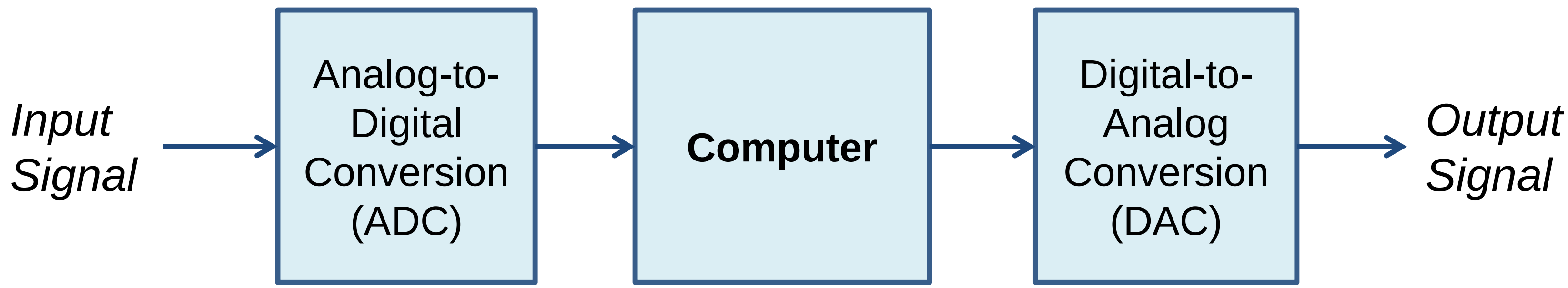
Lecture 02:
[Rabiner] Chapter 2
Fundamentals of Digital Signal Processing

DEEE725 음성신호처리실습
Speech Signal Processing Lab
Instructor: 장길진

What is DSP?

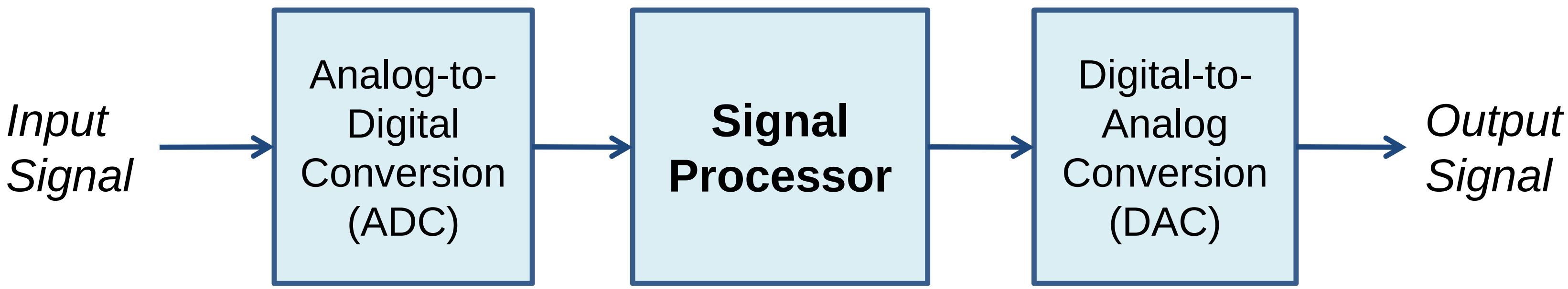


- **Digital**
  - Method to represent a quantity, a phenomenon or an event
- **Signal**
  - something (e.g., a sound, gesture, or object) that carries information
  - a detectable physical quantity (e.g., a voltage, current, or magnetic field strength) by which messages or information can be transmitted
- **Processing**
  - What kind of processing do we need and encounter almost everyday?
  - Related to **Computing**

