^{j\omega})| |X(e^{j\omega})|$$

- And a phase response

$$\angle Y(e^{j\omega}) = \angle H(e^{j\omega}) + \angle X(e^{j\omega})$$

Phase Distortion and Delay

- Remember the ideal delay system

$$h_i[n] = \delta[n - n_d] \xrightarrow{\text{DTFT}} H_i(e^{j\omega}) = e^{-j\omega n_d}$$

- In terms of magnitude and phase response

$$\left| H_{id}(e^{j\omega}) \right| = 1$$

$$\angle H_{id}(e^{j\omega}) = -\omega n_d \quad \left| \omega \right| < \pi$$

- Delay distortion is generally acceptable form of distortion
 - Translates into a simple delay in time
- Also called a linear phase response
 - Generally used as target phase response in system design
- Ideal lowpass or highpass filters have zero phase response
 - Not implementable in practice

System Functions for Difference Equations

- Ideal systems are conceptually useful but not implementable
- Constant-coefficient difference equations are
 - general to represent most useful systems
 - Implementable
 - LTI and causal with zero initial conditions

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

- The z-transform is useful in analyzing difference equations
- Let's take the z-transform of both sides

$$\sum_{k=0}^N a_k z^{-k} Y(z) = \sum_{k=0}^M b_k z^{-k} X(z)$$
$$\left(\sum_{k=0}^N a_k z^{-k} \right) Y(z) = \left(\sum_{k=0}^M b_k z^{-k} \right) X(z)$$

System Function

- Systems described as difference equations have system functions of the form

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^M b_k z^{-k}}{\sum_{k=0}^N a_k z^{-k}} = \left(\frac{b_0}{a_0} \right) \frac{\prod_{k=1}^M (1 - c_k z^{-1})}{\prod_{k=1}^N (1 - d_k z^{-1})}$$

- Example

$$H(z) = \frac{(1 + z^{-1})^2}{\left(1 - \frac{1}{2}z^{-1}\right)\left(1 + \frac{3}{4}z^{-1}\right)} = \frac{1 + 2z^{-1} + z^{-2}}{1 + \frac{1}{4}z^{-1} + \frac{3}{8}z^{-2}} = \frac{Y(z)}{X(z)}$$

$$\left(1 + \frac{1}{4}z^{-1} + \frac{3}{8}z^{-2}\right)Y(z) = (1 + 2z^{-1} + z^{-2})X(z)$$

$$y[n] + \frac{1}{4}y[n-1] + \frac{3}{8}y[n-2] = x[n] + 2x[n-1] + x[n-2]$$

Stability and Causality

- A system function does not uniquely specify a system
 - Need to know the ROC
- Properties of system gives clues about the ROC
- Causal systems must be right sided
 - ROC is outside the outermost pole
- Stable system requires absolute summable impulse response

$$\sum_{k=-\infty}^{\infty} |h[n]| < \infty$$

- Absolute summability implies existence of DTFT
 - DTFT exists if unit circle is in the ROC
 - Therefore, stability implies that the ROC includes the unit circle
- Causal AND stable systems have all poles inside unit circle
 - Causal hence the ROC is outside outermost pole
 - Stable hence unit circle included in ROC
 - This means outermost pole is inside unit circle
 - Hence all poles are inside unit circle

Example

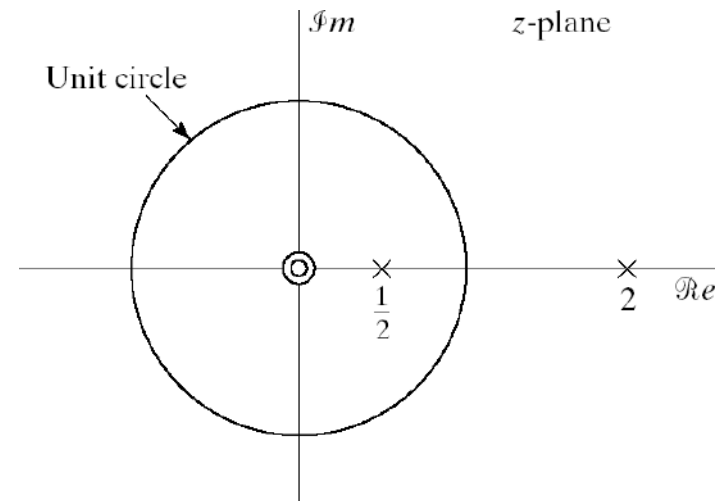
- Let's consider the following LTI system

$$y[n] - \frac{5}{2}y[n-1] + y[n-2] = x[n]$$

- System function can be written as

$$H(z) = \frac{1}{\left(1 - \frac{1}{2}z^{-1}\right)\left(1 - 2z^{-1}\right)}$$

- Three possibilities for ROC
 - If causal ROC_1 but not stable
 - If stable ROC_2 but not causal
 - If not causal neither stable ROC_3



$$ROC_1 : |z| > 2$$

$$ROC_2 : \frac{1}{2} < |z| < 2$$

$$ROC_3 : |z| < \frac{1}{2}$$

Structures for Discrete-Time Systems

- Block Diagram Representation of Linear Constant-Coefficient Difference Equations
- Signal Flow Graph Representation of Linear Constant-Coefficient Difference Equations
- Basic Structures for IIR Systems
- Transposed Forms
- Basic Network Structures for FIR Systems
- Lattice Structures

Introduction

- Example: The system function of a discrete-time system is

$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 - a z^{-1}}, \quad |z| > |a|$$

- Its impulse response will be

$$h[n] = b_0 a^n u[n] + b_1 a^{n-1} u[n-1]$$

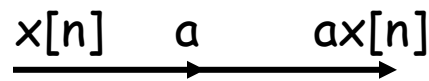
- Its difference equation will be

$$y[n] - a y[n-1] = b_0 x[n] + b_1 x[n-1]$$

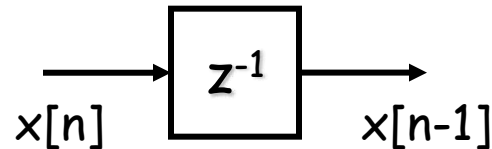
Since this system has an infinite-duration impulse response, it is not possible to implement the system by discrete convolution. However, it can be rewritten in a form that provides the basis for an algorithm for recursive computation.

$$y[n] = a y[n-1] + b_0 x[n] + b_1 x[n-1]$$

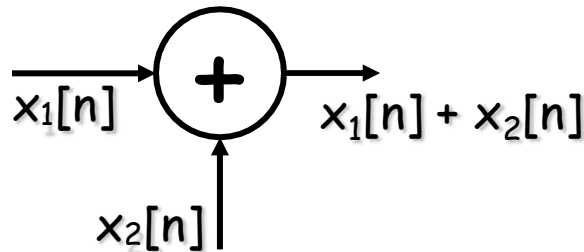
Block Diagram Representation of Linear Constant-coefficient Difference Equations



Multiplication of a sequence by a constant



Unit delay

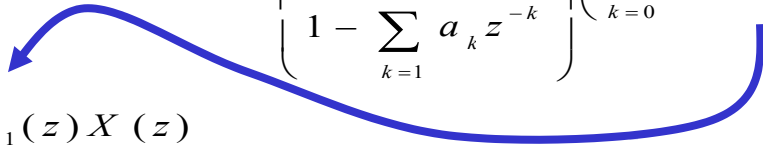


Addition of two sequences

Block Diagram Representation of Linear Constant-coefficient Difference Equations

$$H(z) = \frac{\sum_{k=0}^N b_k z^{-k}}{1 - \sum_{k=1}^N a_k z^{-k}} = \frac{Y(z)}{X(z)}$$

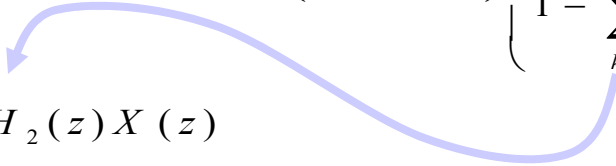
1

$$H(z) = H_2(z)H_1(z) = \left\{ \frac{1}{1 - \sum_{k=1}^N a_k z^{-k}} \right\} \left(\sum_{k=0}^M b_k z^{-k} \right)$$


$$V(z) = H_1(z)X(z)$$

$$Y(z) = H_2(z)V(z)$$

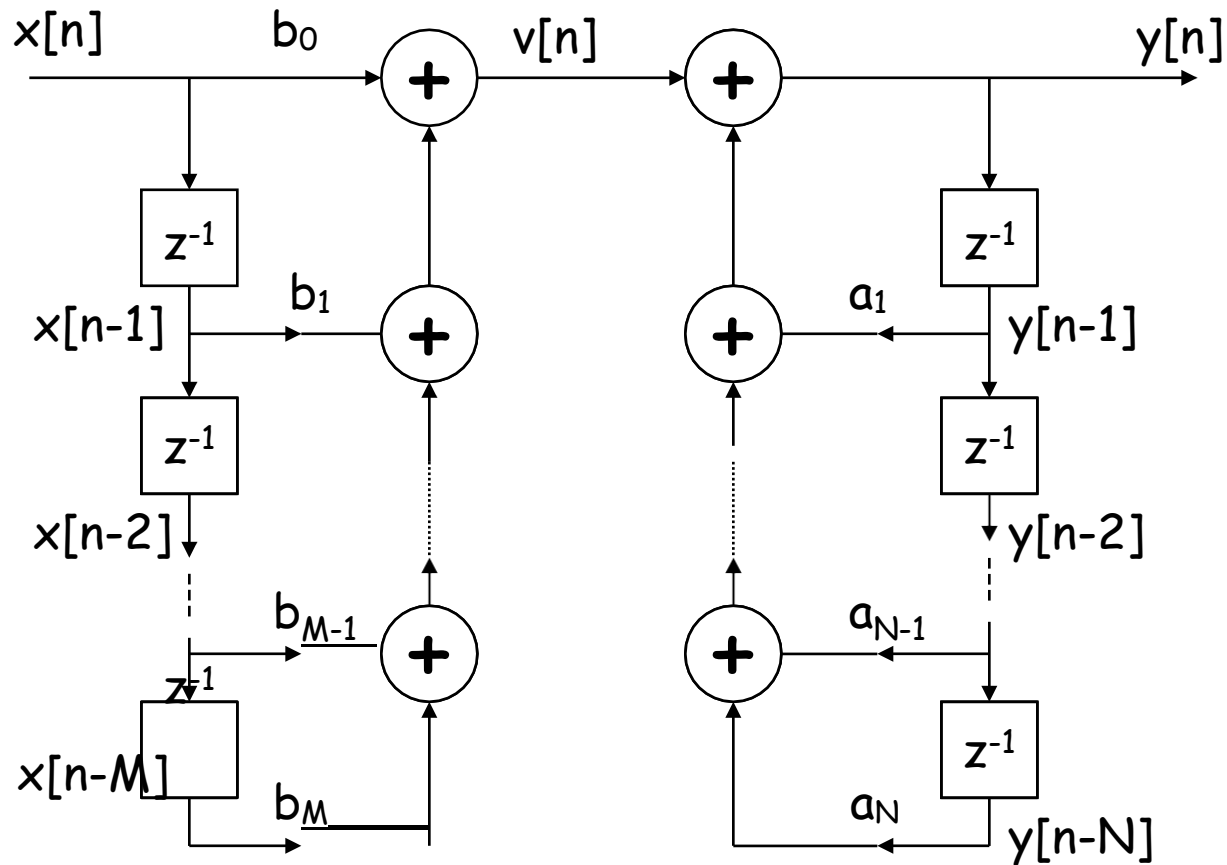
2

$$H(z) = H_1(z)H_2(z) = \left(\sum_{k=0}^M b_k z^{-k} \right) \left\{ \frac{1}{1 - \sum_{k=1}^N a_k z^{-k}} \right\}$$


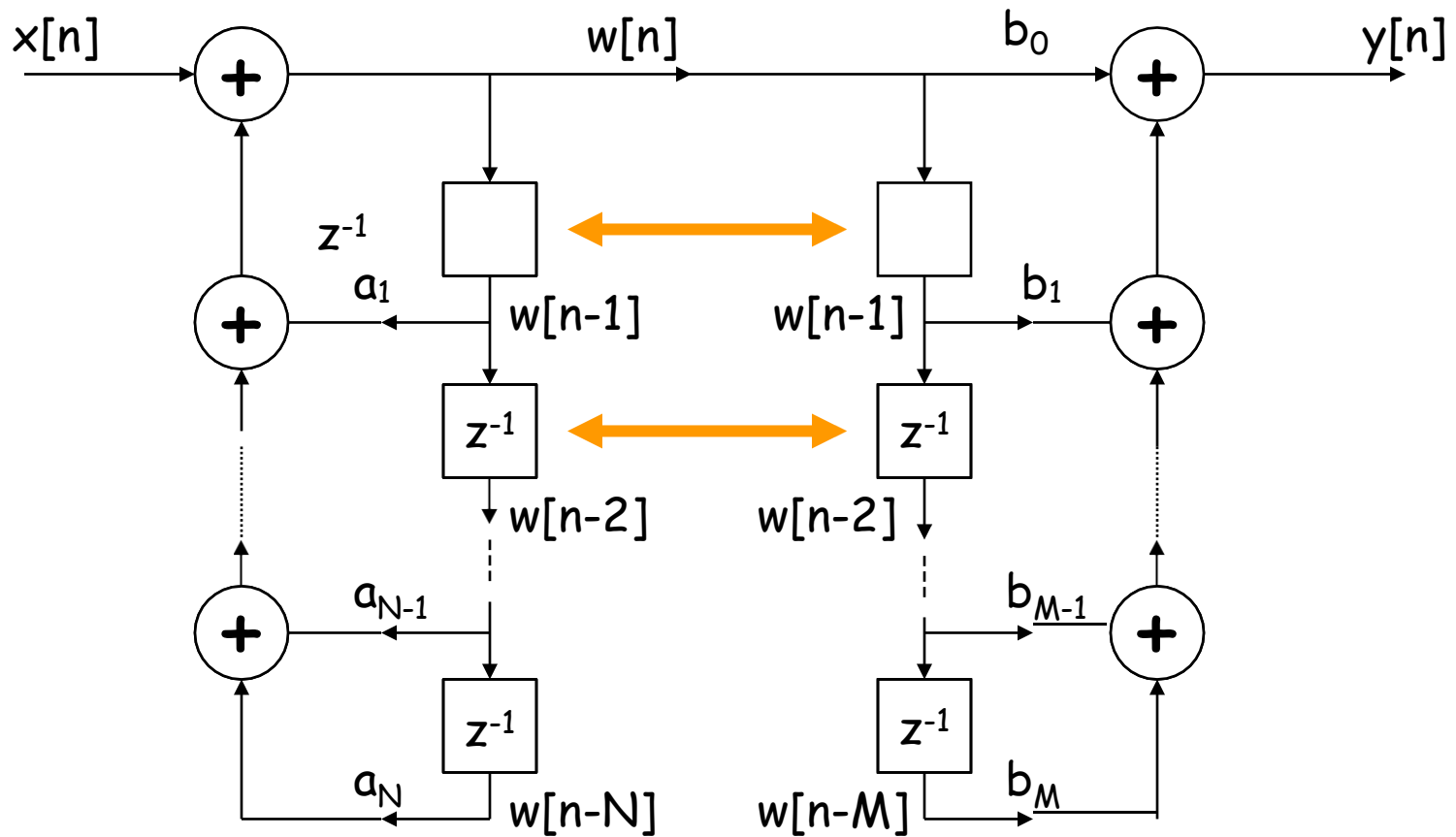
$$W(z) = H_2(z)X(z)$$

$$Y(z) = H_1(z)W(z)$$

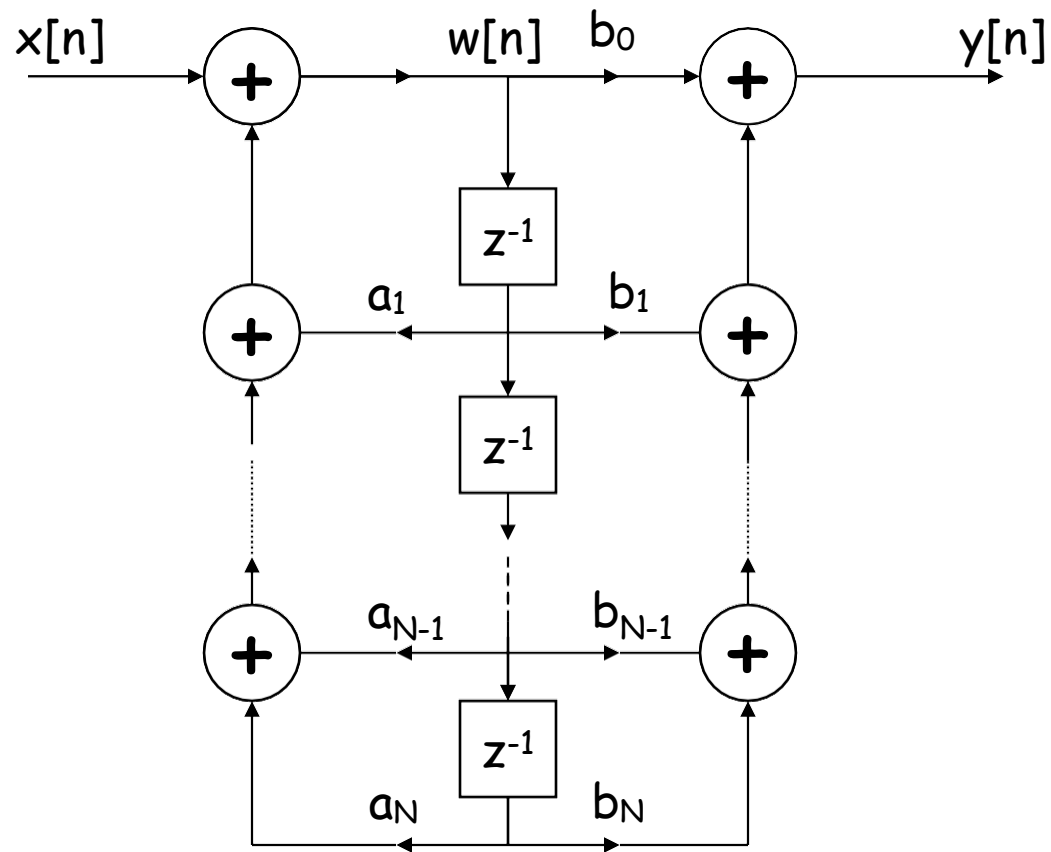
Block diagram representation for a general Nth-order difference equation: Direct Form 127



Block diagram representation for a general Nth-order difference equation: Direct Form 128



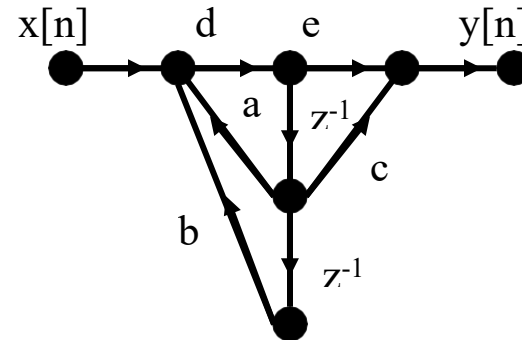
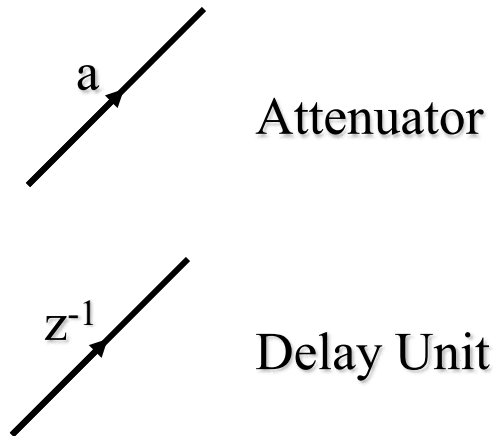
Combination of delay units (in case $N = M$)



Block Diagram Representation of Linear Constant-coefficient Difference Equations 2

- An implementation with the minimum number of delay elements is commonly referred to as a *canonic form* implementation.
- The *direct form I* is a direct realization of the difference equation satisfied by the input $x[n]$ and the output $y[n]$, which in turn can be written directly from the system function by inspection.
- The *direct form II* or canonic direct form is an rearrangement of the direct form I in order to combine the delay units together.

Signal Flow Graph Representation of Linear Constant-coefficient Difference Equations



- Node: Adder, Separator, Source, or Sink

Basic Structures for IIR Systems

- Direct Forms
- Cascade Form
- Parallel Form
- Feedback in IIR Systems

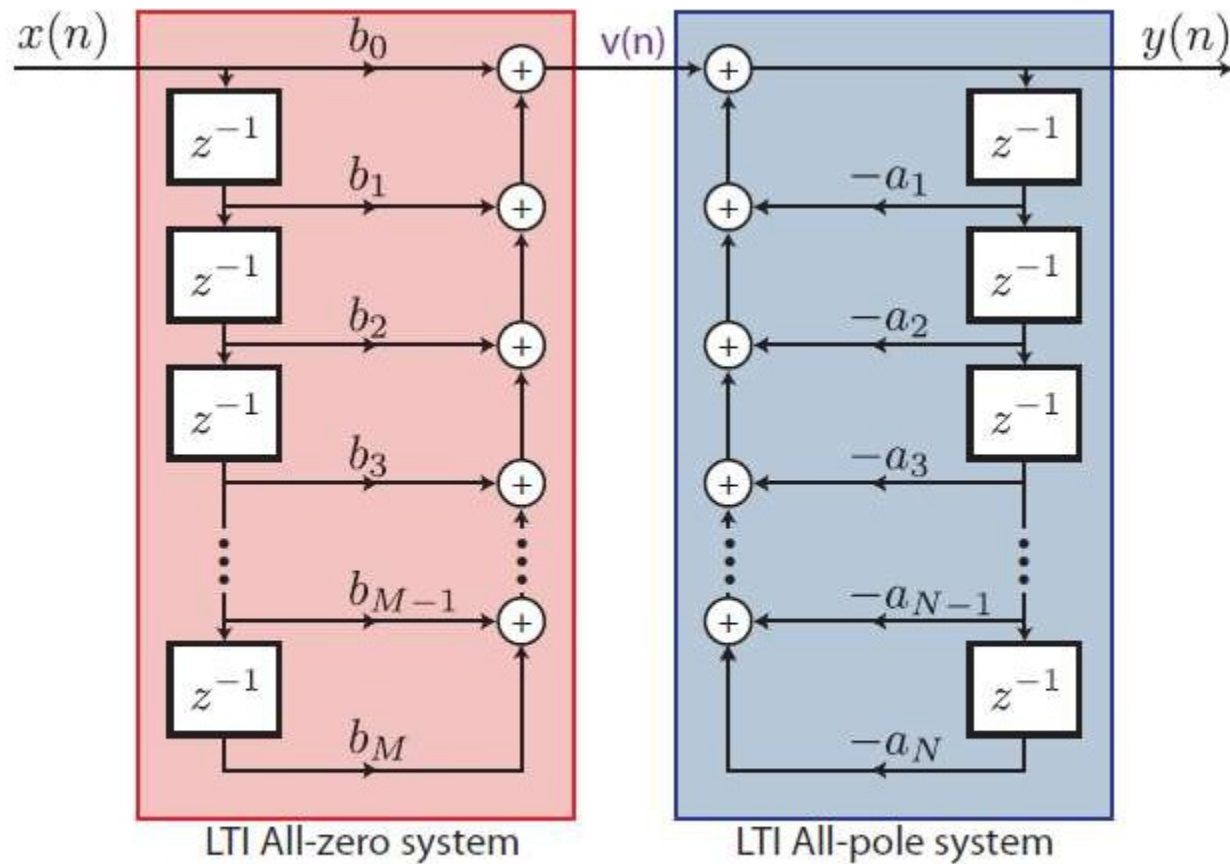
Basic Structures for IIR Systems

- **Direct Forms**

$$y[n] - \sum_{k=1}^N a_k y[n-k] = \sum_{k=0}^M b_k x[n-k]$$

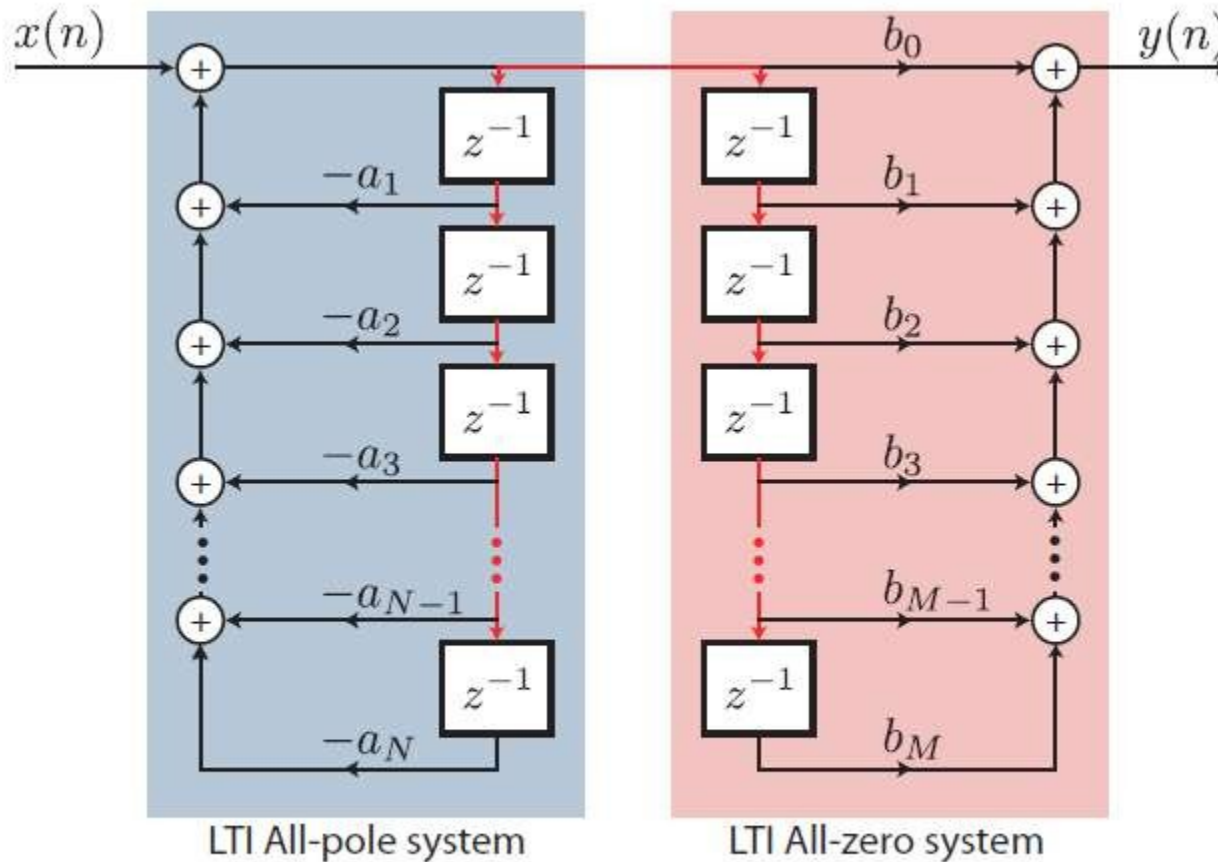
$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 - \sum_{k=1}^N a_k z^{-k}}$$

Direct Form I ($M = N$)



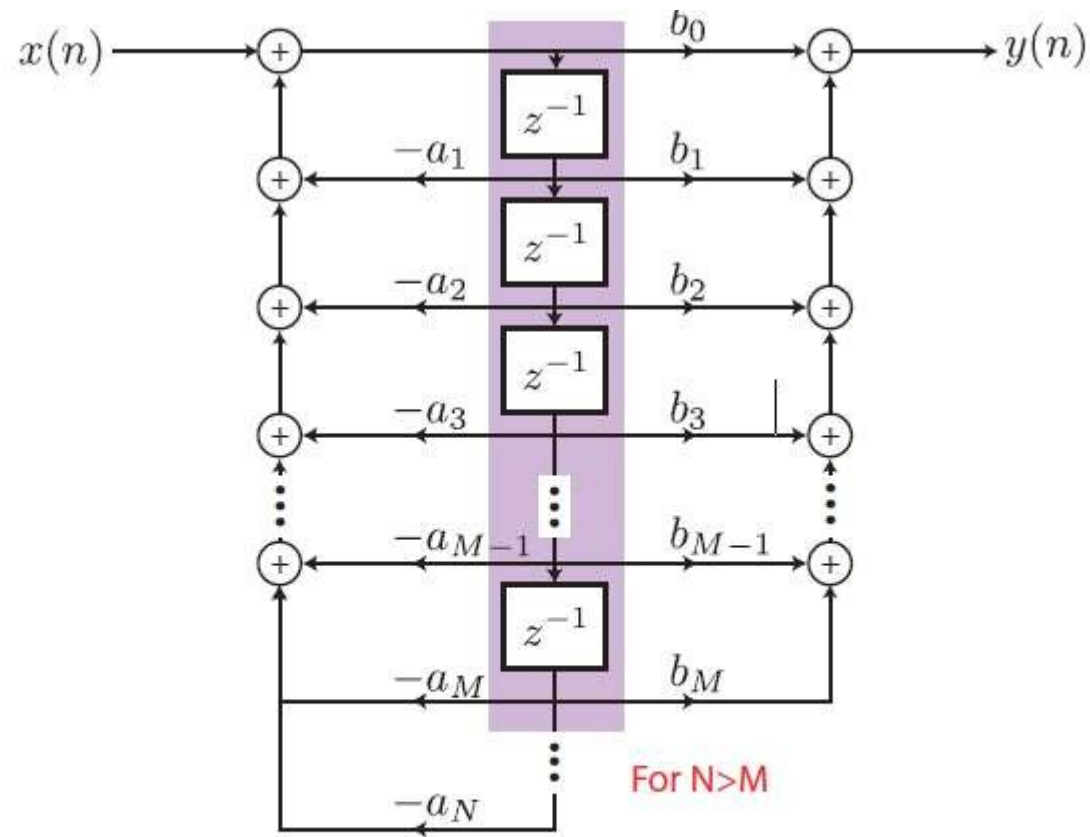
Requires: $M + N + 1$ multiplications, $M + N$ additions, $M + N$ memory locations

Direct Form II (M = N)

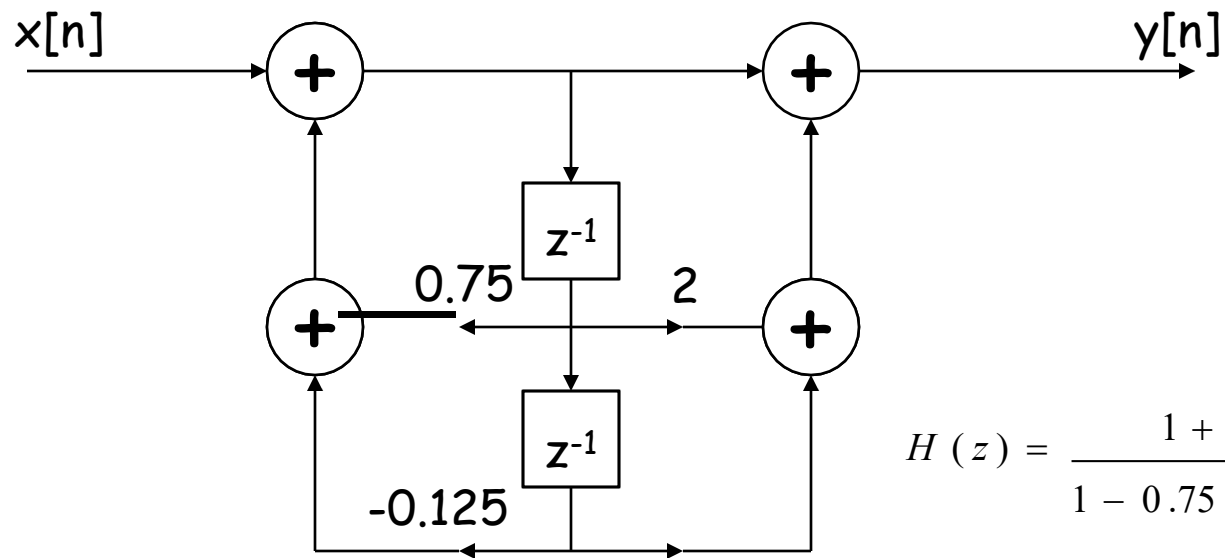
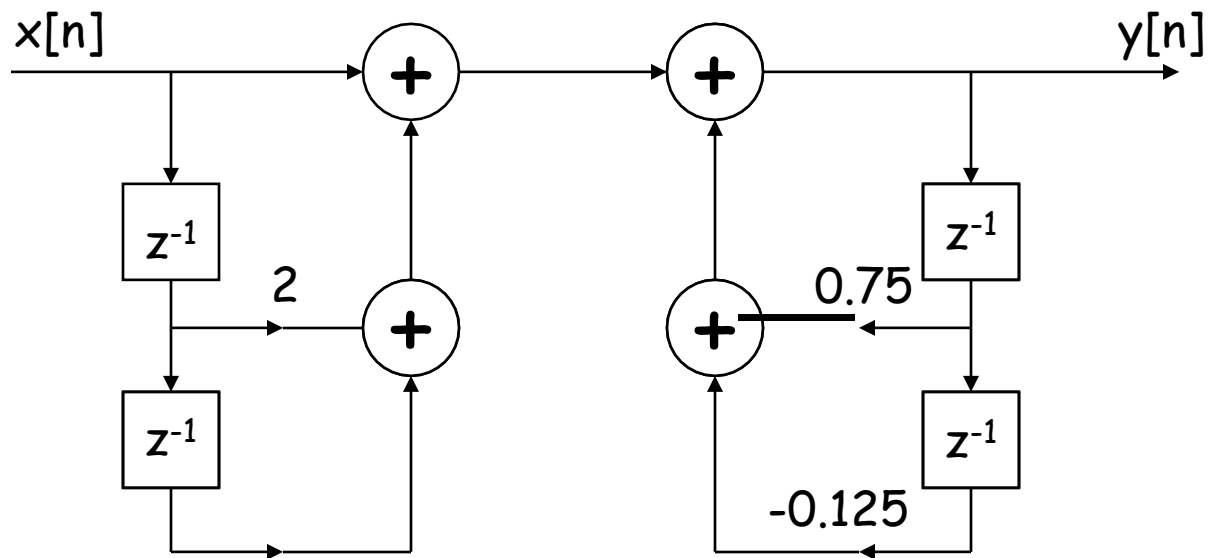


Requires: $M + N + 1$ multiplications, $M + N$ additions, $M + N$ memory locations

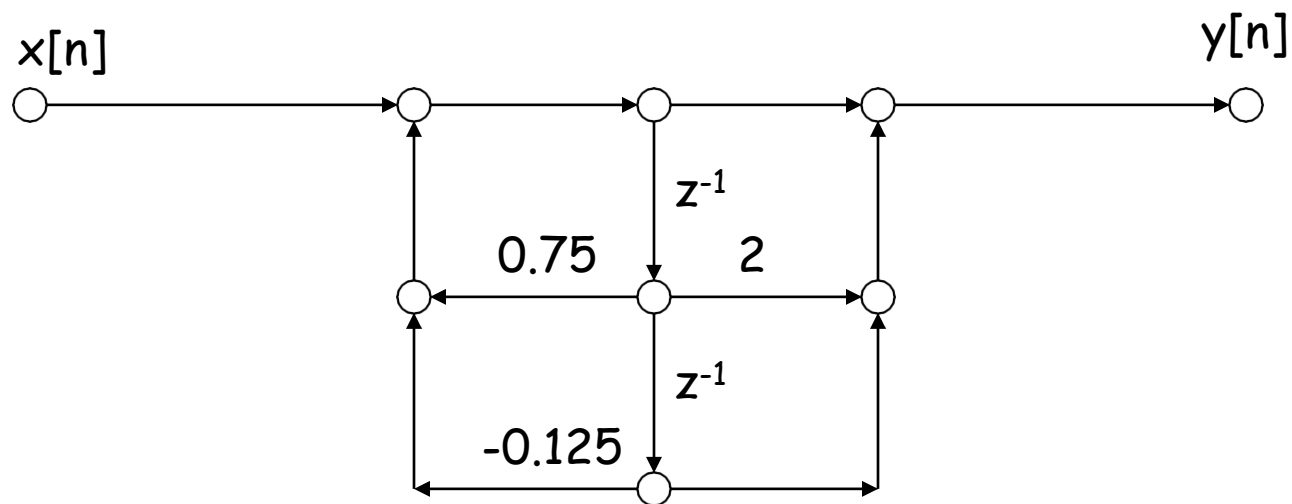
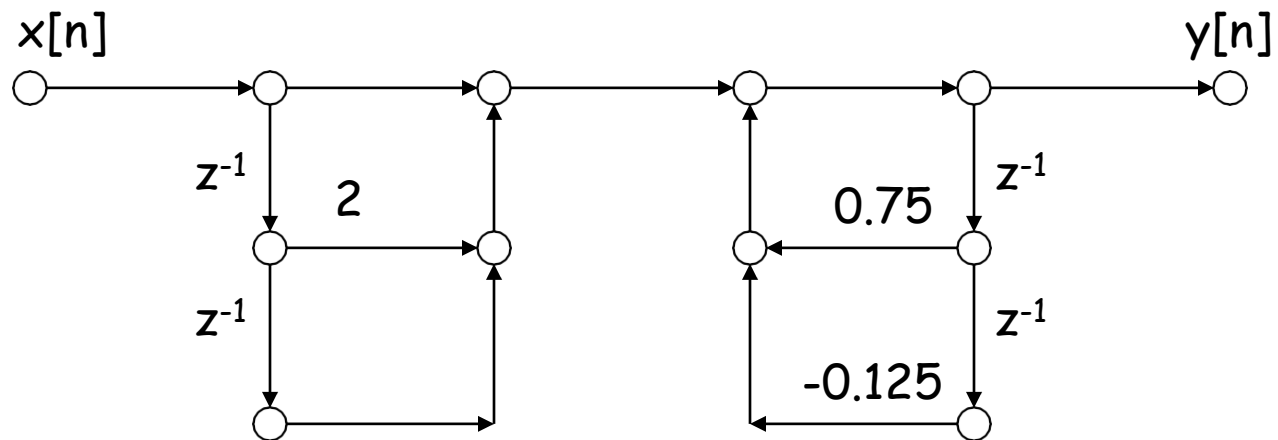
Direct Form II



Requires: $M + N + 1$ multiplications, $M + N$ additions, $\max(M, N)$ memory locations



$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$



Basic Structures for IIR Systems 2

- Cascade Form

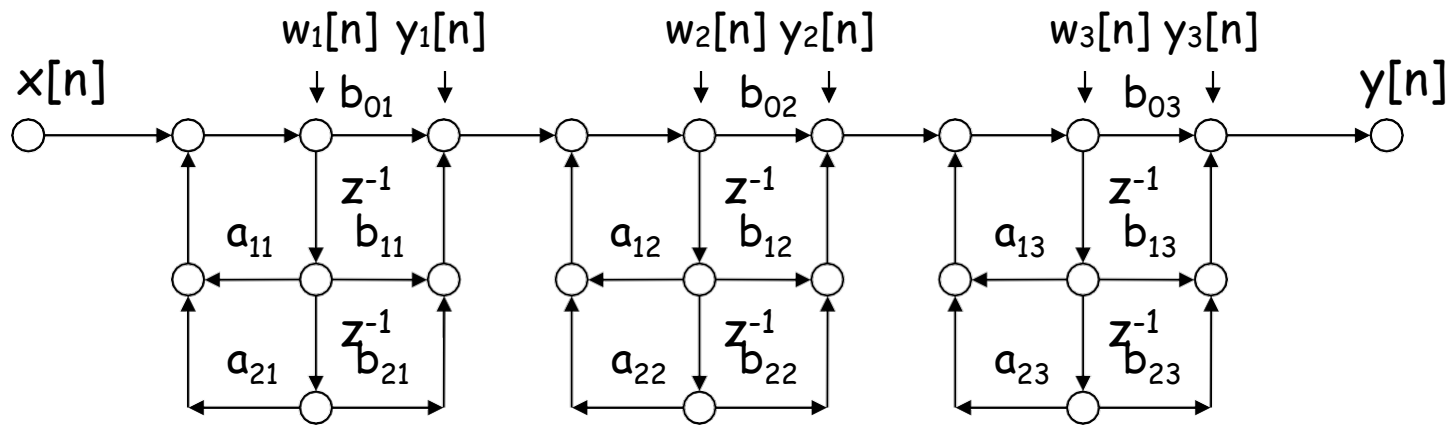
$$H(z) = A \frac{\prod_{k=1}^{M_1} (1 - g_k z^{-1}) \prod_{k=1}^{M_2} (1 - h_k z^{-1})(1 - h_k^* z^{-1})}{\prod_{k=1}^{N_1} (1 - c_k z^{-1}) \prod_{k=1}^{N_2} (1 - d_k z^{-1})(1 - d_k^* z^{-1})}$$

where $M = M_1 + 2M_2$ and $N = N_1 + 2N_2$.

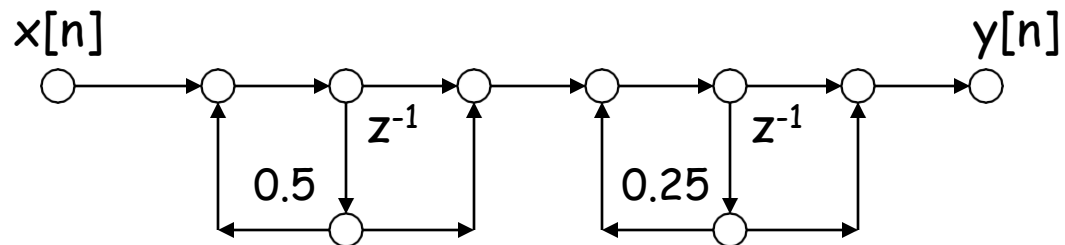
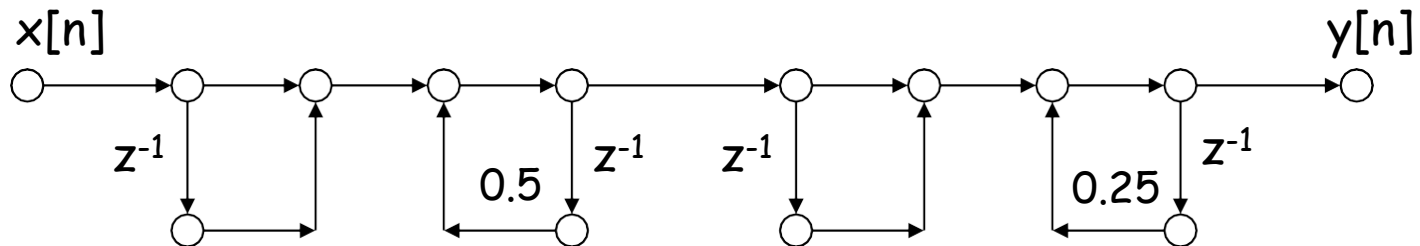
- A modular structure that is advantageous for many types of implementations is obtained by combining pairs of real factors and complex conjugate pairs into second-order factors.

$$H(z) = \prod_{k=1}^{N_s} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}}$$

where N_s is the largest integer contained in $(N+1)/2$.



$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = \frac{(1 + z^{-1})(1 + z^{-1})}{(1 - 0.5z^{-1})(1 - 0.25z^{-1})}$$



Basic Structures for IIR Systems 3

- Parallel Form

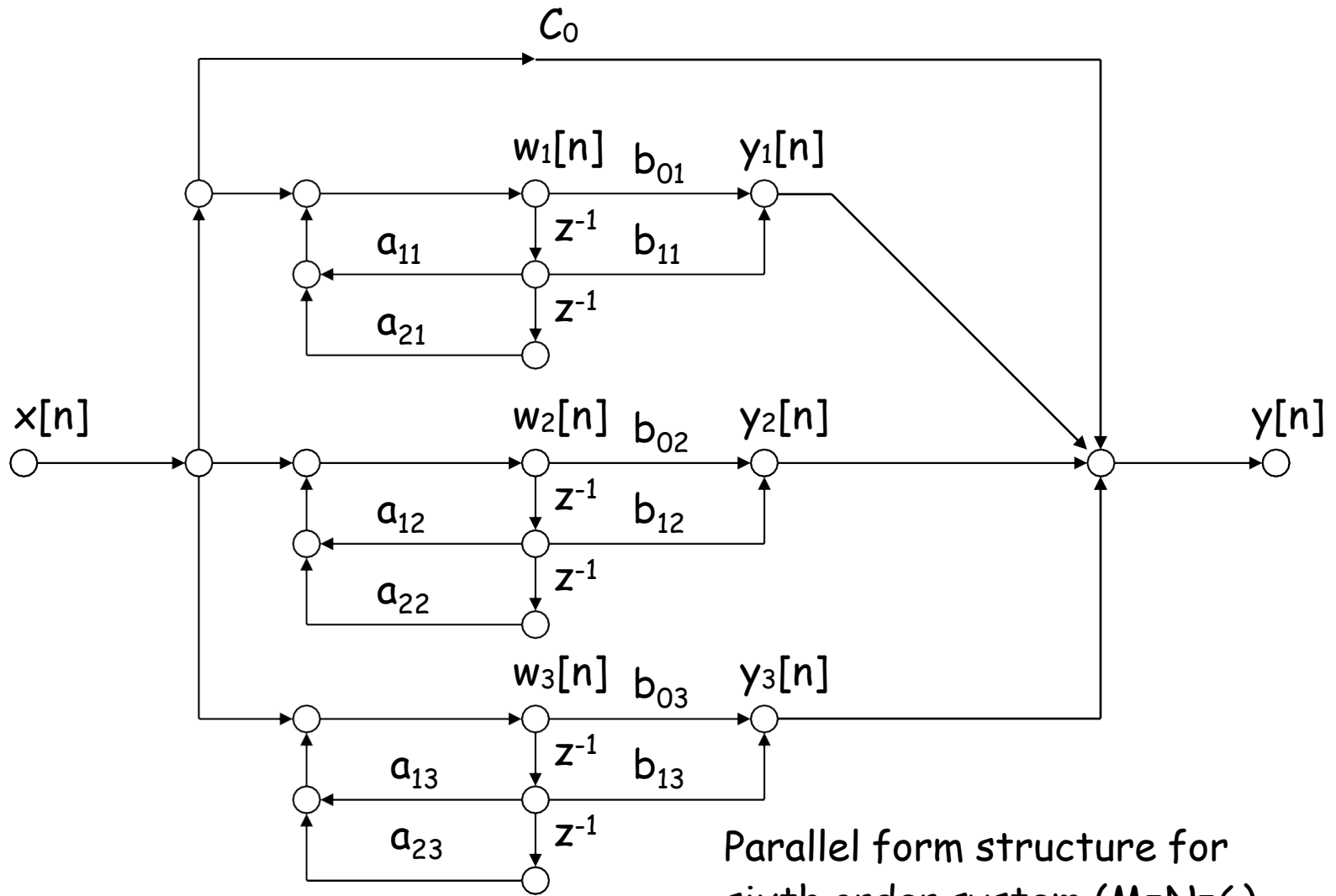
$$H(z) = \sum_{k=0}^{N_P} C_k z^{-k} + \sum_{k=1}^{N_1} \frac{A_k}{1 - c_k z^{-1}} + \sum_{k=1}^{N_2} \frac{B_k (1 - e_k z^{-1})}{(1 - d_k z^{-1})(1 - d_k^* z^{-1})}$$

where $N = N_1 + 2N_2$. If $M \leq N$, then $N_P = M - N$; otherwise, the first summation in right hand side of equation above is not included.

- Alternatively, the real poles of $H(z)$ can be grouped in pairs :

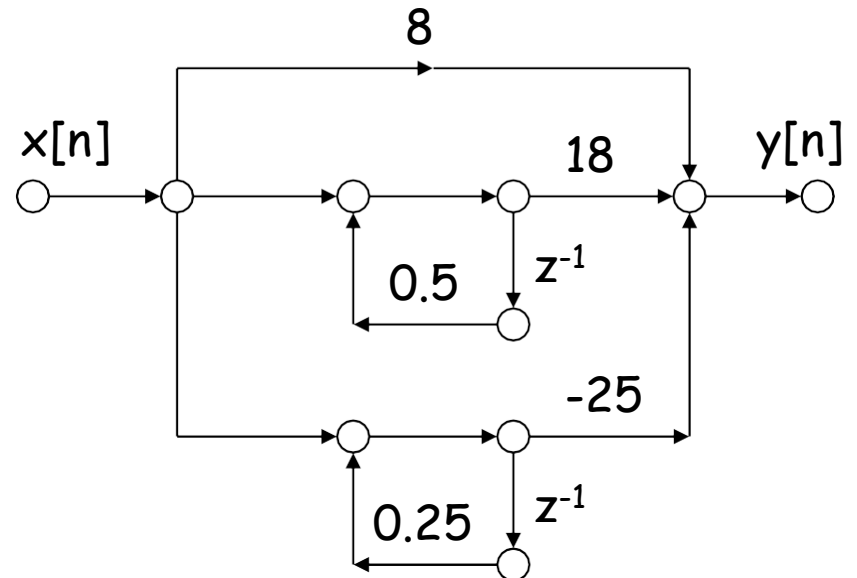
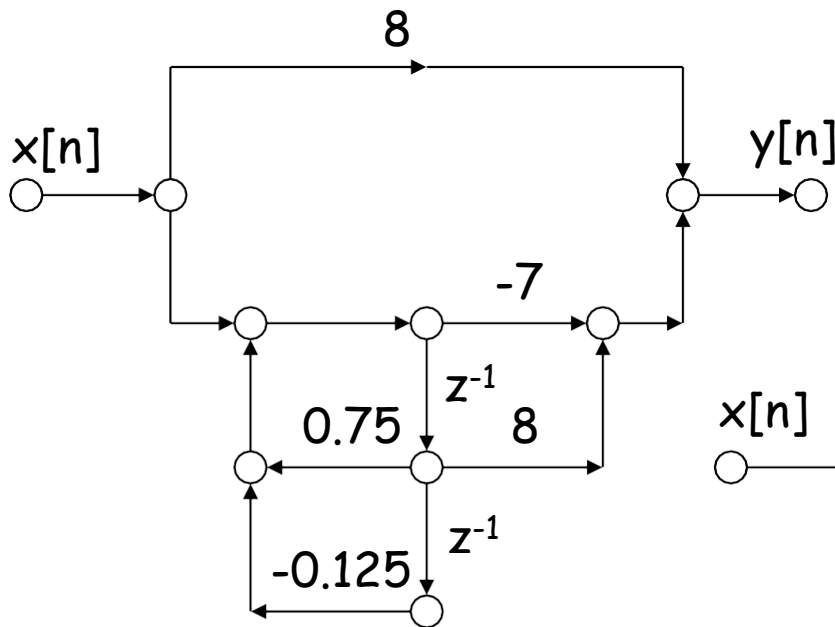
$$H(z) = \sum_{k=0}^{N_P} C_k z^{-k} + \sum_{k=1}^{N_S} \frac{e_{0k} + e_{1k} z^{-1}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}}$$

where N_S is the largest integer contained in $(N+1)/2$, and if $N_P = M - N$ is negative, the first sum is not present.



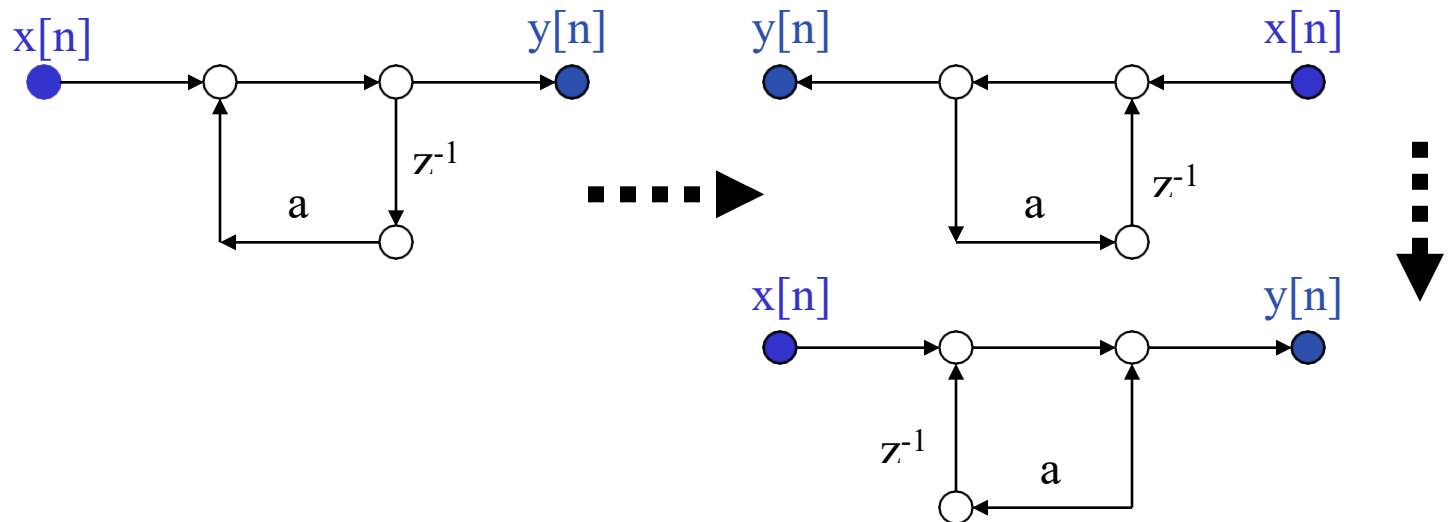
Parallel form structure for sixth order system ($M=N=6$).

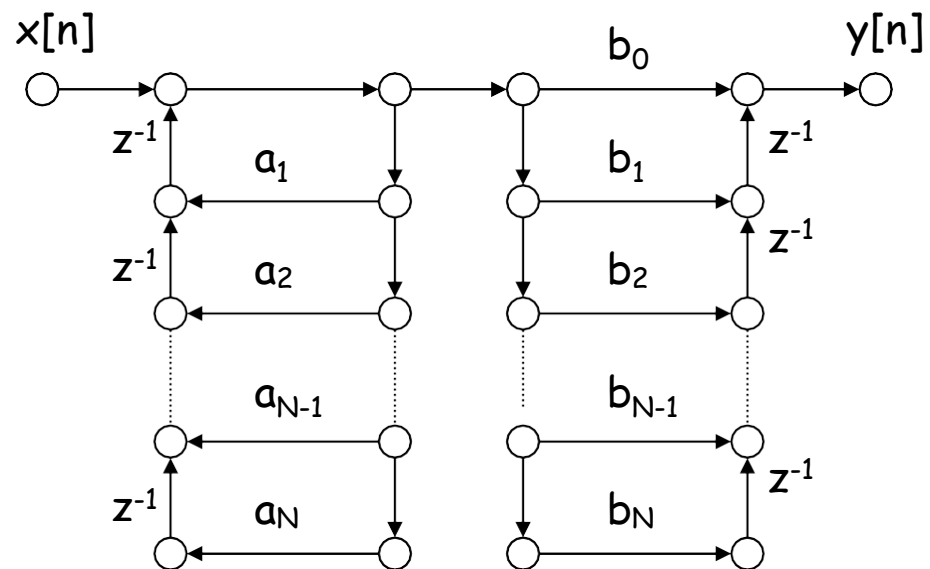
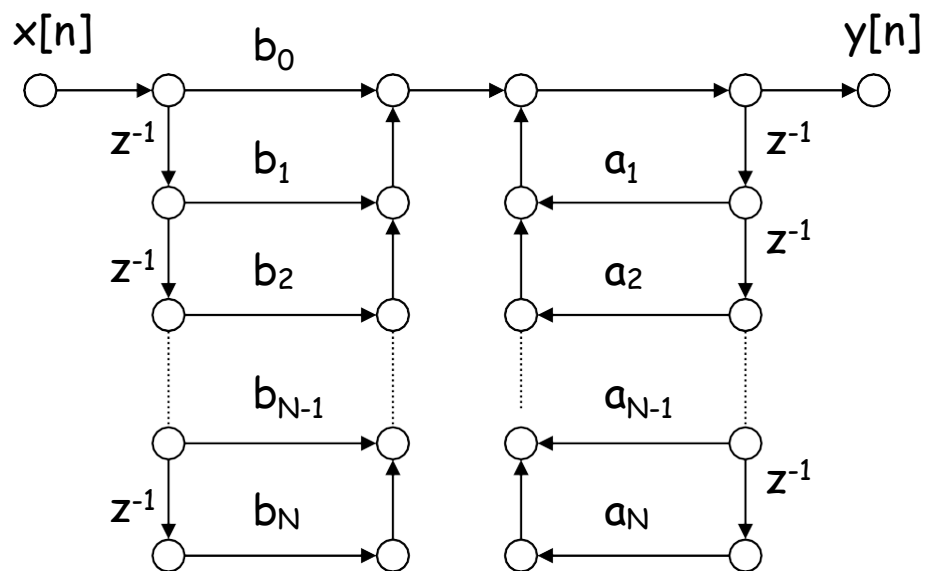
$$\begin{aligned}
 H(z) &= \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = 8 + \frac{-7 + 8z^{-1}}{1 - 0.75z^{-1} + 0.125z^{-2}} \\
 &= 8 + \frac{18}{1 - 0.5z^{-1}} - \frac{25}{1 - 0.25z^{-1}}
 \end{aligned}$$

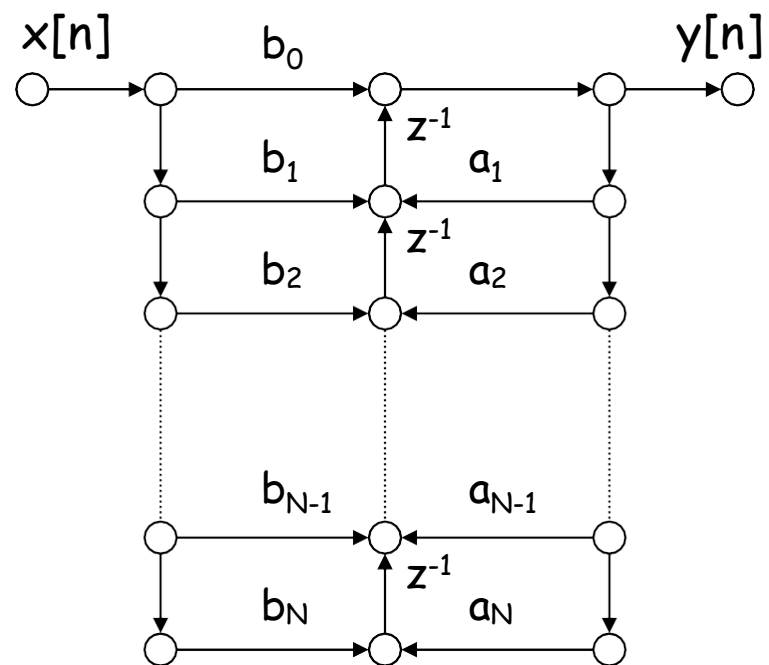
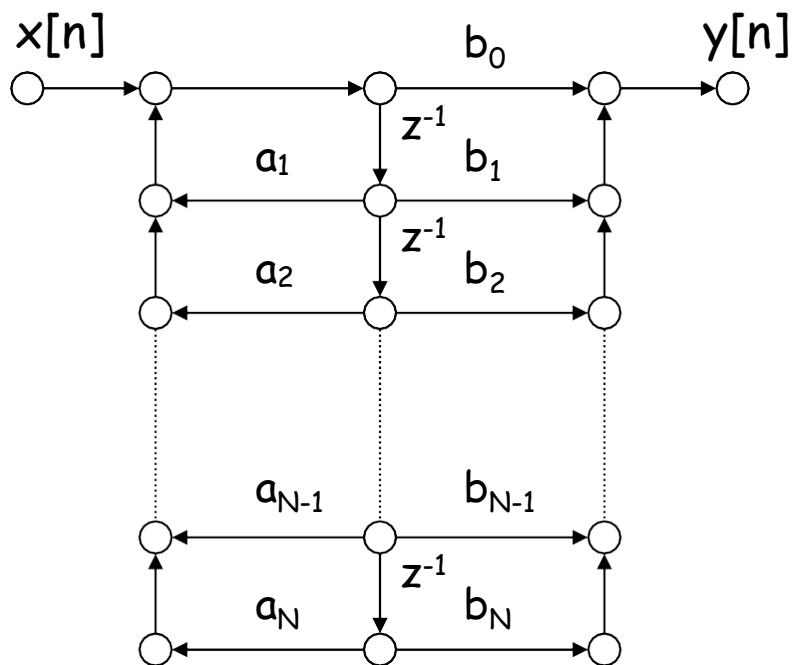


Transposed Forms

- Transposition (or flow graph reversal) of a flow graph is accomplished by reversing the directions of all branches in the network while keeping the branch transmittances as they were and reversing the roles of the input and output so that source nodes become sink nodes and vice versa.







Basic Network Structures for FIR Systems

- Direct Form
 - It is also referred to as a *tapped delay line* structure or a *transversal filter* structure.
- Transposed Form
- Cascade Form

$$H(z) = \sum_{n=0}^M h[n]z^{-n} = \prod_{k=1}^{M_s} (b_{0k} + b_{1k}z^{-1} + b_{2k}z^{-2})$$

where M_s is the largest integer contained in $(M + 1)/2$. If M is odd, one of coefficients b_{2k} will be zero.

Direct Form

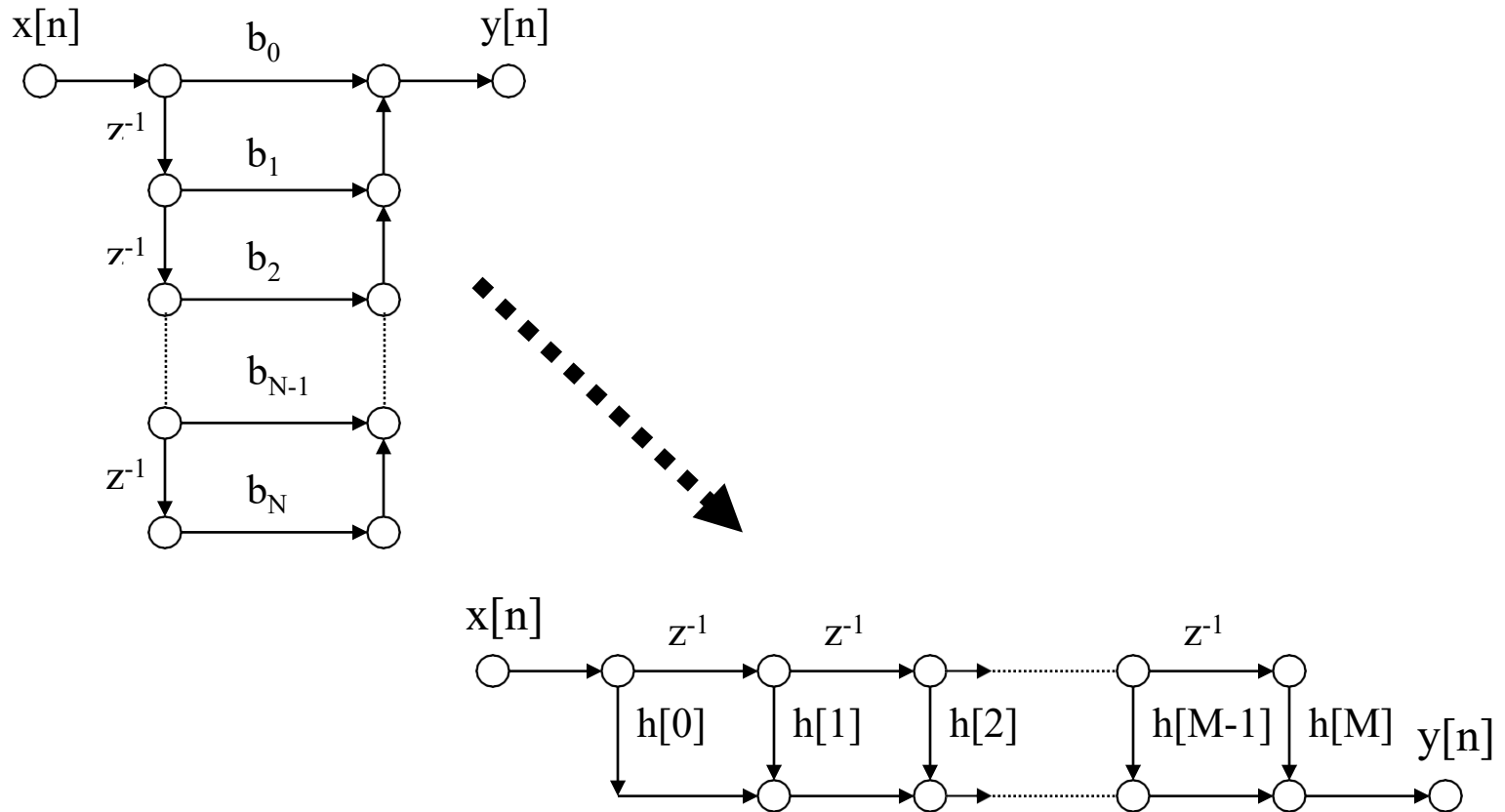
- For causal FIR system, the system function has only zeros (except for poles at $z = 0$) with the difference equation:

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

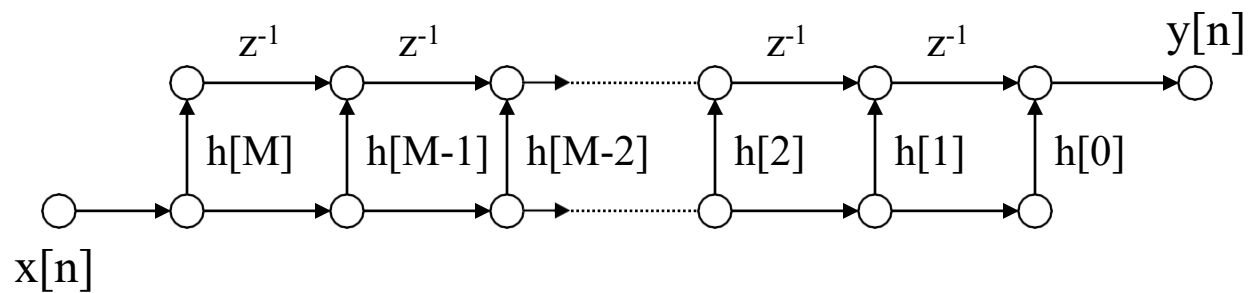
- It can be interpreted as the discrete convolution of $x[n]$ with the impulse response

$$h[n] = \begin{cases} b_n, & n = 0, 1, \dots, M, \\ 0, & \text{otherwise.} \end{cases}$$

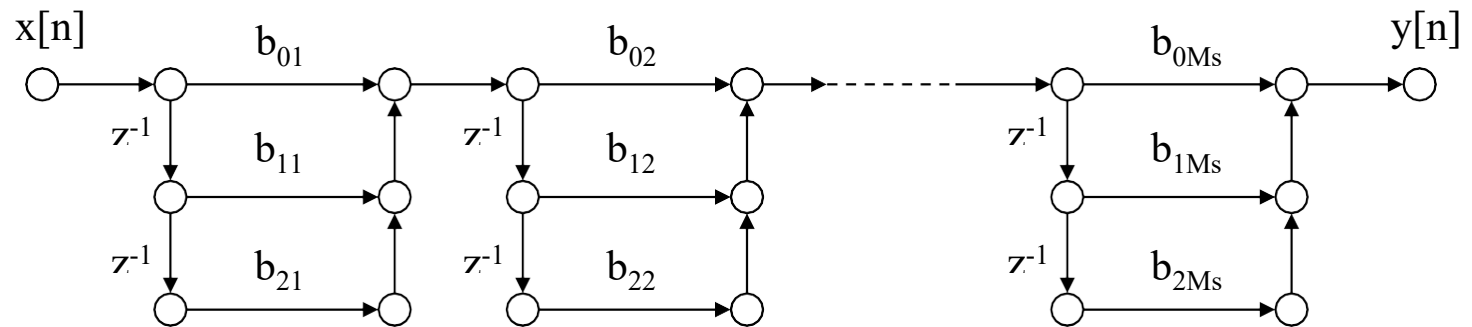
Direct Form (Tapped Delay Line or Transversal Filter)



Transposed Form of FIR Network



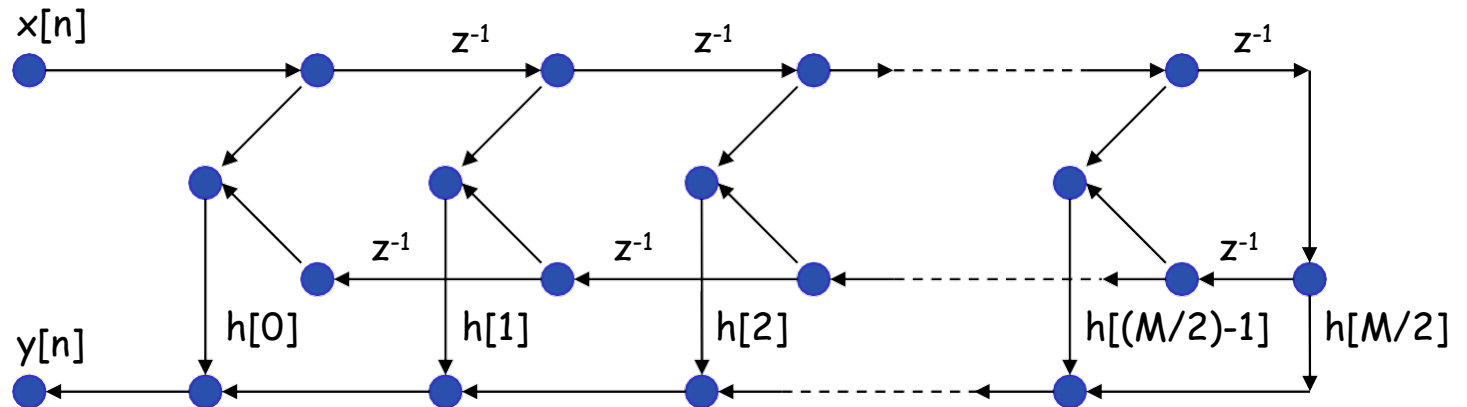
Cascade Form Structure of a FIR System



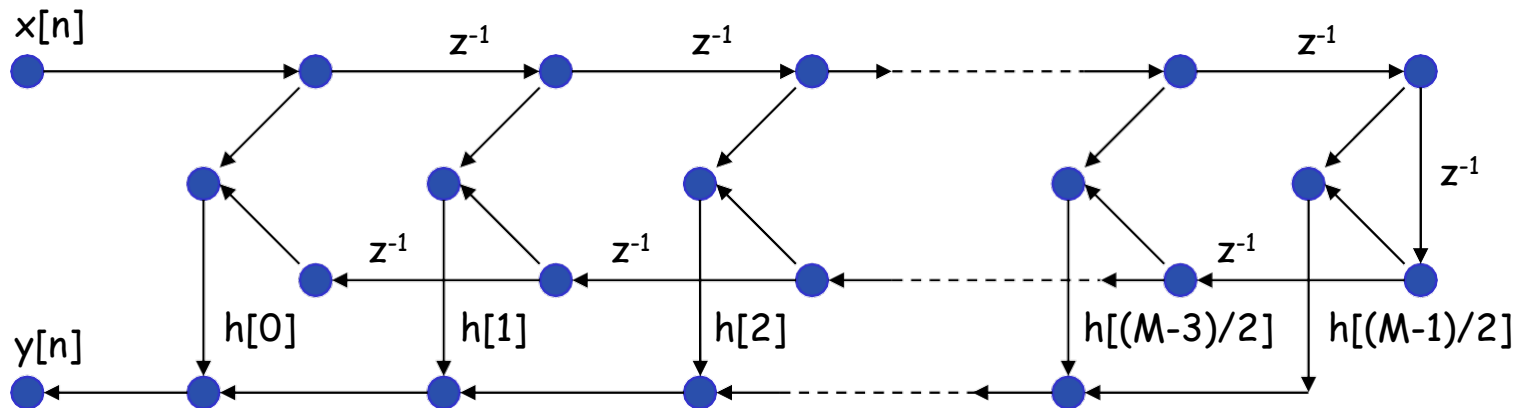
Structures for Linear-Phase FIR Systems

- Structures for Linear Phase FIR Systems :
 - $h[M-n] = h[n]$ for $n = 0, 1, \dots, M$
 - For M is an even integer : Type I
$$y[n] = \sum_{k=0}^{(M/2)-1} h[k](x[n-k] + x[n-M+k]) + h[M/2]x[n-M/2]$$
 - For M is an odd integer : Type II
$$y[n] = \sum_{k=0}^{(M-1)/2} h[k](x[n-k] + x[n-M+k])$$
 - $h[M-n] = -h[n]$ for $n = 0, 1, \dots, M$
 - For M is an even integer : Type III
$$y[n] = \sum_{k=0}^{(M/2)-1} h[k](x[n-k] - x[n-M+k])$$
 - For M is an odd integer : Type IV
$$y[n] = \sum_{k=0}^{(M-1)/2} h[k](x[n-k] - x[n-M+k])$$

Direct form structure for an FIR linear-phase when M is even.



Direct form structure for an FIR linear-phase when M is odd.



Lattice Structures

- Theory of autoregressive signal modeling :
 - *Lattice Structure*
- Development of digital filter structures that are analogous to analog filter structures :
 - *Wave Digital Filters*
- Another structure development approach is based on state-variable representations and linear transformations.

Lattice Structures 2

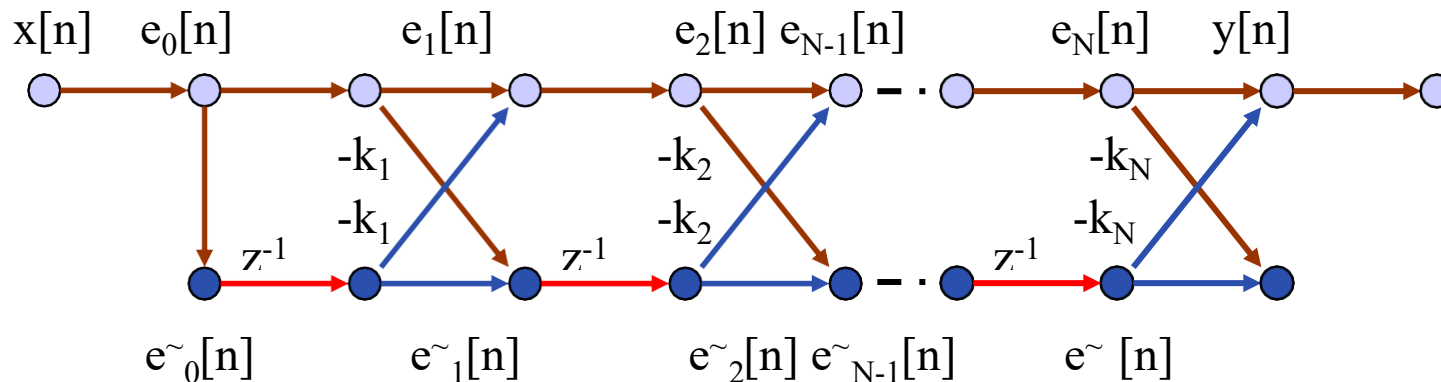
- FIR Lattice

$$H(z) = \frac{Y(z)}{X(z)} = A(z) = \left[1 - \sum_{m=1}^N a_m z^{-m} \right]$$

The coefficients $\{k_i\}$ in the lattice structures are referred to as the k-parameters, which are called *reflection coefficients* or *PARCOR coefficients*.

- When used in signal modeling, the k-parameters are estimated from a data signal .
- Given a set of k-parameters, the system function can be found and therefore the impulse response : by recurrence formula for computing $A(z)$ in terms of the intermediate system functions

Reflection coefficients or PARCOR coefficients structure



Signal flow graph of an FIR lattice system

$$e_0[n] = e_0^{\sim}[n] = x[n]$$

$$\begin{aligned} e_i[n] &= e_{i-1}[n] - k_i e_{i-1}^{\sim}[n-1] \\ e_i^{\sim}[n] &= -k_i e_i[n] + e_{i-1}^{\sim}[n-1] \end{aligned} \quad i = 1, 2, \dots, N,$$

$$y[n] = e_N[n]$$

A recurrence formula for computing $A(z) = H(z) = Y(z)/X(z)$ can be obtained in terms of intermediate system functions:

$$A_i(z) = \frac{E_i(z)}{E_o(z)} = \frac{1}{1 - \sum_{m=1}^i a_m^{(i)} z^{-m}}$$

By recursive technique:

$$a_i^{(i)} = k_i$$

$$a_m^{(i)} = a_m^{(i-1)} - k_i a_{i-m}^{(i-1)},$$

$$m = 1, 2, \dots, (i-1)$$

Or by reverse recursive technique:

$$k_i = a_i^{(i)}$$

$$a_m^{(i-1)} = [a_m^{(i)} + k_i a_{i-m}^{(i)}] / [1 - k_i^2], \quad m = 1, 2, \dots, (i-1)$$

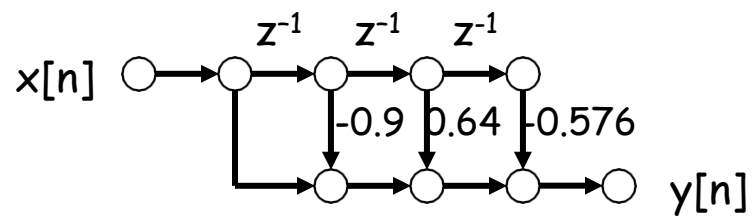
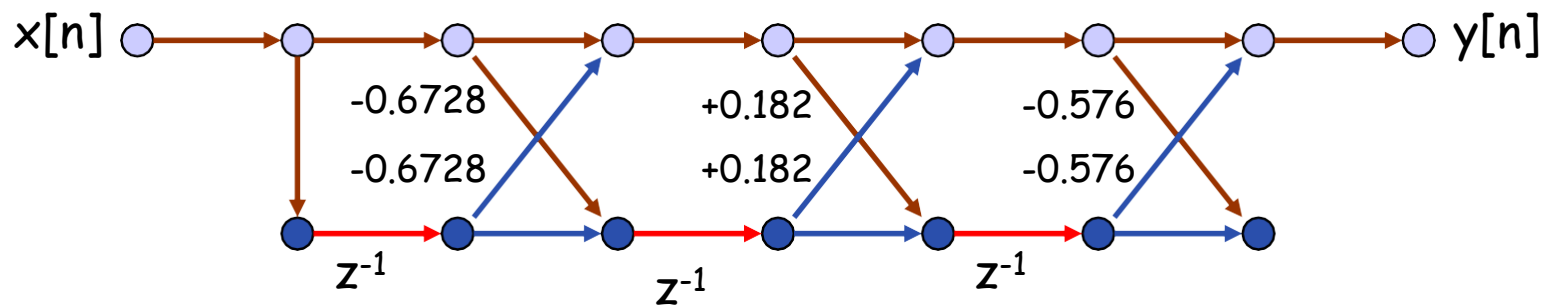
Example:

$$A(z) = (1-0.8jz^{-1})(1+0.8jz^{-1})(1-0.9z^{-1}) = 1 - 0.9z^{-1} + 0.64z^{-2} - 0.576z^{-3}.$$

Then, $a_1^{(3)} = 0.9$, $a_2^{(3)} = -0.64$, and $a_3^{(3)} = 0.576$

The k-parameter can be computed as follow:

$$\begin{aligned} k_3 &= a_3^{(3)} = 0.576 \\ a_1^{(2)} &= [a_1^{(3)} + k_3 a_2^{(3)}] / [1 - k_3^2] = 0.79518245 \\ a_2^{(2)} &= [a_2^{(3)} + k_3 a_1^{(3)}] / [1 - k_3^2] = -0.18197491 \\ k_2 &= a_2^{(2)} = -0.18197491 \\ a_1^{(1)} &= [a_1^{(2)} + k_2 a_2^{(2)}] / [1 - k_2^2] = 0.67275747 \\ k_1 &= a_1^{(1)} = 0.67275747 \end{aligned}$$



All-Pole Lattice

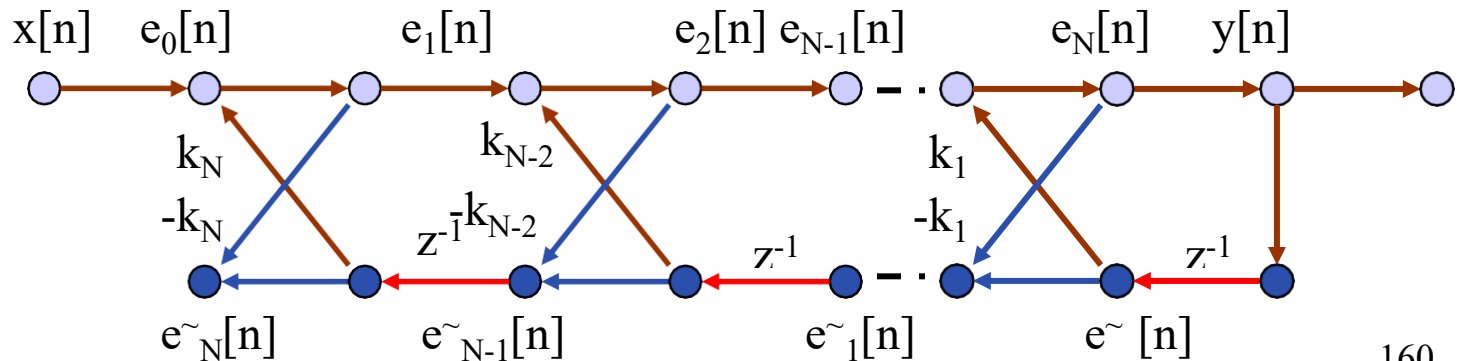
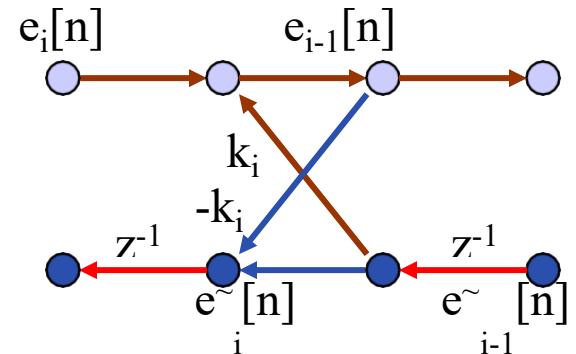
- A lattice system with an all-pole system function $H(z) = 1/A(z)$ can be developed from the FIR lattice .
 - all roots of $A(z)$ must be inside the unit circle: $|k_i| < 1, i = 1, \dots, N$

$$e_N[n] = x[n]$$

$$E_{i-1}[n] = e_i[n] + k_i \tilde{e}_{i-1}[n-1], \quad i = N, (N-1), \dots, 1,$$

$$\tilde{e}_i[n] = -k_i e_i[n] + \tilde{e}_{i-1}[n-1]$$

$$y[n] = e_0[n] = \tilde{e}_0[n]$$



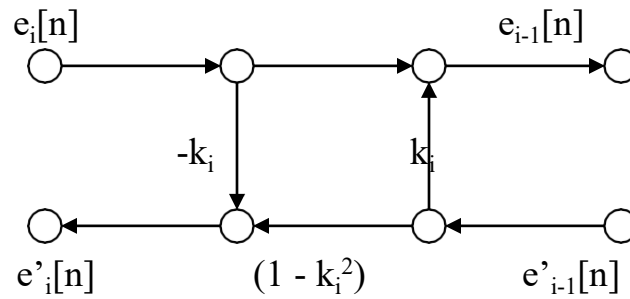
Basic all-pole lattice structures

- Three-multiplier form
- Four-multiplier, normalized form

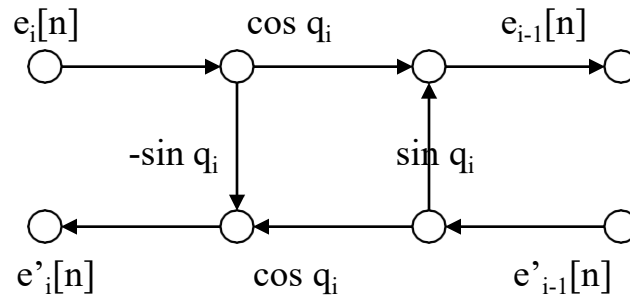
$$H(z) = \frac{\prod_{i=1}^N \cos \theta_i}{A(z)}$$

- Four-multiplier, Kelly-Lochbaum form : was first derived as an acoustic tube model for speech synthesis.

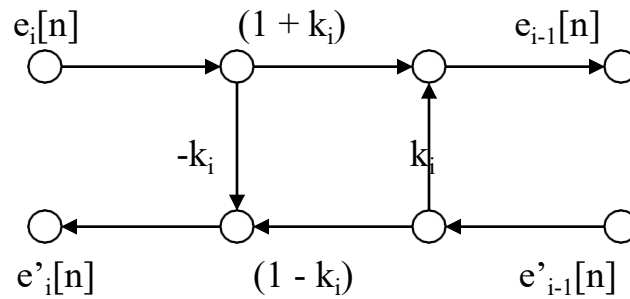
$$H(z) = \frac{\prod_{i=1}^N (1 + k_i)}{A(z)}$$



Three-multiplier form



Four-multiplier, normalized form



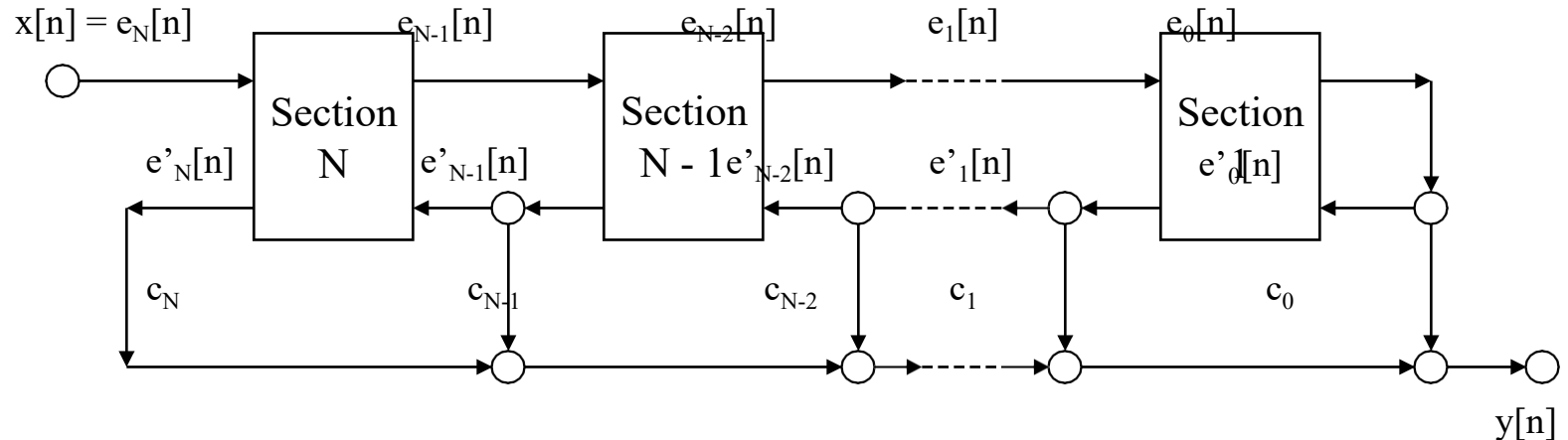
Four-multiplier, Kelly-Lochbaum form

Lattice Systems with Poles and Zeros

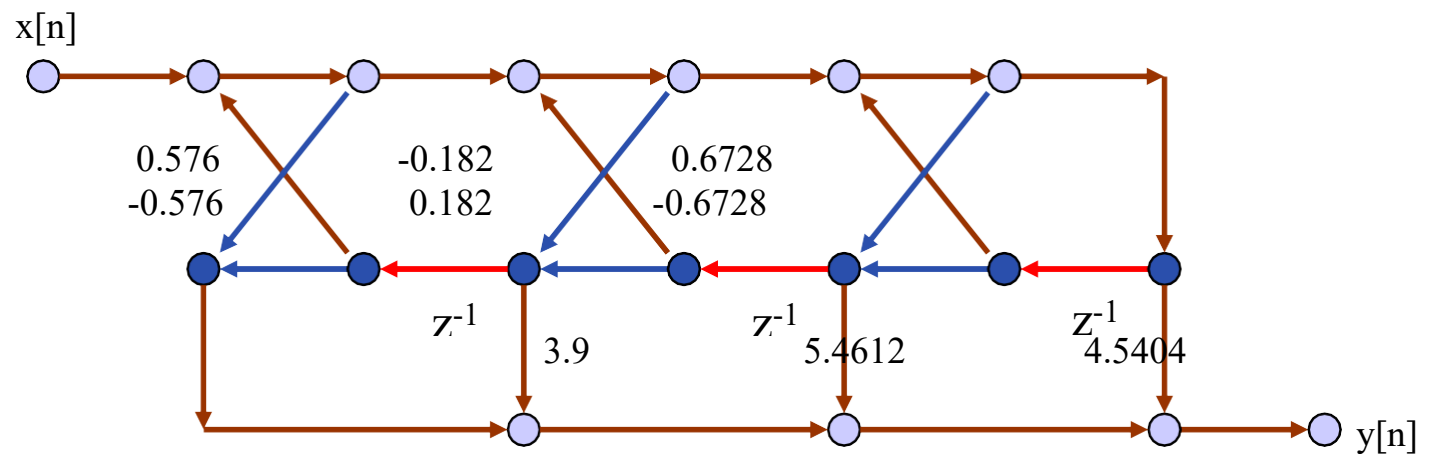
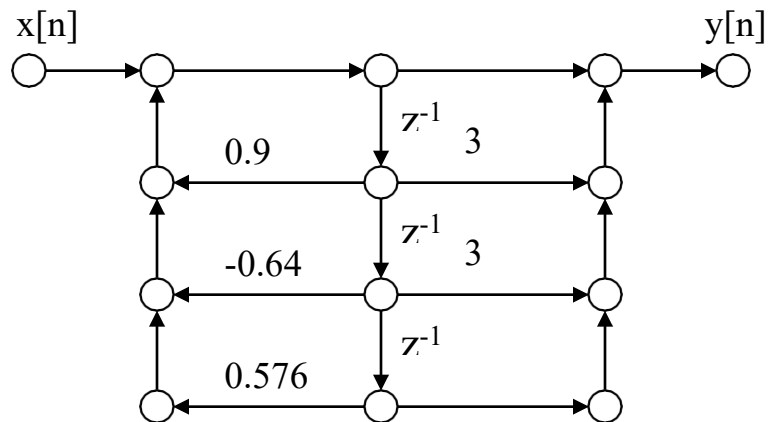
$$H(z) = \frac{Y(z)}{X(z)} = \sum_{i=0}^N \frac{c_i z^{-i} A(z^{-1})}{A(z)} = \frac{B(z)}{A(z)}$$

$$B(z) = \sum_{m=0}^N b_m z^{-m}$$

$$b_m = c_m - \sum_{i=m+1}^N c_i a_{i-m}^{(i)}$$



Example of lattice IIR filter with poles and zeros



UNIT-2

DFS , DFT & FFT

Fourier representation of signals

- A *discrete-time sinusoidal signal* $x[n]$ is obtained by sampling a continuous-time sinusoid $x(t) = \cos(2\pi F_0 t + \theta)$ at equally spaced points $t = nT$, which results in

$$x[n] = A \cos(2\pi F_0 nT + \theta) \Big|_{T=1/F_s} = A \cos \left(2\pi \frac{F_0}{F_s} n + \theta \right)$$

where F_0 (Hz) is the fundamental frequency of $x(t)$ and F_s is the *sampling frequency*.

- The *normalized frequency* variable is defined as

$$f \triangleq \frac{F}{F_s} = FT$$

where T is the *sampling period*.

- Similarly, the *normalized angular frequency* variable is defined as

$$\omega \triangleq 2\pi f = 2\pi \frac{F}{F_s} = \Omega T$$

- In this case, the discrete sinusoidal signal can be expressed as

$$x[n] = A \cos(2\pi f_0 n + \theta) = A \cos(\omega_0 n + \theta)$$

Fourier representation of signals

- **Periodicity in time:** By definition $x[n]$ is *periodic* if $x[n + N] = x[n], \forall n$.

$$x[n + N] = A \cos(2\pi f_0 n + 2\pi f_0 N + \theta) = A \cos(2\pi f_0 n + \theta) = x[n]$$

which is possible if and only if $2\pi f_0 N = 2\pi k$, with $k \in \mathbb{Z}$.

Result

The sequence $x[n] = A \cos(2\pi f_0 n + \theta)$ is periodic iff $f_0 = k/N$, that is, f_0 is a rational number. If k and N are a pair of *mutually prime* integers, then N is a fundamental period of $x[n]$.

- **Periodicity in frequency:** We can see that

$$A \cos[(\omega_0 + k 2\pi)n + \theta] = A \cos(\omega_0 n + \underbrace{kn}_{\in \mathbb{Z}} 2\pi + \theta) = A \cos(\omega_0 n + \theta)$$

Result

The sequence $x[n] = A \cos(\omega_0 n + \theta)$ is periodic in ω_0 with fundamental period 2π and periodic in f_0 with fundamental period one.

Fourier representation of signals

- All distinct sinusoidal sequences have frequencies within an interval of 2π radians. We shall use the *fundamental frequency ranges*

$$-\pi \leq \omega < \pi \quad \text{or} \quad 0 \leq \omega < 2\pi$$

Therefore, if $0 \leq \omega_0 < 2\pi$, the frequencies ω_0 and $\omega_0 + m 2\pi$ are indistinguishable in terms of their values.

- Since $A \cos(\omega_0[n + n_0] + \theta) = A \cos(\omega_0 n + (\omega_0 n_0 + \theta))$, a time shift is equivalent to a phase change.
- The rate of oscillation of a discrete-time sinusoid increases as ω_0 goes from $\omega_0 = 0$ to $\omega_0 = \pi$. Yet, as ω_0 increases from $\omega_0 = \pi$ to $\omega_0 = 2\pi$, the oscillations become slower. Therefore:

$$\begin{array}{ll} \text{Vicinity of } \omega_0 = k 2\pi & \implies \text{Low frequencies} \\ \text{Vicinity of } \omega_0 = \pi + k 2\pi & \implies \text{High frequencies} \end{array}$$

Discrete complex exponentials

- Similar properties hold for the *discrete-time complex exponentials*

$$s_k = A_k e^{j\omega_k n}$$

- For $s_k[n]$ to be periodic with fundamental period N , the frequency ω_k should be a rational multiple of 2π , that is $\omega_k = 2\pi k/N$.

All distinct complex exponentials with period N and frequency in the fundamental range, have frequencies equal to $\{\omega_k = 2\pi k/N\}_{k=0}^{N-1}$.

- The discrete complex exponentials are N -periodic in both the n - and k -variables.

$$s_k[n + N] = s_k[n] \quad (\text{periodic in time})$$

$$s_{k+N}[n] = s_k[n] \quad (\text{periodic in frequency})$$

- The complex exponentials are also *orthogonal*, viz.

$$\langle s_k, s_m \rangle \triangleq \sum_{n=0}^{N-1} s_k[n] s_m^*[n] = \begin{cases} N, & k = m \\ 0, & k \neq m \end{cases}$$

Discrete Fourier Series

- Given a periodic sequence $\tilde{x}[n]$ with period N so that

$$\tilde{x}[n] = \tilde{x}[n + rN]$$

- The Fourier series representation can be written as

$$\tilde{x}[n] = \frac{1}{N} \sum_k \tilde{x}[k] e^{j(2\pi/N)kn}$$

- The Fourier series representation of continuous-time periodic signals require infinite many complex exponentials
- Not that for discrete-time periodic signals we have

$$e^{j(2\pi/N)(k+mN)n} = e^{j(2\pi/N)kn} e^{j(2\pi mn)} = e^{j(2\pi/N)kn}$$

- Due to the periodicity of the complex exponential we only need N exponentials for discrete time Fourier series

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{x}[k] e^{j(2\pi/N)kn}$$

Discrete Fourier Series Pair

- A periodic sequence in terms of Fourier series coefficients

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{x}[k] e^{j(2\pi/N)kn}$$

- The Fourier series coefficients can be obtained via

$$\tilde{x}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j(2\pi/N)kn}$$

- For convenience we sometimes use

- Analysis equation $W_N = e^{-j(2\pi/N)}$

$$\tilde{x}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] W_N^{kn}$$

- Synthesis equation

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{x}[k] W_N^{-kn}$$

Fourier series for discrete-time periodic signals

- Consider a linear combination of N complex exponentials

$$x[n] = \sum_{k=0}^{N-1} c_k e^{j\frac{2\pi}{N}kn} = \sum_{k=0}^{N-1} c_k s_k[n]$$

which is periodic with fundamental period N .

- To determine the series expansion coefficients c_k , we exploit the orthogonality of $s_k[n]$ as follows

$$\langle x, s_m \rangle = \sum_{n=0}^{N-1} x[n] s_m^*[n] = \sum_{k=0}^{N-1} c_k \langle s_k, s_m \rangle = N c_m, \quad m = 0, \dots, N-1$$

- Therefore, we have

$$c_k = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}$$

which is periodic in k with the fundamental period equal to N .

Discrete-time Fourier series (DTFS)

DTFS

The *Discrete Time Fourier Series (DTFS)* pair is defined as

$$x[n] = \sum_{k=0}^{N-1} c_k e^{j\frac{2\pi}{N}kn} \iff c_k = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}$$

- **Parseval's relation:** The average power in one period of $x[n]$ can be expressed in terms of the Fourier series coefficients as

$$P_{\text{av}} = \frac{1}{N} \sum_{n=0}^{N-1} |x[n]|^2 = \sum_{k=0}^{N-1} |c_k|^2$$

- The value of $|c_k|^2$ provides the portion of the average power P_{av} of $x[n]$ that is contributed by its k -th harmonic component. Since $c_{k+N} = c_k$, there are only N distinct harmonic components.
- The graph of $|c_k|^2$ (as a function of either $f = k/N$, $\omega = 2\pi k/N$, or simply k) is known as the *power spectrum* of $x[n]$.

Fourier representation of aperiodic signals

- Consider a finite duration sequence $x[n]$, such that $x[n] = 0$ outside the range $-L_1 \leq n \leq L_2$. Define a *periodized version* $x_p[n]$ of $x[n]$ as

$$x_p[n] = \sum_{l \in \mathbb{Z}} x[n - lN], \quad \text{with } N > L_1 + L_2 + 1$$

- The DTFS of $x_p[n]$ is given by

$$x_p[n] = \sum_{k=0}^{N-1} c_k e^{j\frac{2\pi}{N}kn}$$

where

$$c_k = \frac{1}{N} \sum_{n=0}^{N-1} x_p[n] e^{-j\frac{2\pi}{N}kn} = \frac{1}{N} \sum_{n=-\infty}^{\infty} x[n] e^{-j\frac{2\pi}{N}kn}$$

- Define the “envelope” function $X(e^{j\omega})$ as

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

Discrete-time Fourier transform (DTFT)

- Noticing that $1/N = \Delta\omega/2\pi$, we have

$$x_p[n] = \sum_{k=0}^{N-1} c_k e^{j\frac{2\pi}{N}kn} = \frac{1}{2\pi} \sum_{k=0}^{N-1} X(e^{jk\Delta\omega}) e^{j(k\Delta\omega)n} \Delta\omega$$

- As $N \rightarrow \infty$, $x_p[n] = x[n]$, $\forall n$. Also, as $N \rightarrow \infty$, $\Delta\omega \rightarrow 0$, and the summation above passes to an integral over the frequency range from 0 to 2π . As a result, we have

DTFT

The *Discrete Time Fourier Transform (DTFT)* pair is defined as

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

- The quantities $X(e^{j\omega})$, $|X(e^{j\omega})|$, and $\angle X(e^{j\omega})$ are known as the *spectrum*, *magnitude spectrum*, and *phase spectrum* of $x[n]$.
- **Parseval's relation:**

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

Discrete Fourier Transform

- Periodic sequence and DFS coefficients

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] W_N^{kn}$$

$$\tilde{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}[k] W_N^{-kn}$$

- Since summations are calculated btw 0 and (N-1)

$$X[k] = \begin{cases} \sum_{n=0}^{N-1} x[n] W_N^{kn}, & 0 \leq k \leq N-1 \\ 0, & \text{otherwise} \end{cases}$$

$$x[n] = \begin{cases} \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}, & 0 \leq n \leq N-1 \\ 0, & \text{otherwise} \end{cases}$$

Generally

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}$$

Discrete Fourier Transform

- Given N samples $x[n]$, $0 \leq n \leq N - 1$ of an N -length sequence, its *Discrete Fourier Transform* (DFT) $X[k]$ is defined by

$$X[k] \triangleq \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi}{N} kn}, \quad 0 \leq k \leq N - 1$$

- Given N DFT coefficients $X[k]$, $0 \leq k \leq N - 1$, their related N “time-domain” samples $x[n]$, $0 \leq n \leq N - 1$ can be recovered by the *inverse DFT* (IDFT) given by

$$x[n] \triangleq \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j \frac{2\pi}{N} kn}$$

- Note that $X[k]$ is a function of the discrete frequency index k , which corresponds to $\omega_k = 2\pi/N$, $k = 0, 1, \dots, N - 1$.

In summary: The DFT pair

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad \xleftrightarrow{\text{DFT}} \quad x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}, \quad W_N \triangleq e^{-j \frac{2\pi}{N}}$$

Discrete Fourier Transform

- The correctness of the DFT formulas can be validated through:

$$\begin{aligned} X[k] &= \sum_{n=0}^{N-1} x[n] W_N^{nk} = \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{m=0}^{N-1} X[m] W_N^{-mn} \right] W_N^{nk} = \\ &= \sum_{m=0}^{N-1} X[m] \left[\frac{1}{N} \sum_{n=0}^{N-1} W_N^{(k-m)n} \right] \end{aligned}$$

- The orthogonality of discrete complex exponentials suggests

$$\frac{1}{N} \sum_{n=0}^{N-1} W_N^{(k-m)n} = \frac{1}{N} \langle W_N^k, W_N^m \rangle = \begin{cases} 1, & k - m = rN \\ 0, & \text{otherwise} \end{cases}$$

which concludes the proof.

- Note that the N complex numbers $\{W_N^{-k}\}_{k=0}^{N-1}$ satisfy

$$(W_N^{-k})^N = e^{j2\pi k} = 1$$

and therefore they form the *roots of unity* (i.e., the N solutions of $z^N - 1 = 0$). Note that these roots are equally spaced around the unit circle with the angular spacing of $2\pi/N$ radians.

Discrete Fourier Transform

- The N equations for the DFT coefficients can be expressed in matrix form as

$$\underbrace{\begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix}}_{X_N} = \underbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & W_N & \dots & W_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & \dots & W_N^{(N-1)(N-1)} \end{bmatrix}}_{W_N} \underbrace{\begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}}_{x_N}$$

- Thus, we have

$$X_N = W_N x_N$$

- Note that W_N is symmetric ($W_N = W_N^T$) and *orthogonal*, viz.

$$W_N^H W_N = N I_N \implies W_N^{-1} = \frac{1}{N} W_N^H = \frac{1}{N} W_N^*$$

- Therefore, x_N can be recovered (synthesized) from X_N according to

$$x_N = W_N^{-1} X_N = \frac{1}{N} W_N^* X_N$$

which is nothing else but a matrix representation of IDFT.

Discrete Fourier Transform

- The *twiddle factor* $W_N^{kn} = e^{-j\frac{2\pi}{N}kn}$ is periodic in both k and n with fundamental period N , namely

$$W_N^{(k+N)n} = W_N^{kn} \quad \text{and} \quad W_N^{k(N+n)} = W_N^{kn}$$

- Letting $k \in \mathbb{Z}$ results in the *Discrete Fourier Series* (DFS):

$$\tilde{X}[k] = \tilde{X}[k + N] = \sum_{n=0}^{N-1} x[n]W_N^{kn}, \quad \forall k \in \mathbb{Z}$$

- If n is allowed to take upon any integer value, the values of $x[n]$ will repeat with fundamental period N , resulting in the *Inverse Discrete Fourier Series* (IDFS).

$$\tilde{x}[n] = \tilde{x}[n + N], \quad \forall n \in \mathbb{Z}$$

- These periodicities are an inherent property of DFT, which stem from the discrete nature of time and frequency variables.

Summary of properties

	Property	N -point sequence	N -point DFT
		$x[n], h[n], v[n]$	$X[k], H[k], V[k]$
		$x_1[n], x_2[n]$	$X_1[k], X_2[k]$
1.	Linearity	$a_1 x_1[n] + a_2 x_2[n]$	$a_1 X_1[k] + a_2 X_2[k]$
2.	Time shifting	$x[\langle n - m \rangle_N]$	$W_N^{km} X[k]$
3.	Frequency shifting	$W_N^{-mn} x[n]$	$X[\langle k - m \rangle_N]$
4.	Modulation	$x[n] \cos(2\pi/N)k_0 n$	$\frac{1}{2} X[\langle k + k_0 \rangle_N] + \frac{1}{2} X[\langle k - k_0 \rangle_N]$
5.	Folding	$x[\langle -n \rangle_N]$	$X^*[k]$
6.	Conjugation	$x^*[n]$	$X^*[\langle -k \rangle_N]$
7.	Duality	$X[n]$	$Nx[\langle -k \rangle_N]$
8.	Convolution	$h[n] \bigcirc_N x[n]$	$H[k]X[k]$
9.	Correlation	$x[n] \bigcirc_N y[\langle -n \rangle_N]$	$X[k]Y^*[k]$
10.	Windowing	$v[n]x[n]$	$\frac{1}{N} V[k] \bigcirc_N X[k]$
11.	Parseval's theorem	$\sum_{n=0}^{N-1} x[n]y^*[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]Y^*[k]$	
12.	Parseval's relation	$\sum_{n=0}^{N-1} x[n] ^2 = \frac{1}{N} \sum_{k=0}^{N-1} X[k] ^2$	

DFT Pair & Properties

- The Discrete Fourier Transform pair

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j(2\pi/N)kn}$$

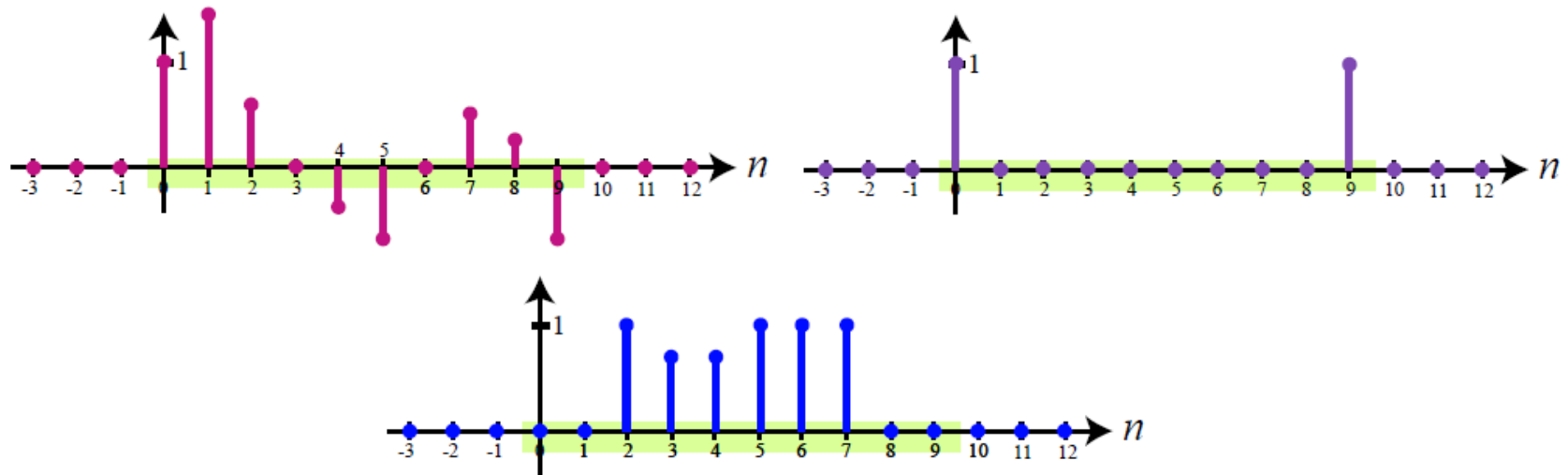
$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j(2\pi/N)kn}$$

Property	Time Domain	Frequency Domain
Notation:	$x(n)$ $x_1(n)$ $x_2(n)$	$X(\omega)$ $X_1(\omega)$ $X_2(\omega)$
Linearity:	$a_1 x_1(n) + a_2 x_2(n)$	$a_1 X_1(\omega) + a_2 X_2(\omega)$
Time shifting:	$x(n-k)$	$e^{-j\omega k} X(\omega)$
Time reversal	$x(-n)$	$X(-\omega)$
Convolution:	$x_1(n) * x_2(n)$	$X_1(\omega) X_2(\omega)$
Correlation:	$r_{x_1 x_2}(l) = x_1(l) * x_2(-l)$	$S_{x_1 x_2}(\omega) = X_1(\omega) X_2^*(-\omega)$ $= X_1(\omega) X_2^*(\omega)$ [if $x_2(n)$ real]
Wiener-Khintchine:	$r_{xx}(l) = x(l) * x(-l)$	$S_{xx}(\omega) = X(\omega) ^2$

Circular convolution

Assume: $x_1(n)$ and $x_2(n)$ have support $n = 0, 1, \dots, N - 1$.