Common Forms of Computing

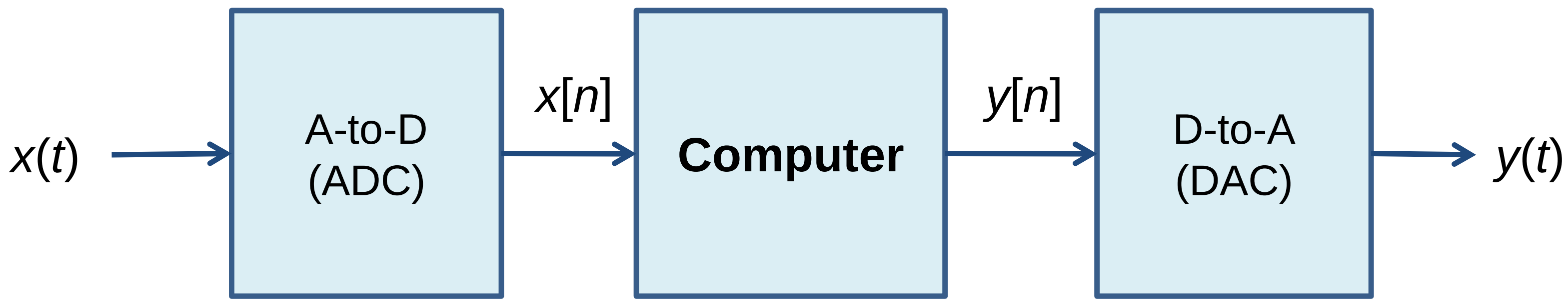
- **Text processing**
  - handling of text, tables, basic arithmetic and logic operations (i.e., calculator functions)
  - word processing, language processing, spreadsheet processing, presentation processing
- **Signal Processing**
  - a more general form of information processing, including handling of speech, audio, image, video, etc.
  - Filtering/spectral analysis
  - Analysis, recognition, synthesis and coding of real world signals
  - Detection and estimation of signals in the presence of noise or interference

Advantages of Digital Representations



- Permanence and robustness of signal representations; zero-distortion reproduction may be achievable
- Advanced IC technology works well for digital systems
- Virtually infinite flexibility with digital systems
  - Multi-functionality
  - Multi-input/multi-output
- Indispensable in telecommunications which is virtually all digital at the present time

Digital Processing of Analog Signals



- **A-to-D conversion**: bandwidth control, sampling and quantization
- **Computational processing**: implemented on computers or ASICs with finite-precision arithmetic
  - basic numerical processing: add, subtract, multiply (scaling, amplification, attenuation), mute, etc.
  - algorithmic numerical processing: convolution or linear filtering, non-linear filtering (e.g., median filtering), difference equations, DFT, inverse filtering, MAX/MIN, etc.
- **D-to-A conversion**: re-quantification and filtering (or interpolation) for reconstruction

Discrete-Time Signals

- A sequence of numbers
- Mathematical representation;

  $x = \{x[n]\}, -\infty < n < \infty$
- Sampled from an analog signal, $x(t)$, at time t=nT;

  $x[n] = x(nT), -\infty < n < \infty$
- T is called the **sampling period**, and its reciprocal, $F_s = 1/T$, is called the **sampling frequency**

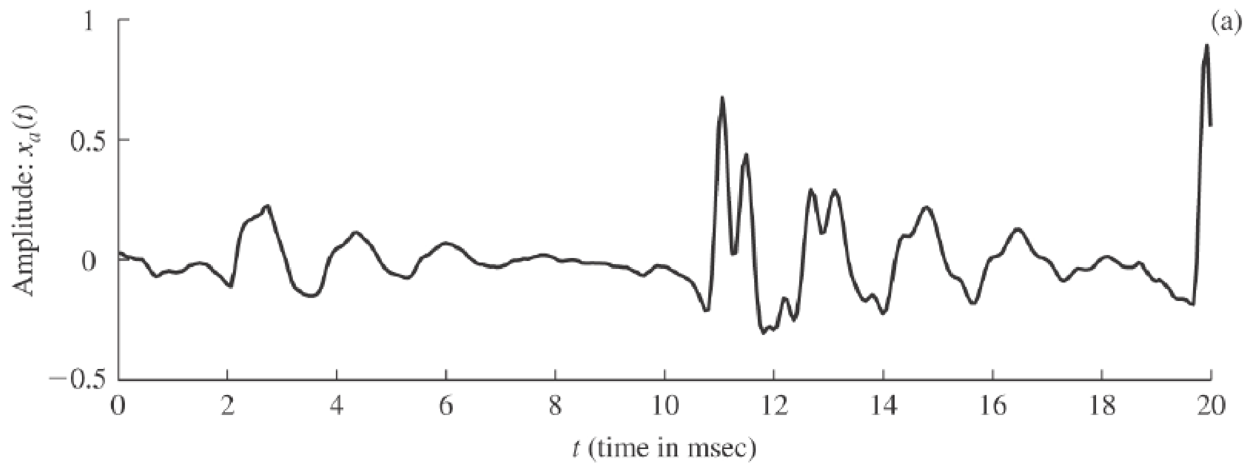
  F_s = 8 kHz $\leftrightarrow$ T = 1/8000 = 125 μsec

  F_s = 10 kHz $\leftrightarrow$ T = 1/10000 = 100 μsec

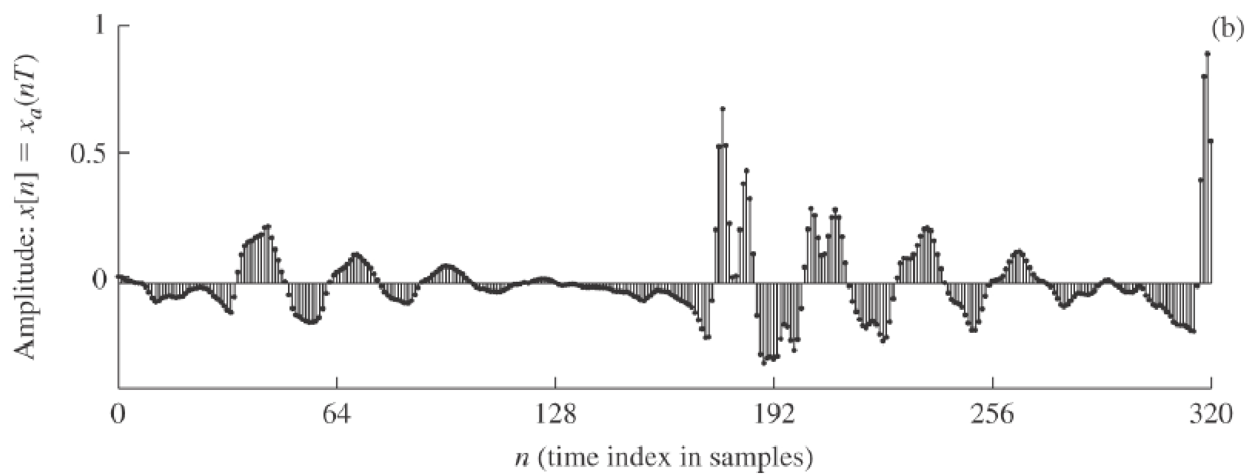
  F_s = 16 kHz $\leftrightarrow$ T = 1/16000 = 62.5 μsec

  F_s = 44.1 kHz $\leftrightarrow$ T = 1/44100 = 22.676 μsec

Speech Waveform Display & Varying Sample Rates

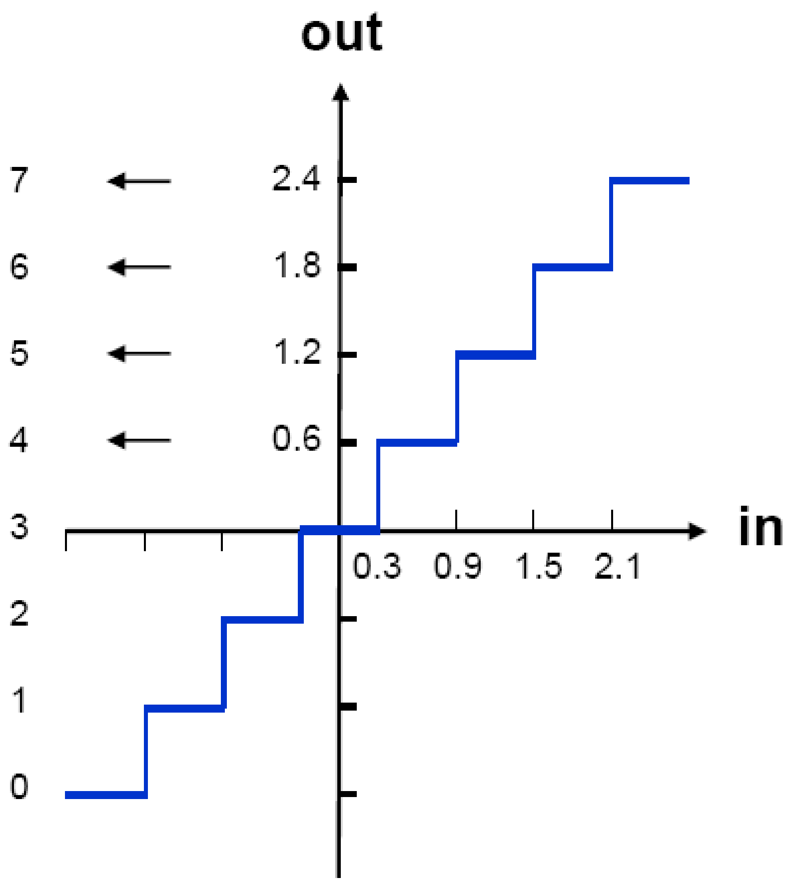


plot( );



stem( )