Examples: $N = 10$ and support: $n = 0, 1, \dots, 9$



Modulo Indices and Periodic Repetition

$$(n)_N = n \bmod N = \text{remainder of } n/N$$

Example: $N = 4$

n	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8
$(n)_4$	0	1	2	3	0	1	2	3	0	1	2	3	0

$$\frac{n}{N} = \text{integer} + \frac{\boxed{\text{nonneg integer} < N}}{N}$$

$$\frac{5}{4} = 1 + \frac{1}{4}$$

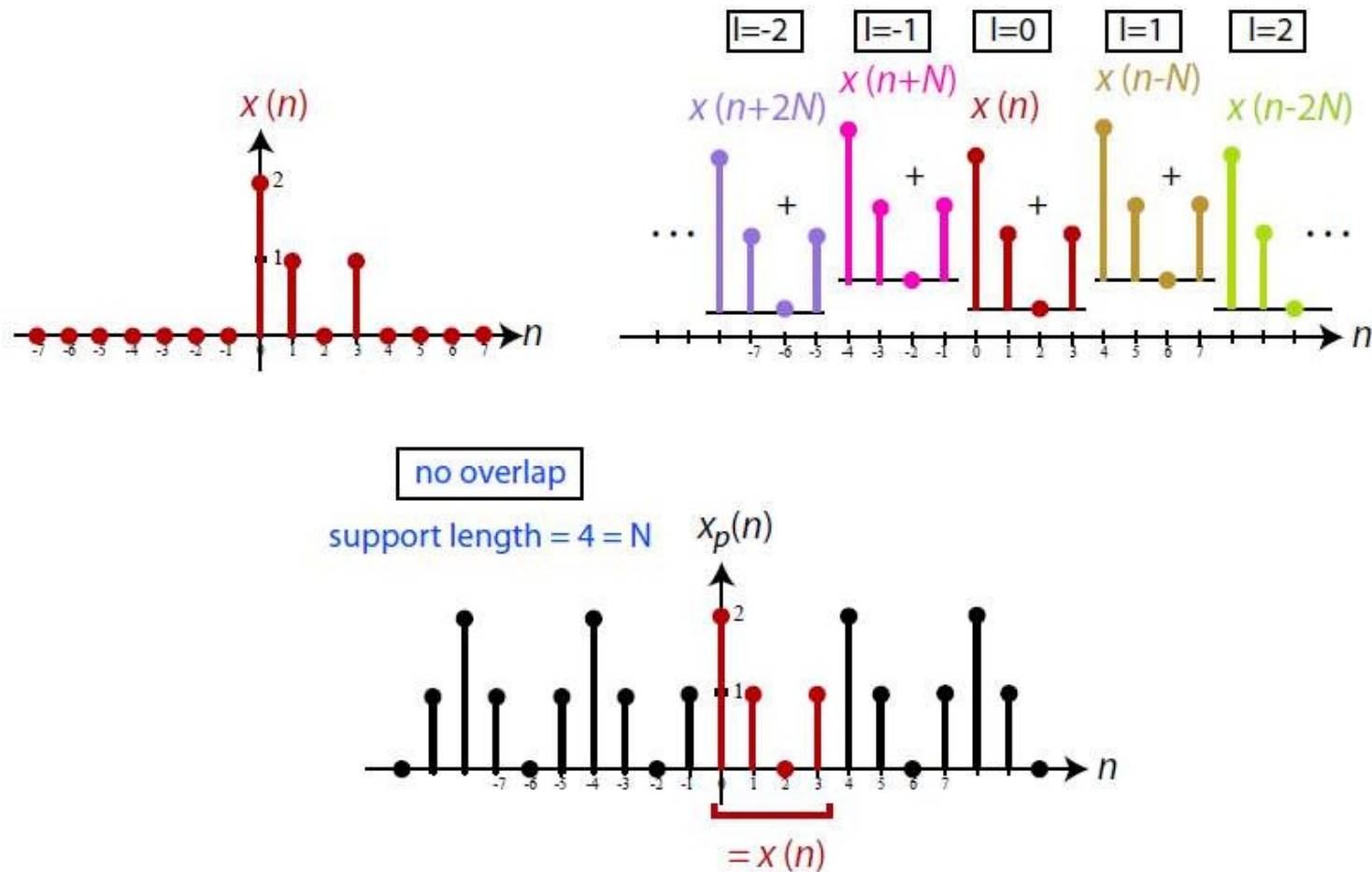
$$\frac{-2}{4} = -1 + \frac{2}{4}$$

Overlap During Periodic Repetition

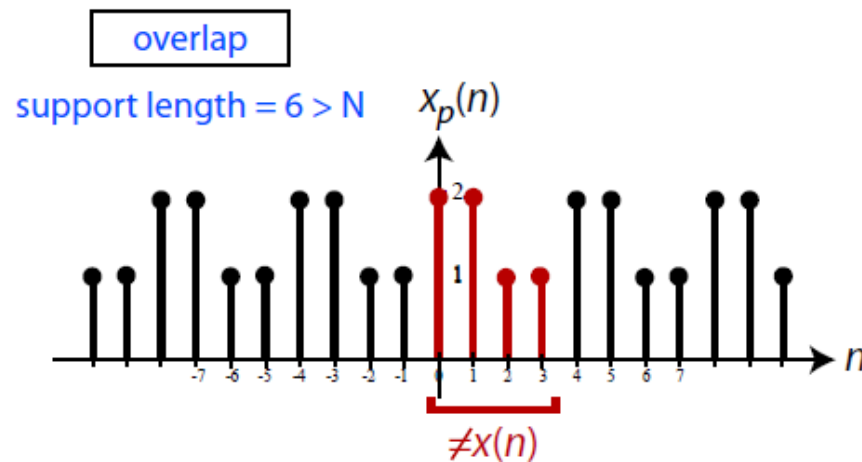
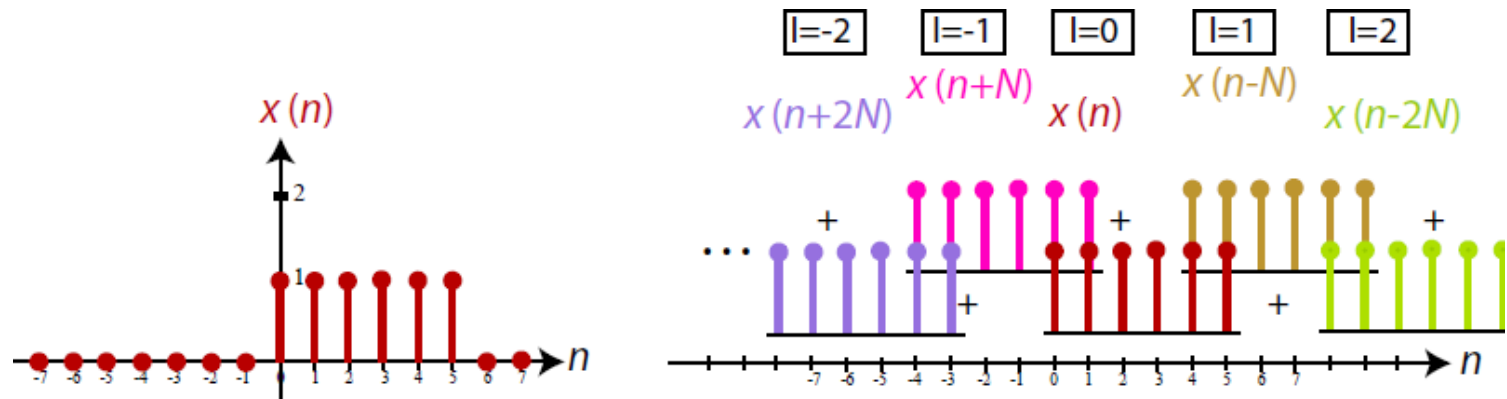
A periodic repetition makes an aperiodic signal $x(n)$, periodic to produce $x_p(n)$.

$$x_p(n) = \sum_{l=-\infty}^{\infty} x(n - lN)$$

Periodic repetition: $N=4$



Periodic repetition: $N=4$



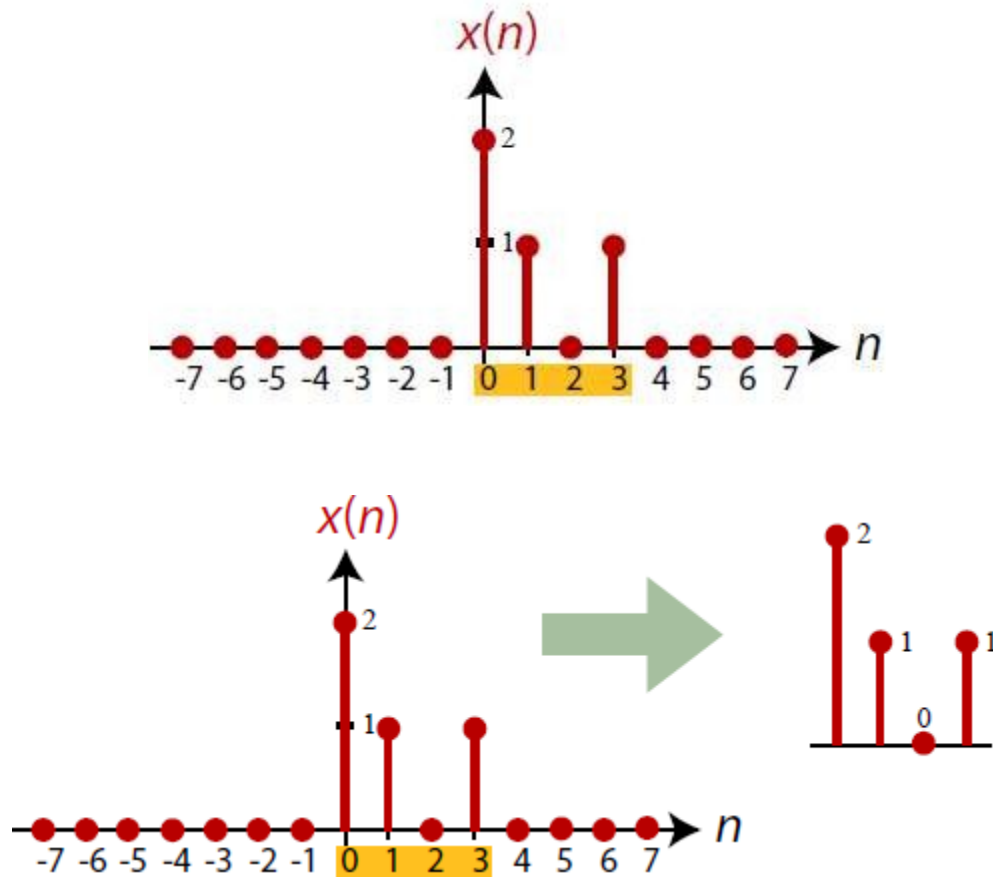
Modulo Indices and the Periodic Repetition

Assume: $x(n)$ has support $n = 0, 1, \dots, N - 1$.

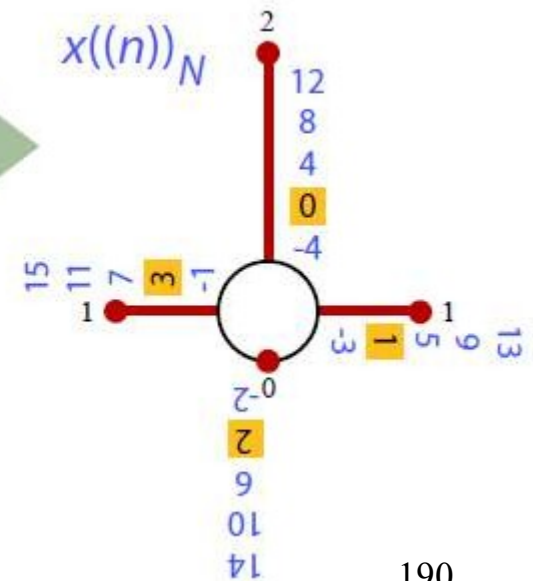
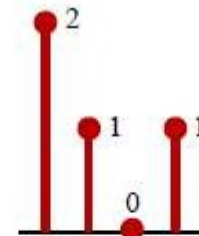
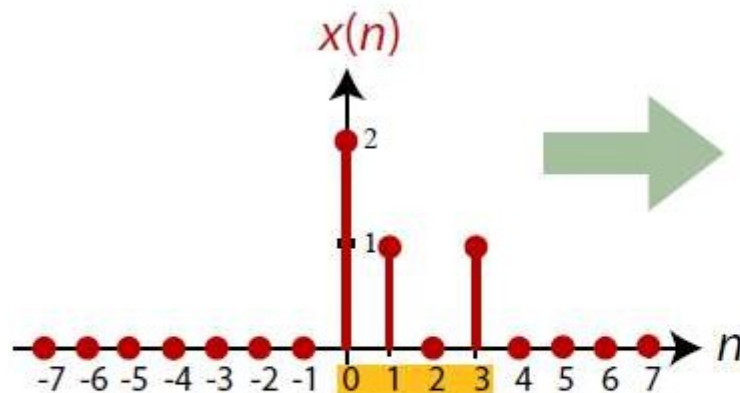
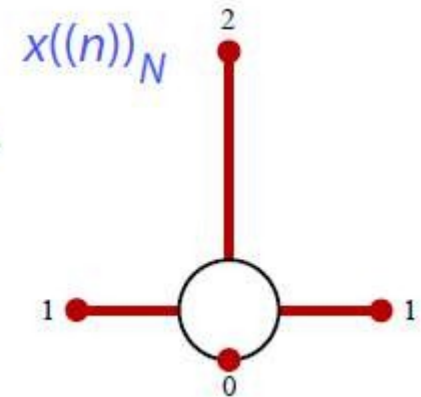
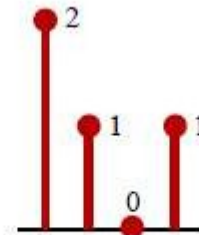
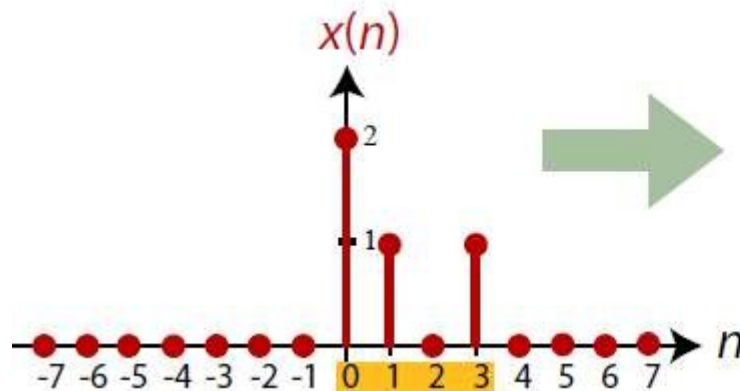
$$x((n))_N = x(n \bmod N) = x_p(n) = \sum_{l=-\infty}^{\infty} x(n - lN)$$

Note: Because the support size and period size are the same, there is **no overlap** when taking the periodic repetition $x((n))_N$.

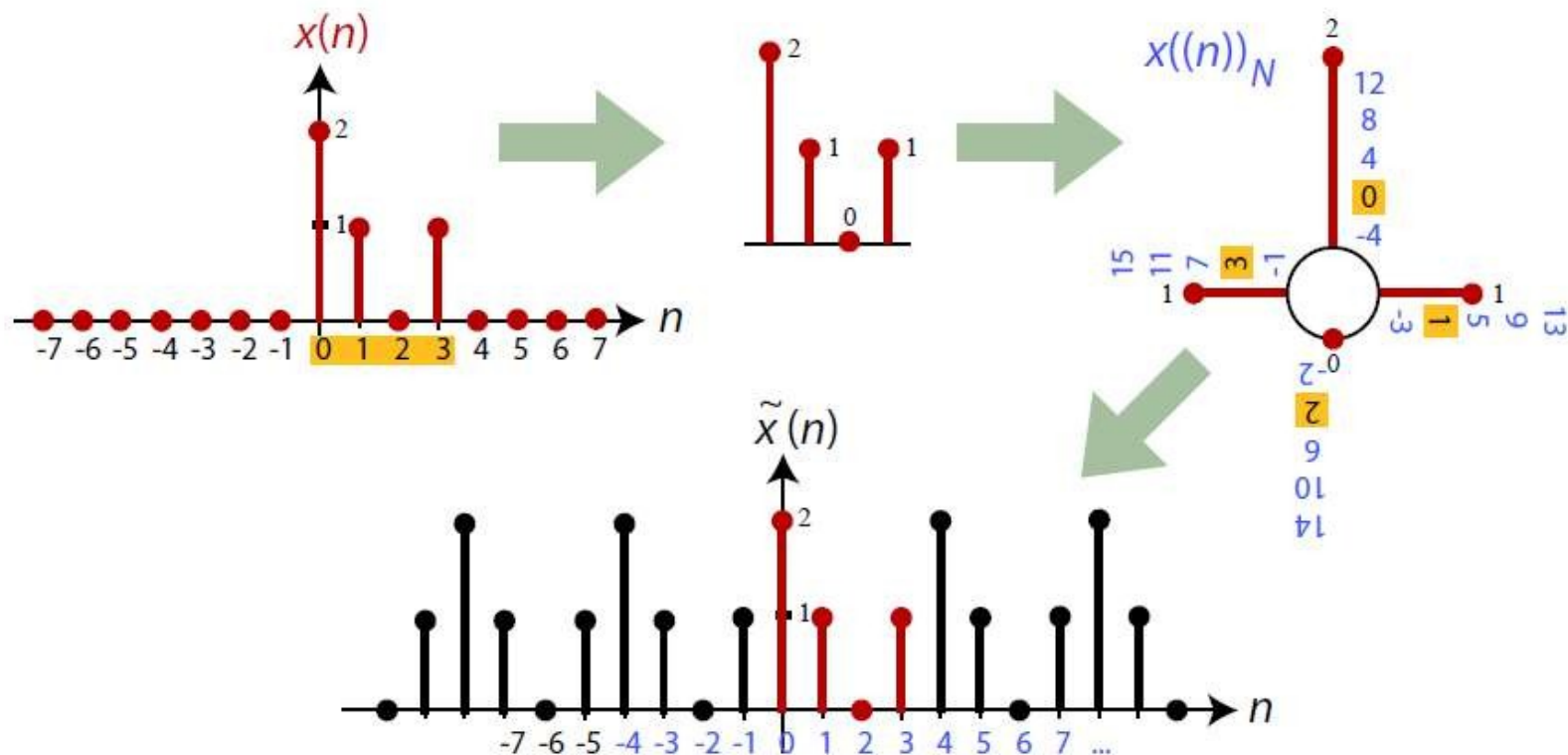
Modulo Indices and the Periodic Repetition



Modulo Indices and the Periodic Repetition



Modulo Indices and the Periodic Repetition



Therefore $x((n))_N = x_p(n)$.

Circular convolution

Assume: $x_1(n)$ and $x_2(n)$ have support $n = 0, 1, \dots, N - 1$.

To compute $\sum_{k=0}^{N-1} x_1(k)x_2((n-k))_N$ (or $\sum_{k=0}^{N-1} x_2(k)x_1((n-k))_N$):

1. Take the periodic repetition of $x_2(n)$ with period N :

$$x_{2p}(n) = \sum_{l=-\infty}^{\infty} x_2(n - lN)$$

2. Conduct a standard linear convolution of $x_1(n)$ and $x_{2p}(n)$ for $n = 0, 1, \dots, N - 1$:

$$x_1(n) \otimes x_2(n) = x_1(n) * x_{2p}(n) = \sum_{k=-\infty}^{\infty} x_1(k)x_{2p}(n-k) = \sum_{k=0}^{N-1} x_1(k)x_{2p}(n-k)$$

Note: $x_1(n) \otimes x_2(n) = 0$ for $n < 0$ and $n \geq N$.

Circular convolution

$$\sum_{k=0}^{N-1} x_1(k) \boxed{x_2((n-k))_N} = \sum_{k=0}^{N-1} x_1(k) \boxed{x_{2p}(n-k)}$$

... which makes sense, since $x((n))_N = x_p(n)$.

Circular convolution-another interpretation

Assume: $x_1(n)$ and $x_2(n)$ have support $n = 0, 1, \dots, N - 1$.

To compute $\sum_{k=0}^{N-1} x_1(k)x_2((n-k))_N$ (or $\sum_{k=0}^{N-1} x_2(k)x_1((n-k))_N$):

1. Conduct a linear convolution of $x_1(n)$ and $x_2(n)$ for all n :

$$x_L(n) = x_1(n) * x_2(n) = \sum_{k=-\infty}^{\infty} x_1(k)x_2(n-k) = \sum_{k=0}^{N-1} x_1(k)x_2(n-k)$$

2. Compute the periodic repetition of $x_L(n)$ and window the result for $n = 0, 1, \dots, N - 1$:

$$x_1(n) \otimes x_2(n) = \sum_{l=-\infty}^{\infty} x_L(n - lN), \quad n = 0, 1, \dots, N - 1$$

Using DFT for Linear Convolution

Therefore, circular convolution and linear convolution are related as follows:

$$x_C(n) = x_1(n) \otimes x_2(n) = \sum_{l=-\infty}^{\infty} x_L(n - lN)$$

for $n = 0, 1, \dots, N - 1$

Q: When can one recover $x_L(n)$ from $x_C(n)$?

When can one use the DFT to compute linear convolution?

A: When there is no overlap in the periodic repetition of $x_L(n)$.

When support length of $x_L(n) \leq N$.

Using DFT for Linear Convolution

- The *linear convolution* of two finite-length sequences $\{x[n]\}_{n=0}^{L-1}$ and $\{h[n]\}_{n=0}^{M-1}$ is a sequence $y[n]$ of length $L + M - 1$, given by

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k], \quad n = 0, 1, \dots, L + M - 2$$

- The convolution sequence $y[n]$ has DTFT given by

$$Y(e^{j\omega}) = X(e^{j\omega}) H(e^{j\omega})$$

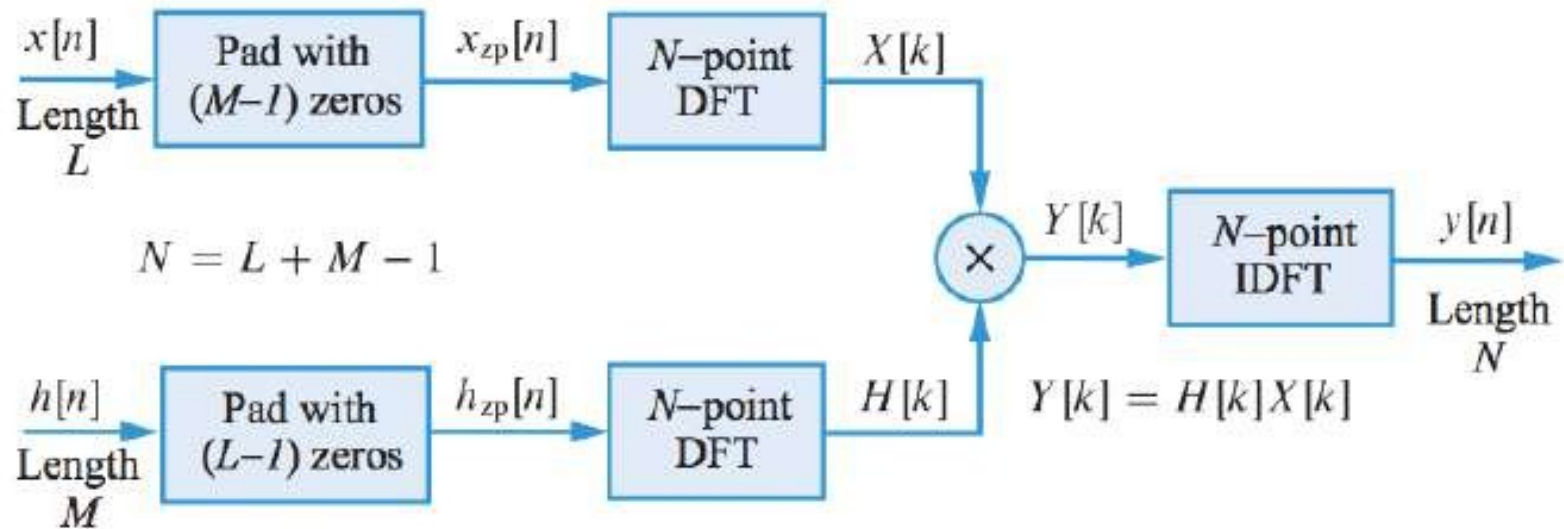
- If we sample $Y(e^{j\omega})$ at $\omega_k = 2\pi k/N$, where $N \geq L + M - 1$, we can uniquely recover $y[n]$ from $Y[k] = Y(e^{j2\pi k/N})$.
- On the other hand, the IDFTs of $H(e^{j\frac{2\pi k}{N}})$ and $X(e^{j\frac{2\pi k}{N}})$ yield the sequences $h[n]$ and $x[n]$ padded with $(N - M)$ and $(N - L)$ zeros, respectively. As a result,

$$y_{zp}[n] = x_{zp}[n] \circledast h_{zp}[n] \iff Y[k] = X[k]H[k]$$

- Note that if $N \geq L + M - 1$, $y[n] = y_{zp}[n]$, $0 \leq n \leq L + M - 2$, that is, circular convolution is identical to linear convolution. 196

Using DFT for Linear Convolution

- Thus, linear convolution can be implemented by means of the DFT as shown below.



- The length M of the impulse response at which the DFT based approach is more efficient than direct computation of convolution depends on the hardware and software available to implement the computations.

Using DFT for Linear Convolution

Let $x(n)$ have support $n = 0, 1, \dots, L - 1$.

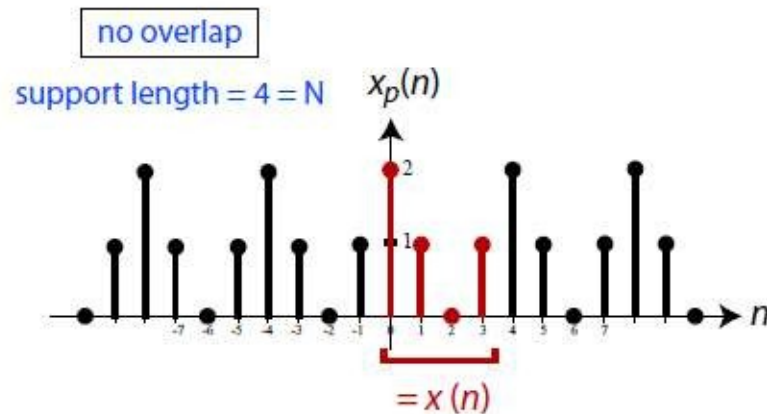
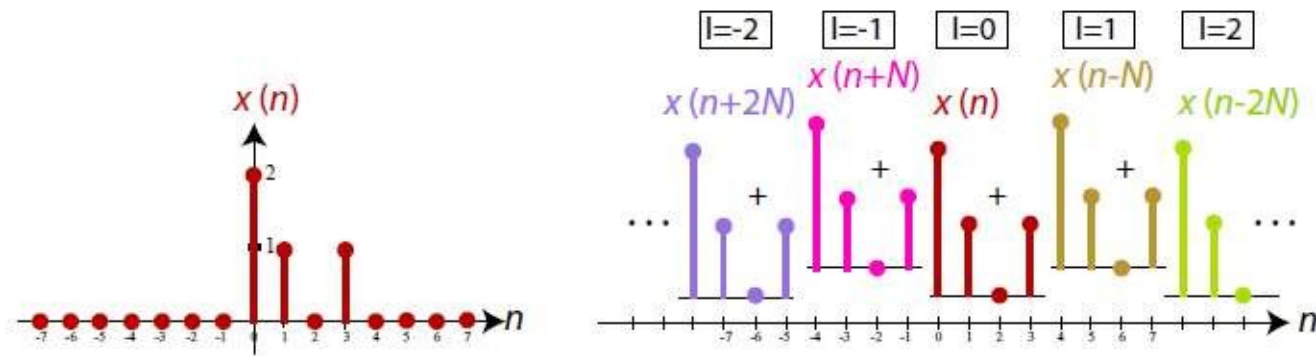
Let $h(n)$ have support $n = 0, 1, \dots, M - 1$.

We can set $N \geq L + M - 1$ and zero pad $x(n)$ and $h(n)$ to have support $n = 0, 1, \dots, N - 1$.

1. Take N -DFT of $x(n)$ to give $X(k)$, $k = 0, 1, \dots, N - 1$.
2. Take N -DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \dots, N - 1$.
3. Multiply: $Y(k) = X(k) \cdot H(k)$, $k = 0, 1, \dots, N - 1$.
4. Take N -IDFT of $Y(k)$ to give $y(n)$, $n = 0, 1, \dots, N - 1$.

Using DFT for Linear Convolution

Length of linear convolution result = Length of DFT



Using DFT for circular Convolution

$$N = L + M - 1.$$

Let $x_m(n)$ have support $n = 0, 1, \dots, N - 1$.

Let $h(n)$ have support $n = 0, 1, \dots, M - 1$.

We zero pad $h(n)$ to have support $n = 0, 1, \dots, N - 1$.

1. Take N -DFT of $x_m(n)$ to give $X_m(k)$, $k = 0, 1, \dots, N - 1$.
2. Take N -DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \dots, N - 1$.
3. Multiply: $Y_m(k) = X_m(k) \cdot H(k)$, $k = 0, 1, \dots, N - 1$.
4. Take N -IDFT of $Y_m(k)$ to give $y_{C,m}(n)$, $n = 0, 1, \dots, N - 1$.

Using DFT for circular Convolution

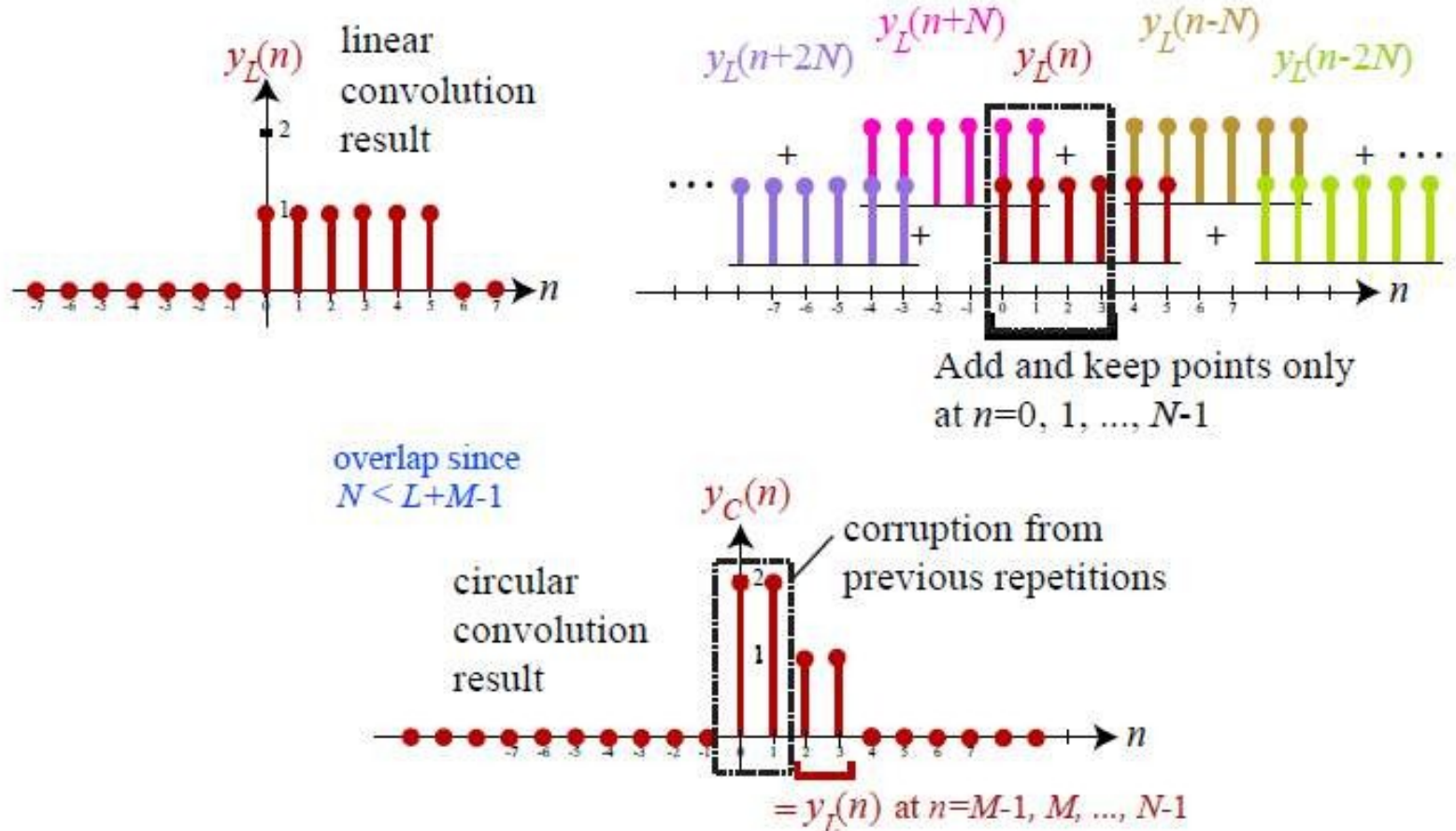
- Let's compute $y[n]$ for the case of $N = 4$. We have

$$\begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \end{bmatrix} = \underbrace{\begin{bmatrix} x[0] & x[3] & x[2] & x[1] \\ x[1] & x[0] & x[3] & x[2] \\ x[2] & x[1] & x[0] & x[3] \\ x[3] & x[2] & x[1] & x[0] \end{bmatrix}}_{\mathbf{X}_N} \begin{bmatrix} h[0] \\ h[1] \\ h[2] \\ h[3] \end{bmatrix}$$

- We note that the column of \mathbf{X}_N are generated by circularly shifting $x[n]$. A matrix generated by this process is called a *circulant matrix*.

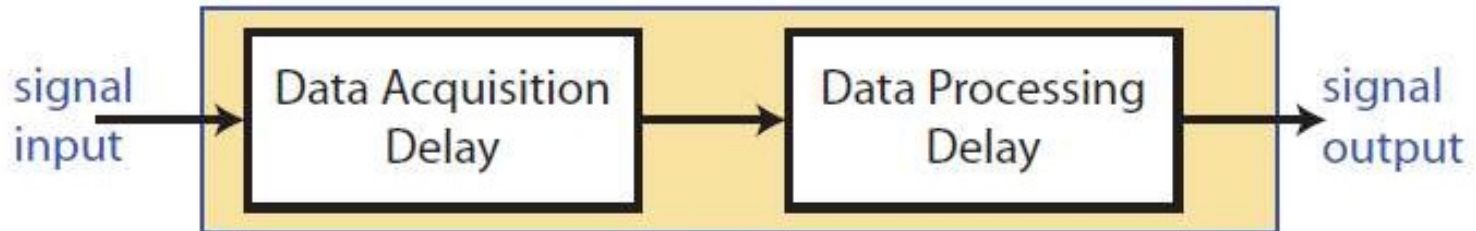
Using DFT for circular Convolution

Length of linear convolution result $>$ Length of DFT



Filtering of Long Data Sequences

- ▶ All N -input samples are required **simultaneously** by the DFT operator.
- ▶ If N is too large as for long data sequences, then there is a **significant delay** in processing that precludes real-time processing.



Filtering of Long Data Sequences

- ▶ Strategy:
 1. **Segment the input** signal into fixed-size blocks prior to processing.
 2. Compute DFT-based linear filtering of each block separately.
 3. **Fit the output blocks together** in such a way that the overall output is equivalent to the linear filtering of $x(n)$ directly.
- ▶ **Main advantage:** samples of the output $y(n) = h(n) * x(n)$ will
- ▶ Goal: FIR filtering: $y(n) = x(n) * h(n)$
- ▶ Two approaches to real-time linear filtering of long inputs:
 - ▶ **Overlap-Add** Method
 - ▶ **Overlap-Save** Method
- ▶ Assumptions:
 - ▶ FIR filter $h(n)$ length = M
 - ▶ Block length = $L \gg M$

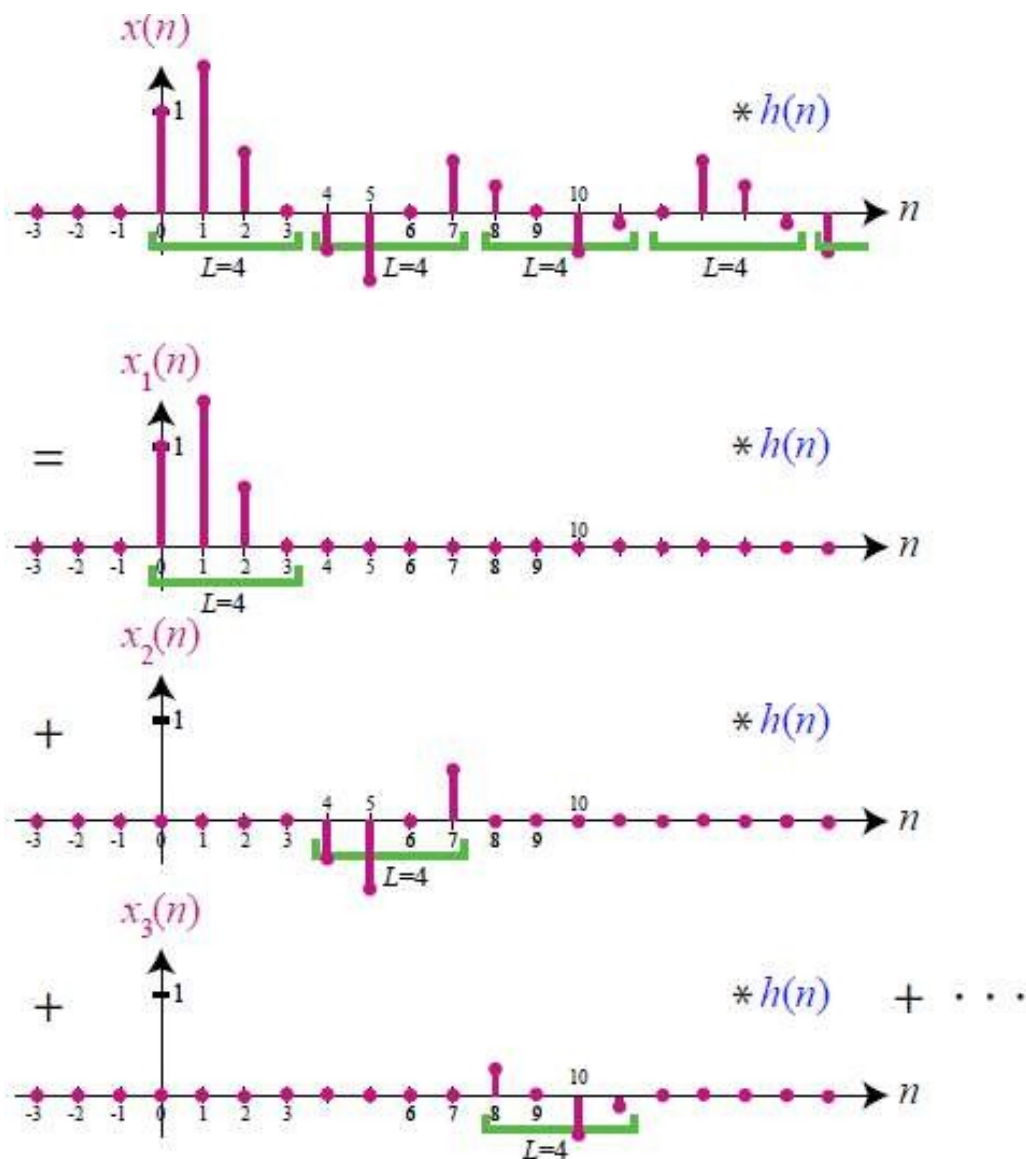
Over-lap Add

Deals with the following signal processing principles:

- ▶ The linear convolution of a discrete-time signal of length L and a discrete-time signal of length M produces a discrete-time convolved result of length $L + M - 1$.
- ▶ Additivity:

$$(x_1(n) + x_2(n)) * h(n) = x_1(n) * h(n) + x_2(n) * h(n)$$

Over-lap Add



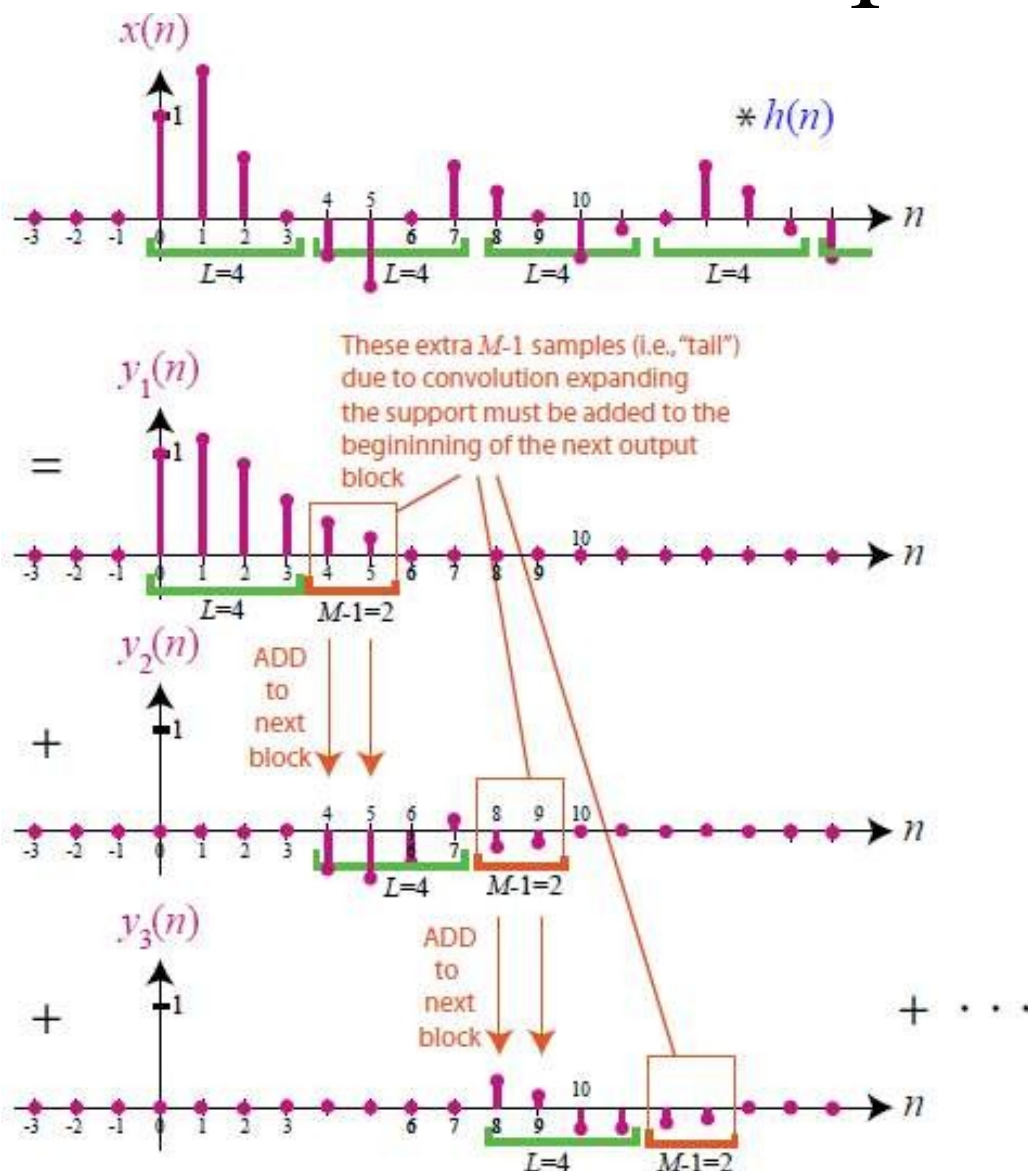
Input $x(n)$ is divided into **non-overlapping** blocks $x_m(n)$ each of length L .

Each input block $x_m(n)$ is **individually** filtered **as it is received** to produce the output block $y_m(n)$.

Over-lap Add

- ▶ makes use of the N -DFT and N -IDFT where: $N = L + M - 1$
 - ▶ Thus, zero-padding of $x(n)$ and $h(n)$ that are of length $L, M < N$ is required.
 - ▶ The actual implementation of the DFT/IDFT will use the **fast Fourier Transform** (FFT) for computational simplicity.

Over-lap Add



Output blocks $y_m(n)$ must be fitted together **appropriately** to generate:

$$y(n) = x(n) * h(n)$$

The support **overlap** amongst the $y_m(n)$ blocks must be accounted for.

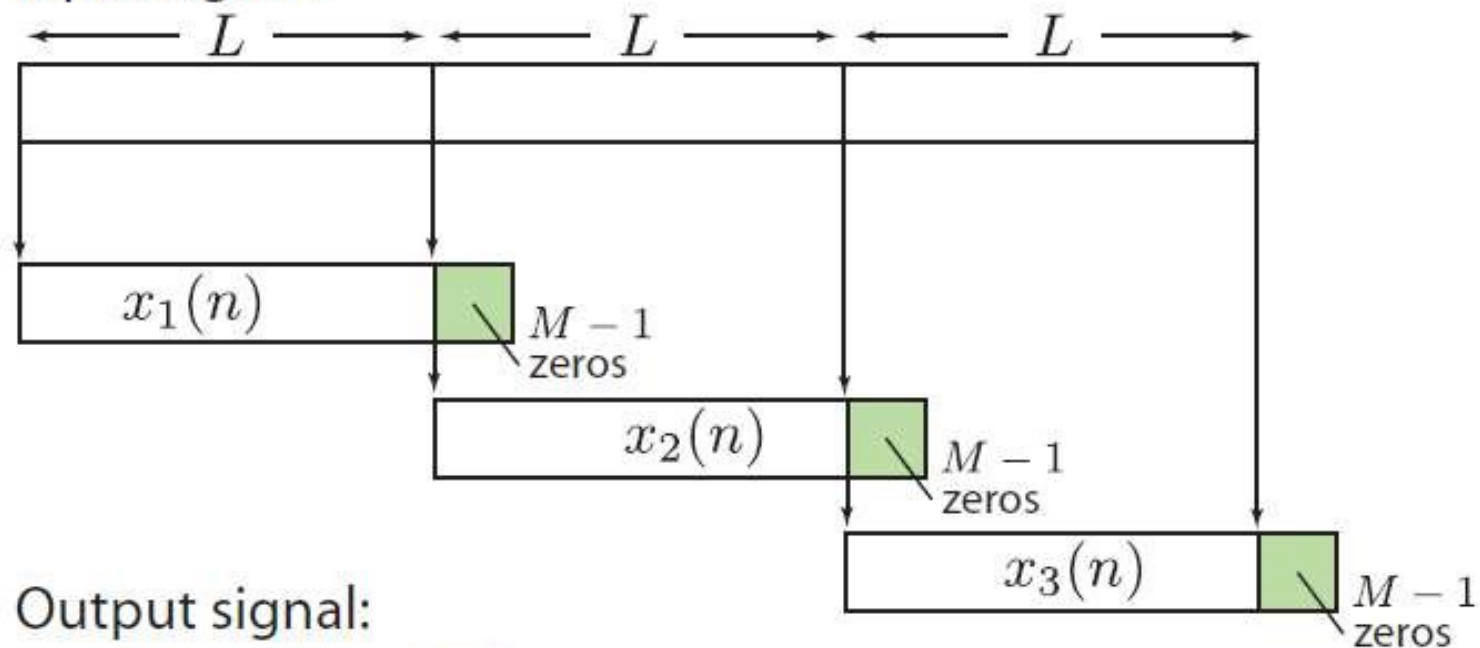
Over-lap Add

From the Additivity property, since:

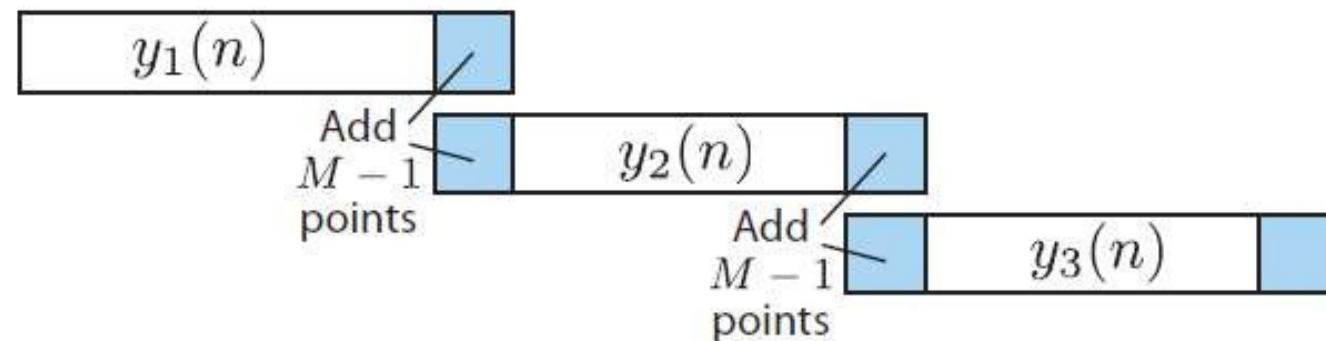
$$\begin{aligned}x(n) &= x_1(n) + x_2(n) + x_3(n) + \cdots = \sum_{m=1}^{\infty} x_m(n) \\x(n) * h(n) &= (x_1(n) + x_2(n) + x_3(n) + \cdots) * h(n) \\&= x_1(n) * h(n) + x_2(n) * h(n) + x_3(n) * h(n) + \cdots \\&= \sum_{m=1}^{\infty} x_m(n) * h(n) = \sum_{m=1}^{\infty} y_m(n)\end{aligned}$$

Over-lap Add

Input signal:



Output signal:



Over-lap Add

1. Break the input signal $x(n)$ into **non-overlapping** blocks $x_m(n)$ of length L .
2. Zero pad $h(n)$ to be of length $N = L + M - 1$.
3. Take N -DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \dots, N - 1$.
4. For each block m :
 - 4.1 Zero pad $x_m(n)$ to be of length $N = L + M - 1$.
 - 4.2 Take N -DFT of $x_m(n)$ to give $X_m(k)$, $k = 0, 1, \dots, N - 1$.
 - 4.3 Multiply: $Y_m(k) = X_m(k) \cdot H(k)$, $k = 0, 1, \dots, N - 1$.
 - 4.4 Take N -IDFT of $Y_m(k)$ to give $y_m(n)$, $n = 0, 1, \dots, N - 1$.
5. Form $y(n)$ by overlapping the **last** $M - 1$ samples of $y_m(n)$ with the **first** $M - 1$ samples of $y_{m+1}(n)$ and adding the result.

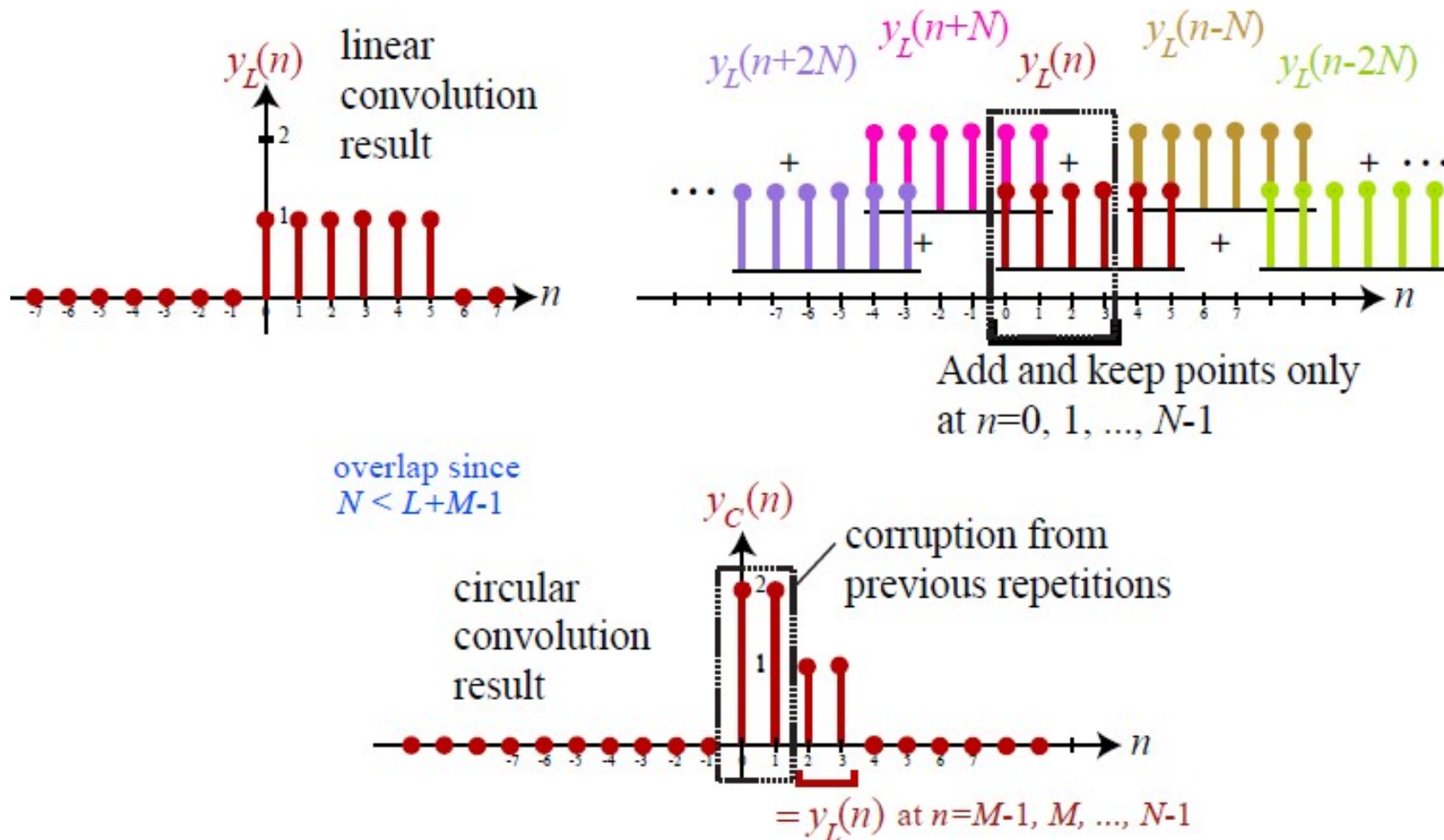
Over-lap save

Deals with the following signal processing principles:

- ▶ The $N = (L + M - 1)$ -circular convolution of a discrete-time signal of length N and a discrete-time signal of length M using an N -DFT and N -IDFT.
- ▶ Time-Domain Aliasing:

$$x_C(n) = \sum_{l=-\infty}^{\infty} \underbrace{x_L(n - lN)}_{\text{support} = M + N - 1}, \quad n = 0, 1, \dots, N - 1$$

Over-lap save



Over-lap save

- Convolution of $x_m(n)$ with support $n = 0, 1, \dots, N - 1$ and $h(n)$ with support $n = 0, 1, \dots, M - 1$ via the N -DFT will produce a result $y_{C,m}(n)$ such that:

$$y_{C,m}(n) = \begin{cases} \text{aliasing corruption} & n = 0, 1, \dots, M - 2 \\ y_{L,m}(n) & n = M - 1, M, \dots, N - 1 \end{cases}$$

where $y_{L,m}(n) = x_m(n) * h(n)$ is the desired output.