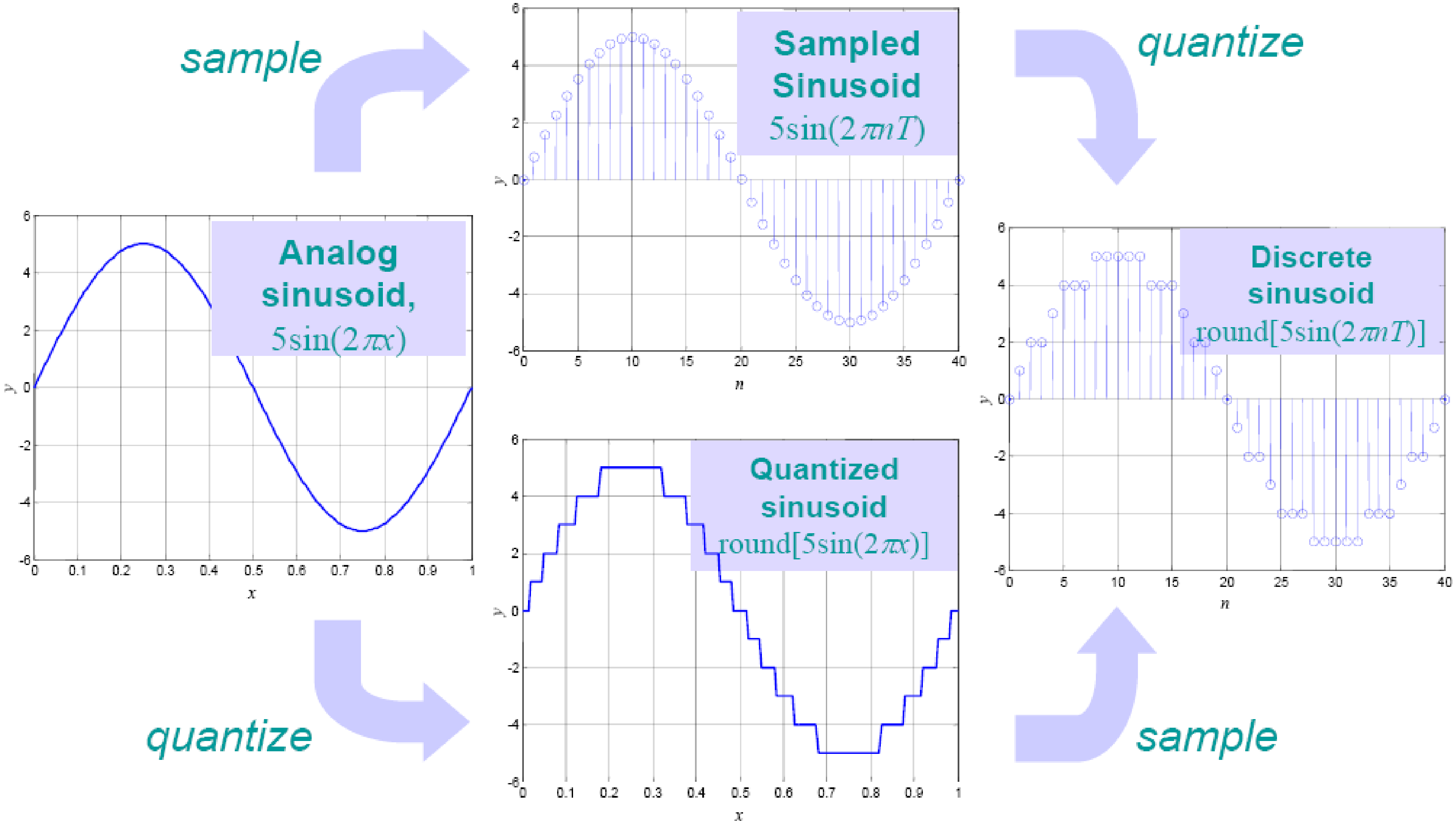
Quantization



A 3-bit uniform quantizer

- Transforming a continuously-valued input into a representation that assumes one out of a finite set of values
- The finite set of output values is indexed; e.g., the value 1.8 has an index of 6, or $(110)_2$ in binary representation
- Storage or transmission uses binary representation; a quantization (mapping) table is needed