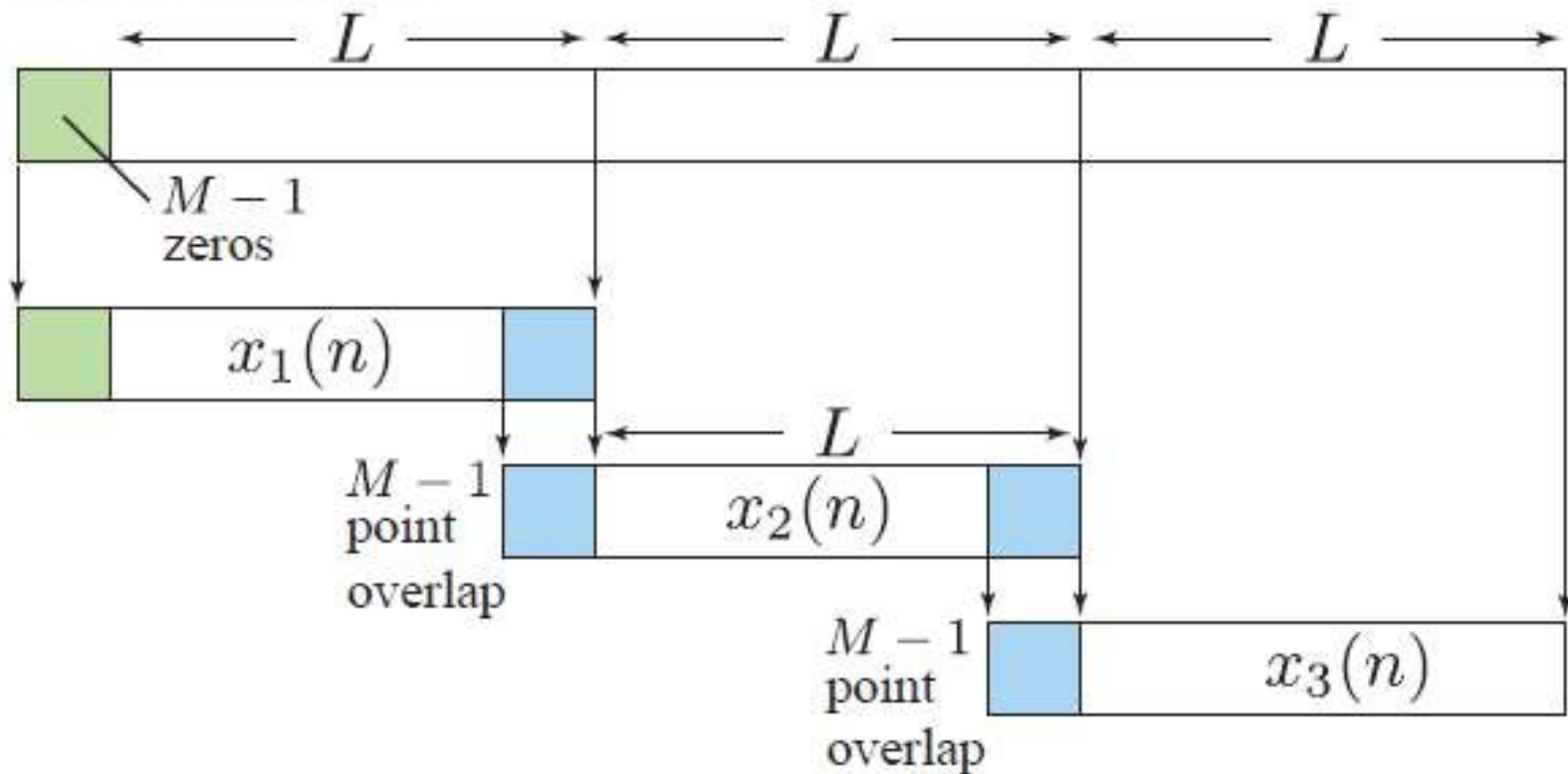
- The first $M - 1$ points of a the **current** filtered output block $y_m(n)$ must be **discarded**.
- The **previous** filtered block $y_{m-1}(n)$ must **compensate** by providing these output samples.

Over-lap save input segment stage

1. All input blocks $x_m(n)$ are of length $N = (L + M - 1)$ and contain sequential samples from $x(n)$.
2. Input block $x_m(n)$ for $m > 1$ overlaps containing the first $M - 1$ points of the previous block $x_{m-1}(n)$ to deal with aliasing corruption.
3. For $m = 1$, there is no previous block, so the first $M - 1$ points are zeros.

Over-lap save input segment stage

Input signal blocks:



Over-lap save input segment stage

$$x_1(n) = \underbrace{\{0, 0, \dots, 0\}}_{M-1 \text{ zeros}}, x(0), x(1), \dots, x(L-1)\}$$

$$x_2(n) = \underbrace{\{x(L-M+1), \dots, x(L-1)\}}_{\text{last } M-1 \text{ points from } x_1(n)}, x(L), \dots, x(2L-1)\}$$

$$x_3(n) = \underbrace{\{x(2L-M+1), \dots, x(2L-1)\}}_{\text{last } M-1 \text{ points from } x_2(n)}, x(2L), \dots, x(3L-1)\}$$

\vdots

The last $M-1$ points from the previous input block must be saved for use in the current input block.

Over-lap save filtering stage

- ▶ makes use of the N -DFT and N -IDFT where: $N = L + M - 1$
- ▶ Only a **one-time** zero-padding of $h(n)$ of length $M \ll L < N$ is required to give it support $n = 0, 1, \dots, N - 1$.
- ▶ The input blocks $x_m(n)$ are of length N to start, so no zero-padding is necessary.
- ▶ The actual implementation of the DFT/IDFT will use the **fast Fourier Transform** (FFT) for computational simplicity.

Over-lap save output blocks

$$y_{C,m}(n) = \begin{cases} \text{aliasing} & n = 0, 1, \dots, M-2 \\ y_{L,m}(n) & n = M-1, M, \dots, N-1 \end{cases}$$

where $y_{L,m}(n) = x_m(n) * h(n)$ is the desired output.

Over-lap save output blocks

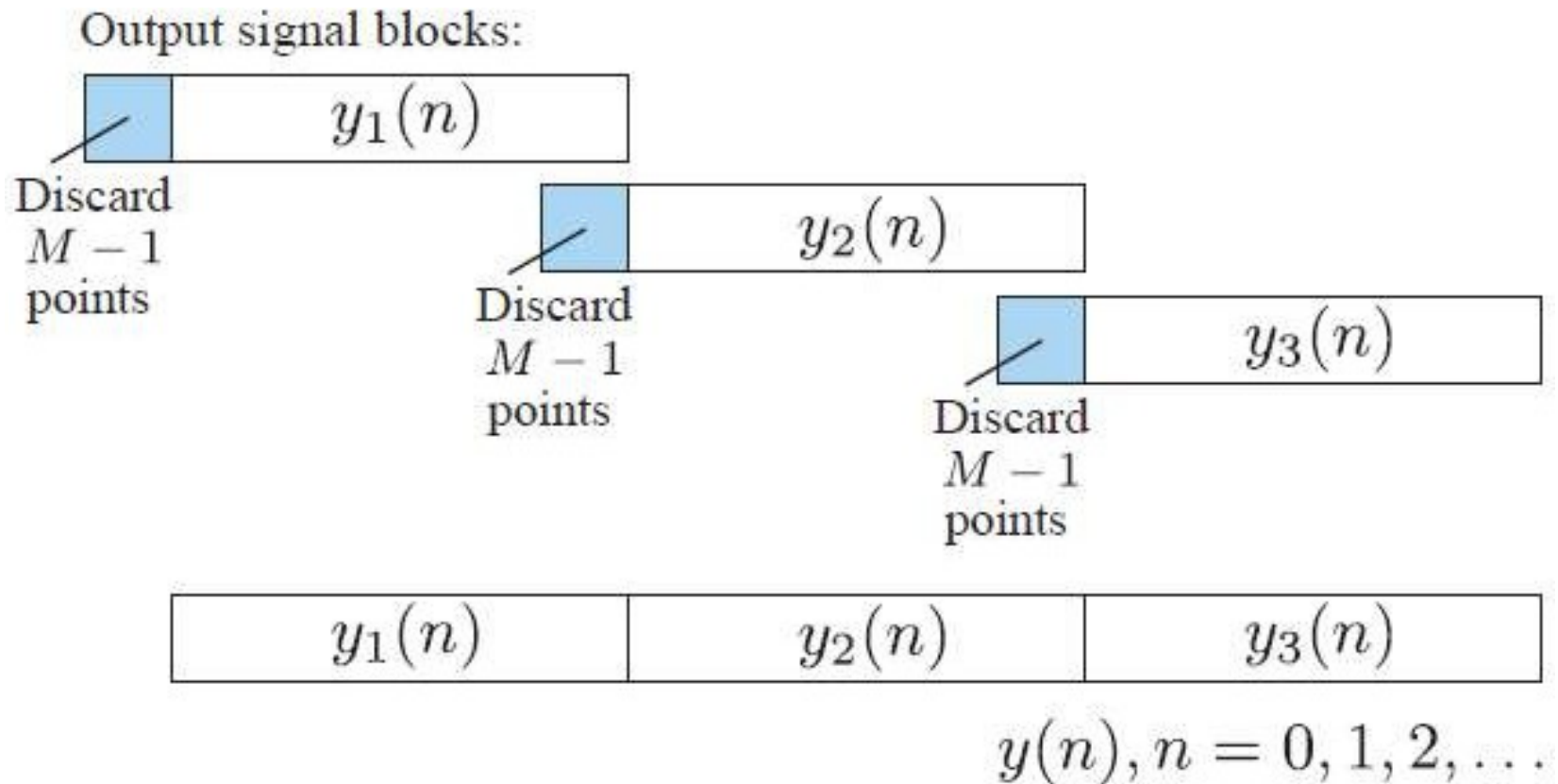
$$\begin{aligned}y_1(n) &= \underbrace{\{y_1(0), y_1(1), \dots, y_1(M-2)\}}_{M-1 \text{ points corrupted from aliasing}}, y(0), \dots, y(L-1)\} \\y_2(n) &= \underbrace{\{y_2(0), y_2(1), \dots, y_2(M-2)\}}_{M-1 \text{ points corrupted from aliasing}}, y(L), \dots, y(2L-1)\} \\y_3(n) &= \underbrace{\{y_3(0), y_3(1), \dots, y_3(M-2)\}}_{M-1 \text{ points corrupted from aliasing}}, y(2L), \dots, y(3L-1)\}\end{aligned}$$

where $y(n) = x(n) * h(n)$ is the desired output.

The first $M - 1$ points of each output block are discarded.

The remaining L points of each output block are **appended** to form $y(n)$.

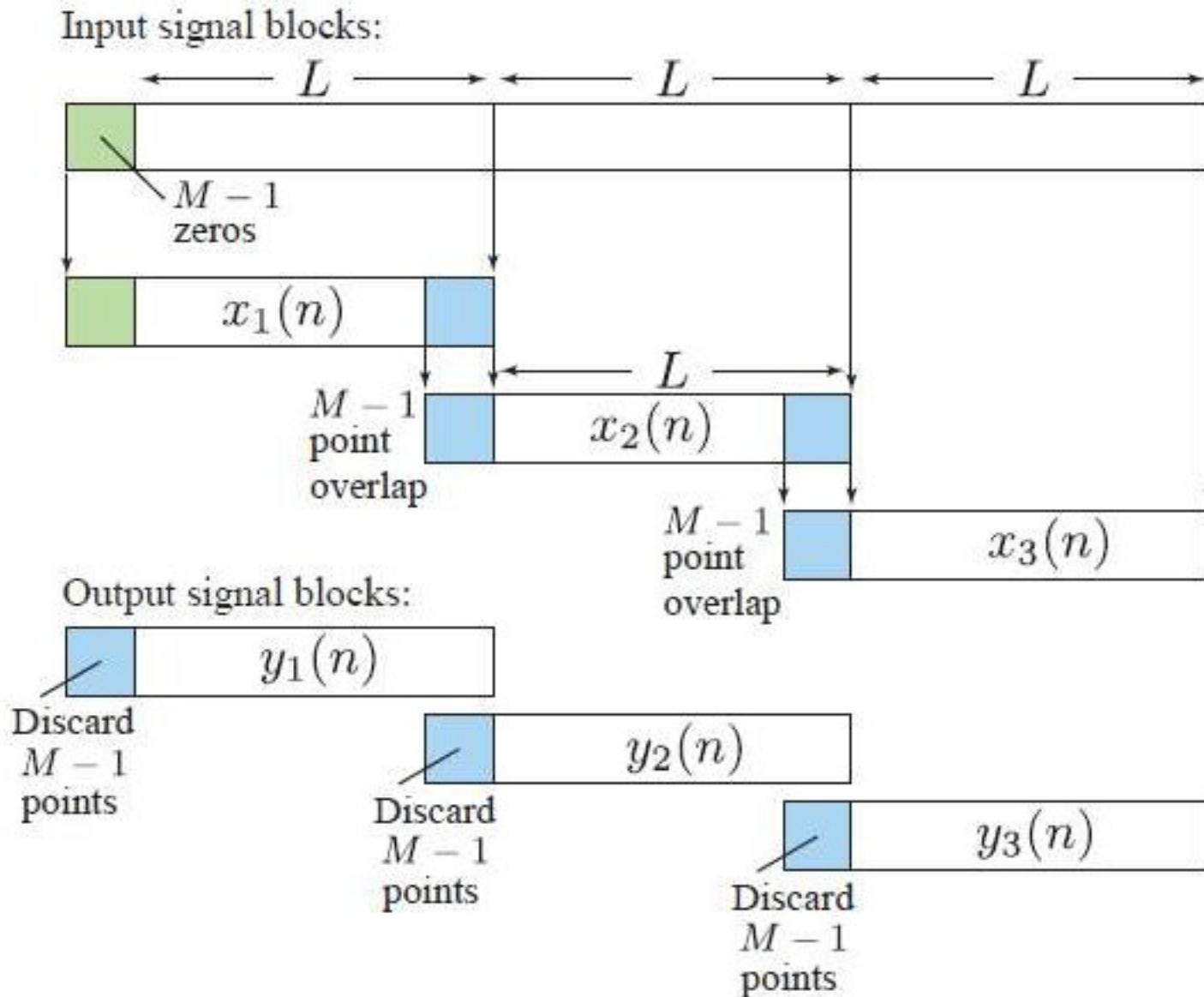
Over-lap save output blocks



Over-lap save

1. Insert $M - 1$ zeros at the beginning of the input sequence $x(n)$.
2. Break the padded input signal into **overlapping** blocks $x_m(n)$ of length $N = L + M - 1$ where the overlap length is $M - 1$.
3. Zero pad $h(n)$ to be of length $N = L + M - 1$.
4. Take N -DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \dots, N - 1$.
5. For each block m :
 - 5.1 Take N -DFT of $x_m(n)$ to give $X_m(k)$, $k = 0, 1, \dots, N - 1$.
 - 5.2 Multiply: $Y_m(k) = X_m(k) \cdot H(k)$, $k = 0, 1, \dots, N - 1$.
 - 5.3 Take N -IDFT of $Y_m(k)$ to give $y_m(n)$, $n = 0, 1, \dots, N - 1$.
 - 5.4 **Discard** the first $M - 1$ points of each output block $y_m(n)$.
6. Form $y(n)$ by appending the remaining (i.e., **last**) L samples of each block $y_m(n)$.

Over-lap save



Relationships between CTFT, DTFT, & DFT

- The N -point DFT provides a unique representation of the N -samples of a finite duration sequence.
- The DFT provides samples of the DTFT of the sequence at a set of equally spaced frequencies.
- Suppose that we are given a continuous-time signal $x_c(t)$ with Fourier transform $X_c(j\Omega)$.
- Its related discrete signal $x[n] = x_c(nT)$ has the DTFT given by

$$X(e^{j\Omega T}) = \sum_n x_c(nT) e^{-j\Omega T n} = \frac{1}{T} \sum_m X_c \left(j \left(\Omega - \frac{2\pi}{T} m \right) \right)$$

- Since $\omega = \Omega T$, the N -point DFT $X[k]$ is obtained by sampling $X(e^{j\omega})$ at $\omega_k = \frac{2\pi}{N} k$ (or, alternatively, by sampling $X(e^{j\Omega T})$ at $\Omega_k = \frac{2\pi}{TN} k$). Formally,

$$X[k] = \frac{1}{T} \sum_m X_c \left(j \left(\frac{2\pi}{NT} k - \frac{2\pi}{T} m \right) \right), \quad k = 0, 1, \dots, N-1$$

Relationships between CTFT, DTFT, & DFT

- Sampling the DTFT of $x[n]$ is equivalent to the periodic repetition of $x[n]$ with period N or equivalently of $x_c(nT)$ with period NT . The result is

$$\tilde{x}[n] = \sum_k x_c(nT - NTk)$$

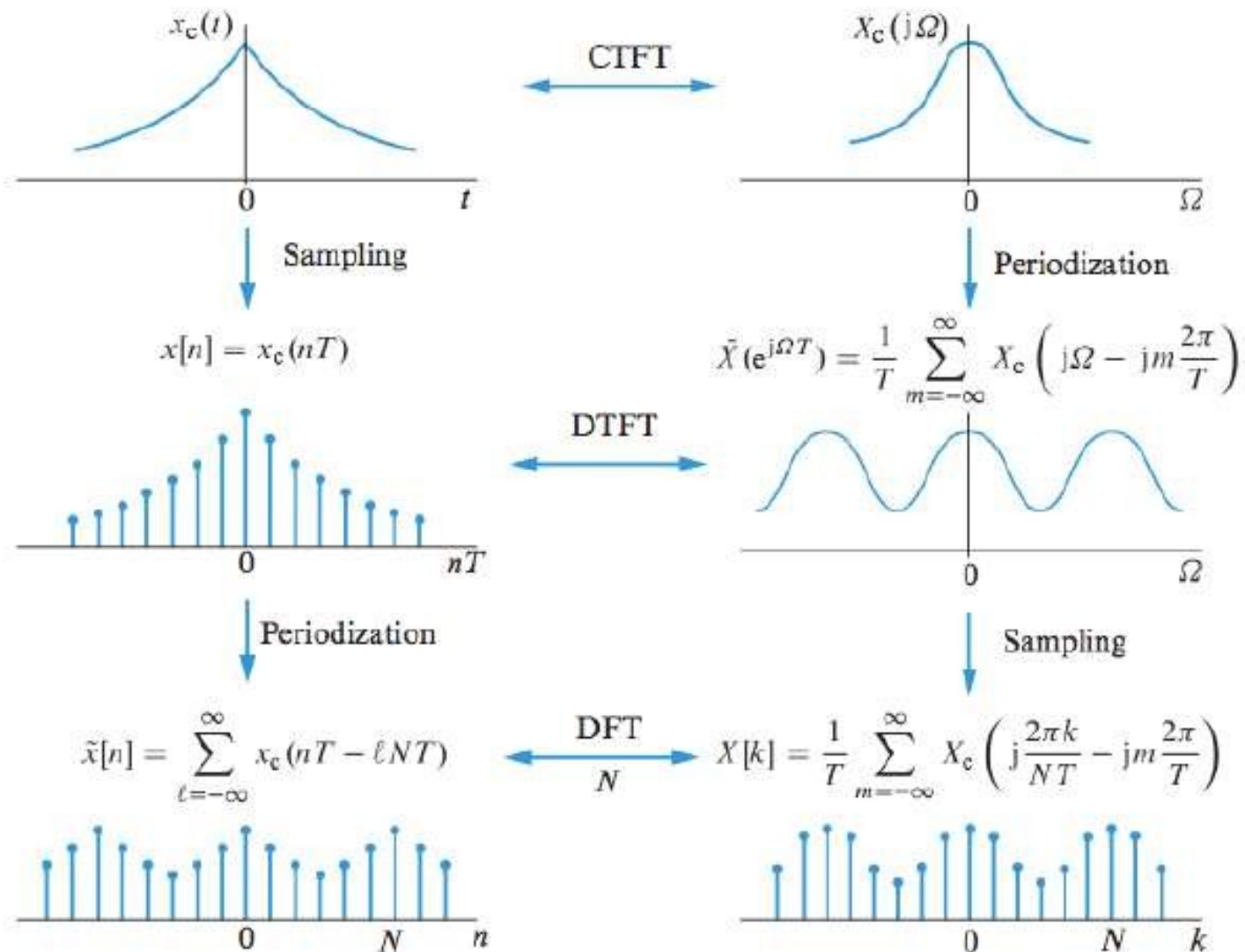
- Therefore, we have the following N -point DFT pair

$$\sum_k x_c(nT - NTk) \Longleftrightarrow \frac{1}{T} \sum_k X_c \left(j \left(\frac{2\pi}{NT} k - \frac{2\pi}{T} m \right) \right)$$

where $0 \leq n \leq N - 1$ and $0 \leq k \leq N - 1$.

- The above relation reveals a frequency-domain aliasing caused by time-domain sampling and a time-domain aliasing caused by frequency-domain sampling (which, in turn, explains the inherent periodicity of the DFT).

Relationships between CTFT, DTFT, & DFT



Fast Fourier Transform

Discrete Fourier Transform

- The DFT pair was given as

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j(2\pi/N)kn} \quad x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j(2\pi/N)kn}$$

- Baseline for computational complexity:

- Each DFT coefficient requires
 - N complex multiplications
 - N-1 complex additions
- All N DFT coefficients require
 - N^2 complex multiplications
 - $N(N-1)$ complex additions

- Complexity in terms of real operations


- $4N^2$ real multiplications
- $2N(N-1)$ real additions

- Most fast methods are based on symmetry properties

- Conjugate symmetry $e^{-j(2\pi/N)k(N-n)} = e^{-j(2\pi/N)kN} e^{-j(2\pi/N)k(-n)} = e^{j(2\pi/N)kn}$
- Periodicity in n and k $e^{-j(2\pi/N)kn} = e^{-j(2\pi/N)k(n+N)} = e^{j(2\pi/N)(k+N)n}$

Direct computation of DFT

- The DFT of a finite-length sequence of length N

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad k = 0, 1, \dots, N-1$$


- Direct computation: N^2 complex multiplications and $N(N-1)$ complex additions

- Compute and store (only over one period)

$$W_N^k = e^{-j(2\pi/N)k}$$

$$= \cos(2\pi k / N) + j \sin(2\pi k / N), \quad k = 0, 1, \dots, N-1$$

- Compute the DFT using stored W_N^k and input $x[n]$

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad k = 0, 1, \dots, N-1$$

W_N^k and $x[n]$ may be complex

Direct computation of DFT

- For each k

$$X[k] = \sum_{n=0}^{N-1} [(\operatorname{Re}\{x[n]\} \operatorname{Re}\{W_N^{kn}\} - (\operatorname{Im}\{x[n]\} \operatorname{Im}\{W_N^{kn}\})) \\ + j(\operatorname{Re}\{x[n]\} \operatorname{Im}\{W_N^{kn}\} + \operatorname{Im}\{x[n]\} \operatorname{Re}\{W_N^{kn}\})], \quad k = 0, 1, \dots, N-1$$

- Therefore, for each value of k , the direct computation of $X[k]$ requires $4N$ real multiplications and $(4N-2)$ real additions.
- The direct computation of the DFT requires $4N^2$ real multiplications and $N(4N-2)$ real additions.
- The efficiency can be improved by exploiting the symmetry and periodicity properties of W_N^{kn}

- Complex conjugate symmetry

$$W_N^{k[N-n]} = W_N^{-kn} = (W_N^{kn})^* = \text{Re}\{W_N^{kn}\} - j \text{Im}\{W_N^{kn}\}$$

- Periodicity in n and k

$$W_N^{kn} = W_N^{k(n+N)} = W_N^{(k+N)n}$$

- For example

$$\begin{aligned} & \text{Re}\{x[n]\} \text{Re}\{W_N^{kn}\} + \text{Re}\{x[N-n]\} \text{Re}\{W_N^{k[N-n]}\} \\ &= (\text{Re}\{x[n]\} + \text{Re}\{x[N-n]\}) \text{Re}\{W_N^{kn}\} \end{aligned}$$

- The number of multiplications is reduced by a factor of 2.

FFT

- Cooley and Tukey (1965) published an algorithm for the computation of the DFT that is applicable when N is a composite number, i.e., the product of two or more integers. Later, it resulted in a number of highly efficient computational algorithms.
- The entire set of such algorithms are called the fast Fourier transform, FFT.
- FFT decomposes the computation of the DFT of a sequence of length N into successively smaller DFTs.

Decimation-In-Time FFT Algorithms

- Makes use of both symmetry and periodicity
- Consider special case of N an integer power of 2
- Separate $x[n]$ into two sequence of length $N/2$
 - Even indexed samples in the first sequence
 - Odd indexed samples in the other sequence

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j(2\pi/N)kn} = \sum_{n \text{ even}}^{N-1} x[n] e^{-j(2\pi/N)kn} + \sum_{n \text{ odd}}^{N-1} x[n] e^{-j(2\pi/N)kn}$$

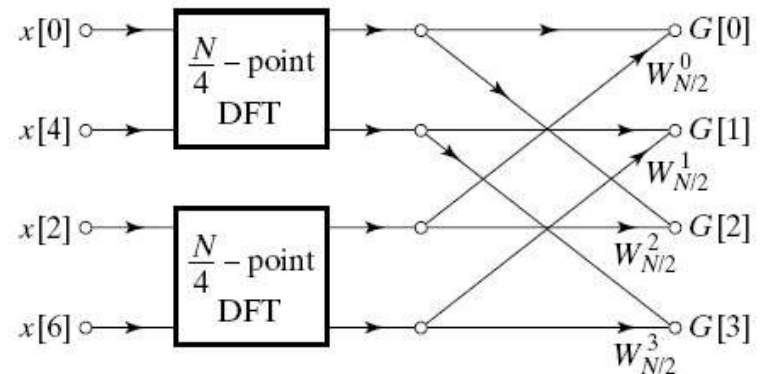
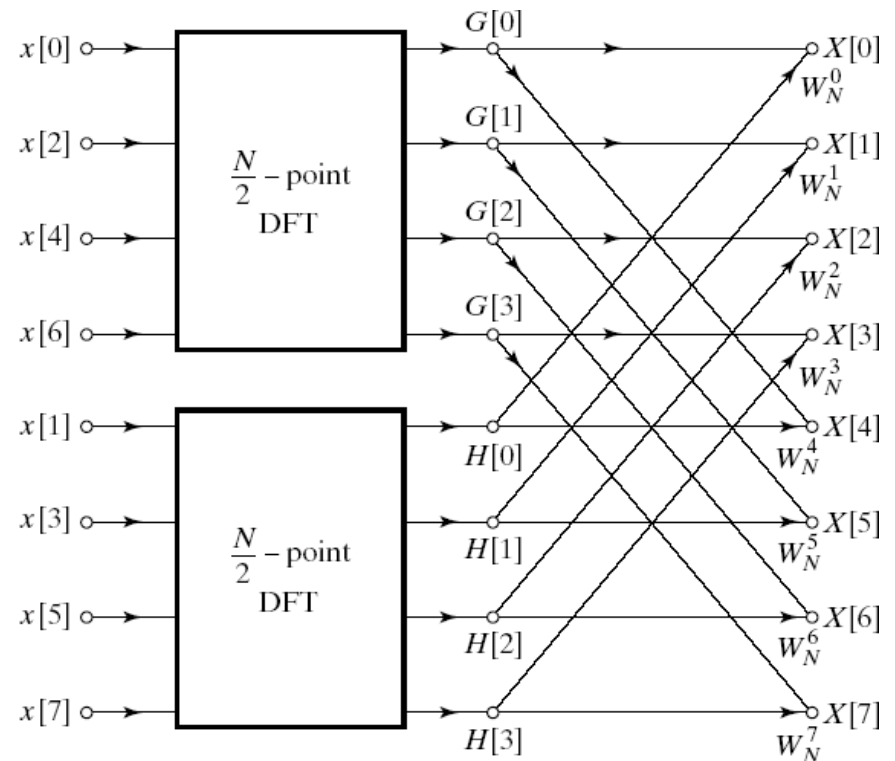
- Substitute variables $n=2r$ for n even and $n=2r+1$ for odd

$$\begin{aligned} X[k] &= \sum_{r=0}^{N/2-1} x[2r] W_N^{2rk} + \sum_{r=0}^{N/2-1} x[2r+1] W_N^{(2r+1)k} \\ &= \sum_{r=0}^{N/2-1} x[2r] W_{N/2}^{rk} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_{N/2}^{rk} \\ &= G[k] + W_N^k H[k] \end{aligned}$$

- $G[k]$ and $H[k]$ are the $N/2$ -point DFT's of each subsequence

Decimation In Time

- 8-point DFT example using decimation-in-time
- Two $N/2$ -point DFTs
 - $2(N/2)^2$ complex multiplications
 - $2(N/2)^2$ complex additions
- Combining the DFT outputs
 - N complex multiplications
 - N complex additions
- Total complexity
 - $N^2/2 + N$ complex multiplications
 - $N^2/2 + N$ complex additions
 - More efficient than direct DFT
- Repeat same process
 - Divide $N/2$ -point DFTs into
 - Two $N/4$ -point DFTs
 - Combine outputs



Decimation In Time Cont'd

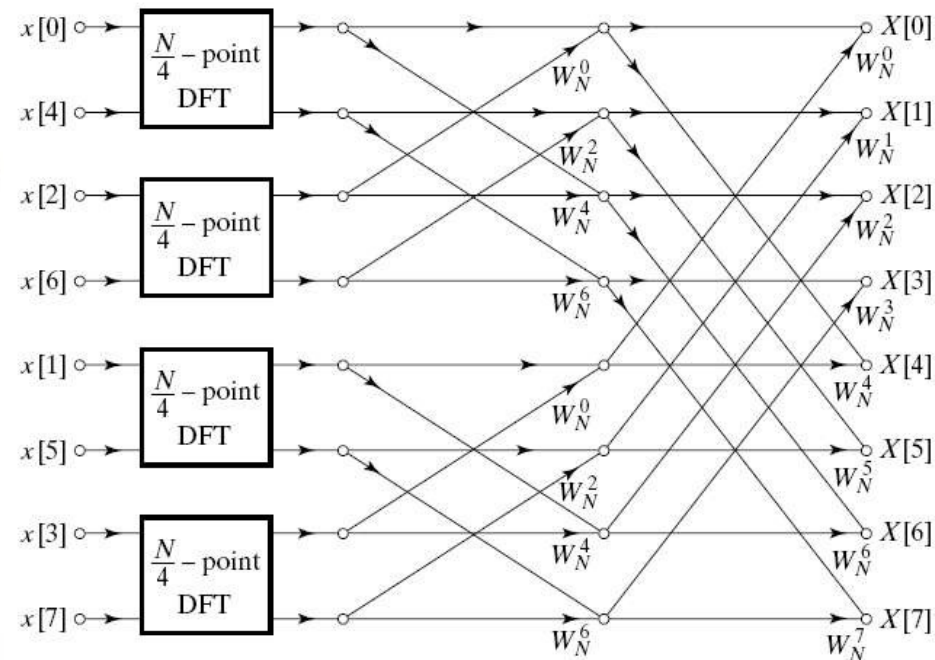
- After two steps of decimation in time

Further break down

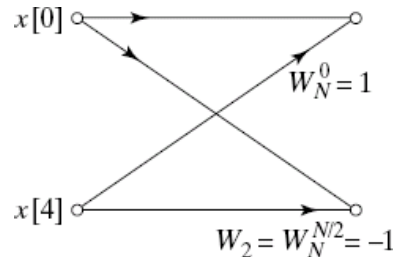
$$G[k] = \sum_{r=0}^{(N/2)-1} g[r] W_{N/2}^{rk} = \sum_{l=0}^{(N/4)-1} g[2l] W_{N/2}^{2lk} + \sum_{l=0}^{(N/4)-1} g[2l+1] W_{N/2}^{(2l+1)k}$$

$$= \sum_{l=0}^{(N/4)-1} g[2l] W_{N/4}^{lk} + W_{N/2}^k \sum_{l=0}^{(N/4)-1} g[2l+1] W_{N/4}^{lk}$$

$$H[k] = \sum_{l=0}^{(N/4)-1} h[2l] W_{N/4}^{lk} + W_{N/2}^k \sum_{l=0}^{(N/4)-1} h[2l+1] W_{N/4}^{lk}$$

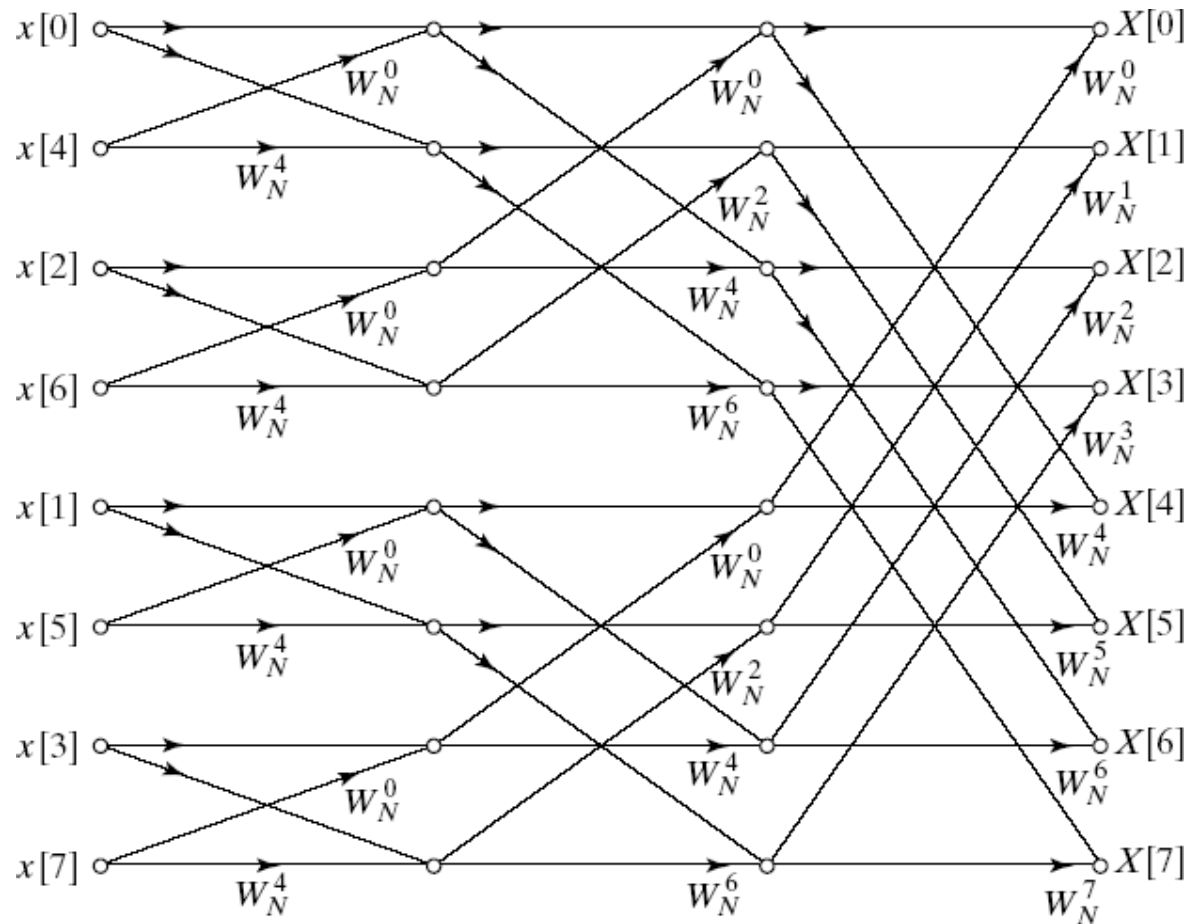


- Repeat until we're left with two-point DFT's



Decimation-In-Time FFT Algorithm

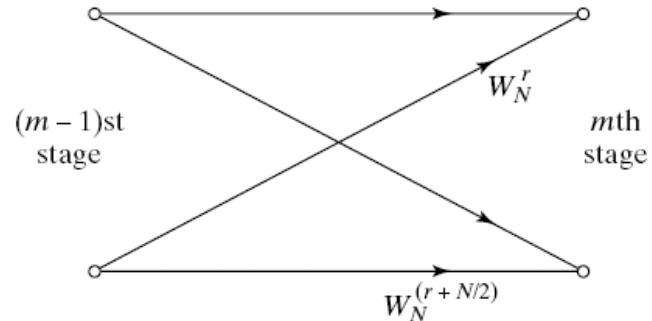
- Final flow graph for 8-point decimation in time



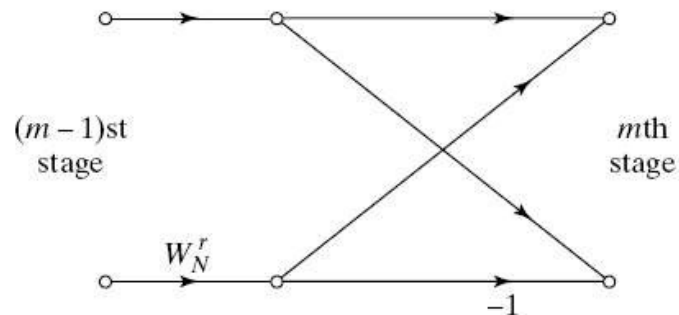
- Complexity:
 - $N \log_2 N$ complex multiplications and additions

Butterfly Computation

- Flow graph constitutes of butterflies



- We can implement each butterfly with one multiplication



- Final complexity for decimation-in-time FFT
 - $(N/2)\log_2 N$ complex multiplications and additions

In-Place Computation

- Decimation-in-time flow graphs require two sets of registers
 - Input and output for each stage
- Note the arrangement of the input indices
 - Bit reversed indexing

$$X_0[0] = x[0] \leftrightarrow X_0[000] = x[000]$$

$$X_0[1] = x[4] \leftrightarrow X_0[001] = x[100]$$

$$X_0[2] = x[2] \leftrightarrow X_0[010] = x[010]$$

$$X_0[3] = x[6] \leftrightarrow X_0[011] = x[110]$$

$$X_0[4] = x[1] \leftrightarrow X_0[100] = x[001]$$

$$X_0[5] = x[5] \leftrightarrow X_0[101] = x[101]$$

$$X_0[6] = x[3] \leftrightarrow X_0[110] = x[011]$$

$$X_0[7] = x[7] \leftrightarrow X_0[111] = x[111]$$

Decimation-In-Frequency FFT Algorithm

- The DFT equation

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk}$$

- Split the DFT equation into even and odd frequency indexes

$$X[2r] = \sum_{n=0}^{N-1} x[n] W_N^{n2r} = \sum_{n=0}^{N/2-1} x[n] W_N^{n2r} + \sum_{n=N/2}^{N-1} x[n] W_N^{n2r}$$

- Substitute variables to get

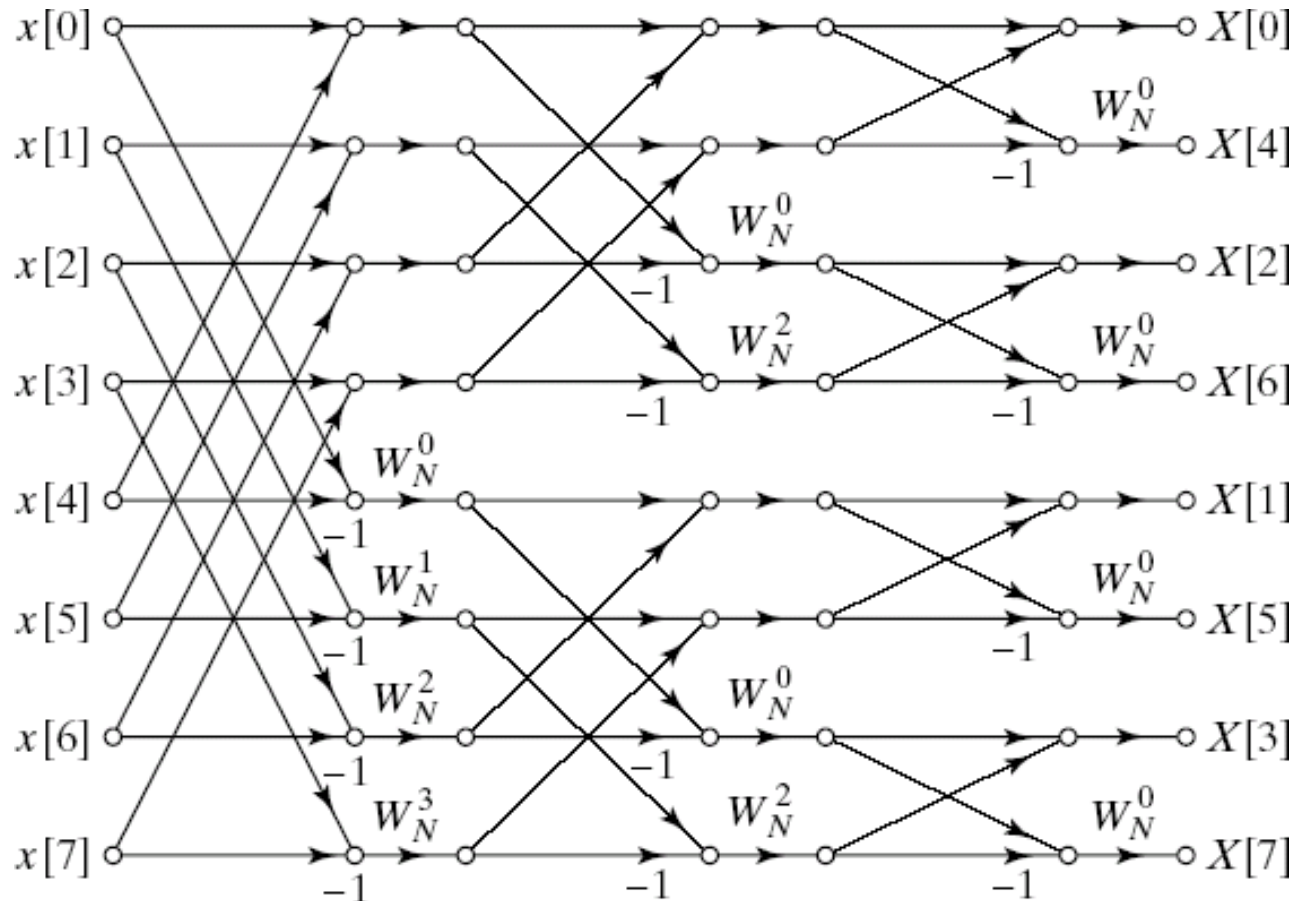
$$X[2r] = \sum_{n=0}^{N/2-1} x[n] W_N^{n2r} + \sum_{n=0}^{N/2-1} x[n + N/2] W_N^{(n+N/2)2r} = \sum_{n=0}^{N/2-1} (x[n] + x[n + N/2]) W_{N/2}^{nr}$$

- Similarly for odd-numbered frequencies

$$X[2r + 1] = \sum_{n=0}^{N/2-1} (x[n] - x[n + N/2]) W_{N/2}^{n(2r+1)}$$

Decimation-In-Frequency FFT Algorithm

- Final flow graph for 8-point decimation in frequency

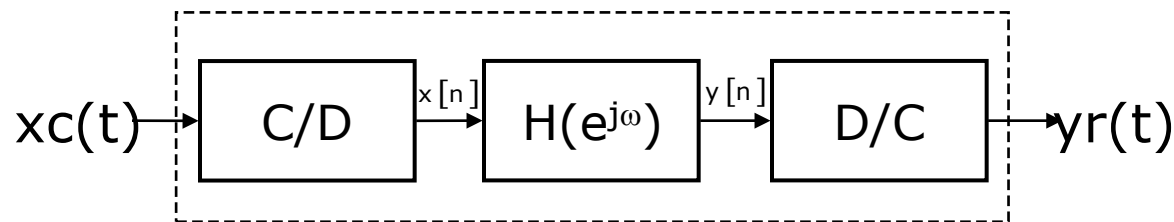


UNIT-3

IIR filters

Filter Design Techniques

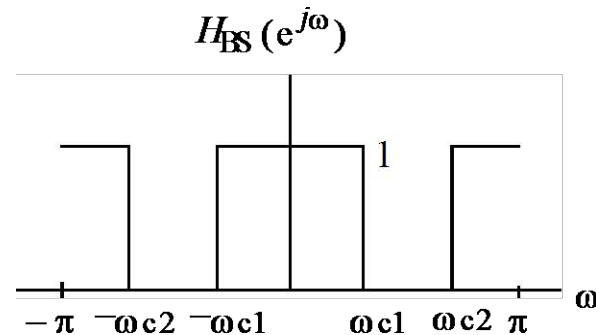
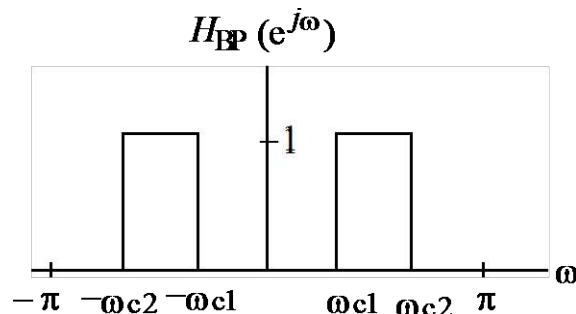
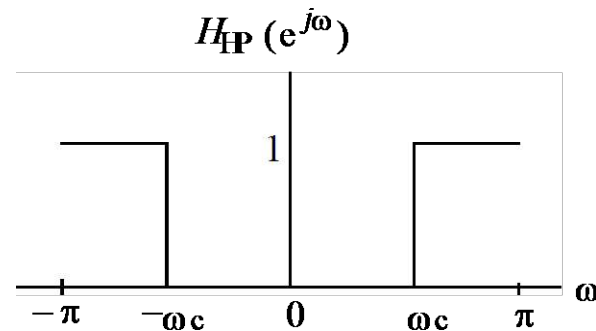
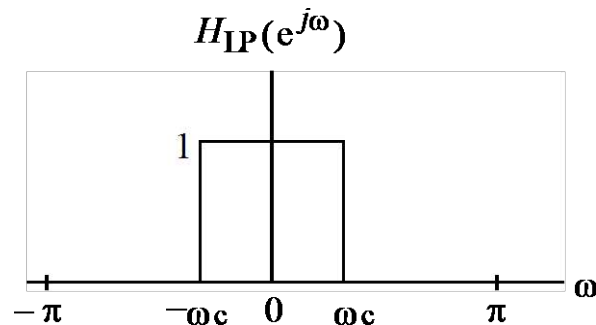
- Any discrete-time system that modifies certain frequencies
- Frequency-selective filters pass only certain frequencies
- Filter Design Steps
 - Specification
 - Problem or application specific
 - Approximation of specification with a discrete-time system
 - Our focus is to go from spec to discrete-time system
 - Implementation
 - Realization of discrete-time systems depends on target technology
- We already studied the use of discrete-time systems to implement a continuous-time system
 - If our specifications are given in continuous time we can use



$$H(e^{j\omega}) = H_c(j\omega / T) \quad \left| \omega \right| < \pi$$

Digital Filter Specifications

- Only the magnitude approximation problem
- Four basic types of ideal filters with magnitude responses as shown below (Piecewise flat)



Digital Filter Specifications

- These filters are unrealisable because (one of the following is sufficient)
 - their impulse responses infinitely long non-causal
 - Their amplitude responses cannot be equal to a constant over a band of frequencies

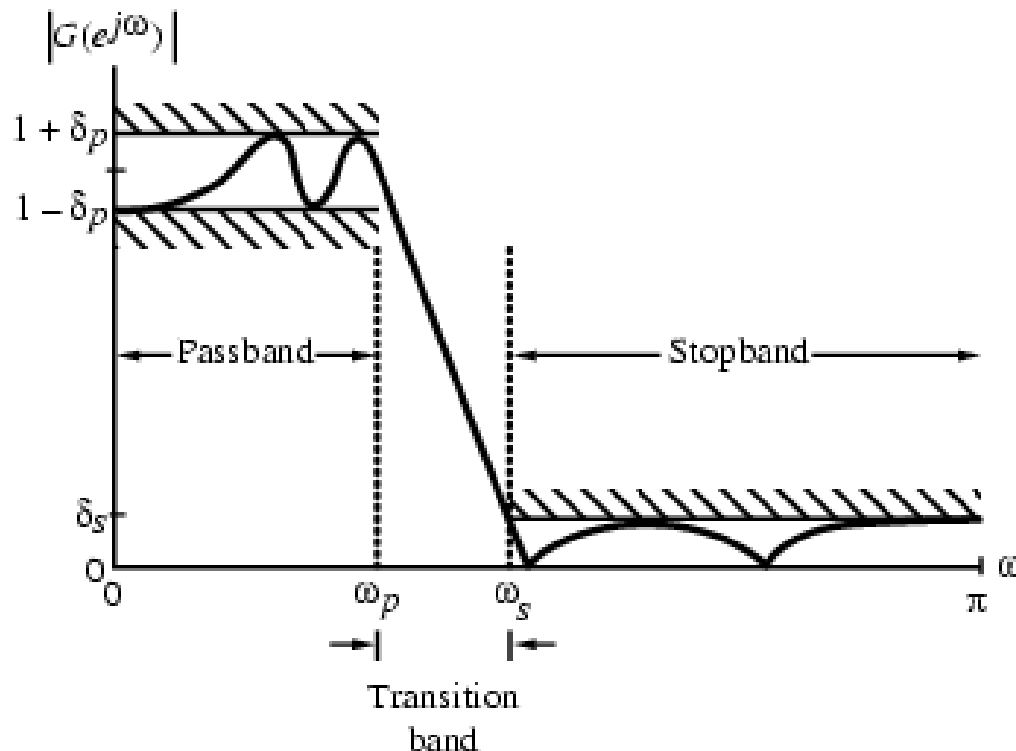
Another perspective that provides some understanding can be obtained by looking at the ideal amplitude squared.

Digital Filter Specifications

- The realisable squared amplitude response transfer function (and its differential) is continuous in
- Such functions ω
 - if IIR can be infinite at point but around that point cannot be zero.
 - if FIR cannot be infinite anywhere.
- Hence previous differential of ideal response is unrealisable

Digital Filter Specifications

- For example the magnitude response of a digital lowpass filter may be given as indicated below



Digital Filter Specifications

- In the **passband** $0 \leq \omega \leq \omega_p$ we require that $|G(e^{j\omega})| \cong 1$ with a deviation $\pm \delta_p$

$$1 - \delta_p \leq |G(e^{j\omega})| \leq 1 + \delta_p, \quad |\omega| \leq \omega_p$$

- In the **stopband** $\omega_s \leq \omega \leq \pi$ we require that $|G(e^{j\omega})| \cong 0$ with a deviation δ_s

$$|G(e^{j\omega})| \leq \delta_s, \quad \omega_s \leq |\omega| \leq \pi$$

Digital Filter Specifications

Filter specification parameters

- ω_p - **passband edge frequency**
- ω_s - **stopband edge frequency**
- δ_p - **peak ripple value in the passband**
- δ_s - **peak ripple value in the stopband**

Digital Filter Specifications

- Practical specifications are often given in terms of **loss function (in dB)**

- $$G(\omega) = -20 \log_{10} |G(e^{j\omega})|$$

- **Peak passband ripple**

$$\alpha_p = -20 \log_{10} (1 - \delta_p) \text{ dB}$$

- **Minimum stopband attenuation**

$$\alpha_s = -20 \log_{10} (\delta_s) \text{ dB}$$

Digital Filter Specifications

- In practice, passband edge frequency F_p and stopband edge frequency F_s are specified in Hz
- For digital filter design, normalized bandedge frequencies need to be computed from specifications in Hz using

$$\omega_p = \frac{\Omega_p}{F_T} = \frac{2\pi F_p}{F_T} = 2\pi F_p T$$

$$\omega_s = \frac{\Omega_s}{F_T} = \frac{2\pi F_s}{F_T} = 2\pi F_s T$$

Digital Filter Specifications

- Example - Let $F_p = 7$ kHz, $F_s = 3$ kHz, and $F_T = 25$ kHz
- Then

$$\omega_p = \frac{2\pi (7 \times 10^3)}{25 \times 10^3} = 0.56 \pi$$

$$\omega_s = \frac{2\pi (3 \times 10^3)}{25 \times 10^3} = 0.24 \pi$$

IIR Digital Filter Design

Standard approach

- (1) Convert the digital filter specifications into an analogue prototype lowpass filter specifications
- (2) Determine the analogue lowpass filter transfer function $H_a(s)$
- (3) Transform $H_a(s)$ by replacing the complex variable to the digital transfer function

$$G(z)$$

IIR Digital Filter Design

- This approach has been widely used for the following reasons:
 - (1) Analogue approximation techniques are highly advanced
 - (2) They usually yield closed-form solutions
 - (3) Extensive tables are available for analogue filter design
 - (4) Very often applications require digital simulation of analogue systems

IIR Digital Filter Design

- Let an analogue transfer function be

$$H_a(s) = \frac{P_a(s)}{D_a(s)}$$

where the subscript “ a ” indicates the analogue domain

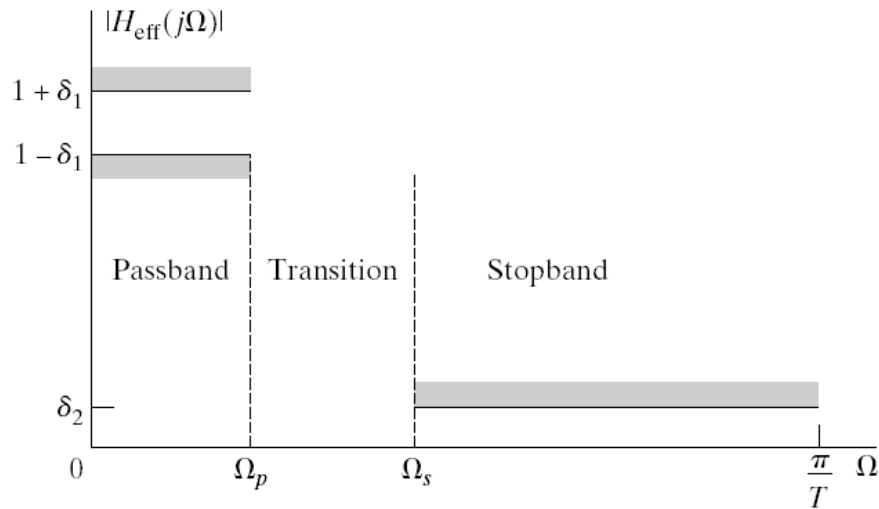
- A digital transfer function derived from this is denoted as

$$G(z) = \frac{P(z)}{D(z)}$$

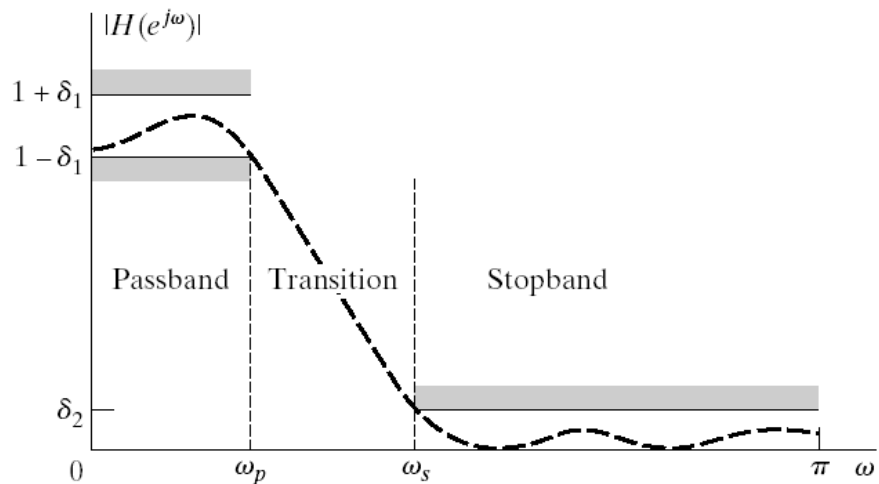
IIR Digital Filter Design

- Basic idea behind the conversion of $H_a(s)$ into $G(z)$ is to apply a mapping from the s -domain to the z -domain so that essential properties of the analogue frequency response are preserved
- Thus mapping function should be such that
 - Imaginary $(j\Omega)$ axis in the s -plane be mapped onto the unit circle of the z -plane
 - A stable analogue transfer function be mapped into a stable digital transfer function

Specification for effective frequency response of a continuous-time lowpass filter and its corresponding specifications for discrete-time system.



(a)



(b)

d_p or d_1 passband ripple

d_s or d_2 stopband ripple

ω_p , ω_p passband edge frequency

ω_s , ω_s stopband edge frequency

e^2 passband ripple parameter

$$1 - d_p = 1/\sqrt{1 + e^2}$$

BW bandwidth = $\omega_u - \omega_l$

ω_c 3-dB cutoff frequency

ω_u , ω_l upper and lower 3-dB cutoff frequencies

$\Delta\omega$ transition band = $|\omega_p - \omega_s|$

A_p passband ripple in dB

$$= \pm 20\log_{10}(1 \pm d_p)$$

A_s stopband attenuation in dB

$$= -20\log_{10}(d_s)$$

Design of Discrete-Time IIR Filters

- From Analog (Continuous-Time) Filters
 - Approximation of Derivatives
 - Impulse Invariance
 - the Bilinear Transformation

Reasons of Design of Discrete-Time IIR Filters from Continuous-Time Filters

- The art of continuous-time IIR filter design is highly advanced and, since useful results can be achieved, it is advantageous to use the design procedures already developed for continuous-time filters.
- Many useful continuous-time IIR design methods have relatively simple closed-form design formulas. Therefore, discrete-time IIR filter design methods based on such standard continuous-time design formulas are rather simple to carry out.
- The standard approximation methods that work well for continuous-time IIR filters do not lead to simple closed-form design formulas when these methods are applied directly to the discrete-time IIR case.

Characteristics of Commonly Used Analog Filters

- Butterworth Filter
- Chebyshev Filter
 - Chebyshev Type I
 - Chebyshev Type II or Inverse Chebyshev Filter

Butterworth Filter

- Lowpass Butterworth filters are all-pole filters characterized by the magnitude-squared frequency response

$$|H(W)|^2 = 1/[1 + (W/W_c)^{2N}] = 1/[1 + e^2(W/W_p)^{2N}]$$

where N is the order of the filter, W_c is its -3 -dB frequency (cutoff frequency), W_p is the bandpass edge frequency, and $1/(1 + e^2)$ is the band-edge value of $|H(W)|^2$.

- At $W = W_s$ (where W_s is the stopband edge frequency) we have

$$1/[1 + e^2(W_s/W_p)^{2N}] = d_2^2$$

and

$$N = (1/2)\log_{10}[(1/d_2^2) - 1]/\log_{10}(W_s/W_c) = \log_{10}(d/e)/\log_{10}(W_s/W_p)$$

where $d_2 = 1/\sqrt{1 + d_2^2}$.

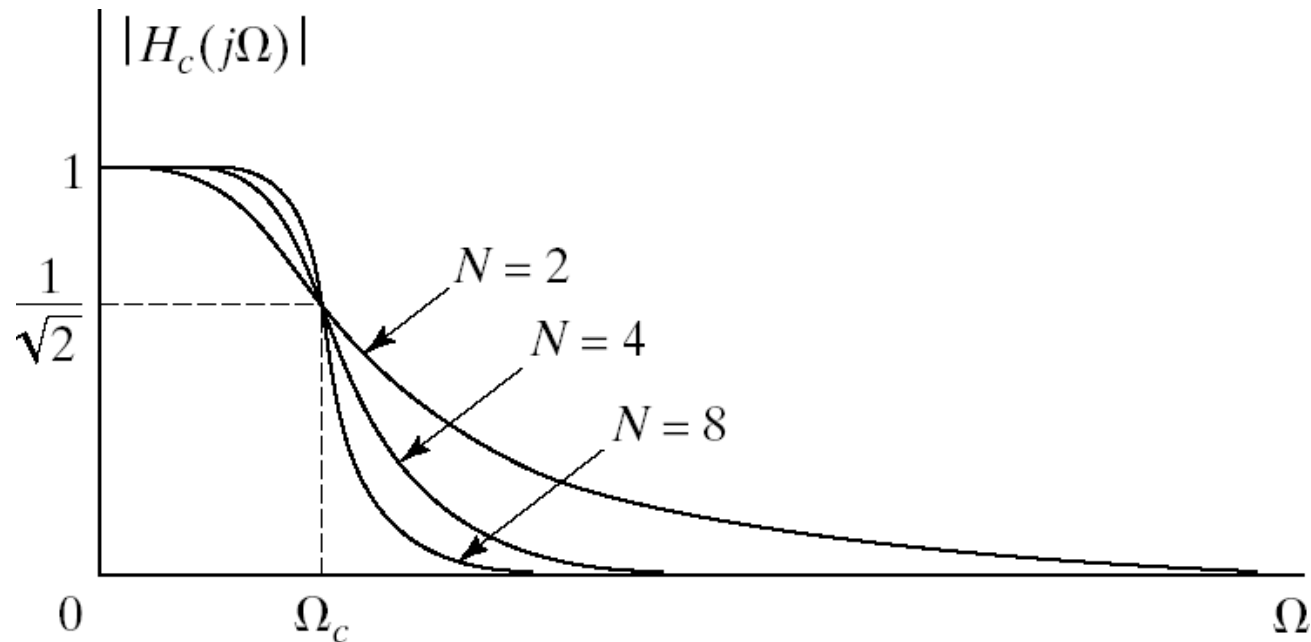
- Thus the Butterworth filter is completely characterized by the parameters N , d_2 , e , and the ratio W_s/W_p .

Butterworth Lowpass Filters

- Passband is designed to be maximally flat
- The magnitude-squared function is of the form

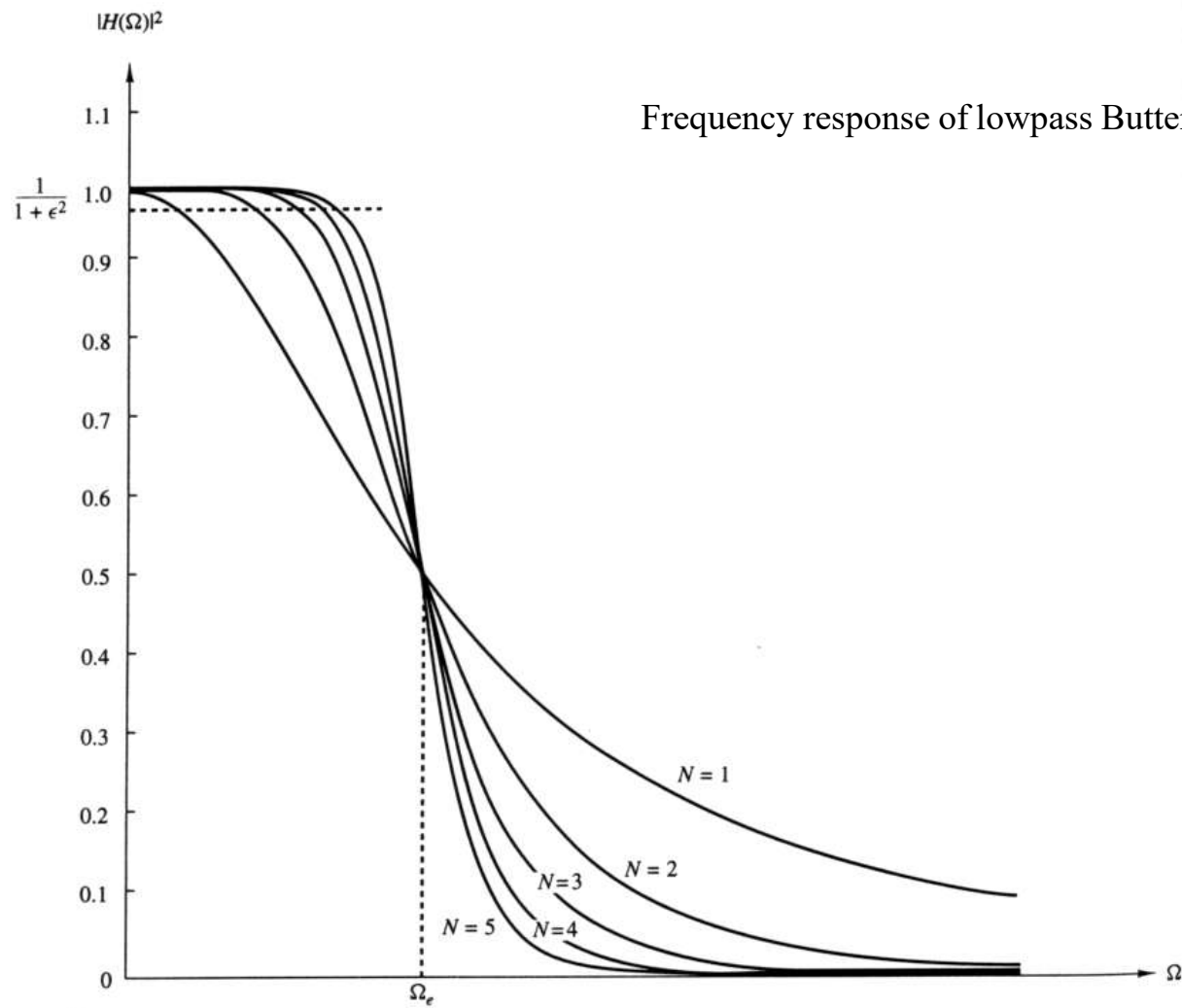
$$|H_c(j\Omega)|^2 = \frac{1}{1 + (j\Omega / j\Omega_c)^{2N}}$$

$$|H_c(s)|^2 = \frac{1}{1 + (s / j\Omega_c)^{2N}}$$



$$s_k = (-1)^{1/2N} (j\Omega_c) = \Omega_c e^{(j\pi/2N)(2k+N-1)} \quad \text{for } k = 0, 1, \dots, 2N - 1$$

Frequency response of lowpass Butterworth filters



Chebyshev Filters

- The magnitude squared response of the analog lowpass Type I Chebyshev filter of Nth order is given by:

$$|H(W)|^2 = 1/[1 + e^2 T_N^2(W/W_c)].$$

where $T_N(W)$ is the Chebyshev polynomial of order N:

$$\begin{aligned} T_N(W) &= \cos(N \cos^{-1} W), & |W| \leq 1, \\ &= \cosh(N \cosh^{-1} W), & |W| > 1. \end{aligned}$$

The polynomial can be derived via a recurrence relation given by

$$T_r(W) = 2WT_{r-1}(W) - T_{r-2}(W), \quad r \geq 2,$$

with $T_0(W) = 1$ and $T_1(W) = W$.

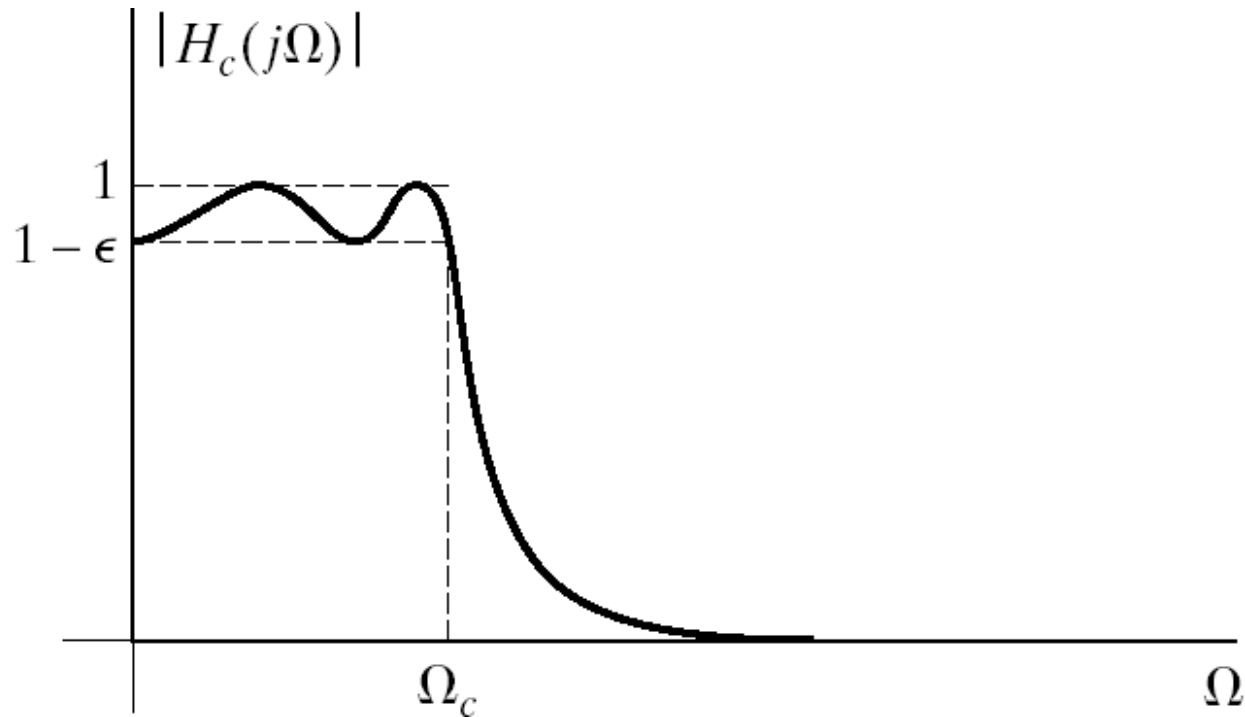
- The magnitude squared response of the analog lowpass Type II or inverse Chebyshev filter of Nth order is given by:

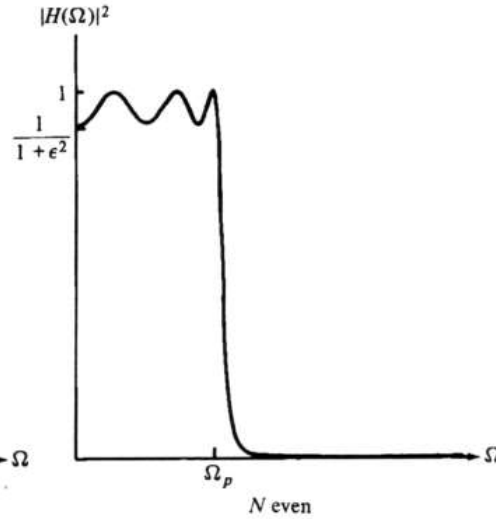
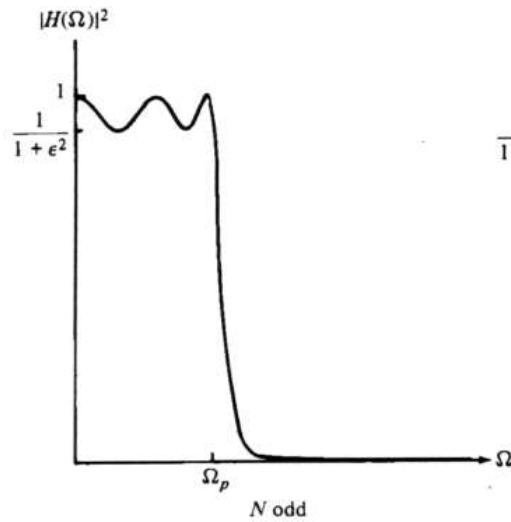
$$|H(W)|^2 = 1/[1 + e^2 \{T_N(W_s/W_p) / T_N(W_s/W)\}^2].$$

Chebyshev Filters

- Equiripple in the passband and monotonic in the stopband
- Or equiripple in the stopband and monotonic in the passband

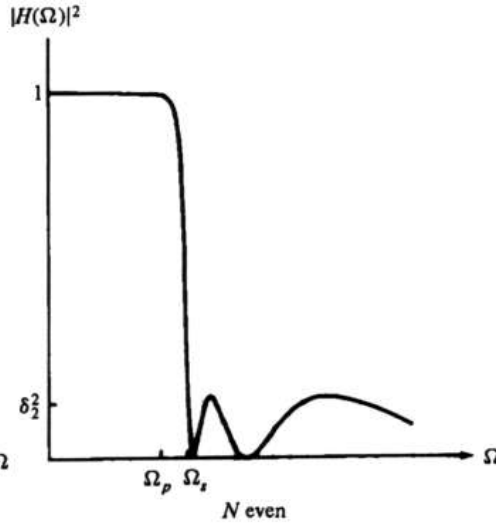
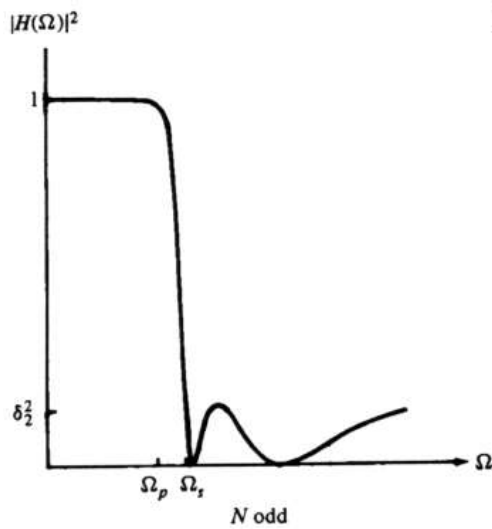
$$|H_c(j\Omega)|^2 = \frac{1}{1 + \epsilon^2 V_N^2(\Omega / \Omega_c)} \quad V_N(x) = \cos(N \cos^{-1} x)$$





Frequency response of
lowpass Type I Chebyshev filter

$$|H(W)|^2 = 1/[1 + \epsilon^2 T_N^2(W/W_p)]$$



Frequency response of
lowpass Type II Chebyshev filter

$$|H(W)|^2 = 1/[1 + \epsilon^2 \{T_N^2(W_s/W_p)/T_N^2(W_s/W)\}]$$

$$N = \log_{10}[(\sqrt{1 - d_2^2} + \sqrt{1 - d_2^2(1 + e^2)})/ed_2] / \log_{10}[(W_s/W_p) + \sqrt{(W_s/W_p)^2 - 1}]$$

$$= [\cosh^{-1}(d/e)] / [\cosh^{-1}(W_s/W_p)]$$

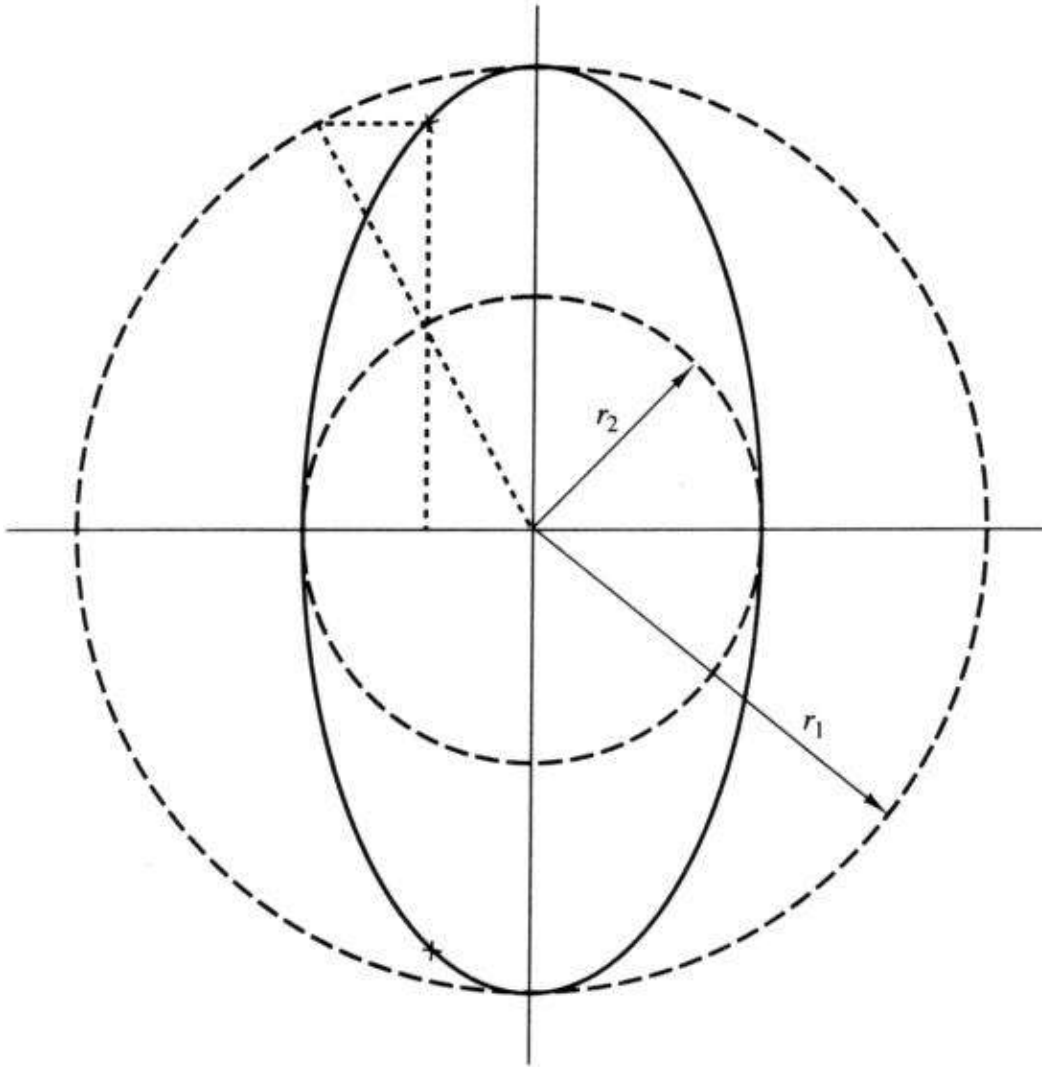
for both Type I and II Chebyshev filters, and where

$$d_2 = 1 / \sqrt{1 + d^2}.$$

- The poles of a Type I Chebyshev filter lie on an ellipse in the s-plane with major axis $r_1 = W_p \{(b^2 + 1)/2b\}$ and minor axis $r_1 = W_p \{(b^2 - 1)/2b\}$ where b is related to e according to

$$b = \{[\sqrt{1 + e^2} + 1]/e\}^{1/N}$$

- The zeros of a Type II Chebyshev filter are located on the imaginary axis.



Determination of the pole locations
for a Chebyshev filter.

Type I: pole positions are

$$x_k = r_2 \cos f_k$$

$$y_k = r_1 \sin f_k$$

$$f_k = [p/2] + [(2k + 1)p/2N]$$

$$r_1 = W_p [b^2 + 1]/2b$$

$$r_2 = W_p [b^2 - 1]/2b$$

$$b = \{[\sqrt{1 + e^2} + 1]/e\}^{1/N}$$

Type II: zero positions are

$$s_k = jW_s / \sin f_k$$

and pole positions are

$$v_k = W_s x_k / \sqrt{x_k^2 + y_k^2}$$

$$w_k = W_s y_k / \sqrt{x_k^2 + y_k^2}$$

$$b = \{[1 + \sqrt{1 - d_2^2}]/d\}^{1/N}$$

$$k = 0, 1, \dots, N-1.$$

Approximation of Derivative Method

- Approximation of derivative method is the simplest one for converting an analog filter into a digital filter by approximating the differential equation by an equivalent difference equation.
 - For the derivative $dy(t)/dt$ at time $t = nT$, we substitute the backward difference $[y(nT) - y(nT - T)]/T$. Thus

$$\left. \frac{dy(t)}{dt} \right|_{t=nT} = \frac{y(nT) - y(nT - T)}{T} \cong \frac{y[n] - y[n - 1]}{T}$$

where T represents the sampling period. Then, $s = (1 - z^{-1})/T$

- The second derivative $d^2y(t)/dt^2$ is derived into second difference as follow:

$$\left. \frac{dy(t)}{dt} \right|_{t=nT} \cong \frac{y[n] - 2y[n - 1] + y[n - 2]}{T^2}$$

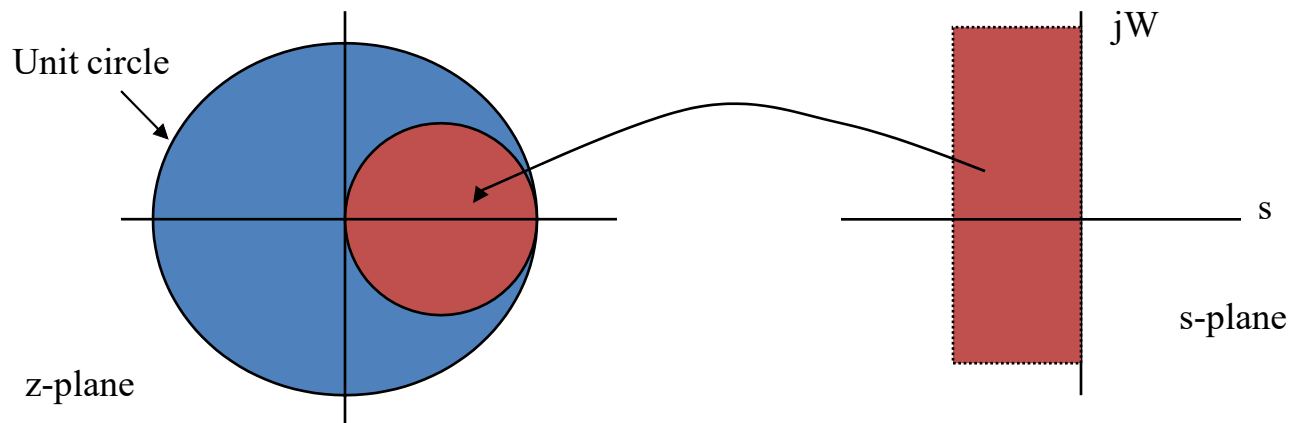
which $s^2 = [(1 - z^{-1})/T]^2$. So, for the k th derivative of $y(t)$, $s^k = [(1 - z^{-1})/T]^k$.

Approximation of Derivative Method

- Hence, the system function for the digital IIR filter obtained as a result of the approximation of the derivatives by finite difference is

$$H(z) = H_a(s)|_{s=(z-1)/Tz}$$

- It is clear that points in the LHP of the s-plane are mapped into the corresponding points inside the unit circle in the z-plane and points in the RHP of the s-plane are mapped into points outside this circle.
 - Consequently, a stable analog filter is transformed into a stable digital filter due to this mapping property.



Example: Approximation of derivative method

Convert the analog bandpass filter with system function

$$H_a(s) = 1/[(s + 0.1)^2 + 9]$$

Into a digital IIR filter by use of the backward difference for the derivative.

Substitute for $s = (1 - z^{-1})/T$ into $H_a(s)$ yields

$$H(z) = 1/[((1 - z^{-1})/T) + 0.1)^2 + 9]$$

$$H(z) = \frac{\frac{T^2}{1 + 0.2T + 9.01T^2}}{1 - \frac{2(1 + 0.1T)}{1 + 0.2T + 9.01T^2}z^{-1} + \frac{1}{1 + 0.2T + 9.01T^2}z^{-2}}$$

T can be selected to satisfied specification of designed filter. For example, if $T = 0.1$, the poles are located at

$$p_{1,2} = 0.91 \pm j0.27 = 0.949\exp[\pm j16.5^\circ]$$

Filter Design by Impulse Invariance

- Remember impulse invariance
 - Mapping a continuous-time impulse response to discrete-time
 - Mapping a continuous-time frequency response to discrete-time

$$h[n] = T_d h_c(nT_d)$$

$$H(e^{j\omega}) = \sum_{k=-\infty}^{\infty} H_c\left(j \frac{\omega}{T_d} + j \frac{2\pi}{T_d} k\right)$$

- If the continuous-time filter is bandlimited to

$$H_c(j\Omega) = 0 \quad \left| \Omega \right| \geq \pi / T_d$$

$$H(e^{j\omega}) = H_c\left(j \frac{\omega}{T_d}\right) \quad \left| \omega \right| \leq \pi$$

- If we start from discrete-time specifications T_d cancels out
 - Start with discrete-time spec in terms of ω
 - Go to continuous-time $\Omega = \omega / T$ and design continuous-time filter
 - Use impulse invariance to map it back to discrete-time $\omega = \Omega T$
- Works best for bandlimited filters due to possible aliasing

Impulse Invariance of System Functions

- Develop impulse invariance relation between system functions
- Partial fraction expansion of transfer function

$$H_c(s) = \sum_{k=1}^N \frac{A_k}{s - s_k}$$

- Corresponding impulse response

$$h_c(t) = \begin{cases} \sum_{k=1}^N A_k e^{s_k t} & t \geq 0 \\ 0 & t < 0 \end{cases}$$

- Impulse response of discrete-time filter

$$h[n] = T_d h_c(nT_d) = \sum_{k=1}^N T_d A_k e^{s_k nT_d} u[n] = \sum_{k=1}^N T_d A_k \left(e^{s_k T_d} \right)^n u[n]$$

- System function

$$H_z(z) = \sum_{k=1}^N \frac{T_d A_k}{1 - e^{s_k T_d} z^{-1}}$$

- Pole $s=s_k$ in s-domain transform into pole at $e^{s_k T_d}$

Impulse Invariant Algorithm

- Step 1: define specifications of filter
 - Ripple in frequency bands
 - Critical frequencies: passband edge, stopband edge, and/or cutoff frequencies.
 - Filter band type: lowpass, highpass, bandpass, bandstop.
- Step 2: linear transform critical frequencies as follow

$$W = w/T_d$$

- Step 3: select filter structure type and its order: Bessel, Butterworth, Chebyshev type I, Chebyshev type II or inverse Chebyshev, elliptic.
- Step 4: convert $H_a(s)$ to $H(z)$ using linear transform in step 2.
- Step 5: verify the result. If it does not meet requirement, return to step 3.

Example: Impulse invariant method

Convert the analog filter with system function

$$H_a(s) = [s + 0.1]/[(s + 0.1)^2 + 9]$$

into a digital IIR filter by means of the impulse invariance method.

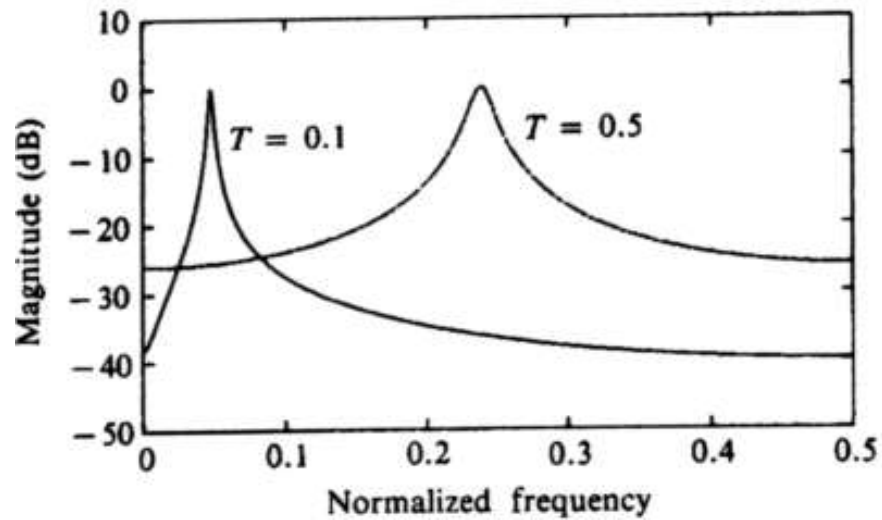
The analog filter has a zero at $s = -0.1$ and a pair of complex conjugate poles at $p_k = -0.1 \pm j3$.

Thus,

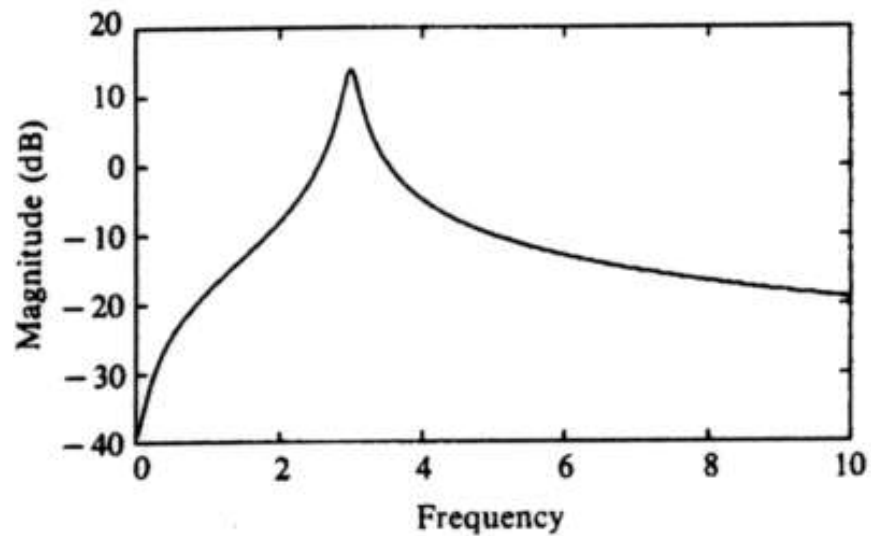
$$H_a(s) = \frac{\frac{1}{2}}{s + 0.1 - j3} + \frac{\frac{1}{2}}{s + 0.1 + j3}$$

Then

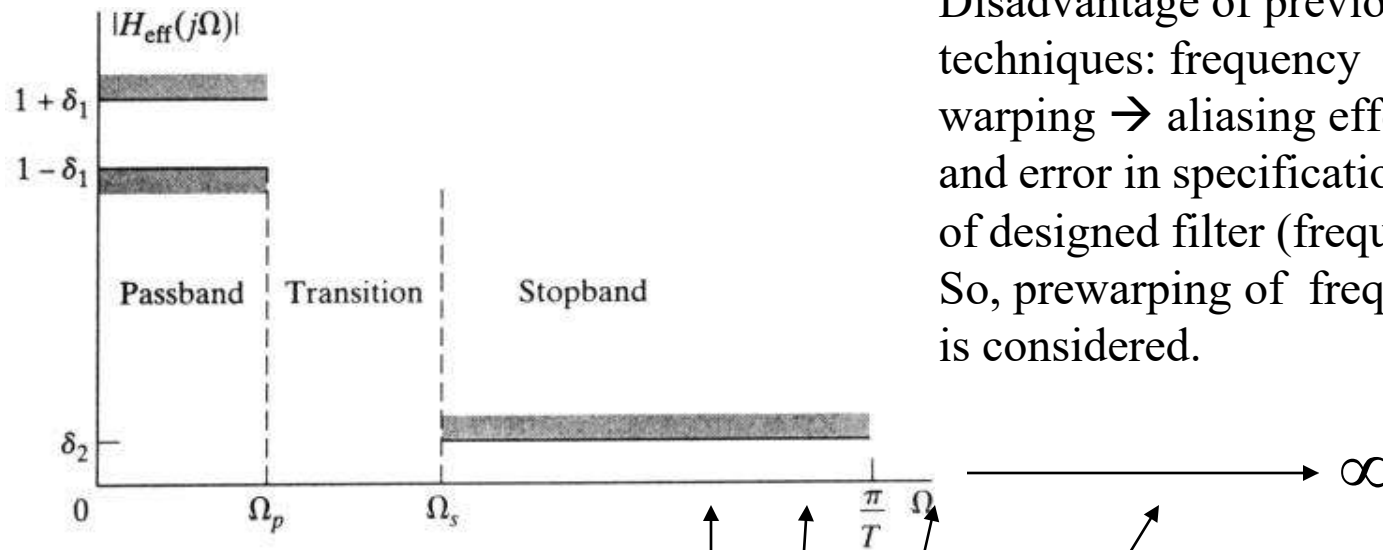
$$H(z) = \frac{\frac{1}{2}}{1 - e^{-0.1T} e^{j3T} z^{-1}} + \frac{\frac{1}{2}}{1 - e^{-0.1T} e^{-j3T} z^{-1}}$$



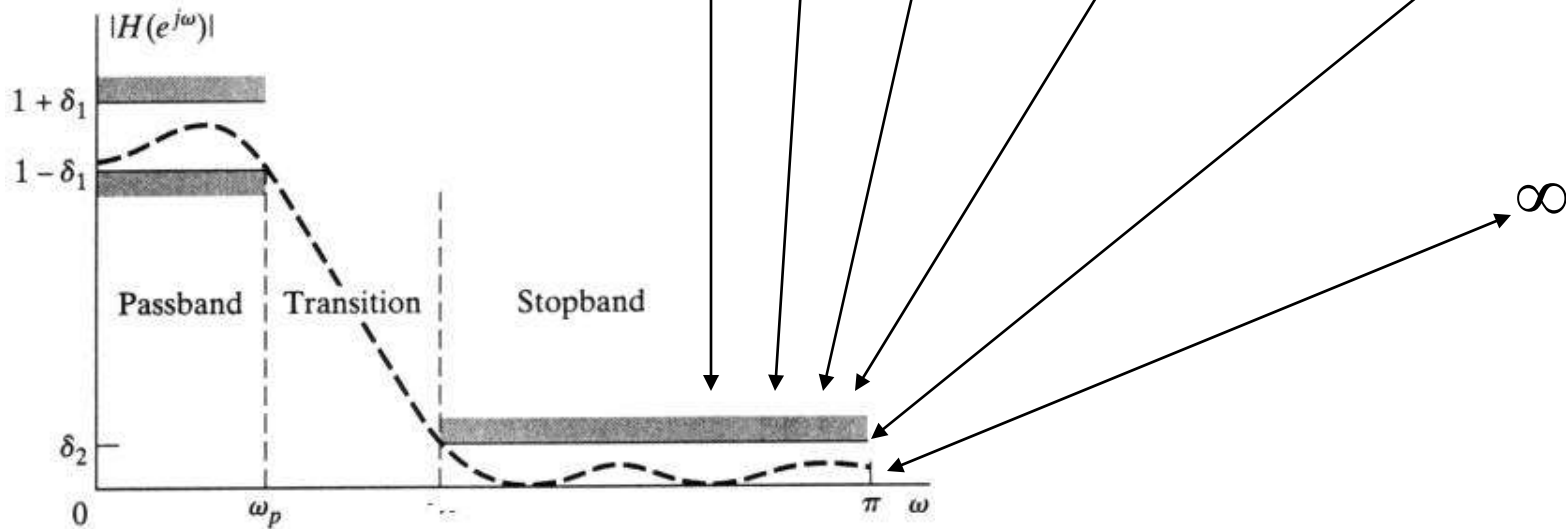
Frequency response
of digital filter.



Frequency response
of analog filter.



Disadvantage of previous techniques: frequency warping \rightarrow aliasing effect and error in specifications of designed filter (frequencies)
 So, prewarping of frequency is considered.



Example

- Impulse invariance applied to Butterworth

$$\begin{aligned} 0.89125 &\leq |H(e^{j\omega})| \leq 1 & 0 \leq |\omega| \leq 0.2\pi \\ |H(e^{j\omega})| &\leq 0.17783 & 0.3\pi \leq |\omega| \leq \pi \end{aligned}$$

- Since sampling rate T_d cancels out we can assume $T_d=1$
- Map spec to continuous time

$$\begin{aligned} 0.89125 &\leq |H(j\Omega)| \leq 1 & 0 \leq |\Omega| \leq 0.2\pi \\ |H(j\Omega)| &\leq 0.17783 & 0.3\pi \leq |\Omega| \leq \pi \end{aligned}$$

- Butterworth filter is monotonic so spec will be satisfied if

$$\begin{aligned} |H_c(j0.2\pi)| &\geq 0.89125 \quad \text{and} \quad |H_c(j0.3\pi)| \leq 0.17783 \\ |H_c(j\Omega)|^2 &= \frac{1}{1 + (j\Omega / j\Omega_c)^{2N}} \end{aligned}$$

- Determine N and Ω_c to satisfy these conditions

Example Cont'd

- Satisfy both constraints

$$1 + \left(\frac{0.2\pi}{\Omega_c} \right)^{2N} = \left(\frac{1}{0.89125} \right)^2 \quad \text{and} \quad 1 + \left(\frac{0.3\pi}{\Omega_c} \right)^{2N} = \left(\frac{1}{0.17783} \right)^2$$

- Solve these equations to get

$$N = 5.8858 \cong 6 \quad \text{and} \quad \Omega_c = 0.70474$$

- N must be an integer so we round it up to meet the spec
- Poles of transfer function

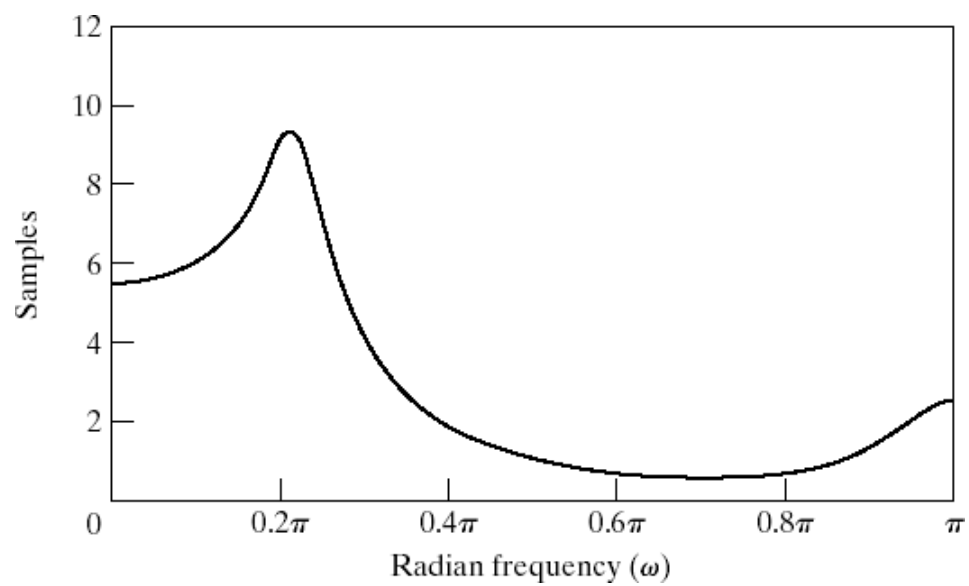
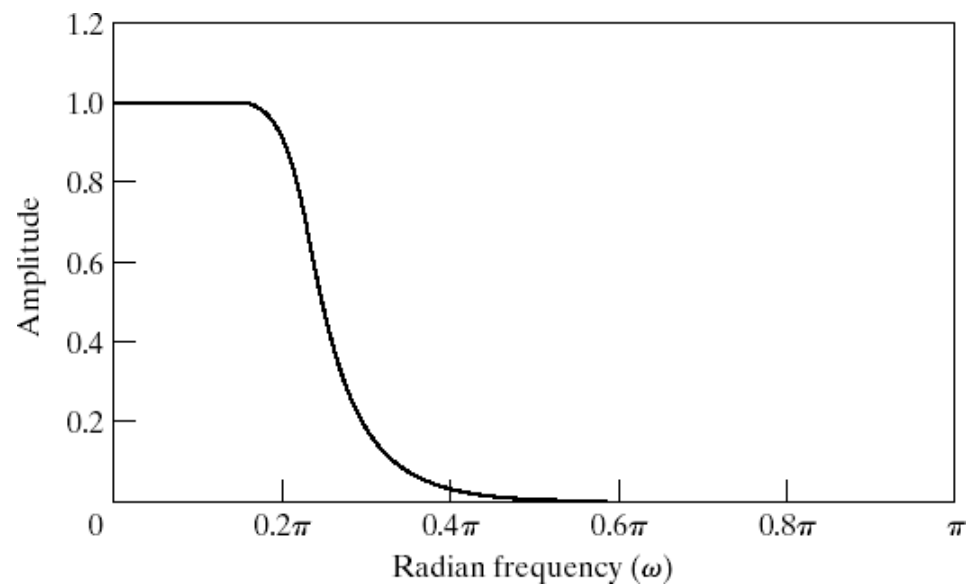
$$s_k = (-1)^{1/12} (j\Omega_c) = \Omega_c e^{(j\pi/12)(2k+11)} \quad \text{for } k = 0, 1, \dots, 11$$

- The transfer function

$$H(s) = \frac{0.12093}{(s^2 + 0.364s + 0.4945)(s^2 + 0.9945s + 0.4945)(s^2 + 1.3585s + 0.4945)}$$

- Mapping to z-domain

$$H(z) = \frac{0.2871 - 0.4466z^{-1}}{1 - 1.2971z^{-1} + 0.6949z^{-2}} + \frac{-2.1428 + 1.1455z^{-1}}{1 - 1.0691z^{-1} + 0.3699z^{-2}} + \frac{1.8557 - 0.6303z^{-1}}{1 - 0.9972z^{-1} + 0.257z^{-2}}$$



Filter Design by Bilinear Transformation

- Get around the aliasing problem of impulse invariance
- Map the entire s-plane onto the unit-circle in the z-plane
 - Nonlinear transformation
 - Frequency response subject to warping
- Bilinear transformation

$$s = \frac{2}{T_d} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right)$$

- Transformed system function

$$H(z) = H_c \left[\frac{2}{T_d} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right) \right]$$

- Again T_d cancels out so we can ignore it
- We can solve the transformation for z as

$$z = \frac{1 + (T_d / 2)s}{1 - (T_d / 2)s} = \frac{1 + \sigma T_d / 2 + j\Omega T_d / 2}{1 - \sigma T_d / 2 - j\Omega T_d / 2} \quad s = \sigma + j\Omega$$

- Maps the left-half s-plane into the inside of the unit-circle in z
 - Stable in one domain would stay in the other

Bilinear Transformation

- On the unit circle the transform becomes

$$z = \frac{1 + j\Omega T_d / 2}{1 - j\Omega T_d / 2} = e^{j\omega}$$

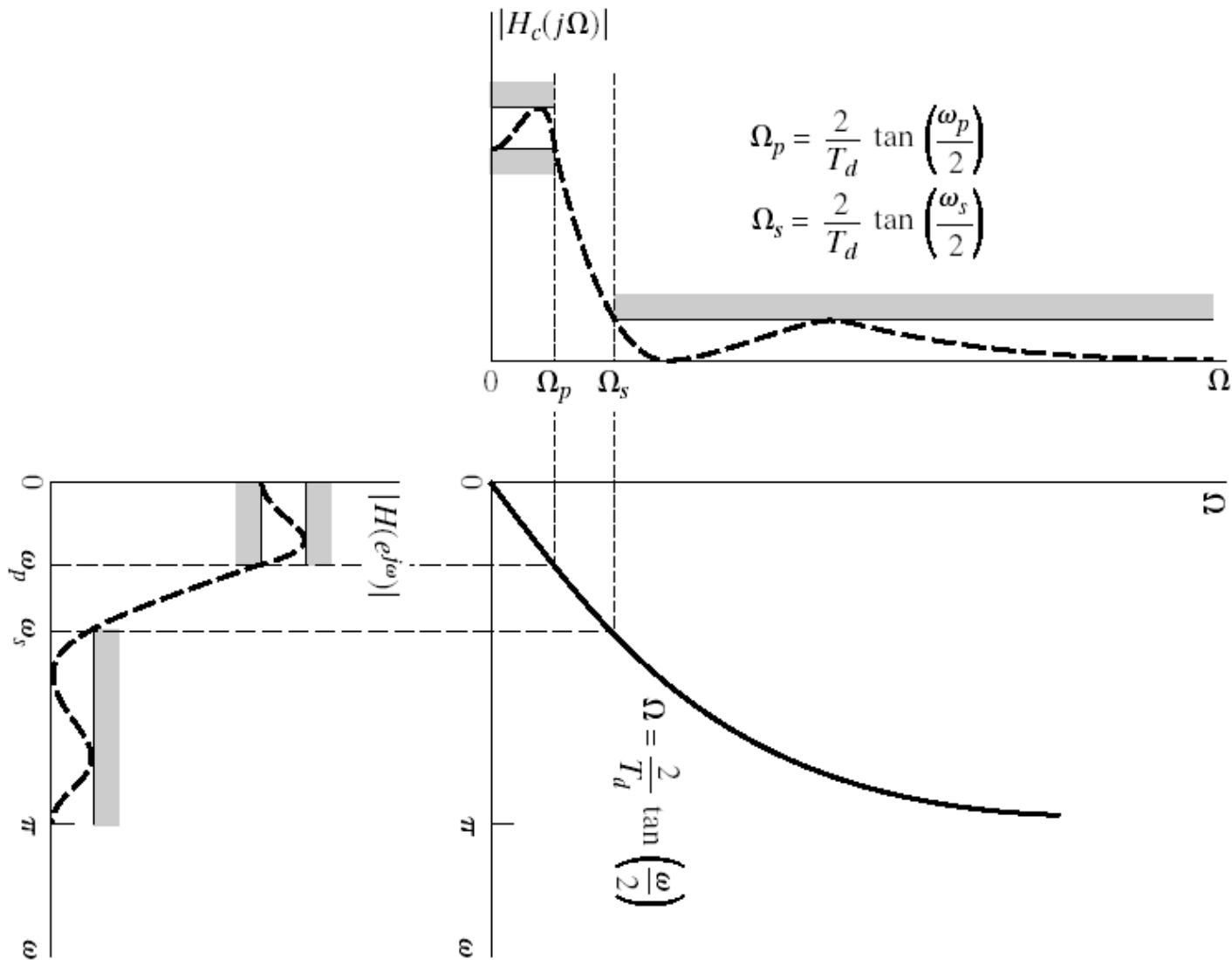
- To derive the relation between ω and Ω

$$s = \frac{2}{T_d} \left(\frac{1 - e^{-j\omega}}{1 + e^{-j\omega}} \right) = \sigma + j = \frac{2}{T_d} \left[\frac{2e^{-j\omega/2} j \sin(\omega/2)}{2e^{-j\omega/2} \cos(\omega/2)} \right] = \frac{2j}{T_d} \tan \left(\frac{\omega}{2} \right)$$

- Which yields

$$\Omega = \frac{2}{T_d} \tan \left(\frac{\omega}{2} \right) \quad \text{or} \quad \omega = 2 \arctan \left(\frac{\Omega T_d}{2} \right)$$

Bilinear Transformation



Example

- Bilinear transform applied to Butterworth

$$0.89125 \leq |H(e^{j\omega})| \leq 1 \quad 0 \leq |\omega| \leq 0.2\pi$$

$$|H(e^{j\omega})| \leq 0.17783 \quad 0.3\pi \leq |\omega| \leq \pi$$

- Apply bilinear transformation to specifications

$$0.89125 \leq |H(j\Omega)| \leq 1 \quad 0 \leq |\Omega| \leq \frac{2}{T_d} \tan\left(\frac{0.2\pi}{2}\right)$$

$$|H(j\Omega)| \leq 0.17783 \quad \frac{2}{T_d} \tan\left(\frac{0.3\pi}{2}\right) \leq |\Omega| < \infty$$

- We can assume $T_d=1$ and apply the specifications to

$$|H_c(j\Omega)|^2 = \frac{1}{1 + (\Omega / \Omega_c)^{2N}}$$

- To get

$$1 + \left| \frac{2 \tan 0.1\pi}{\Omega_c} \right|^{2N} = \left| \frac{1}{0.89125} \right|^2 \quad \text{and} \quad 1 + \left| \frac{2 \tan 0.15\pi}{\Omega_c} \right|^{2N} = \left| \frac{1}{0.17783} \right|^2$$

Example Cont'd

- Solve N and Ω_c

$$N = \frac{\log \left[\left(\left(\frac{1}{0.17783} \right)^2 - 1 \right) / \left(\left(\frac{1}{0.89125} \right)^2 - 1 \right) \right]}{2 \log [\tan (0.15 \pi) / \tan (0.1 \pi)]} = 5.305 \cong 6 \quad \Omega_c = 0.766$$

- The resulting transfer function has the following poles

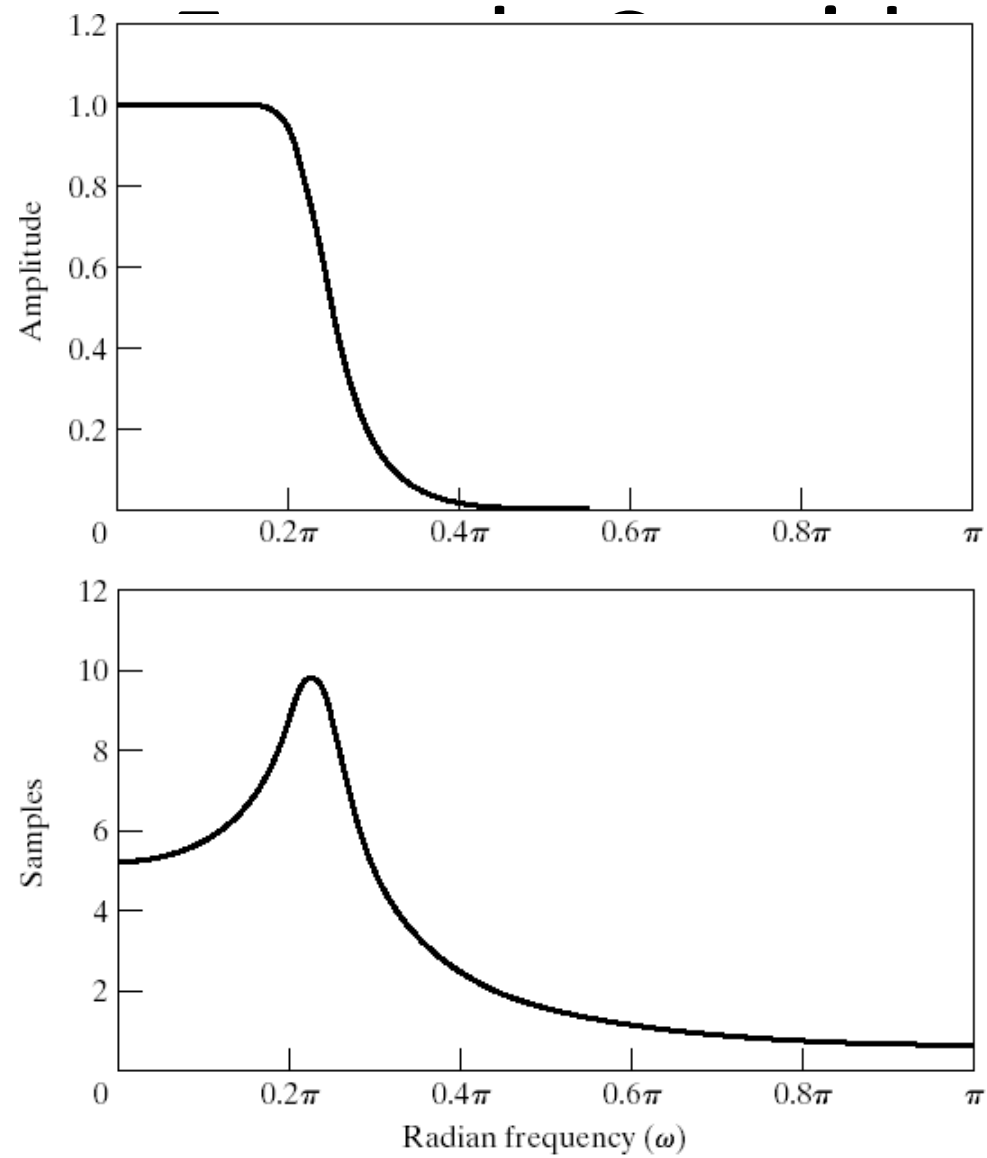
$$s_k = (-1)^{1/12} (j\Omega_c) = \Omega_c e^{(j\pi/12)(2k+11)} \quad \text{for } k = 0, 1, \dots, 11$$