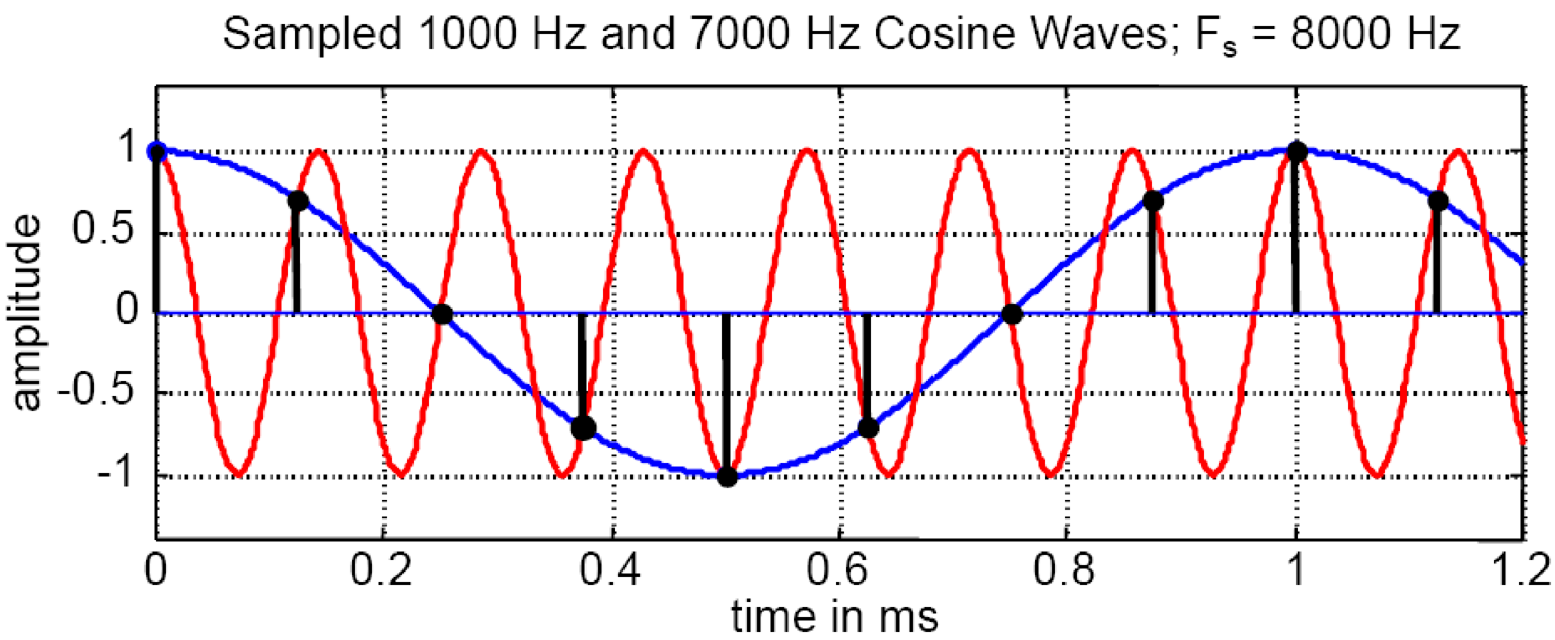
Discrete Signal Representations



Issues with Discrete Signals

- which *sampling rate* is appropriate
  - 6.4 kHz (telephone bandwidth)
  - 8 kHz (extended telephone bandwidth)
  - 11 kHz (extended bandwidth)
  - 16 kHz (hi-fi speech)
  - 44.1 kHz (hi-fi audio)
- how many *quantization levels* are necessary at each bit rate (bits/sample)
  - 16, 12, 8, etc. ➔ ultimately determines the speech-to-noise ratio (SNR) of the speech
  - *speech coding* is concerned with answering this question in an optimal manner

The Sampling Theorem



- A bandlimited signal can be reconstructed exactly from samples taken with sampling frequency

$$\frac{1}{T} = F_s \geq 2f_{\max} \quad \text{or} \quad \frac{2\pi}{T} = W_s \geq 2W_{\max}$$

The Sampling Theorem

- If a signal $x_a(t)$ has a bandlimited Fourier transform $X_a(j\Omega)$ such that $X_a(j\Omega)=0$ for $\Omega \geq 2\pi F_N$, then $x_a(t)$ can be uniquely reconstructed from equally spaced samples $x_a(nT)$, $-\infty<n<\infty$, if $1/T \geq 2F_N$ ($F_S \geq 2F_N$) (A-D or C/D converter)