- Resulting in

$$H_c(s) = \frac{0.20238}{(s^2 + 0.3996s + 0.5871)(s^2 + 1.0836s + 0.5871)(s^2 + 1.4802s + 0.5871)}$$

- Applying the bilinear transform yields

$$H(z) = \frac{0.0007378 (1 + z^{-1})^6}{(1 - 1.2686z^{-1} + 0.7051z^{-2})(1 - 1.0106z^{-1} + 0.3583z^{-2})} \\ \times \frac{1}{(1 - 0.9044z^{-1} + 0.2155z^{-2})}$$



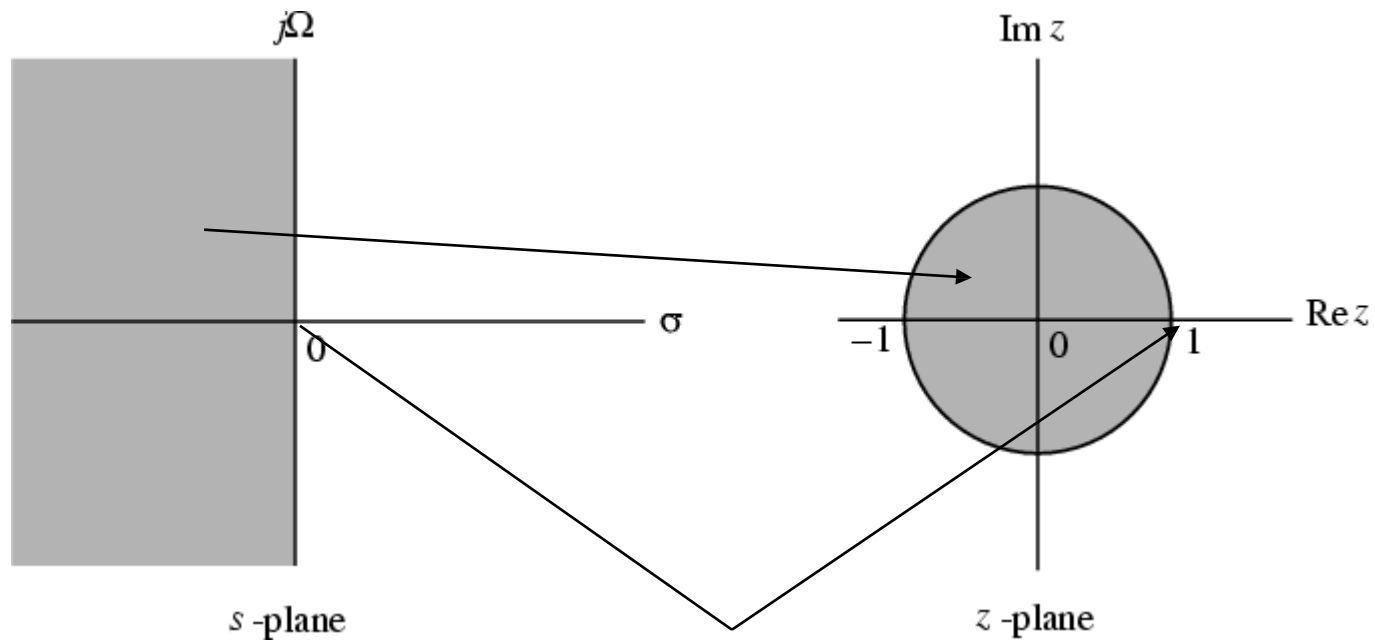
IIR Digital Filter: The bilinear transformation

- To obtain $G(z)$ replace s by $f(z)$ in $H(s)$
- Start with requirements on $G(z)$

<u>$G(z)$</u>	<u>Available $H(s)$</u>
Stable	Stable
Real and Rational in z	Real and Rational in s
Order n	Order n
L.P. (lowpass) cutoff Ω_c	L.P. cutoff $\omega_c T$

Bilinear Transformation

- Mapping of s -plane into the z -plane



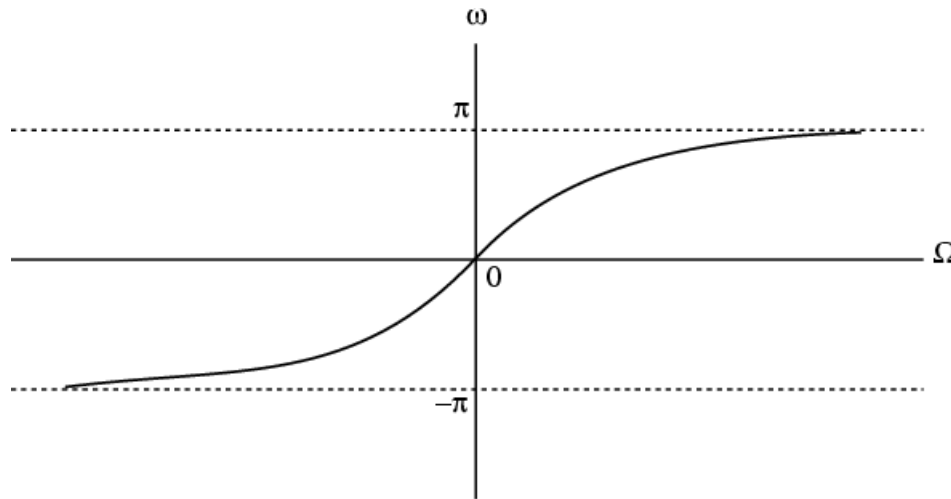
Bilinear Transformation

- For $z = e^{j\omega}$ with unity scalar we have

$$j\Omega = \frac{1 - e^{-j\omega}}{1 + e^{-j\omega}} = j \tan(\omega / 2)$$

or

$$\Omega = \tan(\omega / 2)$$

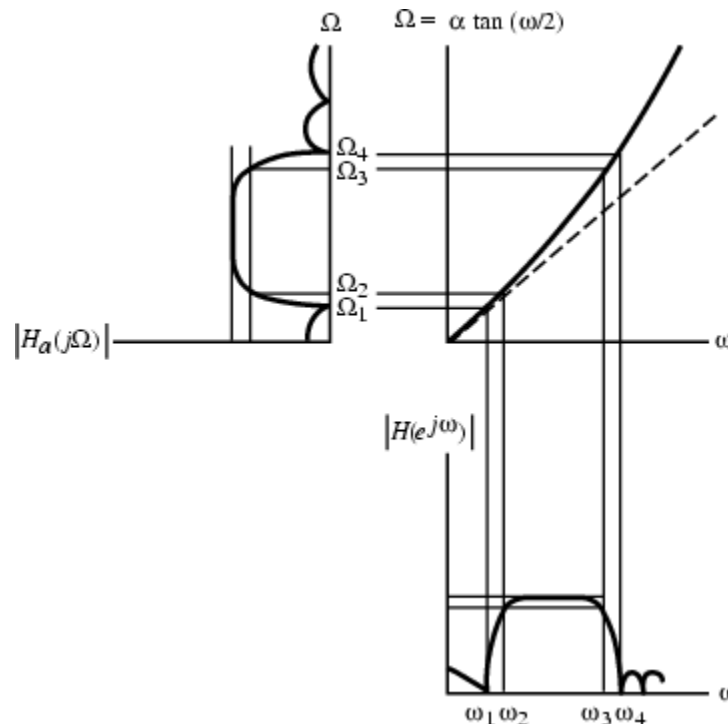


Bilinear Transformation

- Mapping is highly nonlinear
- Complete negative imaginary axis in the s -plane from $\Omega = -\infty$ to $\Omega = 0$ is mapped into the lower half of the unit circle in the z -plane from $z = -1$ to $z = 1$
- Complete positive imaginary axis in the s -plane from $\Omega = 0$ to $\Omega = \infty$ is mapped into the upper half of the unit circle in the z -plane from $z = 1$ to $z = -1$

Bilinear Transformation

- Nonlinear mapping introduces a distortion in the frequency axis called **frequency warping**
- Effect of warping shown below



Spectral Transformations

- To transform $G_L(z)$ a given lowpass transfer function to another transfer function $G_D(\hat{z})$ that may be a lowpass, highpass, bandpass or bandstop filter (solutions given by Constantinides)
- z^{-1} has been used to denote the unit delay in the prototype lowpass filter $G_L(z)$ and \hat{z}^{-1} to denote the unit delay in the transformed filter $G_D(\hat{z})$ to avoid confusion

Spectral Transformations

- Unit circles in z - and \hat{z} -planes defined by

$$z = e^{j\omega} , \quad \hat{z} = e^{j\hat{\omega}}$$

- Transformation from z -domain to \hat{z} -domain given by

- Then

$$z = F(\hat{z})$$

$$G_D(\hat{z}) = G_L\{F(\hat{z})\}$$

Spectral Transformations

- From $z = F(\hat{z})$, thus $|z| = |F(\hat{z})|$, hence

$$|F(\hat{z})| = \begin{cases} > 1, & \text{if } |z| > 1 \\ = 1, & \text{if } |z| = 1 \\ < 1, & \text{if } |z| < 1 \end{cases}$$

- Therefore $1 / F(\hat{z})$ must be a stable allpass function

$$\frac{1}{F(\hat{z})} = \pm \prod_{l=1}^L \left(\frac{1 - \alpha_l^* \hat{z}}{\hat{z} - \alpha_l} \right), \quad |\alpha_l| < 1$$

Lowpass-to-Lowpass Spectral Transformation

- To transform a lowpass filter $G_L(z)$ with a cutoff frequency ω_c to another lowpass filter $G_D(\hat{z})$ with a cutoff frequency ω_c^\wedge , the transformation is

$$z^{-1} = \frac{1}{F(\hat{z})} = \frac{1 - \alpha \hat{z}}{\hat{z} - \alpha}$$

- On the unit circle we have

$$e^{-j\omega} = \frac{e^{-j\omega^\wedge} - \alpha}{1 - \alpha e^{-j\omega^\wedge}}$$

which yields

$$\tan(\omega / 2) = \left(\frac{1 + \alpha}{1 - \alpha} \right) \tan(\omega^\wedge / 2)$$

Lowpass-to-Lowpass Spectral Transformation

- Solving we get

$$\alpha = \frac{\sin((\omega_c - \hat{\omega}_c) / 2)}{\sin((\omega_c + \hat{\omega}_c) / 2)}$$

- Example - Consider the lowpass digital filter

$$G_L(z) = \frac{0.0662 (1 + z^{-1})^3}{(1 - 0.2593 z^{-1})(1 - 0.6763 z^{-1} + 0.3917 z^{-2})}$$

which has a passband from dc to 0.25π with a 0.5 dB ripple

- Redesign the above filter to move the passband edge to

$$0.35\pi$$

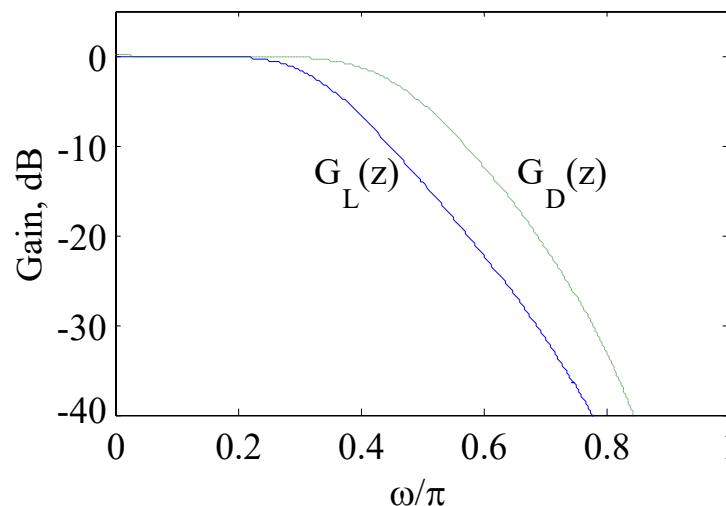
Lowpass-to-Lowpass Spectral Transformation

- Here

$$\alpha = - \frac{\sin(0.05\pi)}{\sin(0.3\pi)} = -0.1934$$

- Hence, the desired lowpass transfer function is

$$G_D(\hat{z}) = G_L(z) \Big|_{z^{-1} = \frac{\hat{z}^{-1} + 0.1934}{1 + 0.1934 \hat{z}^{-1}}}$$



Lowpass-to-Lowpass Spectral Transformation

- The lowpass-to-lowpass transformation

$$z^{-1} = \frac{1}{F(\hat{z})} = \frac{1 - \alpha \hat{z}}{\hat{z} - \alpha}$$

can also be used as highpass-to-highpass,
bandpass-to-bandpass and bandstop-to-
bandstop transformations

Lowpass-to-Highpass Spectral Transformation

- Desired transformation

$$z^{-1} = - \frac{\hat{z}^{-1} + \alpha}{1 + \alpha \hat{z}^{-1}}$$

- The transformation parameter α is given by

$$\alpha = - \frac{\cos \left((\omega_c + \hat{\omega}_c) / 2 \right)}{\cos \left((\omega_c - \hat{\omega}_c) / 2 \right)}$$

where ω_c is the cutoff frequency of the lowpass filter and $\hat{\omega}_c$ is the cutoff frequency of the desired highpass filter

Lowpass-to-Highpass Spectral Transformation

- Example - Transform the lowpass filter

$$G_L(z) = \frac{0.0662 (1 + z^{-1})^3}{(1 - 0.2593 z^{-1})(1 - 0.6763 z^{-1} + 0.3917 z^{-2})}$$

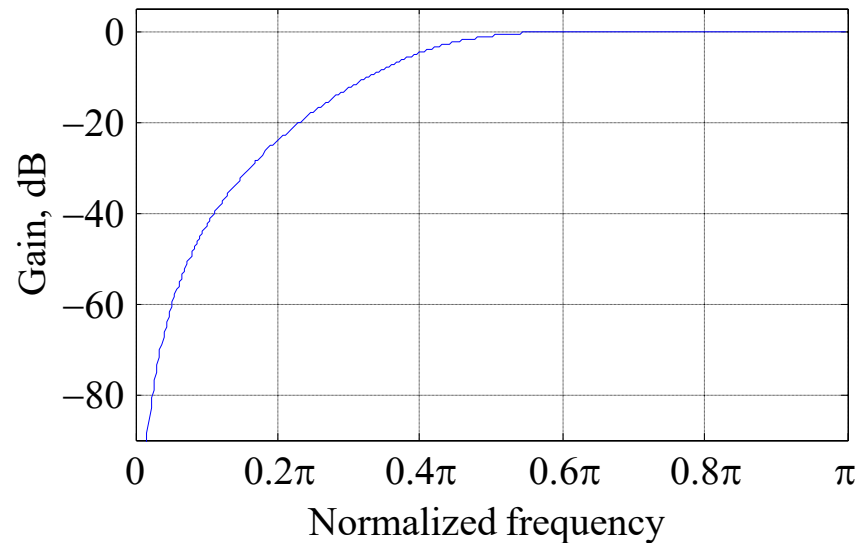
- with a passband edge at 0.25π to a highpass filter with a passband edge at 0.55π
- Here $\alpha = -\cos(0.4\pi) / \cos(0.15\pi) = -0.3468$
- The desired transformation is

$$z^{-1} = -\frac{\hat{z}^{-1} - 0.3468}{1 - 0.3468 \hat{z}^{-1}}$$

Lowpass-to-Highpass Spectral Transformation

- The desired highpass filter is

$$G_D(\hat{z}) = G(z) \Big|_{z^{-1} = -\frac{\hat{z}^{-1} - 0.3468}{1 - 0.3468 \hat{z}^{-1}}}$$



Lowpass-to-Highpass Spectral Transformation

- The lowpass-to-highpass transformation can also be used to transform a highpass filter with a cutoff at ω_c to a lowpass filter with a cutoff at $\hat{\omega}_c$
- and transform a bandpass filter with a center frequency at ω_o to a bandstop filter with a center frequency at $\hat{\omega}_o$

Lowpass-to-Bandpass Spectral Transformation

- Desired transformation

$$z^{-1} = - \frac{\hat{z}^{-2} - \frac{2\alpha\beta}{\beta+1}\hat{z}^{-1} + \frac{\beta-1}{\beta+1}}{\frac{\beta-1}{\beta+1}\hat{z}^{-2} - \frac{2\alpha\beta}{\beta+1}\hat{z}^{-1} + 1}$$

Lowpass-to-Bandpass Spectral Transformation

- The parameters α and β are given by

$$\alpha = \frac{\cos((\omega_{c2} + \omega_{c1}) / 2)}{\cos((\omega_{c2} - \omega_{c1}) / 2)}$$

$$\beta = \cot((\omega_{c2} - \omega_{c1}) / 2) \tan(\omega_c / 2)$$

where ω_c is the cutoff frequency of the lowpass filter, and ω_{c1} and ω_{c2} are the desired upper and lower cutoff frequencies of the bandpass filter

Lowpass-to-Bandpass Spectral Transformation

- Special Case - The transformation can be simplified if $\omega_c = \omega_{c2} - \omega_{c1}$
- Then the transformation reduces to

$$z^{-1} = -\hat{z}^{-1} \frac{\hat{z}^{-1} - \alpha}{1 - \alpha \hat{z}^{-1}}$$

where $\alpha = \cos \omega_o$ with ω_o denoting the desired center frequency of the bandpass filter

Lowpass-to-Bandstop Spectral Transformation

- Desired transformation

$$z^{-1} = \frac{\hat{z}^{-2} - \frac{2\alpha\beta}{1+\beta}\hat{z}^{-1} + \frac{1-\beta}{1+\beta}}{\frac{1-\beta}{1+\beta}\hat{z}^{-2} - \frac{2\alpha\beta}{1+\beta}\hat{z}^{-1} + 1}$$

Lowpass-to-Bandstop Spectral Transformation

- The parameters α and β are given by

$$\alpha = \frac{\cos((\omega_{c2} + \omega_{c1}) / 2)}{\cos((\omega_{c2} - \omega_{c1}) / 2)}$$

$$\beta = \tan((\omega_{c2} - \omega_{c1}) / 2) \tan(\omega_c / 2)$$

where ω_c is the cutoff frequency of the lowpass filter, and ω_{c1} and ω_{c2} are the desired upper and lower cutoff frequencies of the bandstop filter

UNIT-4

FIR Filters

Selection of Filter Type

- The transfer function $H(z)$ meeting the specifications must be a causal transfer function
- For IIR real digital filter the transfer function is a real rational function of z^{-1}

$$H(z) = \frac{p_0 + p_1 z^{-1} + p_2 z^{-2} + \square + p_M z^{-M}}{d_0 + d_1 z^{-1} + d_2 z^{-2} + \square + d_N z^{-N}}$$

- $H(z)$ must be stable and of lowest order N or M for reduced computational complexity

Selection of Filter Type

- FIR real digital filter transfer function is a polynomial in z^{-1} (order N) with real coefficients

$$H(z) = \sum_{n=0}^N h[n] z^{-n}$$

- For reduced computational complexity, degree N of $H(z)$ must be as small as possible
- If a linear phase is desired then we must have:

$$h[n] = \pm h[N - n]$$

Selection of Filter Type

- **Advantages in using an FIR filter** -
 - (1) Can be designed with exact linear phase
 - (2) Filter structure always stable with quantised coefficients
- **Disadvantages in using an FIR filter** - Order of an FIR filter is considerably higher than that of an equivalent IIR filter meeting the same specifications; this leads to higher computational complexity for FIR

FIR Filter Design

Digital filters with finite-duration impulse response (all-zero, or FIR filters) have both advantages and disadvantages compared to infinite-duration impulse response (IIR) filters.

FIR filters have the following primary advantages:

- They can have exactly linear phase.
- They are always stable.
- The design methods are generally linear.
- They can be realized efficiently in hardware.
- The filter startup transients have finite duration.

The primary disadvantage of FIR filters is that they often require a much higher filter order than IIR filters to achieve a given level of performance. Correspondingly, the delay of these filters is often much greater than for an equal performance IIR filter.

FIR Design

FIR Digital Filter Design

Three commonly used approaches to FIR filter design -

- (1) Windowed Fourier series approach
- (2) Frequency sampling approach
- (3) Computer-based optimization methods

Finite Impulse Response Filters

- The transfer function is given by

$$H(z) = \sum_{n=0}^{N-1} h(n) \cdot z^{-n}$$

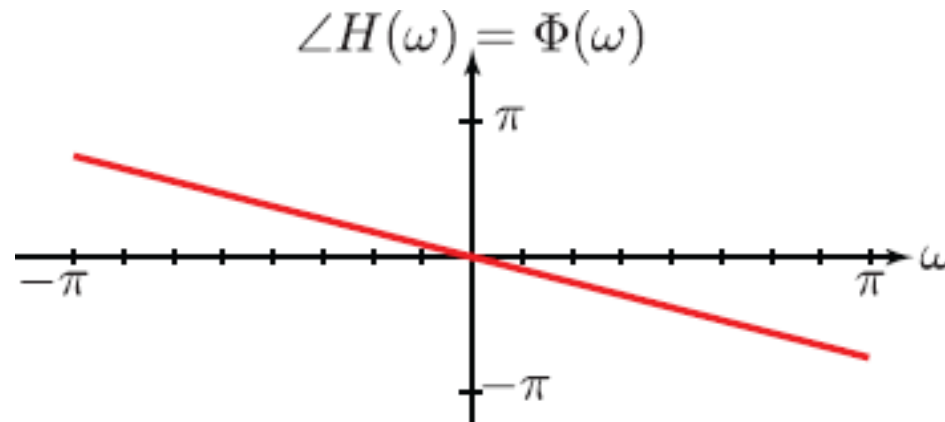
- The length of Impulse Response is N
- All poles are at $z = 0$.
- Zeros can be placed anywhere on the z-plane

FIR: Linear phase

For phase linearity the FIR transfer function **must** have zeros outside the unit circle

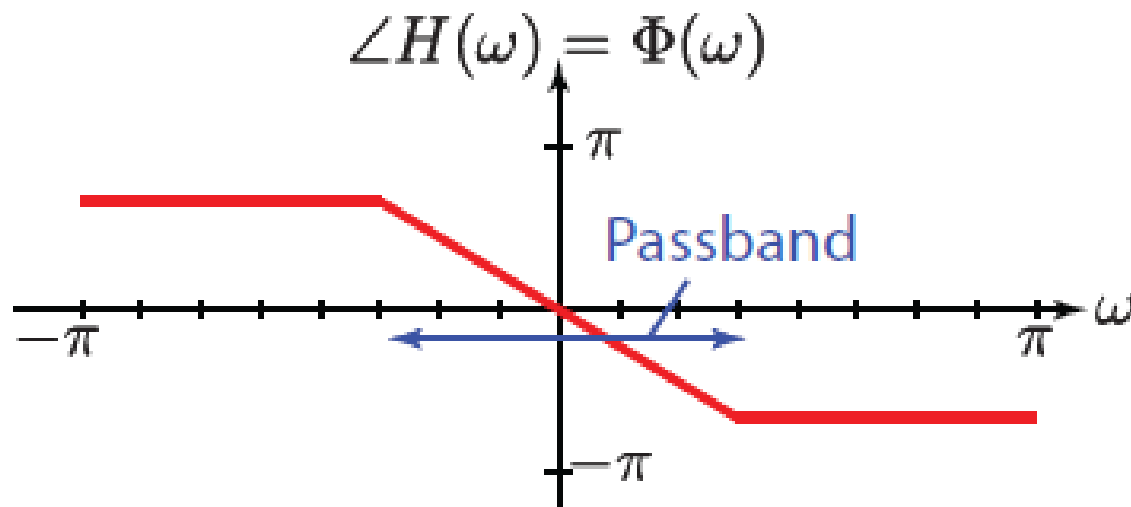
Linear Phase

- What is linear phase?
- Ans: The phase is a straight line in the passband of the system.
- Example: linear phase (all pass system)
- I Group delay is given by the negative of the slope of the line



Linear phase

- linear phase (low pass system)
- Linear characteristics only need to pertain to the passband frequencies only.



FIR: Linear phase

- For Linear Phase t.f. (order $N-1$)

- $$h(n) = \pm h(N-1-n)$$

- so that for N even:

$$\begin{aligned} H(z) &= \sum_{n=0}^{N/2-1} h(n) \cdot z^{-n} \pm \sum_{n=N/2}^{N-1} h(n) \cdot z^{-n} \\ &= \sum_{n=0}^{N/2-1} h(n) \cdot z^{-n} \pm \sum_{n=0}^{N/2-1} h(N-1-n) \cdot z^{-(N-1-n)} \\ &= \sum_{n=0}^{N/2-1} h(n) \left[z^{-n} \pm z^{-m} \right] \quad m = N-1-n \end{aligned}$$

FIR: Linear phase

- for N odd:

$$H(z) = \sum_{n=0}^{\frac{N-1}{2}-1} h(n) \cdot [z^{-n} \pm z^{-m}] + h\left(\frac{N-1}{2}\right) z^{-\left(\frac{N-1}{2}\right)}$$

- I) On $C : |z| = 1$ we have for N even, and +ve sign

$$H(e^{j\omega T}) = e^{-j\omega T \left(\frac{N-1}{2}\right)} \cdot \sum_{n=0}^{\frac{N}{2}-1} 2h(n) \cdot \cos\left(\omega T \left(n - \frac{N-1}{2}\right)\right)$$

FIR: Linear phase

- II) While for –ve sign

$$H(e^{j\omega T}) = e^{-j\omega T \left(\frac{N-1}{2} \right)} \cdot \sum_{n=0}^{N/2-1} j 2 h(n) \cdot \sin \left(\omega T \left(n - \frac{N-1}{2} \right) \right)$$

- [Note: antisymmetric case adds $\pi / 2$ rads to phase, with discontinuity at $\omega = 0$]
- III) For N odd with +ve sign

$$H(e^{j\omega T}) = e^{-j\omega T \left[\frac{N-1}{2} \right]} \left\{ h \left(\frac{N-1}{2} \right) + \sum_{n=0}^{\frac{N-3}{2}} 2 h(n) \cdot \cos \left[\omega T \left(n - \frac{N-1}{2} \right) \right] \right\}$$

FIR: Linear phase

- IV) While with a –ve sign

$$H(e^{j\omega T}) = e^{-j\omega T \left\lceil \frac{N-1}{2} \right\rceil} \left\{ \sum_{n=0}^{\left\lceil \frac{N-3}{2} \right\rceil} 2j \cdot h(n) \cdot \sin \left[\omega T \left(n - \frac{N-1}{2} \right) \right] \right\}$$

- [Notice that for the antisymmetric case to have linear phase we require

$$h\left(\frac{N-1}{2}\right) = 0.$$

The phase discontinuity is as for N even]

FIR: Linear phase

- The cases most commonly used in filter design are (I) and (III), for which the amplitude characteristic can be written as a polynomial in

$$\cos \frac{\omega T}{2}$$

Summary of Properties

$$H(\omega) = e^{j\omega_0} e^{-j\omega N/2} F(\omega) \sum_{k=0}^K a_k \cos(k\omega)$$

Type	I	II	III	IV
Order N	even	odd	even	odd
Symmetry	symmetric	symmetric	anti-symmetric	anti-symmetric
Period	2π	4π	2π	4π
ω_0	0	0	$\pi/2$	$\pi/2$
$F(\omega)$	1	$\cos(\omega/2)$	$\sin(\omega)$	$\sin(\omega/2)$
K	$N/2$	$(N-1)/2$	$(N-2)/2$	$(N-1)/2$
$H(0)$	arbitrary	arbitrary	0	0
$H(\pi)$	arbitrary	0	0	arbitrary

Design of FIR filters: Windows

- (i) Start with ideal infinite duration $\{h(n)\}$
- (ii) Truncate to finite length. (This produces unwanted ripples increasing in height near discontinuity.)
- (iii) Modify to $\tilde{h}(n) = h(n) \cdot w(n)$

Weight $w(n)$ is the window

Design of FIR filters: Windows

- Simplest way of designing FIR filters
- Method is all discrete-time no continuous-time involved
- Start with ideal frequency response

$$H_d(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_d[n] e^{-j\omega n}$$

$$h_d[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(e^{j\omega}) e^{j\omega n} d\omega$$

- Choose ideal frequency response as desired response
- Most ideal impulse responses are of infinite length
- The easiest way to obtain a causal FIR filter from ideal is

$$h[n] = \begin{cases} h_d[n] & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$

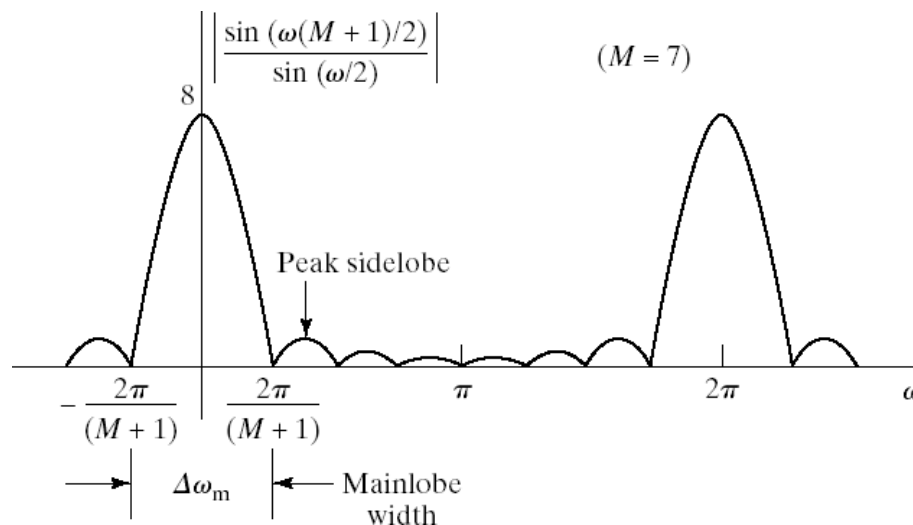
- More generally

$$h[n] = h_d[n]w[n] \quad \text{where} \quad w[n] = \begin{cases} 1 & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$

Properties of Windows

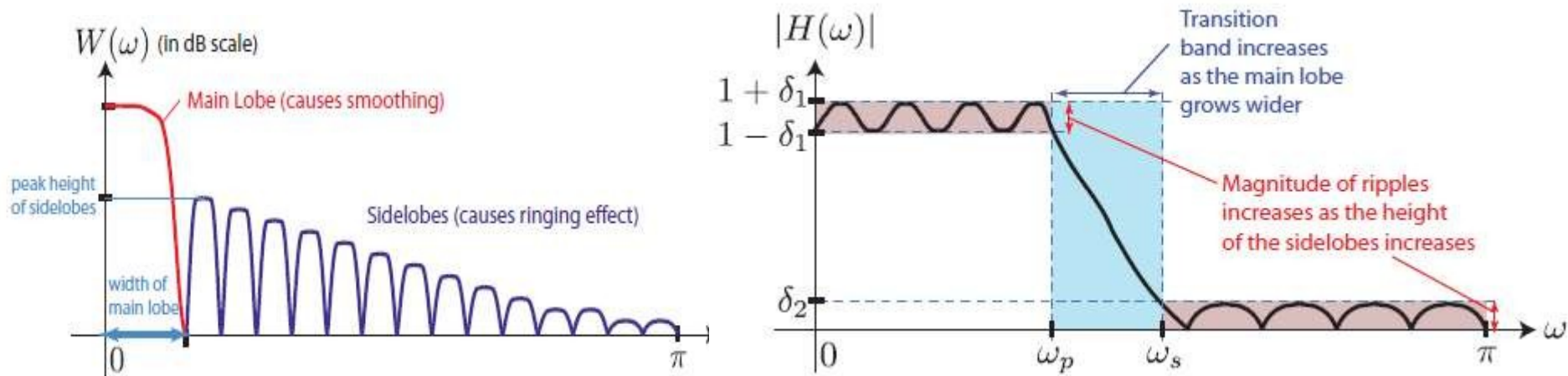
- Prefer windows that concentrate around DC in frequency
 - Less smearing, closer approximation
- Prefer window that has minimal span in time
 - Less coefficient in designed filter, computationally efficient
- So we want concentration in time and in frequency
 - Contradictory requirements
- Example: Rectangular window

$$W(e^{j\omega}) = \sum_{n=0}^M e^{-j\omega n} = \frac{1 - e^{-j\omega(M+1)}}{1 - e^{-j\omega}} = e^{-j\omega M/2} \frac{\sin[\omega(M+1)/2]}{\sin[\omega/2]}$$



Windowing distortion

- increasing window length generally reduces the width of the main lobe
- peak of sidelobes is generally independent of M



Windows

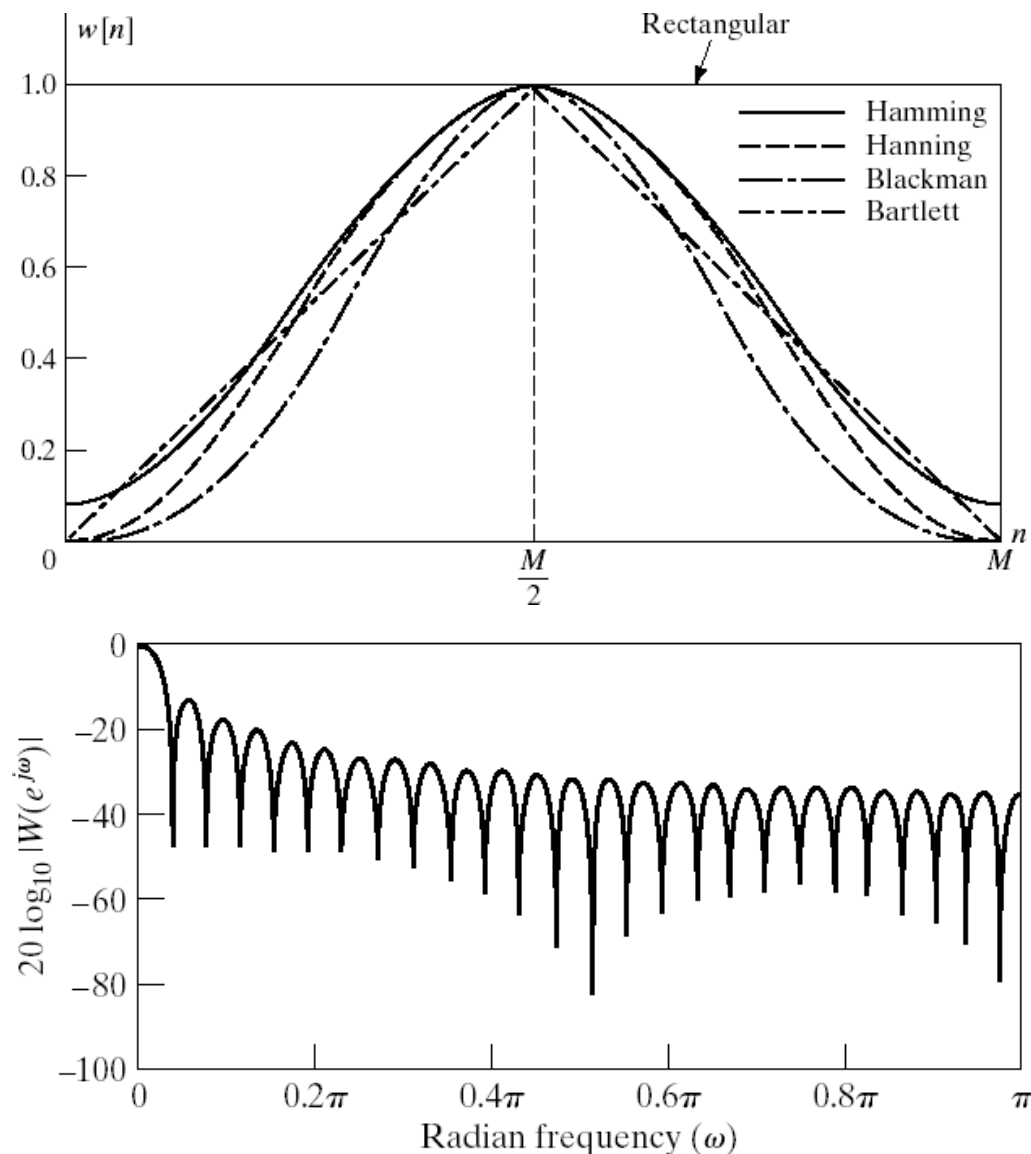
Commonly used windows

- Rectangular
$$1 \quad |n| < \frac{N-1}{2}$$
- Bartlett
$$1 - \frac{2|n|}{N} \quad |n| < \frac{N-1}{2}$$
- Hanning
$$0.5 + 0.5 \cos \left(\frac{2\pi n}{N} \right)$$
- Hamming
$$0.54 + 0.46 \cos \left(\frac{2\pi n}{N} \right)$$
-
- Blackman
$$0.42 + 0.5 \cos \left(\frac{2\pi n}{N} \right) + 0.08 \cos \left(\frac{4\pi n}{N} \right)$$
-
- Kaiser
$$J_0 \left[\beta \sqrt{1 - \left(\frac{2n}{N-1} \right)^2} \right] / J_0(\beta)$$

Rectangular Window

- Narrowest main lobe
 - $4\pi/(M+1)$
 - Sharpest transitions at discontinuities in frequency
- Large side lobes
 - -13 dB
 - Large oscillation around discontinuities
- Simplest window possible

$$w[n] = \begin{cases} 1 & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$



Bartlett (Triangular) Window

- Medium main lobe

$$- 8\pi/M$$

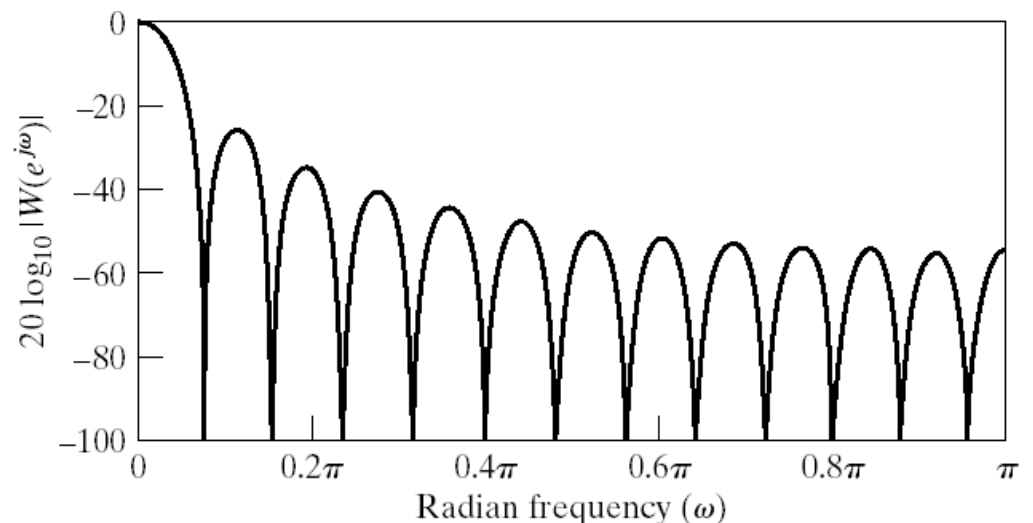
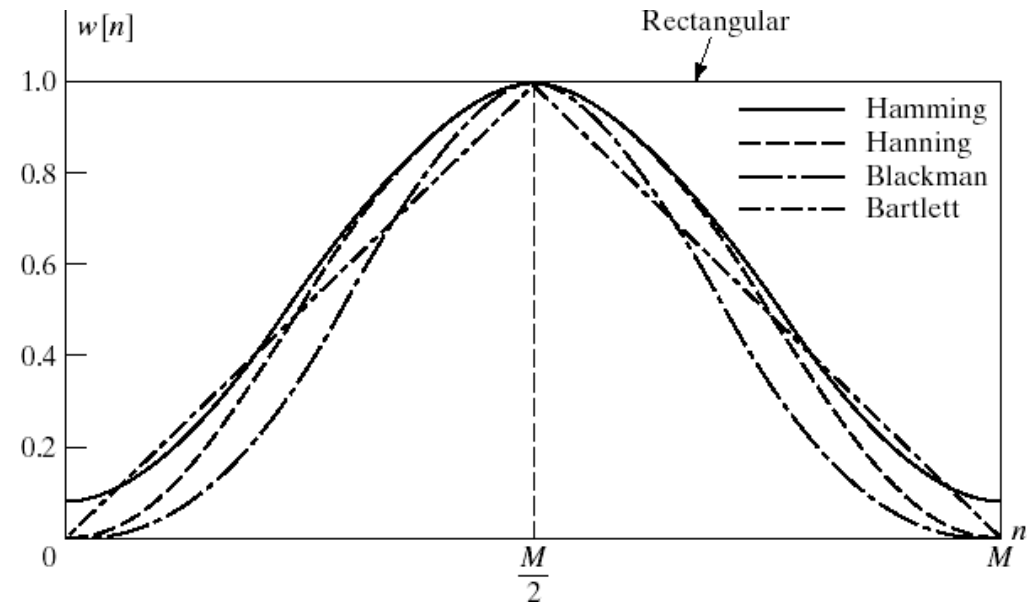
- Side lobes

$$- -25 \text{ dB}$$

- Hamming window performs better

- Simple equation

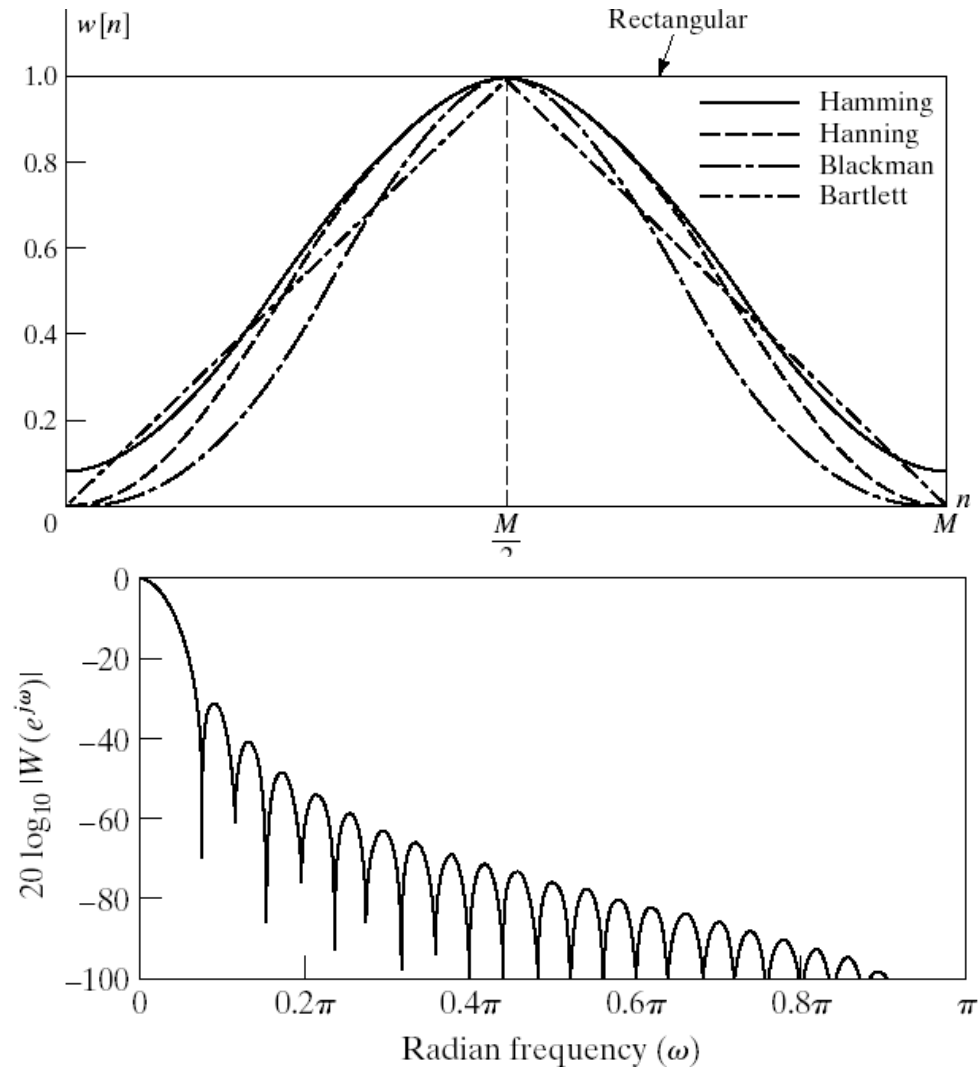
$$w[n] = \begin{cases} 2n/M & 0 \leq n \leq M/2 \\ 2 - 2n/M & M/2 \leq n \leq M \\ 0 & \text{else} \end{cases}$$



Hanning Window

- Medium main lobe
 - $8\pi/M$
- Side lobes
 - -31 dB
- Hamming window performs better
- Same complexity as Hamming

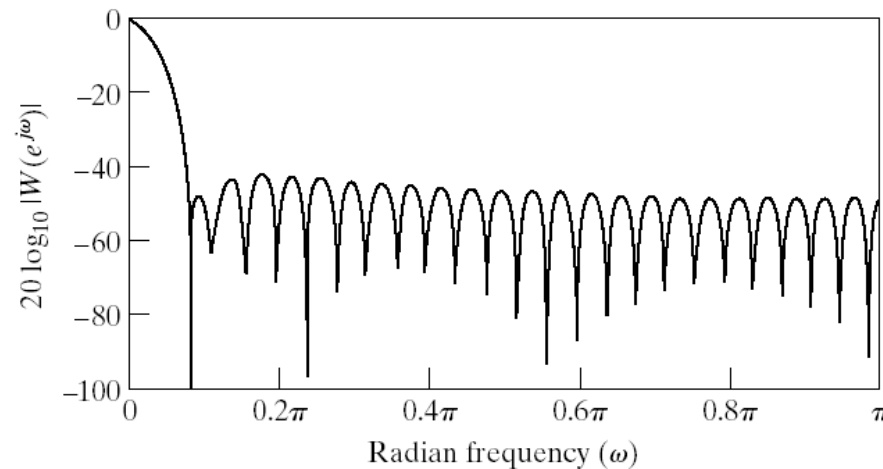
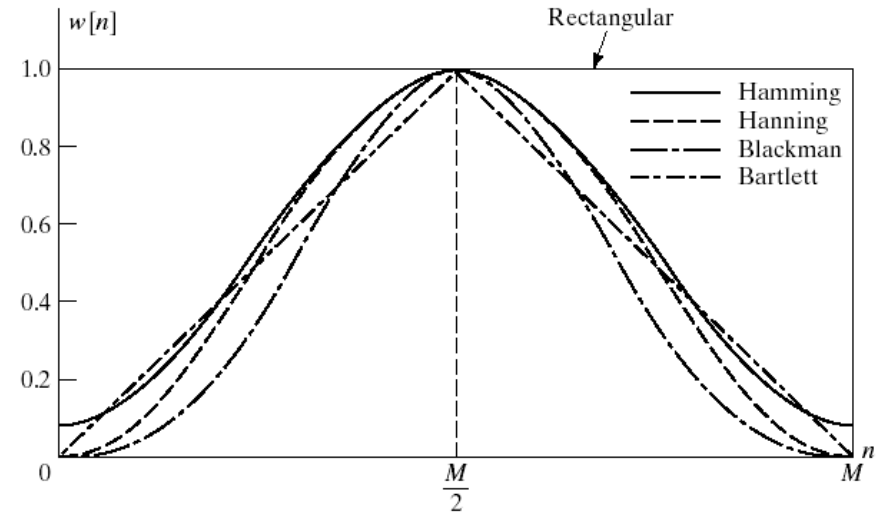
$$w[n] = \begin{cases} \frac{1}{2} \left[1 - \cos \left(\frac{2\pi n}{M} \right) \right] & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$



Hamming Window

- Medium main lobe
 - $8\pi/M$
- Good side lobes
 - -41 dB
- Simpler than Blackman

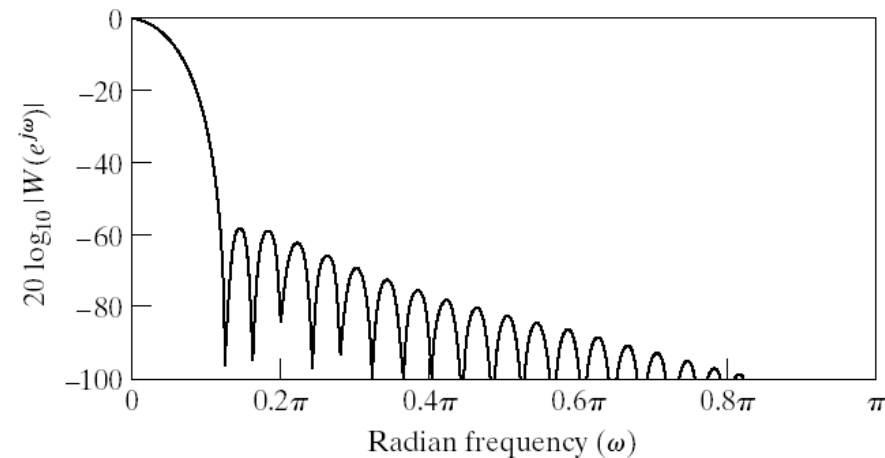
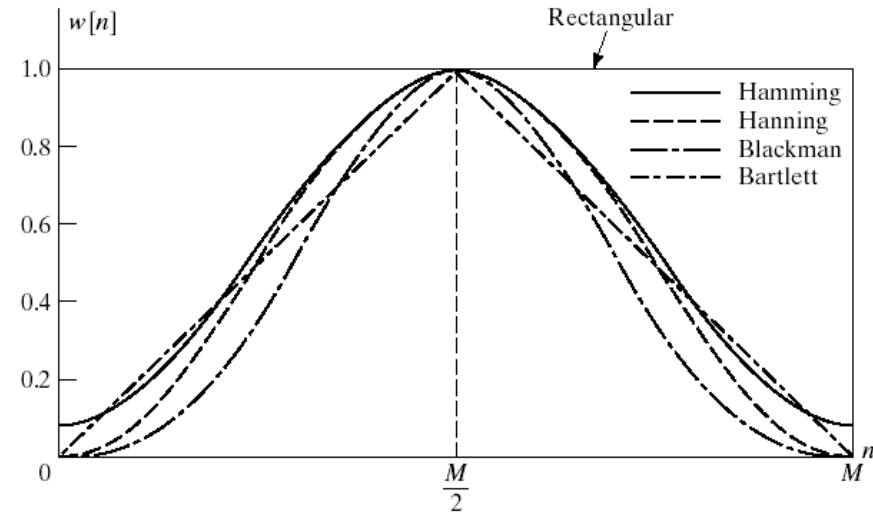
$$w[n] = \begin{cases} 0.54 - 0.46 \cos\left(\frac{2\pi n}{M}\right) & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$



Blackman Window

- Large main lobe
 - $12\pi/M$
- Very good side lobes
 - -57 dB
- Complex equation

$$w[n] = \begin{cases} 0.42 - 0.5 \cos\left(\frac{2\pi n}{M}\right) + 0.08 \cos\left(\frac{4\pi n}{M}\right) & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$

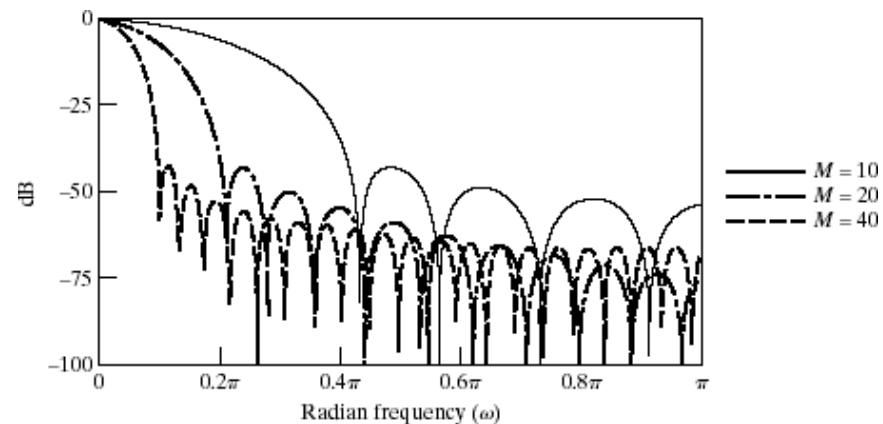
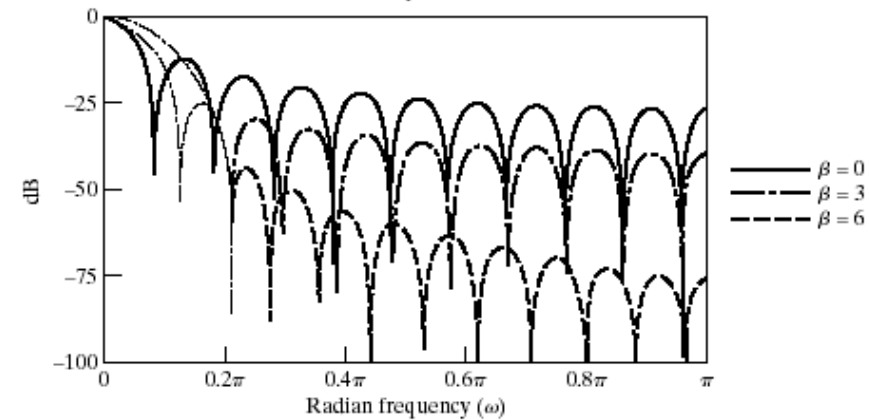
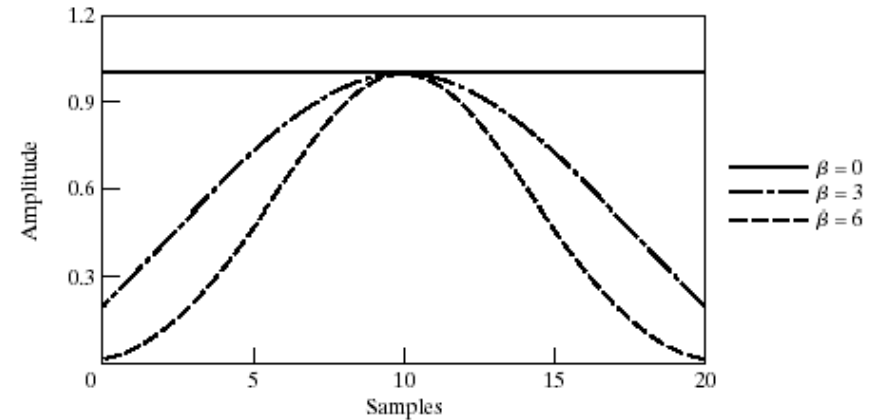


Kaiser Window Filter Design Method

- Parameterized equation forming a set of windows
 - Parameter to change main-lobe width and side-lobe area trade-off

$$w[n] = \begin{cases} \frac{I_0 \left[\beta \sqrt{1 - \left(\frac{n - M/2}{M/2} \right)^2} \right]}{I_0(\beta)} & 0 \leq n \leq M \\ 0 & \text{else} \end{cases}$$

- $I_0(\cdot)$ represents zeroth-order modified Bessel function of 1st kind



Comparison of windows

COMPARISON OF COMMONLY USED WINDOWS

Type of Window	Peak Side-Lobe Amplitude (Relative)	Approximate Width of Main Lobe	Peak Approximation Error, $20 \log_{10} \delta$ (dB)	Equivalent Kaiser Window, β	Transition Width of Equivalent Kaiser Window
Rectangular	-13	$4\pi/(M+1)$	-21	0	$1.81\pi/M$
Bartlett	-25	$8\pi/M$	-25	1.33	$2.37\pi/M$
Hanning	-31	$8\pi/M$	-44	3.86	$5.01\pi/M$
Hamming	-41	$8\pi/M$	-53	4.86	$6.27\pi/M$
Blackman	-57	$12\pi/M$	-74	7.04	$9.19\pi/M$

Kaiser window

- Kaiser window

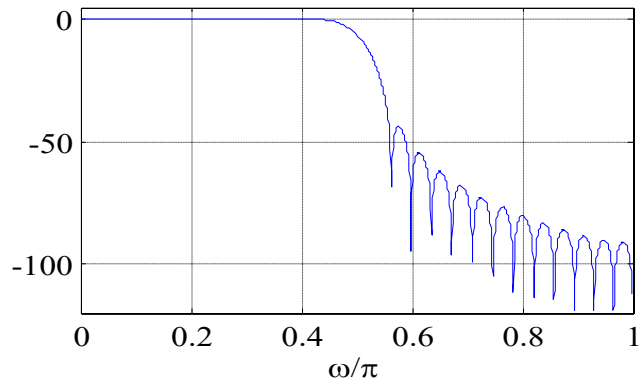
β	Transition width (Hz)	Min. stop attn dB
2.12	$1.5/N$	30
4.54	$2.9/N$	50
6.76	$4.3/N$	70
8.96	$5.7/N$	90

Example

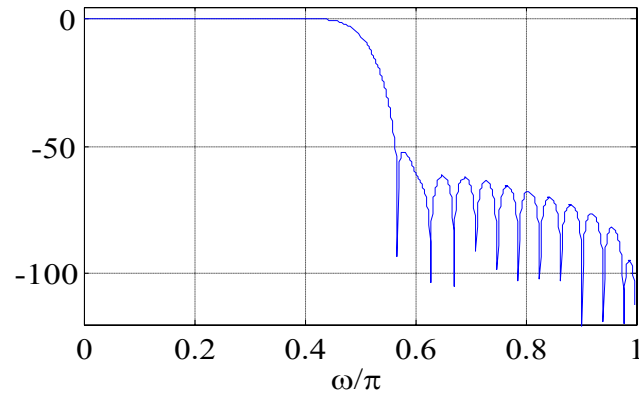
- Lowpass filter of length 51 and

$$\omega_c = \pi / 2$$

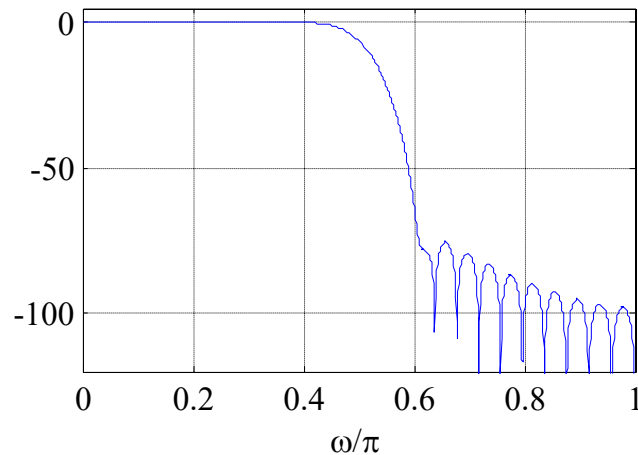
Lowpass Filter Designed Using Hann window



Lowpass Filter Designed Using Hamming window



Lowpass Filter Designed Using Blackman window



Frequency Sampling Method

- In this approach we are given $H(k)$ and need to find $H(z)$
- This is an interpolation problem and the solution is given in the DFT part of the course

$$H(z) = \frac{1}{N} \sum_{k=0}^{N-1} H(k) \cdot \frac{1 - z^{-N}}{1 - e^{j\frac{2\pi}{N}k} \cdot z^{-1}}$$

- It has similar problems to the windowing approach

FIR Digital Filter Order Estimation

Kaiser's Formula:

$$N \cong \frac{-20 \log_{10} (\sqrt{\delta_p \delta_s}) - 13}{14.6 (\omega_s - \omega_p) / 2\pi} + 1$$

- ie N is inversely proportional to transition band width and not on transition band location

UNIT-5

Multirate signal processing & Finite Word length Effects

Single vs Multirate Processing

- **Single-rate processing:** the digital samples before and after processing correspond to the same sampling frequency with respect to (w.r.t.) the analog counterpart.
 - e.g.: LTI filtering can be characterized by the freq. response.
- **The need of multi-rate:**
 - fractional sampling rate conversion in all-digital domain:
 - e.g. 44.1kHz CD rate \iff 48kHz studio rate
- **The advantages of multi-rate signal processing:**
 - Reduce storage and computational cost
 - e.g.: polyphase implementation
 - Perform the processing in all-digital domain without using analog as an intermediate step that can:
 - bring inaccuracies – not perfectly reproducible
 - increase system design / implementation complexity

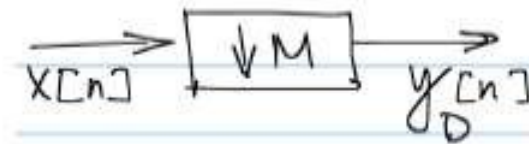
Basic Multirate operations: Decimation and Interpolation

- Building blocks for traditional single-rate digital signal processing: multiplier (with a constant), adder, delay, multiplier (of 2 signals)

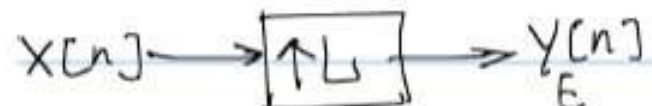


- New building blocks in multi-rate signal processing:

M -fold decimator

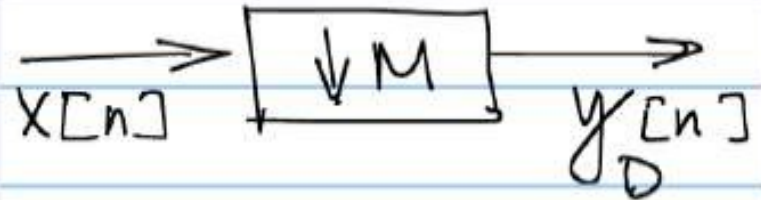


L -fold expander

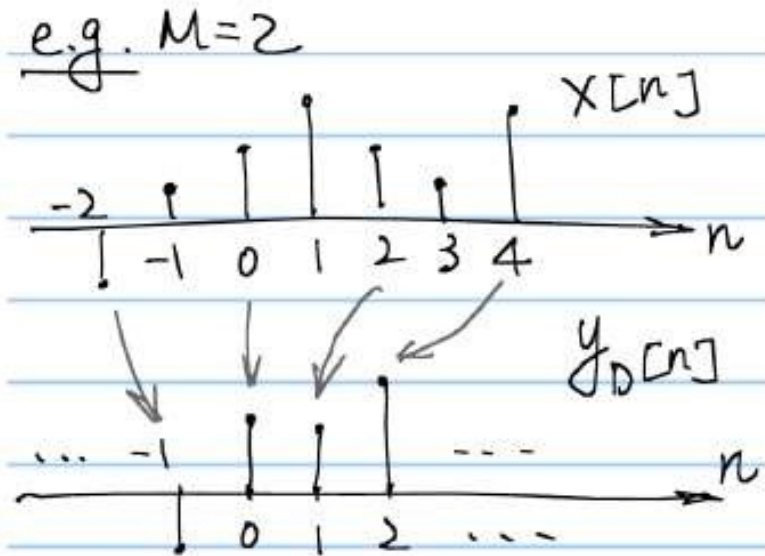


M-fold Decimator

$$y_D[n] = x[Mn], M \in \mathbb{N}$$



Corresponding to the physical time scale, it is as if we sampled the original signal in a slower rate when applying decimation.

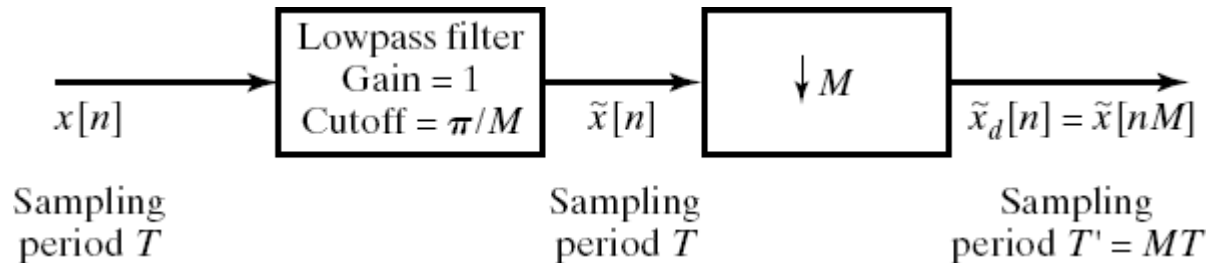


Sampling Rate Reduction by an Integer Factor: Downsampling

- We reduce the sampling rate of a sequence by “sampling” it

$$x_d[n] = x[nM] = x_c(nMT)$$

- This is accomplished with a sampling rate compressor



- We obtain $x_d[n]$ that is identical to what we would get by reconstructing the signal and resampling it with $T'=MT$
- There will be no aliasing if

$$\frac{\pi}{T'} = \frac{\pi}{MT} > \Omega_N$$

Frequency Domain Representation of Downsampling

- Recall the DTFT of $x[n]=x_c(nT)$

$$X(e^{j\omega}) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{T} - \frac{2\pi k}{T} \right) \right)$$

- The DTFT of the downsampled signal can similarly written as

$$X_d(e^{j\omega}) = \frac{1}{T'} \sum_{r=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{T'} - \frac{2\pi r}{T'} \right) \right) = \frac{1}{MT} \sum_{r=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{MT} - \frac{2\pi r}{MT} \right) \right)$$

- Let's represent the summation index as

$$r = i + kM \quad \text{where} \quad -\infty < k < \infty \quad \text{and} \quad 0 \leq i < M$$

$$X_d(e^{j\omega}) = \frac{1}{M} \sum_{i=0}^{M-1} \left[\frac{1}{T} \sum_{r=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{MT} - \frac{2\pi k}{T} - \frac{2\pi i}{MT} \right) \right) \right]$$

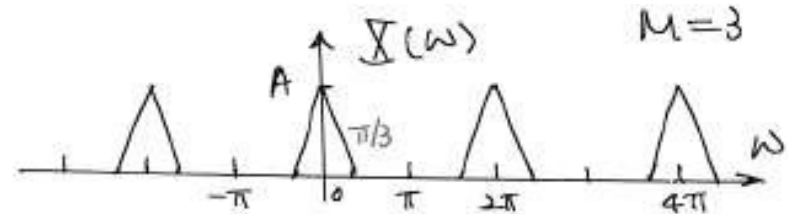
- And finally

$$X_d(e^{j\omega}) = \frac{1}{M} \sum_{i=0}^{M-1} X \left(e^{j \left(\frac{\omega}{M} - \frac{2\pi i}{M} \right)} \right)$$

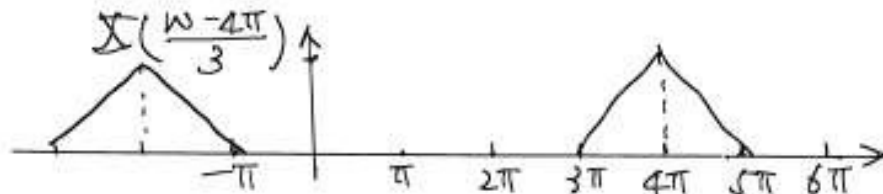
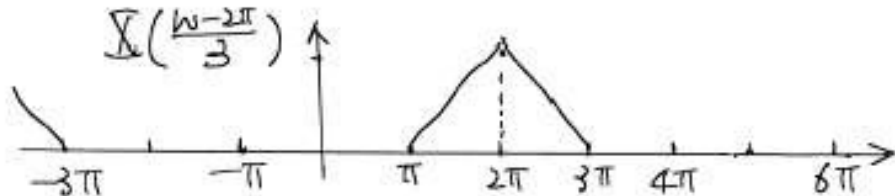
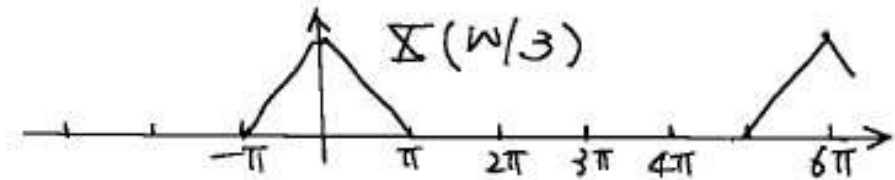
Frequency Domain Representation of Downsampling

Interpretation of $Y_D(\omega)$

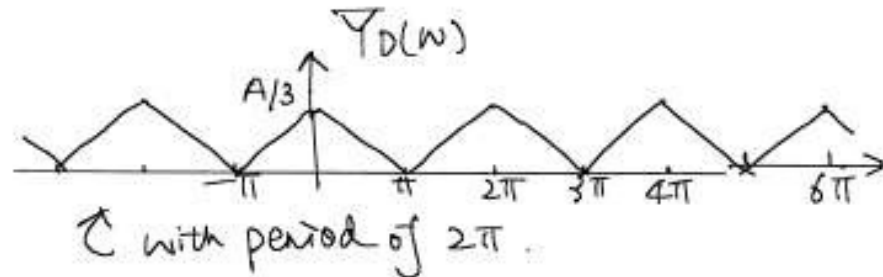
Step-1: stretch $X(\omega)$ by a factor of M to obtain $X(\omega/M)$



Step-2: create $M - 1$ copies and shift them in successive amounts of 2π



Step-3: add all M copies together and multiply by $1/M$.



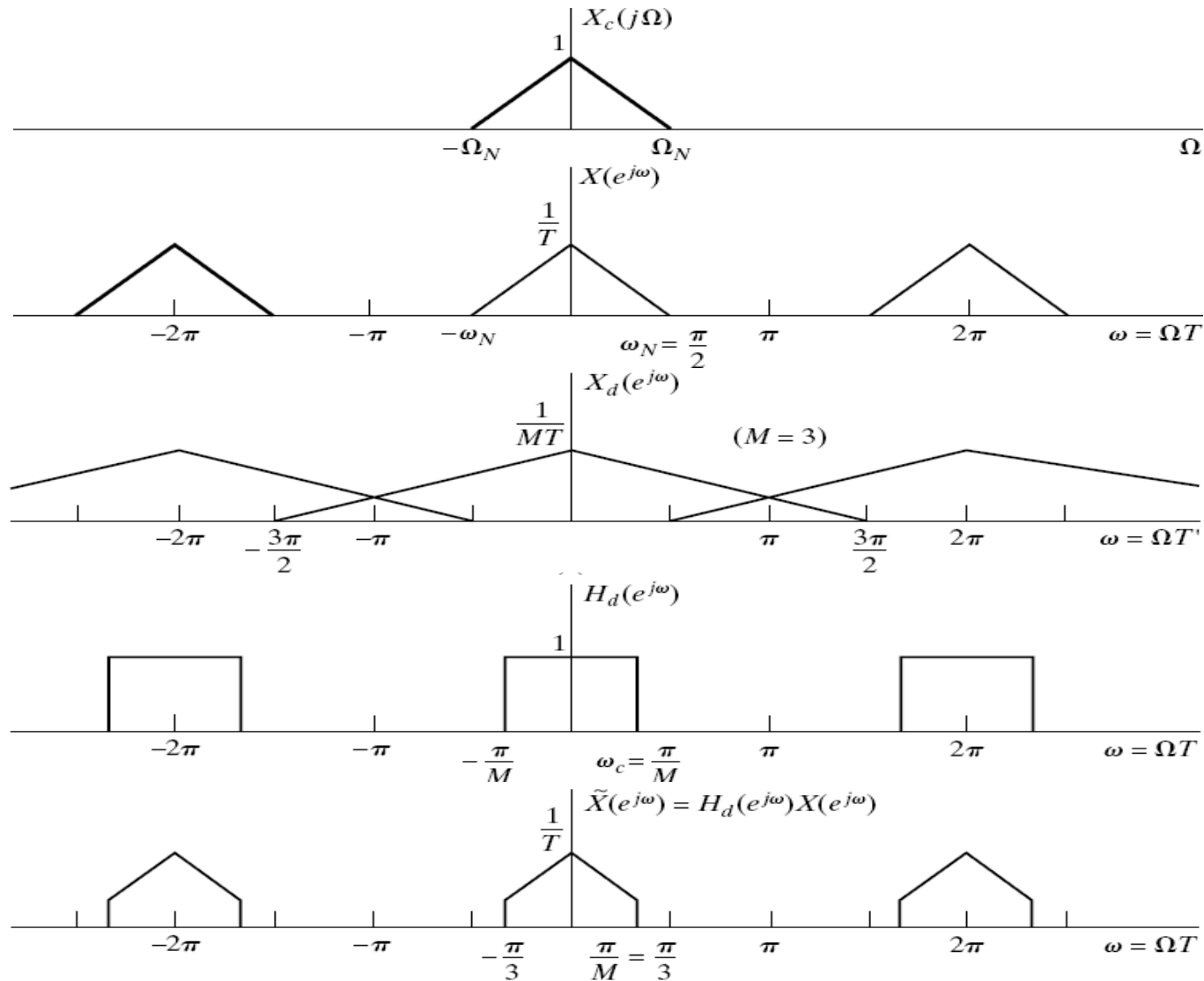
Aliasing

- The stretched version $X(\omega/M)$ can in general overlap with its shifted replicas. This overlap effect is called aliasing.
- When aliasing occurs, we cannot recover $x[n]$ from the decimated version $y_D[n]$, i.e. $\downarrow M$ can be a lossy operation.
- We can avoid aliasing by limiting the bandwidth of $x[n]$ to

$$|\omega| < \pi/M.$$

- When no aliasing, we can recover $x[n]$ from the decimated version $y_D[n]$ by using an expander, followed by filtering of the unwanted spectrum images.

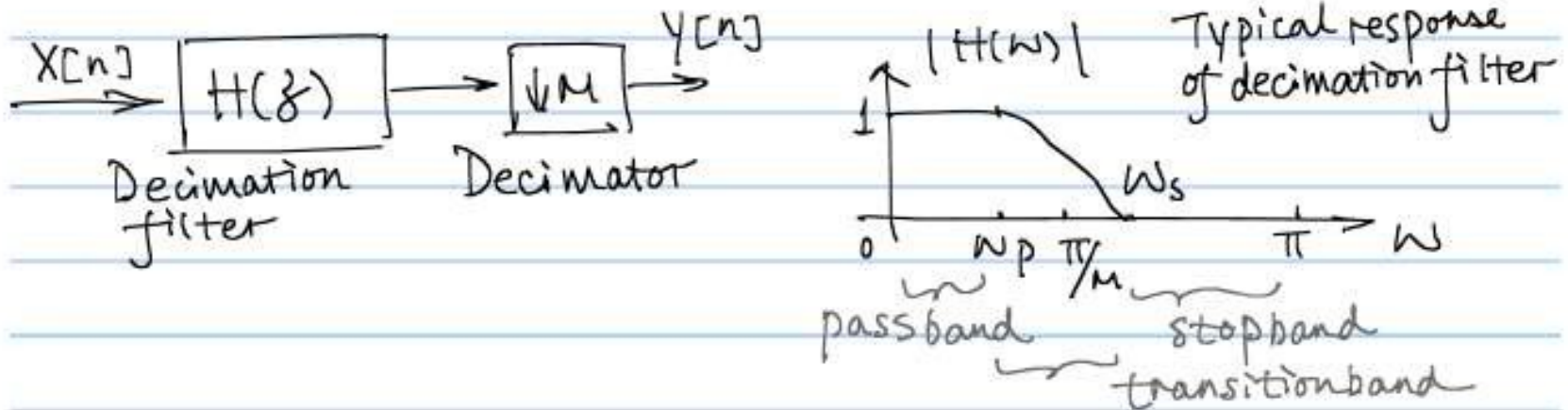
Frequency Domain Representation of Downsampling w/ Prefilter



Decimation filter

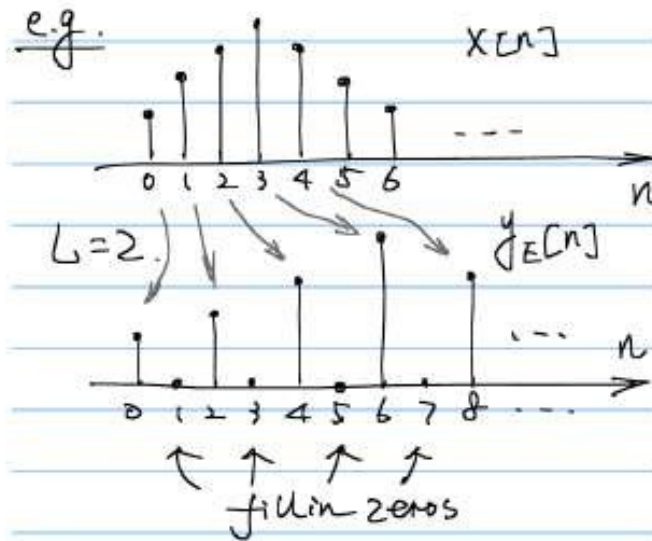
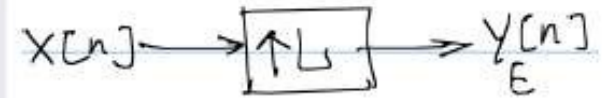
The decimator is normally preceded by a lowpass filter called decimator filter.

Decimator filter ensures the signal to be decimated is bandlimited and controls the extent of aliasing.



L-fold Interpolator

$$y_E[n] = \begin{cases} x[n/L] & \text{if } n \text{ is integer multiple of } L \in \mathbb{N} \\ 0 & \text{otherwise} \end{cases}$$



Question: Can we recover $x[n]$ from $y_E[n]$? \rightarrow Yes.

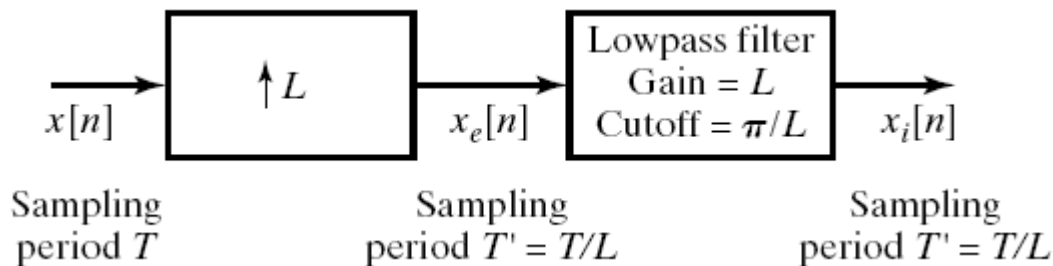
The expander does not cause loss of information.

Increasing the Sampling Rate by an Integer Factor: Upsampling

- We increase the sampling rate of a sequence interpolating it

$$x_i[n] = x[n/L] = x_c(nT/L)$$

- This is accomplished with a sampling rate expander



- We obtain $x_i[n]$ that is identical to what we would get by reconstructing the signal and resampling it with $T' = T/L$
- Upsampling consists of two steps

– Expanding

$$x_e[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{else} \end{cases} = \sum_{k=-\infty}^{\infty} x[k] \delta[n - kL]$$

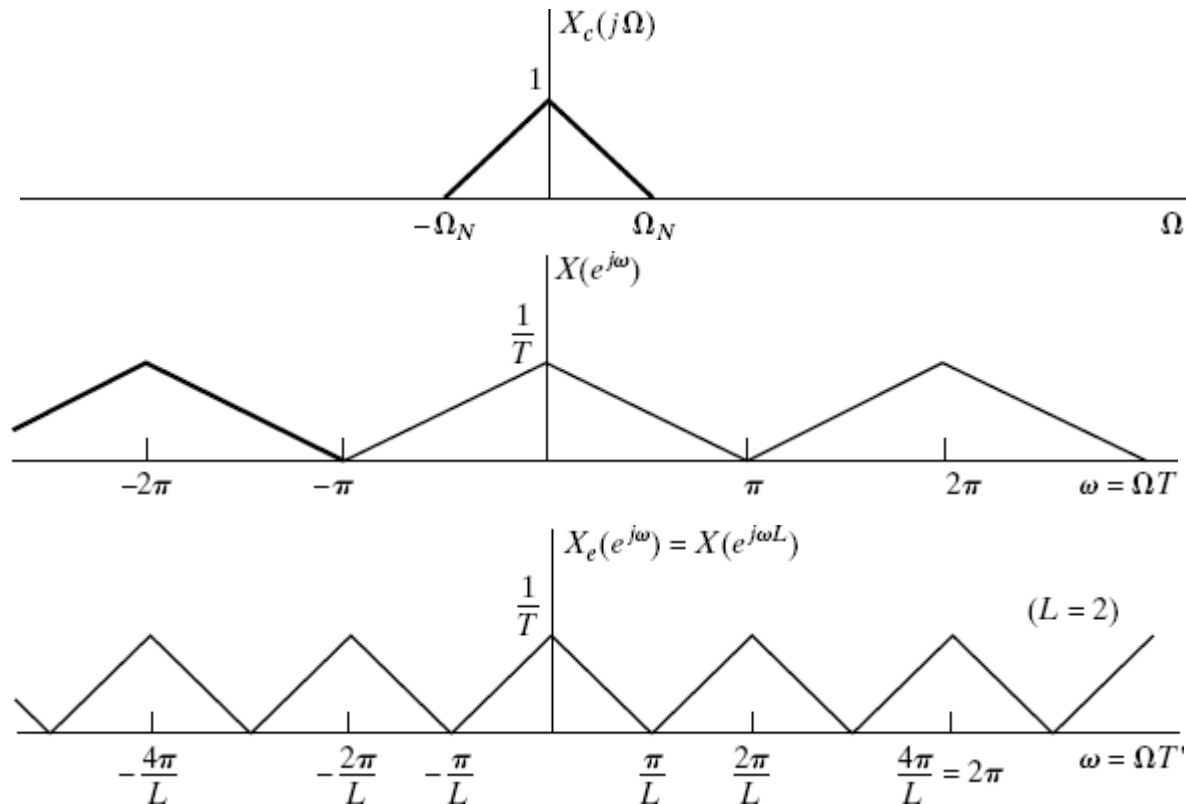
– Interpolating

Frequency Domain Representation of Expander

- The DTFT of $x_e[n]_{-\infty}^{\infty}$ can be written as

$$X_e(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \left[\sum_{k=-\infty}^{\infty} x[k] \delta[n - kL] \right] e^{-j\omega n} = \sum_{k=-\infty}^{\infty} x[k] e^{-j\omega Lk} = X(e^{j\omega L})$$

- The output of the expander is frequency-scaled



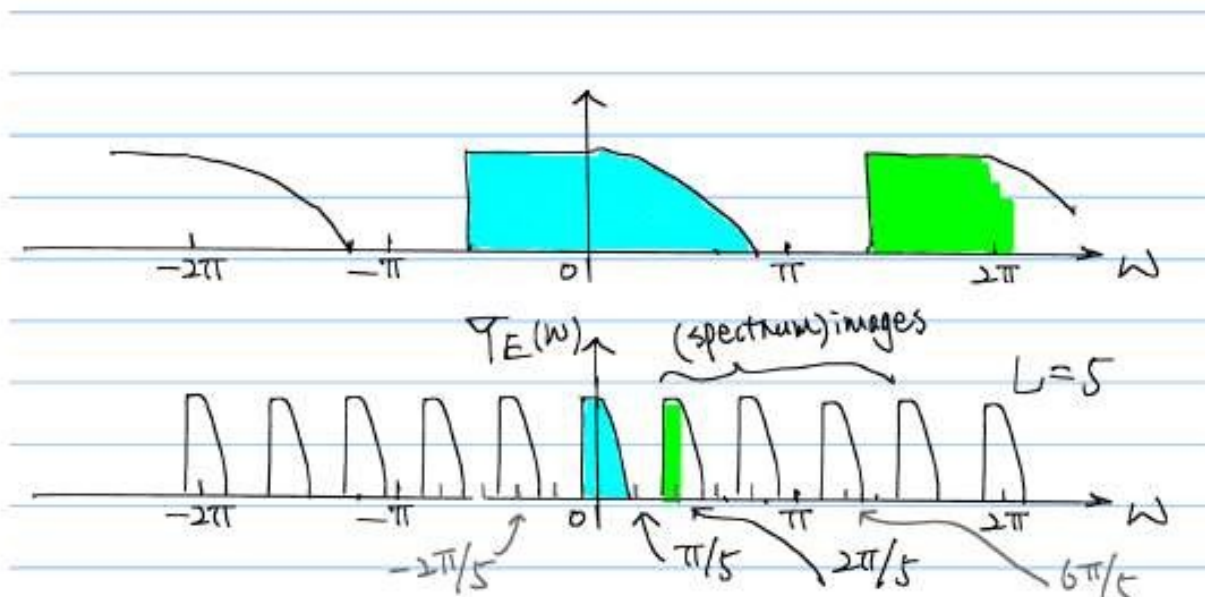
Input-output relation on the Spectrum

$$Y_E(z) = X(z^L)$$

Evaluating on the unit circle, the Fourier Transform relation is:

$$Y_E(e^{j\omega}) = X(e^{j\omega L}) \Rightarrow Y_E(\omega) = X(\omega L)$$

i.e. L -fold compressed version of $X(\omega)$ along ω



Periodicity and spectrum images

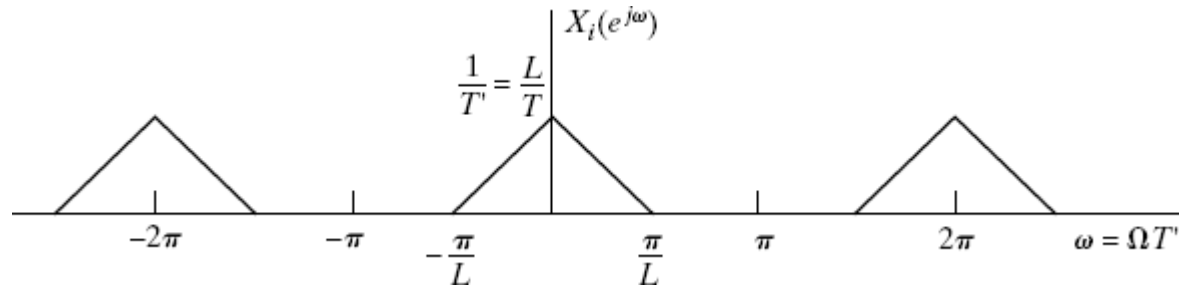
The Fourier Transform of a discrete-time signal has period of 2π .
With expander, $X(\omega L)$ has a period of $2\pi/L$.

The multiple copies of the compressed spectrum over one period of 2π are called images.

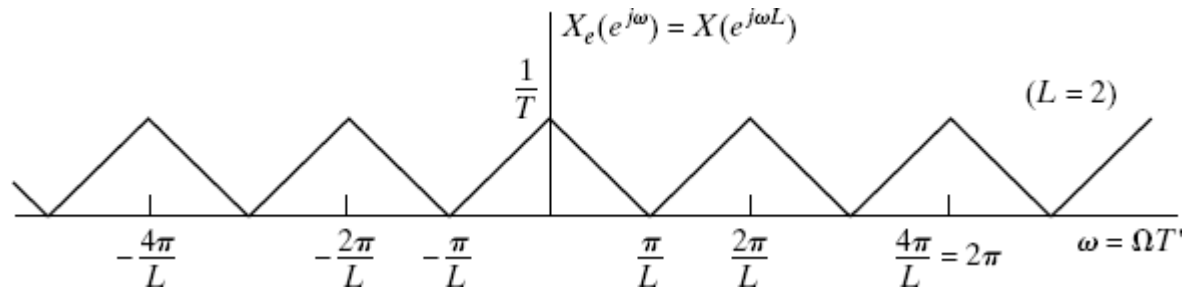
And we say the expander creates an imaging effect.

Frequency Domain Representation of Interpolator

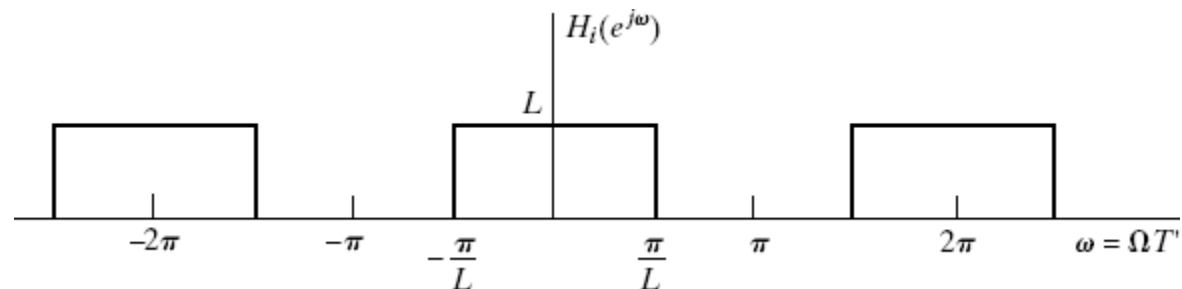
- The DTFT of the desired interpolated signals is



- The extrapolator output is given as

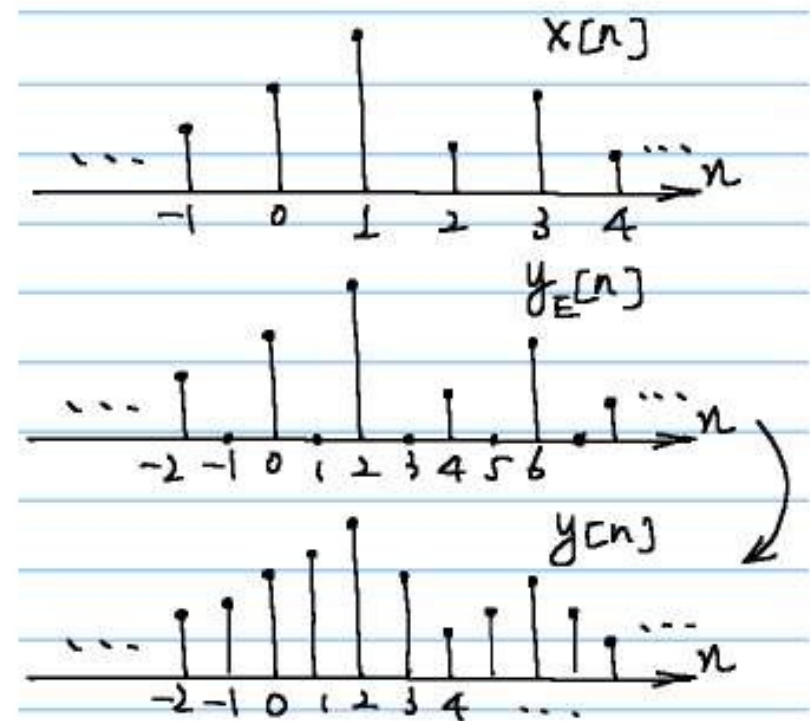
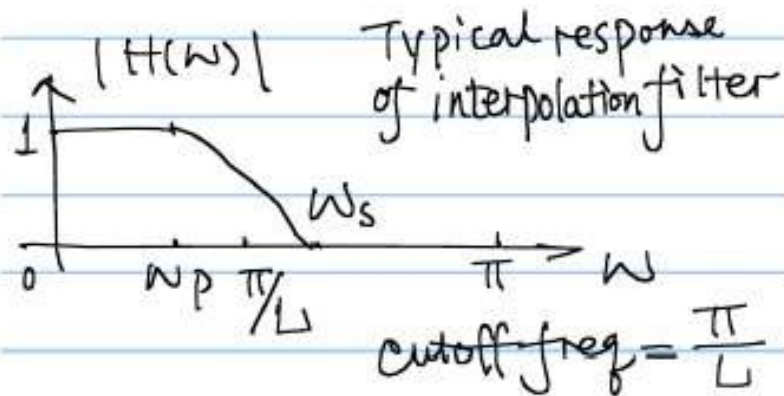
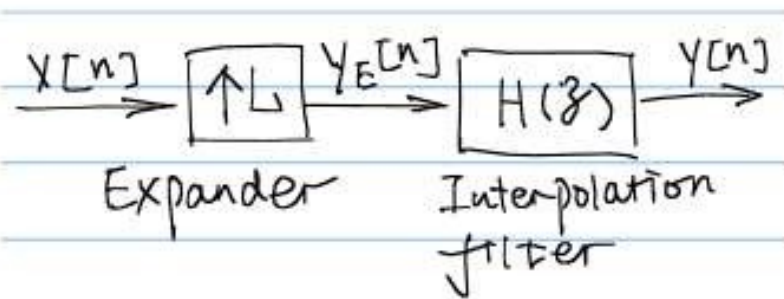


- To get interpolated signal we apply the following LPF



Interpolation filters

An interpolation filter normally follows an expander to suppress all the images in the spectrum.



time-domain interpretation

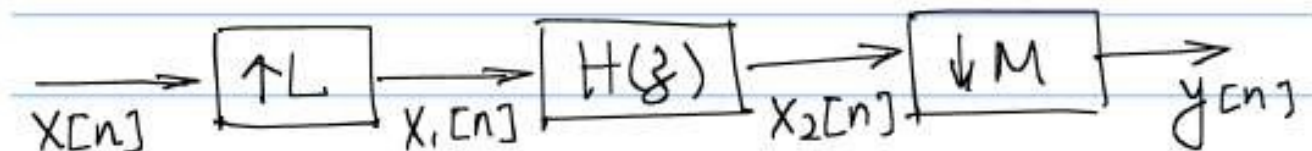
Fractional sampling rate convertor

So far, we have learned how to increase or decrease sampling rate in the digital domain by integer factors.

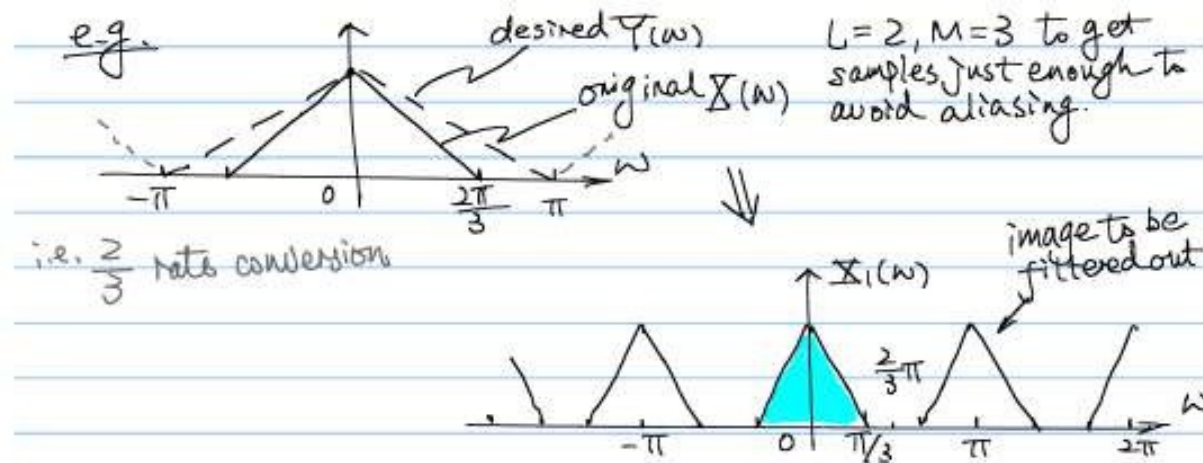
Question: How to change the rate by a rational fraction L/M ?
(e.g.: audio 44.1kHz \longleftrightarrow 48kHz)

- Method-1: convert into an analog signal and resample
- Method-2: directly in digital domain by judicious combination of interpolation and decimation

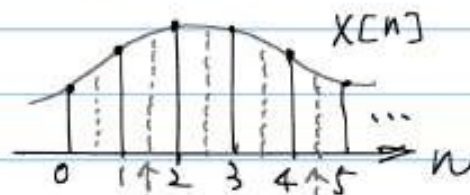
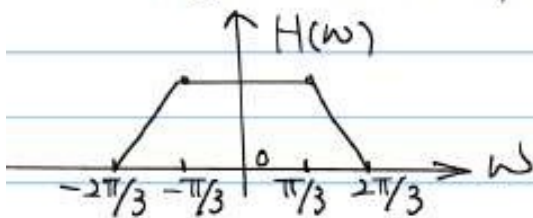
Question: Decimate first or expand first? And why?



Fractional sampling rate convertor



Use a low pass filter with passband greater than $\pi/3$ and stopband edge before $2\pi/3$ to remove images

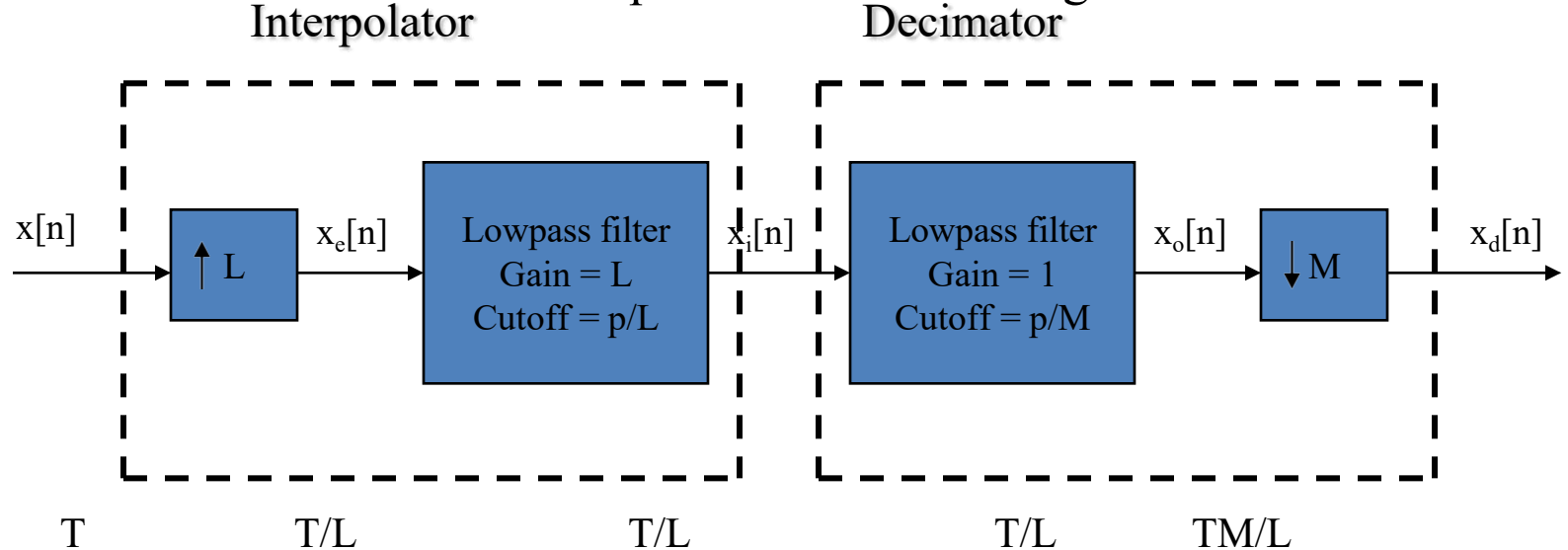


Equiv. to getting 2 samples out of every 3 original samples

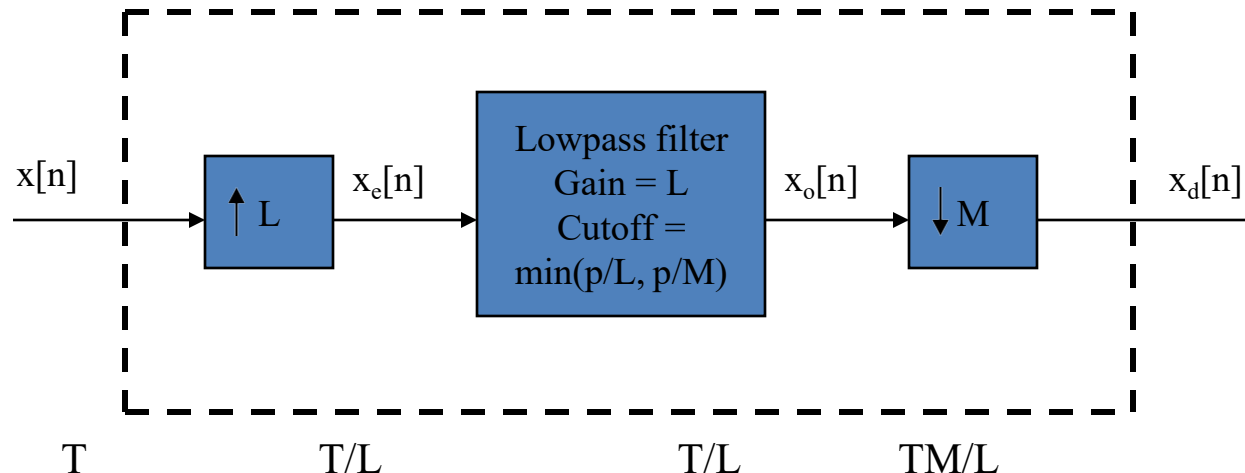
- the signal now is critically sampled
- some samples kept are interpolated from $x[n]$

Changing the Sampling Rate by Non-Integer Factor

- Combine decimation and interpolation for non-integer factors



- The two low-pass filters can be combined into a single one



Time Domain

- $x_i[n]$ is a low-pass filtered version of $x[n]$
- The low-pass filter impulse response is

$$h_i[n] = \frac{\sin(\pi n / L)}{\pi n / L}$$

- Hence the interpolated signal is written as

$$x_i[n] = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin(\pi(n - kL) / L)}{\pi(n - kL) / L}$$

- Note that

$$h_i[0] = 1$$

$$h_i[n] = 0 \quad n = \mp L, \mp 2L, \dots$$

- Therefore the filter output can be written as

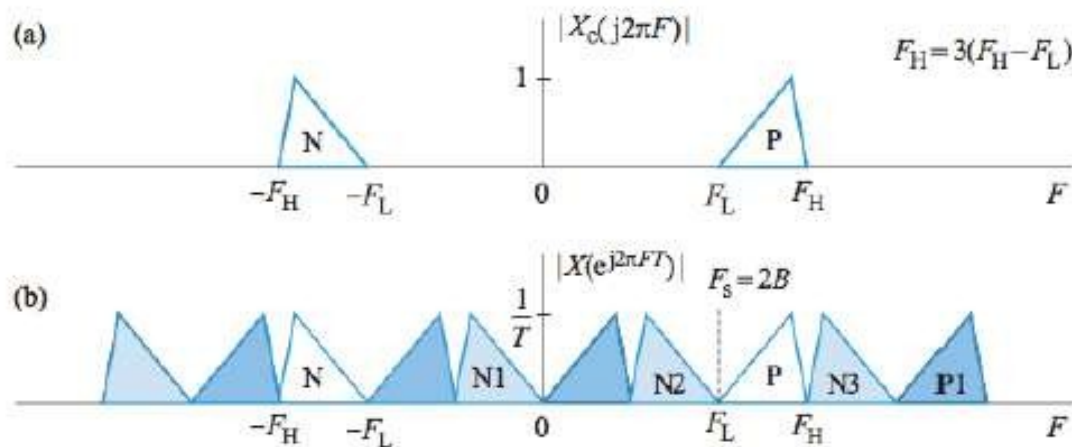
$$x_i[n] = x[n / L] = x_c(nT / L) = x_c(nT') \quad \text{for } n = 0, \mp L, \mp 2L, \dots$$

$$y[n] = \begin{cases} \sum_{k=-\infty}^{\infty} x[k] h[nM - k] & \text{M-fold decimation filter} \\ \sum_{k=-\infty}^{\infty} x[k] h[n - kL] & \text{L-fold interpolation filter} \\ \sum_{k=-\infty}^{\infty} x[k] h[nM - kL] & \text{M/L-fold decimation filter} \end{cases}$$

Sampling of bandpass signals

- Let $x_c(t)$ be a real-valued signal that is band-limited to the range (Ω_L, Ω_H) , viz. $X(j\Omega) = 0$ for $|\Omega| < \Omega_L$ and $|\Omega| > \Omega_H$.
- Such $x_c(t)$ is called a *bandpass signal* with centre frequency $\Omega_C = (\Omega_L + \Omega_H)/2$ and bandwidth (Hz) $B = (\Omega_H - \Omega_L)/2\pi$.
- Let us first assume that there exists an integer $K > 0$ such that $\Omega_H = 2\pi KB$ and let us set $\Omega_s = 2(2\pi B)$.
- Recall that the spectrum of the sampled signal $x[n] = x_c(nT)$ is defined as

$$X(e^{j\Omega T}) = \frac{1}{T} \sum_k X_c(j(\Omega - k\Omega_s))$$



Sampling of bandpass signals

- We notice that if we multiply $X(e^{j\Omega T})$ by the Fourier transform $G_r(j\Omega)$, we can recover $X_c(j\Omega)$, and hence $x_c(t)$, exactly.
- The ideal reconstruction process is given by

$$x_c(t) = \sum_n x_c(nT) g_r(t - nT)$$

where $g_r(t)$ is the *modulated ideal band-limited interpolation function* given by

$$g_r(t) = 2 \frac{\sin(B\pi t)}{\pi t} \cos(\Omega_C t)$$

where B is the bandwidth measured in Hz.

Conclusion

A sampling rate of $F_s = 2(F_H - F_L)$ is adequate for alias-free sampling of a bandpass signal if the ratio $K = F_H/(F_H - F_L)$ is exactly an integer.

Over sampling -ADC

Consider a Nyquist rate ADC in which the signal is sampled at the desired precision and at a rate such that Nyquist's sampling criterion is just satisfied.

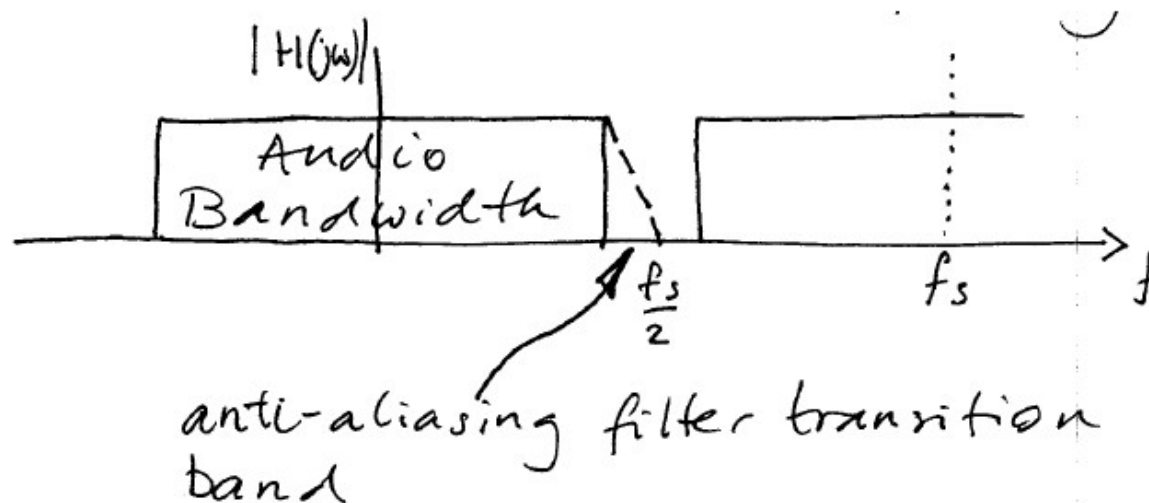
- Bandwidth for audio is $20 \text{ Hz} < f < 20 \text{ kHz}$
- Antialiasing filter required has very demanding specification

$$|H(j\omega)| = 0 \text{ dB}, f < 20 \text{ kHz}$$

$$|H(j\omega)| < 96 \text{ dB}, f \geq \frac{44.1}{2} \text{ kHz}$$

- Requires high order analogue filter such as elliptic filters that have very nonlinear phase characteristics
 - hard to design, expensive and bad for audio quality.

Nyquist Rate Conversion Anti-aliasing Filter.



Consider oversampling the signal at, say, 64 times the Nyquist rate but with lower precision. Then use multirate techniques to convert sample rate back to 44.1 kHz with full precision.