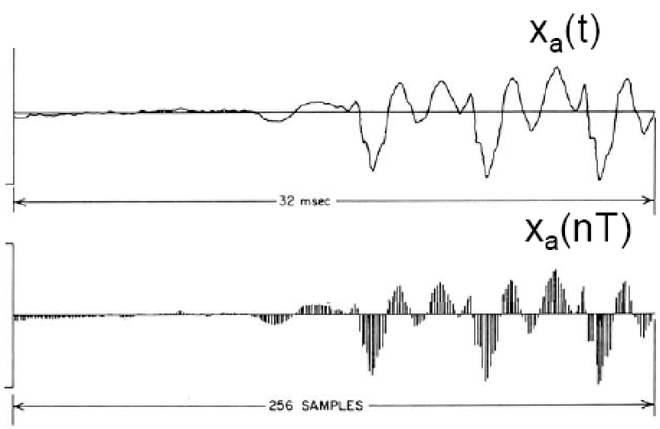
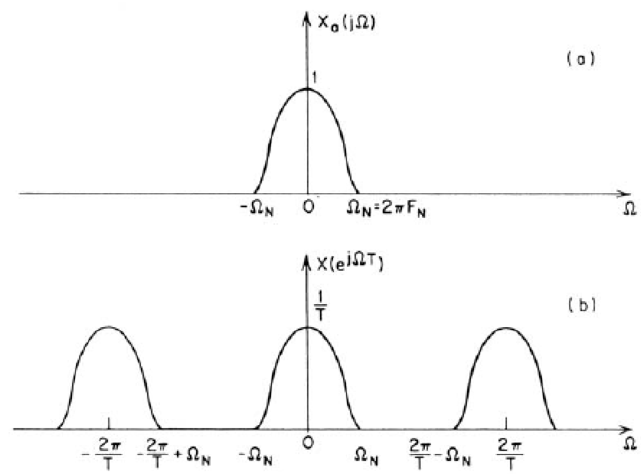


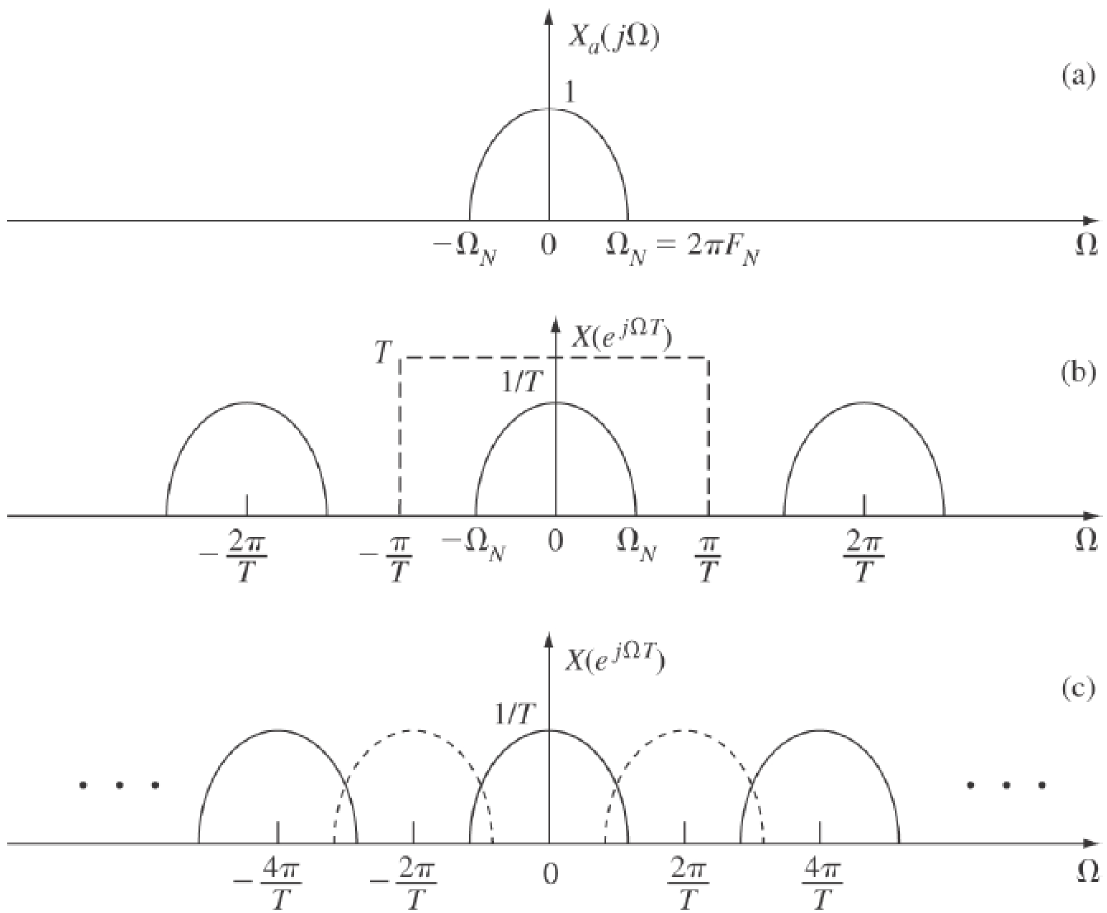
Fig. 2.1 Representations of a speech signal.