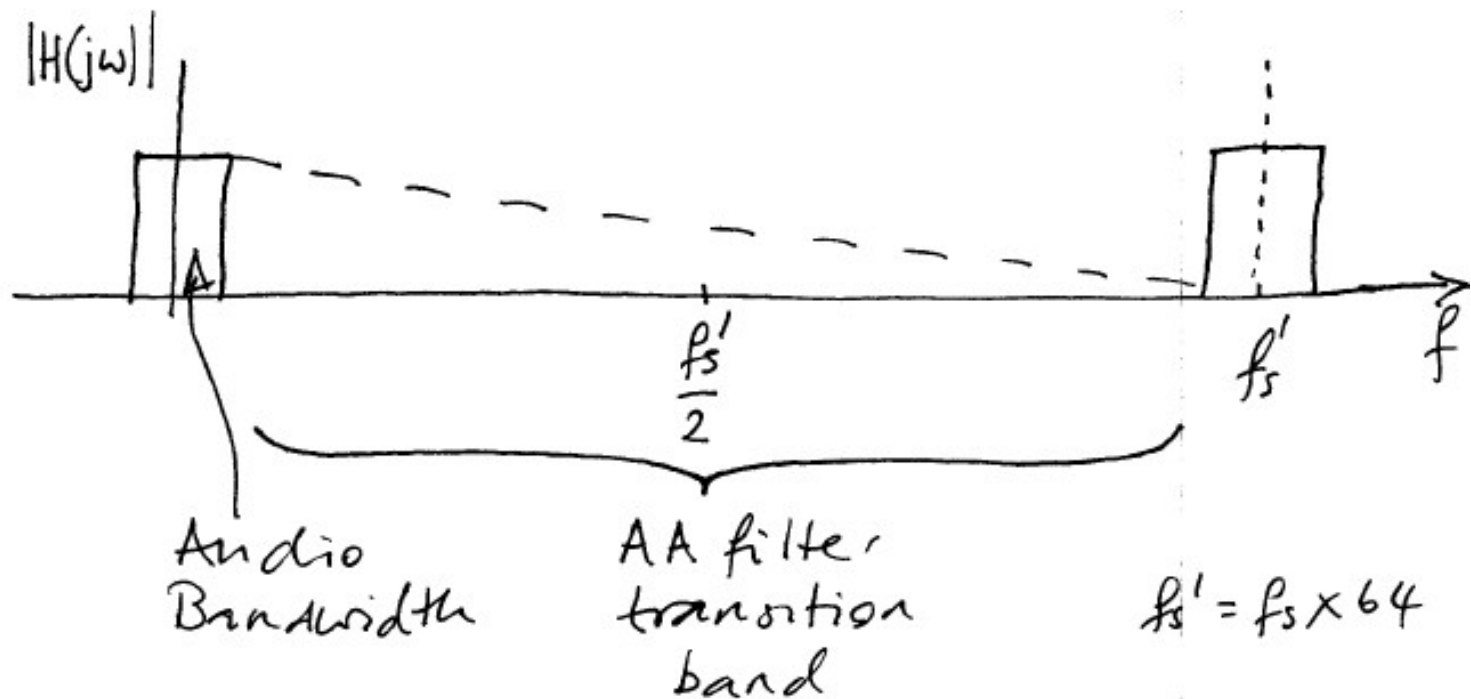
- New (over-sampled) sampling rate is 44.1×64 kHz.
- Requires simple antialiasing filter

$$|H(j\omega)| = 0 \text{ dB}, f < 20 \text{ kHz}$$

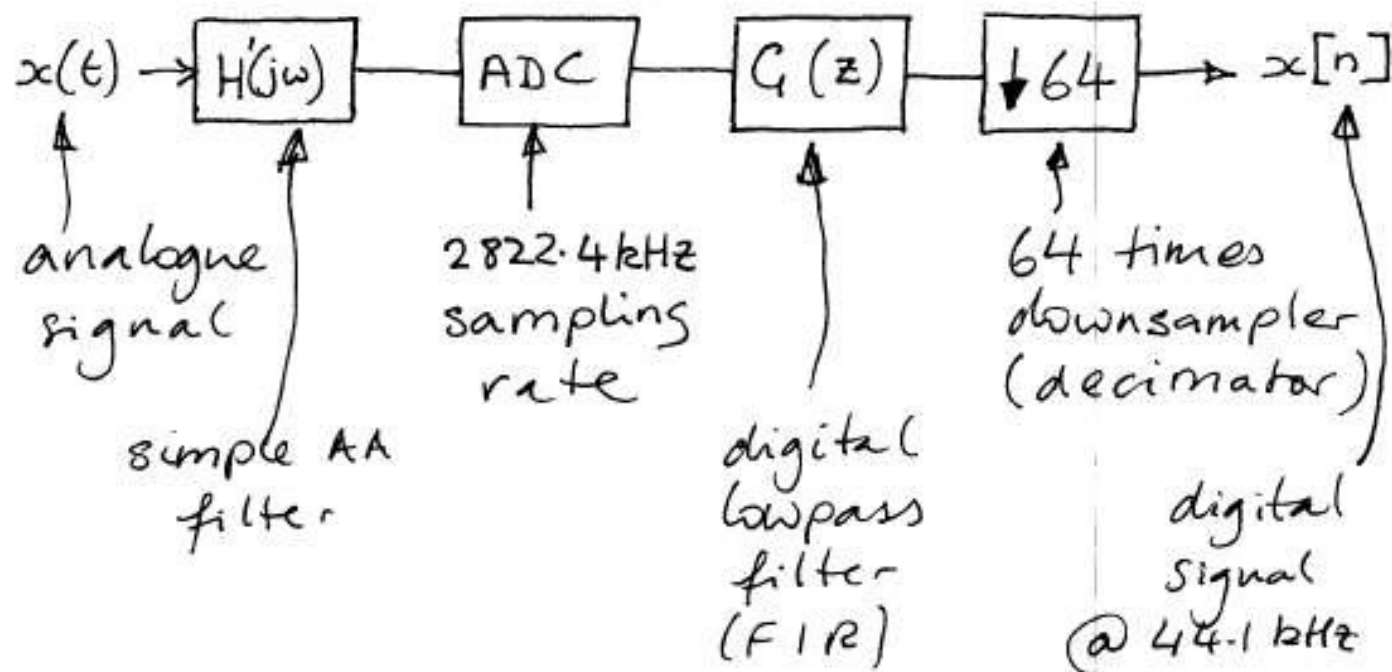
$$|H(j\omega)| < 96 \text{ dB}, f \geq (44.1 \times 64) - \frac{44.1}{2} \text{ kHz}$$

- Could be implemented by simple filter (eg. RC network)
- Recover desired sampling rate by downsampling process.

Oversampled Conversion Antialiasing Filter



Overall System



- This is a simplified version
- In these lectures we will study blocks like $G(z)$ and $\downarrow 64$

Sub band coding

Consider quantizing the samples of a speech signal. How many bits are required?

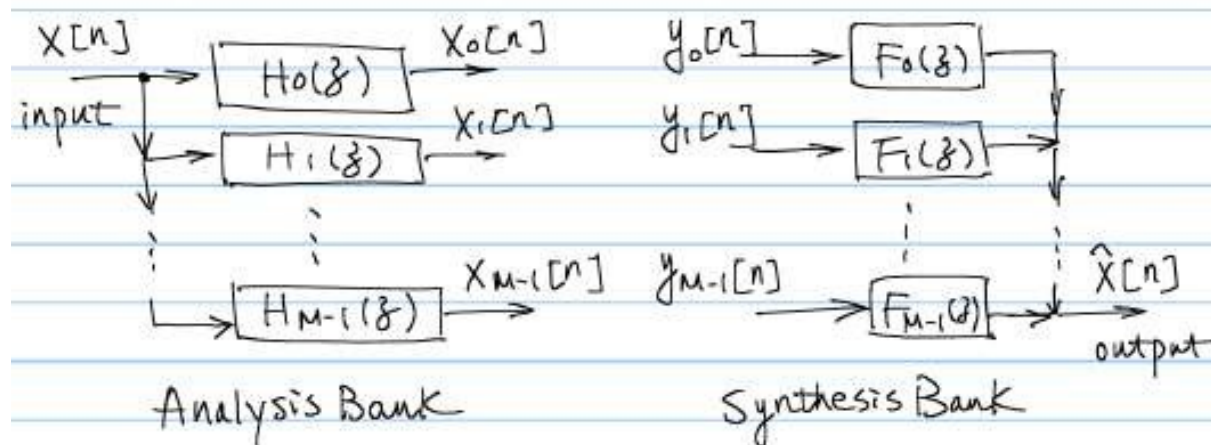
- In general, 16 bits precision per sample is normally used for audio. This gives an adequate dynamic range.
- In practice, certain frequency bands are less important perceptually because they contain less significant information
 - bands with less information or lower perceptual importance may be quantized with lower precision - fewer bits.
- Divide the spectrum of the signal into several subbands then allocate bits to each band appropriately.

Sub band coding

- 16 bits per sample, 10 kHz sampling frequency gives
 - 160 kbits/s
- Divide into 2 bands: high frequency and low frequency subbands.
 - High frequencies of speech are less important to intelligibility.
 - Therefore use only 8 bits per sample
- The sampling frequency can be reduced by a factor of 2 since bandwidth is halved, still satisfying Nyquist criterion.
 - $5 \times 16 + 5 \times 8 = 120$ kbits/s
 - 4:3 compression
- Reconstructed signal has no noticeable reduction in signal quality.

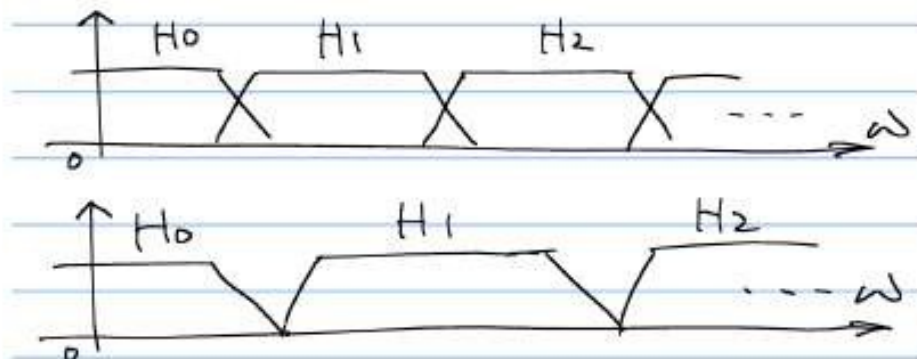
Digital filter banks

A digital filter bank is a collection of digital filters, with a common input or a common output.



- $H_i(z)$: analysis filters
- $x_k[n]$: subband signals
- $F_i(z)$: synthesis filters
- SIMO vs. MISO

- Typical frequency response for analysis filters:



Can be

- marginally overlapping
- non-overlapping
- (substantially) overlapping

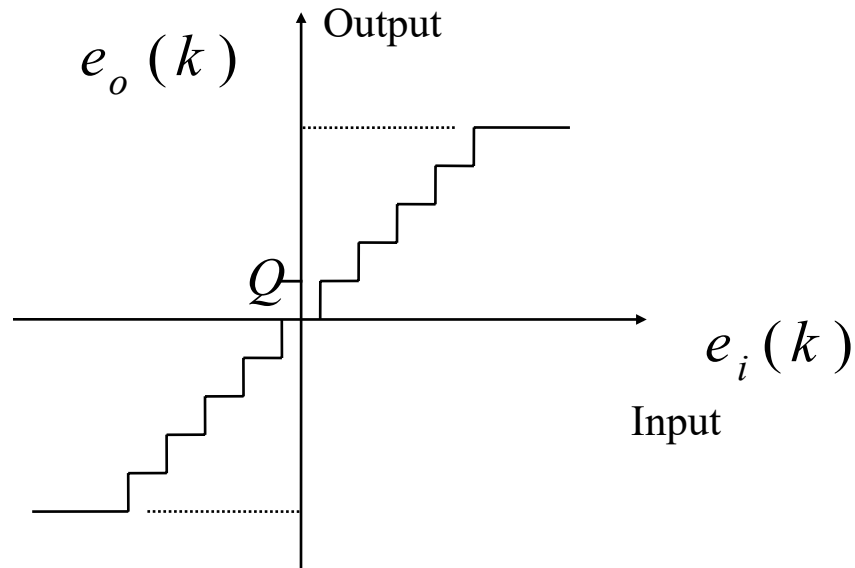
Finite Word length Effects

Finite Wordlength Effects

- Finite register lengths and A/D converters cause errors in:-
 - (i) Input quantisation.
 - (ii) Coefficient (or multiplier) quantisation
 - (iii) Products of multiplication truncated or rounded due to machine length

Finite Wordlength Effects

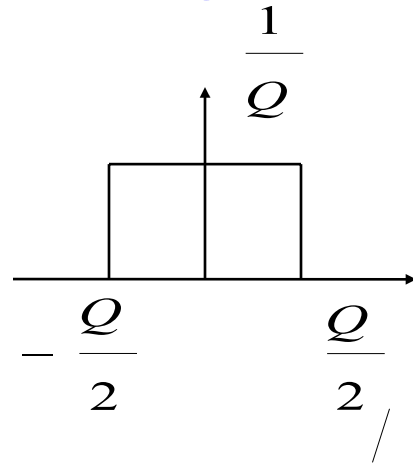
- Quantisation



$$-\frac{Q}{2} \leq e_{i,o}(k) \leq \frac{Q}{2}$$

Finite Wordlength Effects

- The pdf for e using rounding



$$p(e) \cdot de = E\{e^2\}$$

- Noise power
or

$$\sigma^2 = \frac{Q^2}{12}$$

$$\sigma^2 = \int_{-Q/2}^{Q/2} e^2 \cdot p(e) \cdot de$$

Finite Wordlength Effects

- Let input signal be sinusoidal of unity amplitude. Then total signal power $P = \frac{1}{2}$
- If b bits used for binary then $Q = 2/2^b$
so that $\sigma^2 = 2^{-2b}/3$
- Hence $P/\sigma^2 = \frac{3}{2} \cdot 2^{+2b}$
or SNR = 1.8 + 6b dB

Finite Wordlength Effects

- Consider a simple example of finite precision on the coefficients a, b of second order system with poles $\rho e^{\pm j\theta}$

$$H(z) = \frac{1}{1 - az^{-1} + bz^{-2}}$$

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

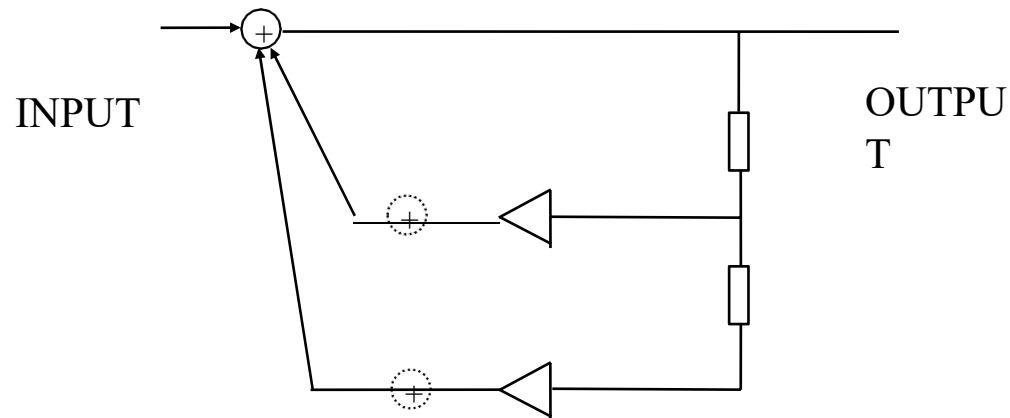
- where $a = 2\rho \cos \theta$ $b = \rho^2$

Finite Wordlength Effects

bit pattern	$2 \rho \cos \theta, \rho^2$	ρ
000	0	0
001	0.125	0.354
010	0.25	0.5
011	0.375	0.611
100	0.5	0.707
101	0.625	0.791
110	0.75	0.866
111	0.875	0.935
1.0	1.0	1.0

Finite Wordlength Effects

- Finite wordlength computations



Limit-cycles; "Effective Pole"

Model; Deadband

- Observe that for $H(z) = \frac{1}{(1 + b_1 z^{-1} + b_2 z^{-2})}$
- instability occurs when $|b_2| \rightarrow 1$
- i.e. poles are
 - (i) either on unit circle when complex
 - (ii) or one real pole is outside unit circle.
- Instability under the "effective pole" model is considered as follows

Finite Wordlength Effects

- In the time domain with $H(z) = \frac{Y(z)}{X(z)}$
- $$y(n) = x(n) - b_1 y(n-1) - b_2 y(n-2)$$
- With $|b_2| \rightarrow 1$ for instability we have $Q[b_2 y(n-2)]$ indistinguishable from $y(n-2)$
- Where $Q[\cdot]$ is quantisation

Finite Wordlength Effects

- With rounding, therefore we have

$$b_2 y(n-2) \pm 0.5 \quad y(n-2)$$

are indistinguishable (for integers)

or

$$b_2 y(n-2) \pm 0.5 = y(n-2)$$

- Hence

$$y(n-2) = \frac{\pm 0.5}{1 - b_2}$$

- With both positive and negative numbers

$$y(n-2) = \frac{\pm 0.5}{1 - |b_2|}$$

Finite Wordlength Effects

- The range of integers $\frac{\pm 0.5}{1 - |b_2|}$

constitutes a set of integers that cannot be individually distinguished as separate or from the asymptotic system behaviour.

- The band of integers $\left(-\frac{0.5}{1 - |b_2|}, +\frac{0.5}{1 - |b_2|} \right)$

is known as the "deadband".

- In the second order system, under rounding, the output assumes a cyclic set of values of the deadband. This is a limit-cycle.

Finite Wordlength Effects

- Consider the transfer function

$$G(z) = \frac{1}{(1 + b_1 z^{-1} + b_2 z^{-2})}$$

$$y_k = x_k - b_1 y_{k-1} - b_2 y_{k-2}$$

- if poles are complex then impulse response is given by h_k

$$h_k = \frac{\rho^k}{\sin \theta} \cdot \sin [(k + 1)\theta]$$

Finite Wordlength Effects

- Where $\rho = \sqrt{b_2}$
- If $b_2 = 1$ then the response is sinusoidal with frequency

$$\omega = \frac{1}{T} \cos^{-1} \left(\frac{-b_1}{2} \right)$$

- Thus product quantisation causes instability implying an "effective" $b_2 = 1$.

Finite Wordlength Effects

- Notice that with infinite precision the response converges to the origin
- With finite precision the response does not converge to the origin but assumes cyclically a set of values –the Limit Cycle

Finite Wordlength Effects

- Assume $\{e_1(k)\}, \{e_2(k)\} \dots$ are not correlated, random processes etc.

$$\sigma_{oi}^2 = \sigma_e^2 \sum_{k=0}^{\infty} h_i^2(k) \quad \sigma_e^2 = \frac{Q^2}{12}$$

Hence total output noise power

$$\sigma_o^2 = \sigma_{o1}^2 + \sigma_{o2}^2 = 2 \cdot \frac{2^{-2b}}{12} \sum_{k=0}^{\infty} \rho^{2k} \cdot \frac{\sin^2[(k+1)\theta]}{\sin^2 \theta}$$

- Where $Q = 2^{-b}$ and

$$h_1(k) = h_2(k) = \rho^k \cdot \frac{\sin[(k+1)\theta]}{\sin \theta}; \quad k \geq 0$$

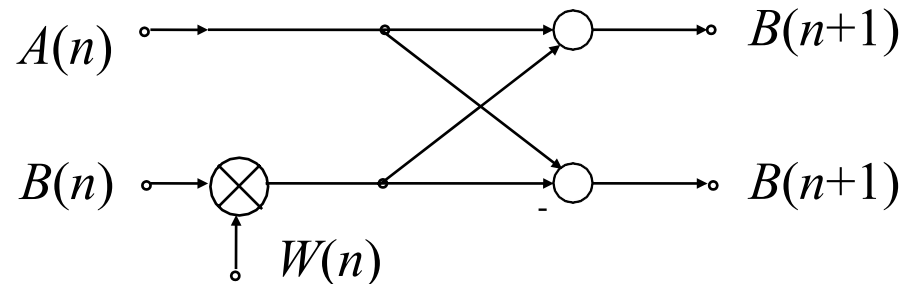
Finite Wordlength Effects

- ie

$$\sigma_0^2 = \frac{2^{-2b}}{6} \left[\frac{1 + \rho^2}{1 - \rho^2} \cdot \frac{1}{1 + \rho^4 - 2\rho^2 \cos 2\theta} \right]$$

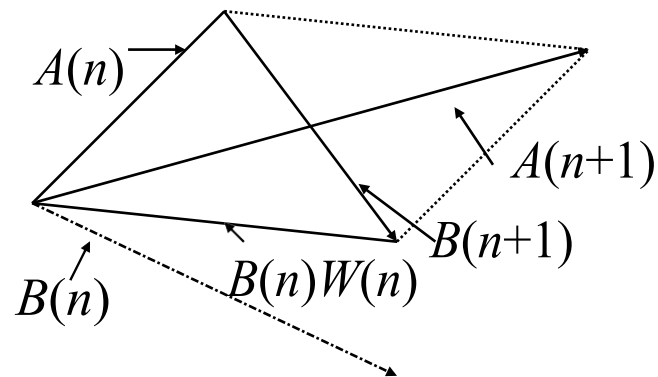
Finite Wordlength Effects

- For FFT



$$A(n+1) = A(n) + W(n) \cdot B(n)$$

$$B(n+1) = A(n) - W(n) \cdot B(n)$$



Finite Wordlength Effects

- FFT

$$\left| A(n+1) \right|^2 + \left| B(n+1) \right|^2 = 2$$

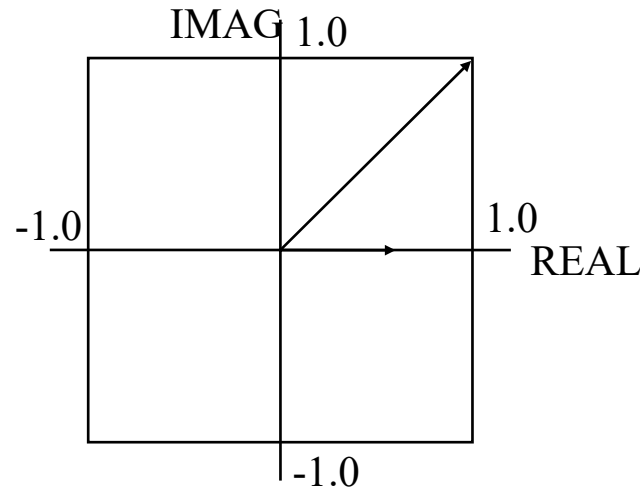
$$\left| A(n+1) \right|^2 = 2 \left| A(n) \right|^2$$

$$\left| A(n) \right| = \sqrt{2} \left| A(n) \right|$$

- AVERAGE GROWTH: 1/2 BIT/PASS

Finite Wordlength Effects

- FFT



$$A_x(n+1) = A_x(n) + B_x(n)C(n) - B_y(n)S(n)$$

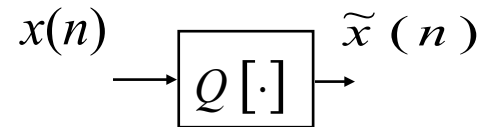
$$|A_x(n+1)| < |A_x(n)| + |B_x(n)||C(n)| - |B_y(n)||S(n)|$$

$$\frac{|A_x(n+1)|}{|A_x(n)|} < 1.0 + |C(n)| - |S(n)| = 2.414 \dots$$

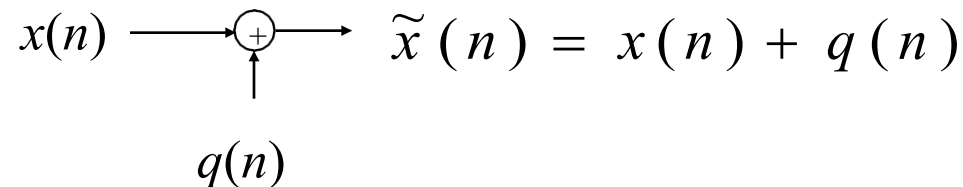
- PEAK GROWTH: 1.21.. BITS/PASS

Finite Wordlength Effects

- Linear modelling of product quantisation



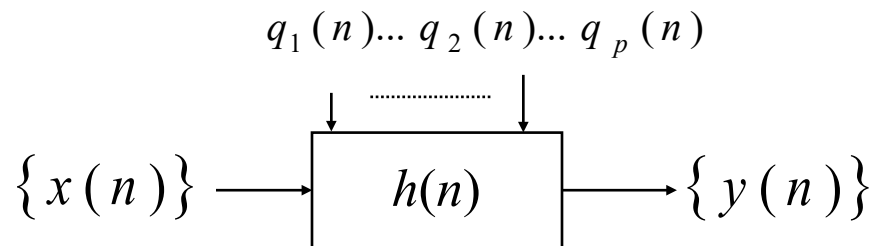
- Modelled as



Finite Wordlength Effects

- For rounding operations $q(n)$ is uniform distributed between $-\frac{Q}{2}, \frac{Q}{2}$ and where Q is the quantisation step (i.e. in a wordlength of bits with sign magnitude representation or mod 2, $Q = 2^{-b}$).
- A discrete-time system with quantisation at the output of each multiplier may be considered as a multi-input linear system

Finite Wordlength Effects



- Then

$$y(n) = \sum_{r=0}^{\infty} x(r) \cdot h(n-r) + \sum_{\lambda=1}^p \left[\sum_{r=0}^{\infty} q_{\lambda}(r) \cdot h_{\lambda}(n-r) \right]$$

- where $h_{\lambda}(n)$ is the impulse response of the system from λ the output of the multiplier to $y(n)$.

Finite Wordlength Effects

- For zero input i.e. $x(n) = 0, \forall n$ we can write

$$|y(n)| \leq \sum_{\lambda=1}^p |\hat{q}_{\lambda}| \cdot \sum_{r=0}^{\infty} |h_{\lambda}(n-r)|$$

- where $|\hat{q}_{\lambda}|$ is the maximum of $|q_{\lambda}(r)|, \forall \lambda, r$
which is not more than $\frac{Q}{2}$

- ie $|y(n)| \leq \frac{Q}{2} \cdot \sum_{\lambda=1}^p \left[\sum_{n=0}^{\infty} |h_{\lambda}(n-r)| \right]$

Finite Wordlength Effects

- However

$$\sum_{n=0}^{\infty} |h_{\lambda}(n)| \leq \sum_{n=0}^{\infty} |h(n)|$$

- And hence

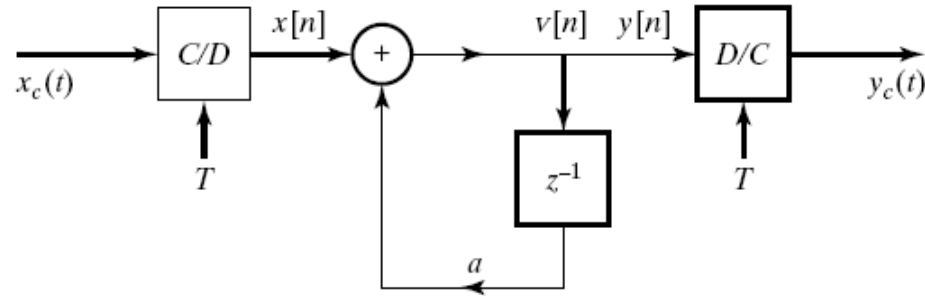
$$|y(n)| \leq \frac{pQ}{2} \cdot \sum_{n=0}^{\infty} |h(n)|$$

- ie we can estimate the maximum swing at the output from the system parameters and quantisation level

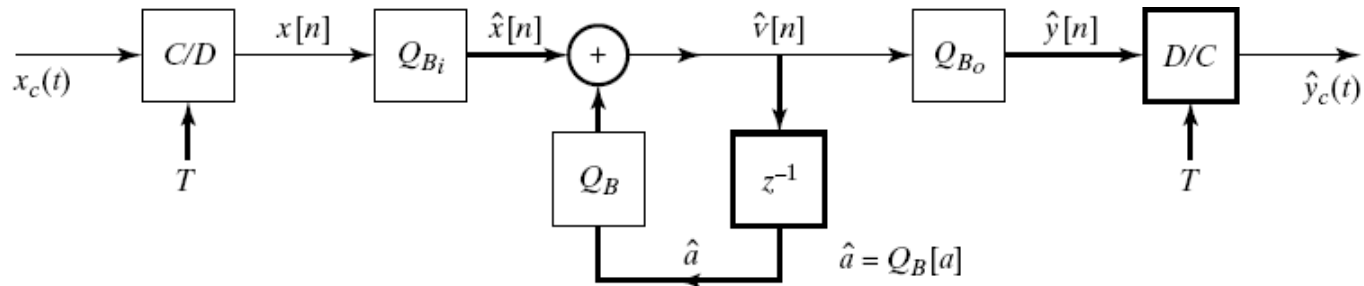
Finite Precision Numerical Effects

Quantization in Implementing Systems

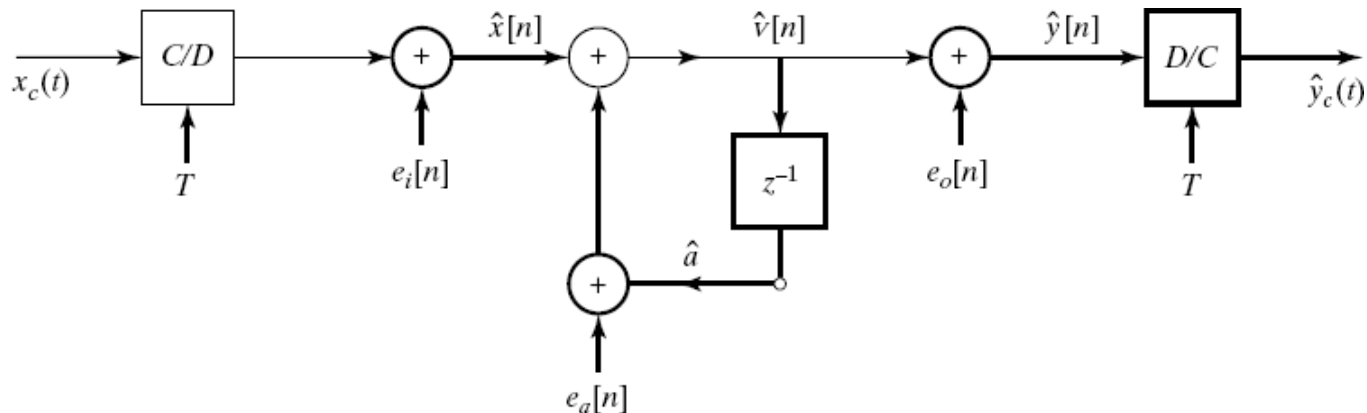
- Consider the following system



- A more realistic model would be



- In order to analyze it we would prefer



Effects of Coefficient Quantization in IIR Systems

- When the parameters of a rational system are quantized
 - The poles and zeros of the system function move
- If the system structure of the system is sensitive to perturbation of coefficients
 - The resulting system may no longer be stable
 - The resulting system may no longer meet the original specs
- We need to do a detailed sensitivity analysis
 - Quantize the coefficients and analyze frequency response
 - Compare frequency response to original response
- We would like to have a general sense of the effect of quantization

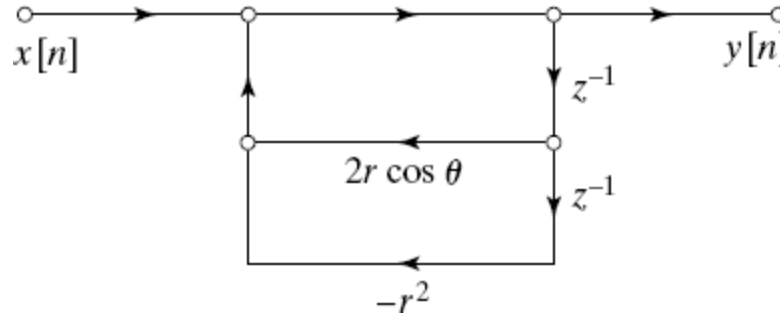
Effects on Roots

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 - \sum_{k=1}^N a_k z^{-k}} \xrightarrow[\text{tion}]{\text{Quantiza}} \hat{H}(z) = \frac{\sum_{k=0}^M \hat{b}_k z^{-k}}{1 - \sum_{k=1}^N \hat{a}_k z^{-k}}$$

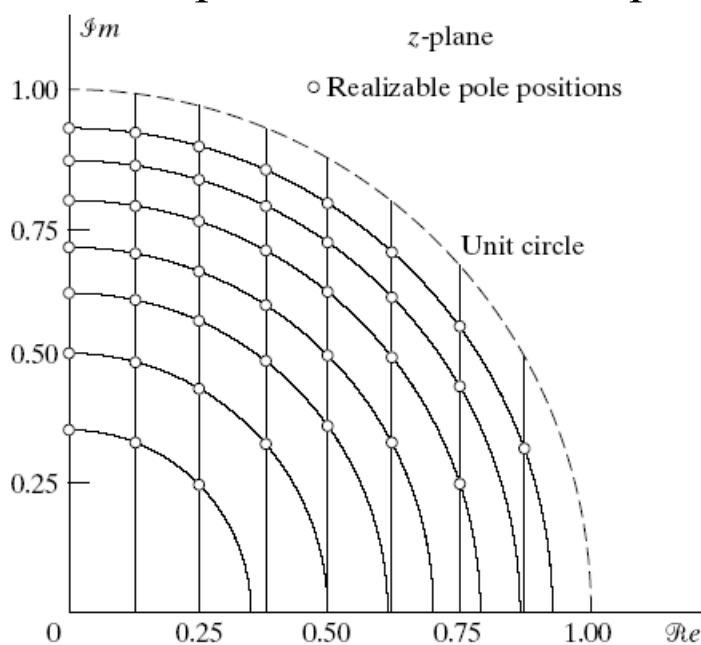
- Each root is affected by quantization errors in ALL coefficient
- Tightly clustered roots can be significantly effected
 - Narrow-bandwidth lowpass or bandpass filters can be very sensitive to quantization noise
- The larger the number of roots in a cluster the more sensitive it becomes
- This is the reason why second order cascade structures are less sensitive to quantization error than higher order system
 - Each second order system is independent from each other

Poles of Quantized Second-Order Sections

- Consider a 2nd order system with complex-conjugate pole pair

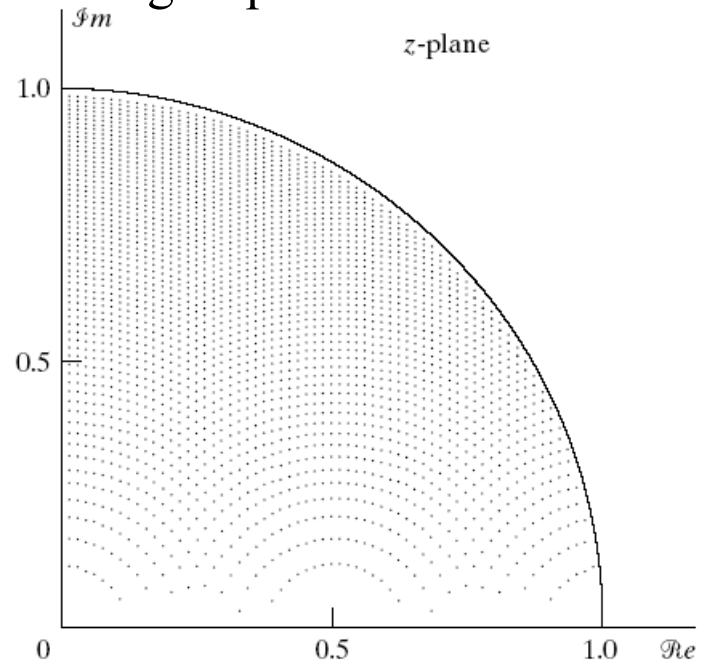


- The pole locations after quantization will be on the grid point



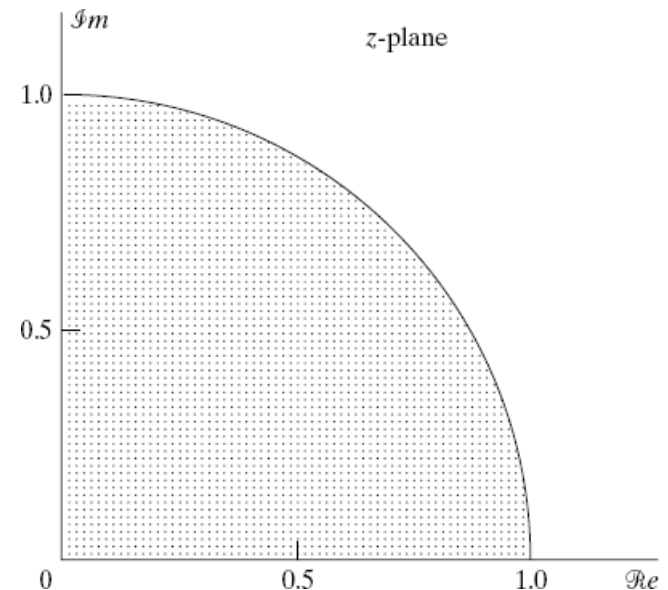
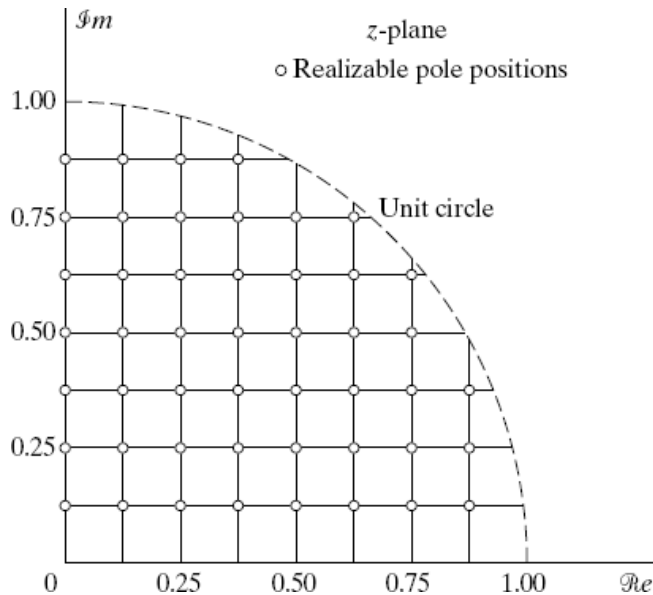
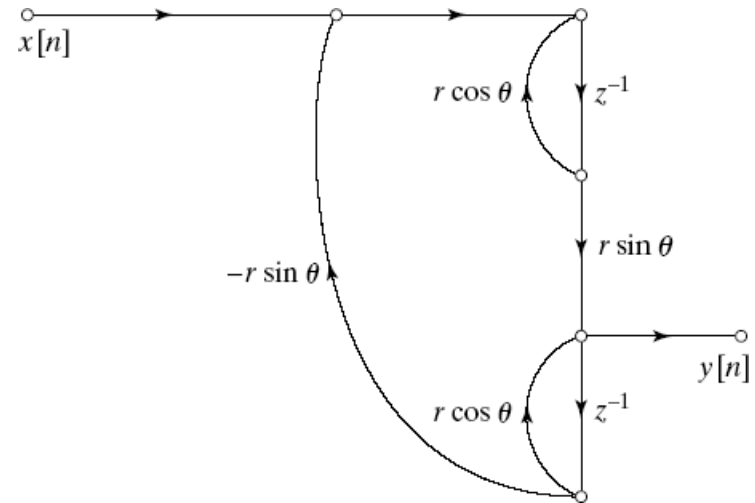
← 3-bits

7-bits →



Coupled-Form Implementation of Complex-Conjugate Pair

- Equivalent implementation of the second order system
- But the quantization grid this time is



Effects of Coefficient Quantization in FIR Systems

- No poles to worry about only zeros
- Direct form is commonly used for FIR systems

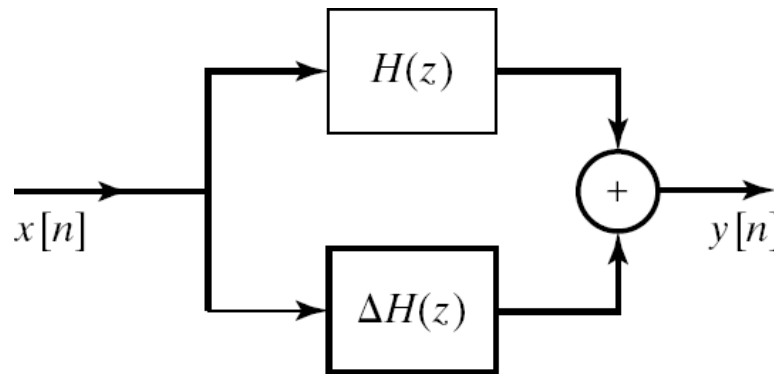
$$H(z) = \sum_{n=0}^M h[n]z^{-n}$$

- Suppose the coefficients are quantized

$$\hat{H}(z) = \sum_{n=0}^M \hat{h}[n]z^{-n} = H(z) + \Delta H(z)$$

$$\Delta H(z) = \sum_{n=0}^M \Delta h[n]z^{-n}$$

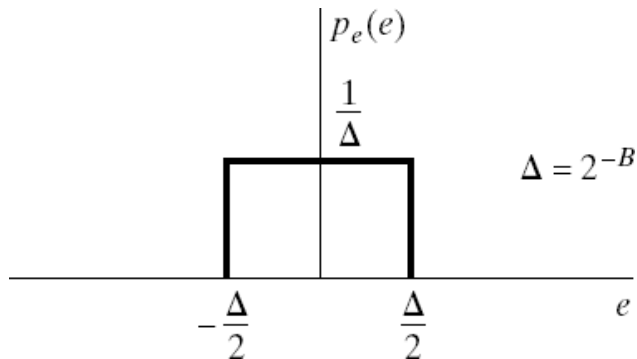
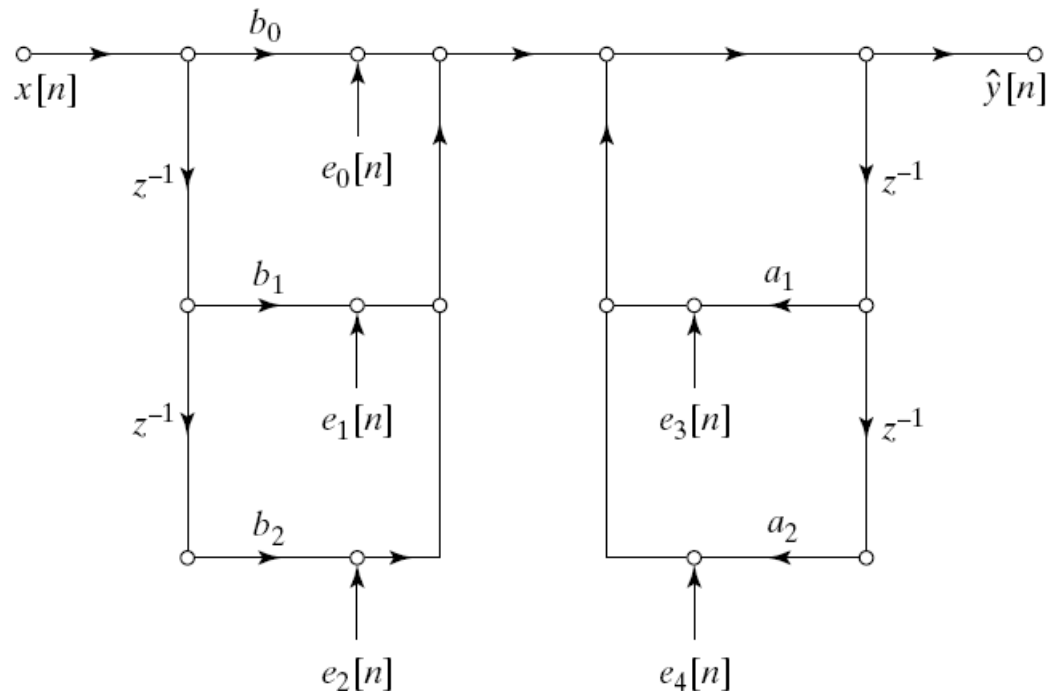
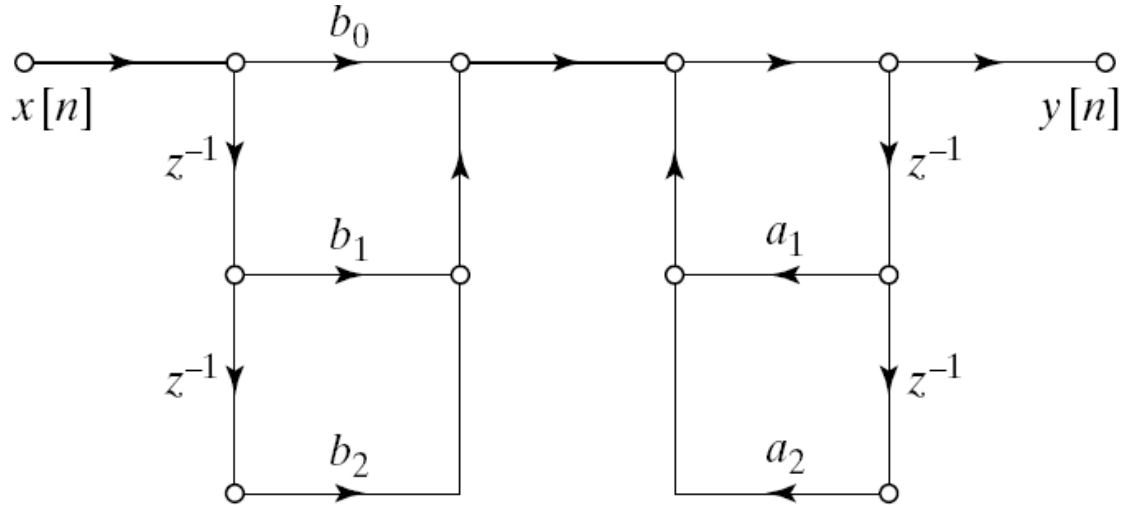
- Quantized system is linearly related to the quantization error



- Again quantization noise is higher for clustered zeros
- However, most FIR filters have spread zeros

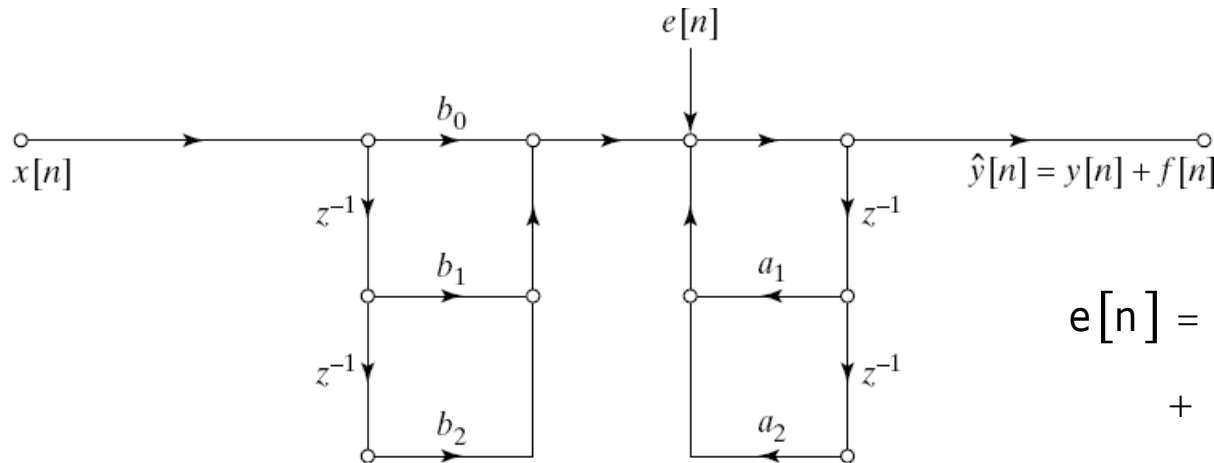
Round-Off Noise in Digital Filters

- Difference equations implemented with finite-precision arithmetic are non-linear systems
- Second order direct form I system
- Model with quantization effect
- Density function error terms for rounding



Analysis of Quantization Error

- Combine all error terms to single location to get



$$e[n] = e_0[n] + e_1[n] + e_2[n] + e_3[n] + e_4[n]$$

- The variance of $e[n]$ in the general case is

$$\sigma_e^2 = (M + 1 + N) \frac{2^{-2B}}{12}$$

- The contribution of $e[n]$ to the output is

$$f[n] = \sum_{k=1}^N a_k f[n - k] + e[n]$$

- The variance of the output error term $f[n]$ is

$$\sigma_f^2 = (M + 1 + N) \frac{2^{-2B}}{12} \sum_{n=-\infty}^{\infty} |h_{ef}[n]|^2$$

$$H_{ef}(z) = 1 / A(z)$$

Round-Off Noise in a First-Order System

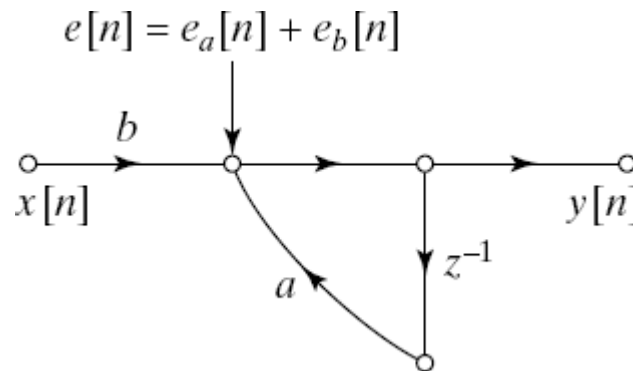
- Suppose we want to implement the following stable system

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

- The quantization error noise variance is

$$\sigma_f^2 = (M + 1 + N) \frac{2^{-2B}}{12} \sum_{n=-\infty}^{\infty} |h_{ef}[n]|^2 = 2 \frac{2^{-2B}}{12} \sum_{n=0}^{\infty} |a|^{2n} = 2 \frac{2^{-2B}}{12} \left(\frac{1}{1 - |a|^2} \right)$$

- Noise variance increases as $|a|$ gets closer to the unit circle
- As $|a|$ gets closer to 1 we have to use more bits to compensate for the increasing error

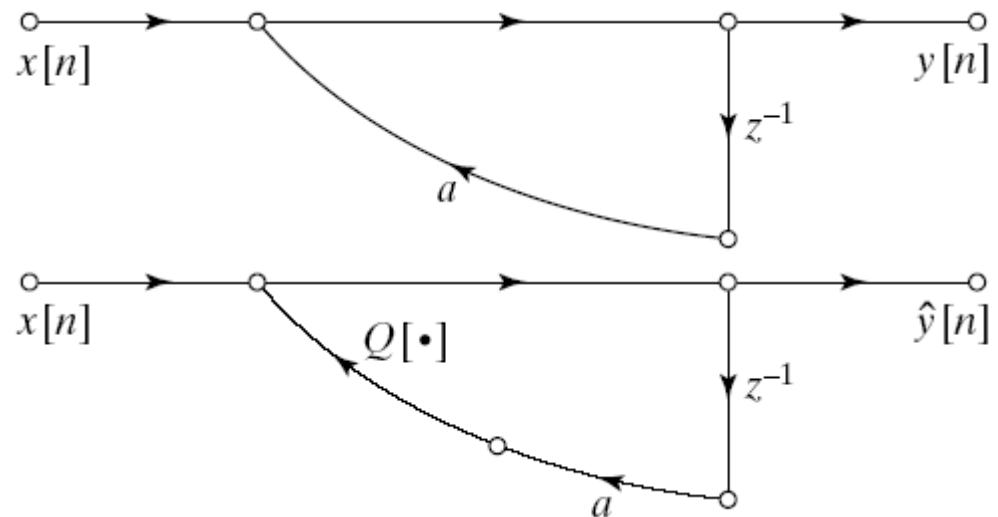


Zero-Input Limit Cycles in Fixed-Point Realization of IIR Filters

- For stable IIR systems the output will decay to zero when the input becomes zero
- A finite-precision implementation, however, may continue to oscillate indefinitely
- Nonlinear behaviour very difficult to analyze so we will study by example
- Example: Limit Cycle Behavior in First-Order Systems

$$y[n] = ay[n-1] + x[n] \quad |a| < 1$$

- Assume $x[n]$ and $y[n-1]$ are implemented by 4 bit



Example Cont'd

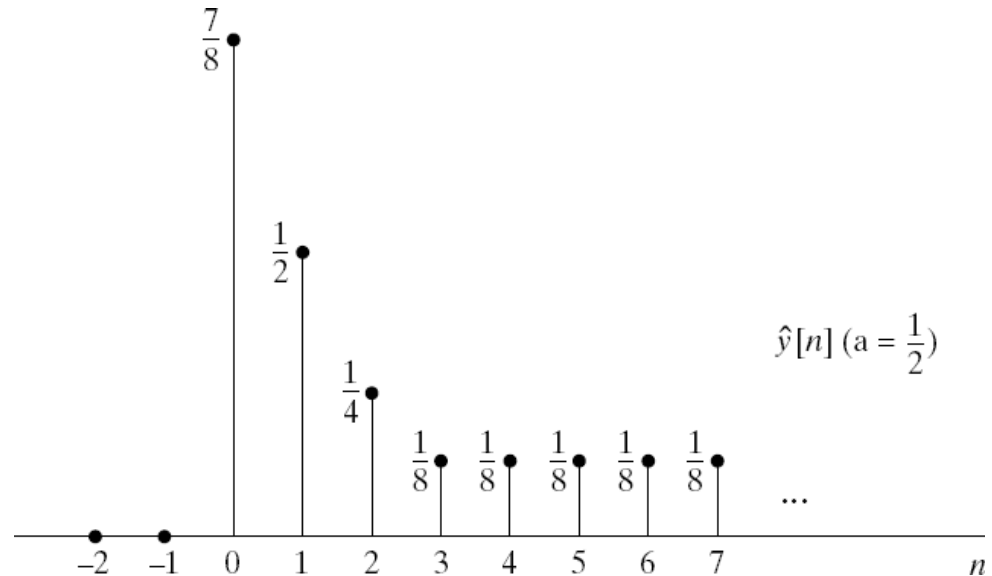
$$y[n] = ay[n-1] + x[n] \quad |a| < 1$$

- Assume that $a=1/2=0.100b$ and the input is

$$x[n] = \frac{7}{8}\delta[n] = (0.111b)\delta[n]$$

- If we calculate the output for values of n

n	y[n]	Q(y[n])
0	$7/8=0.111b$	$7/8=0.111b$
1	$7/16=0.011100b$	$1/2=0.100b$
2	$1/4=0.010000b$	$1/4=0.010b$
3	$1/8=0.001000b$	$1/8=0.001b$
4	$1/16=0.00010b$	$1/8=0.001b$



- A finite input caused an oscillation with period 1

Example: Limit Cycles due to Overflow

- Consider a second-order system realized by

$$\hat{y}[n] = x[n] + Q(a_1 \hat{y}[n-1]) + Q(a_2 \hat{y}[n-2])$$

- Where $Q()$ represents two's complement rounding
- Word length is chosen to be 4 bits

- Assume $a_1 = 3/4 = 0.110b$ and $a_2 = -3/4 = 1.010b$

- Also assume

$$\hat{y}[-1] = 3/4 = 0.110b \quad \text{and} \quad \hat{y}[-2] = -3/4 = 1.010b$$

- The output at sample $n=0$ is

$$\begin{aligned} \hat{y}[0] &= 0.110b \times 0.110b + 1.010b \times 1.010b \\ &= 0.100100b + 0.100100b \end{aligned}$$

- After rounding up we get

$$\hat{y}[0] = 0.101b + 0.101b = 1.010b = -3/4$$

- Binary carry overflows into the sign bit changing the sign
- When repeated for $n=1$

$$\hat{y}[1] = 1.010b + 1.010b = 0.110b = 3/4$$

Avoiding Limit Cycles

- Desirable to get zero output for zero input: Avoid limit-cycles
- Generally adding more bits would avoid overflow
- Using double-length accumulators at addition points would decrease likelihood of limit cycles
- Trade-off between limit-cycle avoidance and complexity
- FIR systems cannot support zero-input limit cycles