$x_a(nT) = x_a(t)\, u_T(nT)$, where $u_T(nT)$ is a periodic pulse train of period T, with periodic spectrum of period $2\pi/T$

116

Sampling Theorem Interpretation



To avoid aliasing need:

$2\pi / T - \Omega_N > \Omega_N$

$\Rightarrow \; 2\pi / T > 2\Omega_N$

$\Rightarrow \; F_s = 1/T > 2F_N$

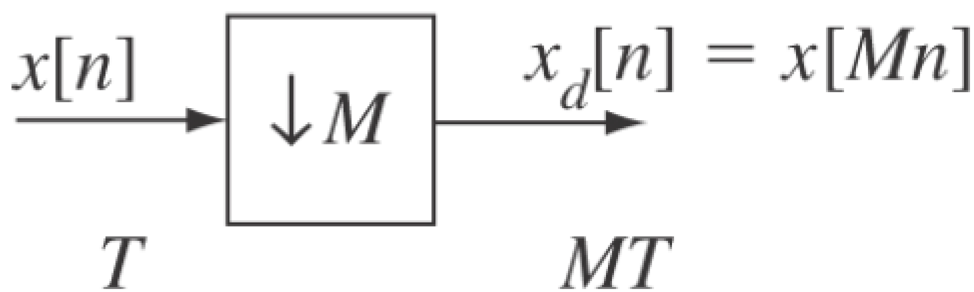
case where $1/T < 2F_N$, aliasing occurs

Nyquist’s Sampling Rates

- F_N = Nyquist frequency (highest frequency with significant spectral level in signal)
- must sample at least twice the Nyquist frequency to prevent aliasing (frequency overlap)
  - telephone speech (300-3200 Hz) → F_S=8000 Hz
  - wideband speech (100-7200 Hz) → F_S=16000 Hz
  - audio signal (50-21000 Hz) → F_S=44100 Hz
  - AM broadcast (100-7500 Hz) → F_S=16000 Hz
- can always sample at rates higher than twice the Nyquist frequency (but that is wasteful of processing)

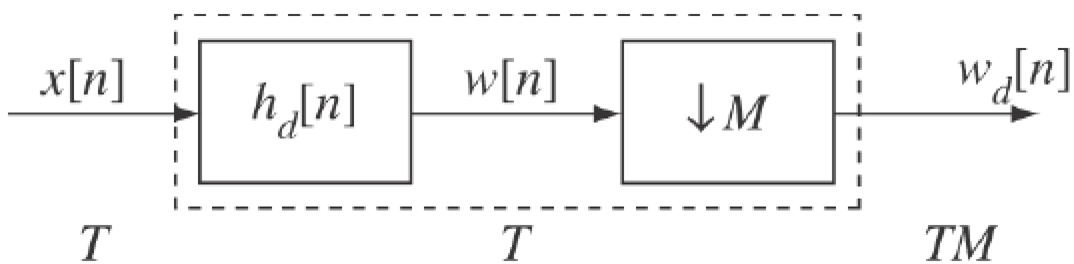
Decimation

- Reducing sampling rate of already sampled signal by factor of $M \geq 2$
- Computing new signal $x_d[n]$ with sampling rate
  $F_S' = 1/T' = 1/(MT) = F_S/M$
  such that $x_d[n] = x_a(nT')$ with no aliasing
- one solution is to downsample $x[n] = x_a(nT)$ by retaining one out of every M samples of $x[n]$, giving $x_d[n] = x[nM]$



Decimation

- need to ensure that the highest frequency in is no greater than $F_S/(2M)$
- thus we need to filter $x[n]$ using an ideal lowpass filter
- using the appropriate lowpass filter, we can downsample the resulting lowpass-filtered signal by a factor of M without aliasing



Interpolation

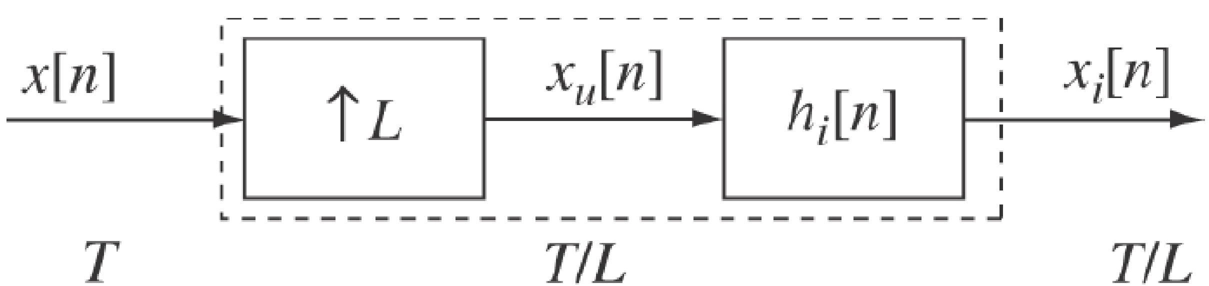
- assume we have $x[n] = x_a(nT)$ with no aliasing and we wish to increase the sampling rate by the integer factor of L
- we need to compute a new sequence of samples of $x_a(t)$ with period $T'' = T/L$, i.e.,

  $x_i[n] = x_a(nT'') = x_a(nT/L)$
- need to fill in the unknown samples by an interpolation process
  - Linear interpolation, sinusoidal interpolation, pulse train, etc.
- **Low-pass filtering is necessary!!**

Interpolation

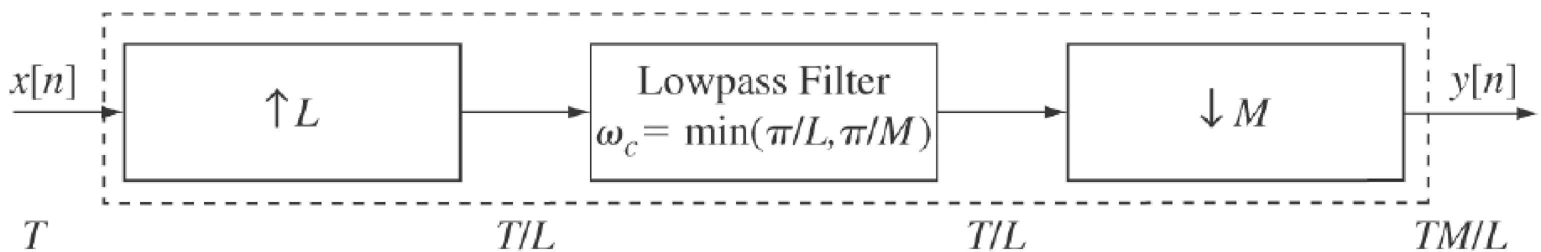
- Original signal, $x[n]$, at sampling period T, is first upsampled to give signal with sampling period $T'' = T/L$
- lowpass filter removes images of original spectrum giving:

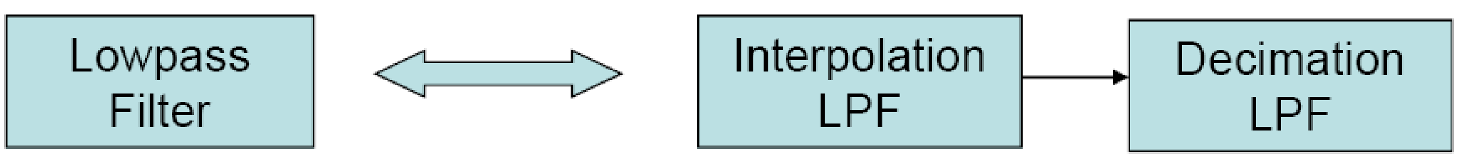
$$x_i[n] = h_i[n] * x_a(nT'') = h_i[n] * x_a(nT/L)$$



Sampling Rate Conversion by Non-Integer Factors

- $T' = MT/L$ → convert rate by factor of M/L
- need to interpolate by L, then decimate by M (why can't it be done in the reverse order?)
- for large values of L, or M, or both, can implement in stages, i.e., L=1024, use $L1$=32 followed by $L2$=32





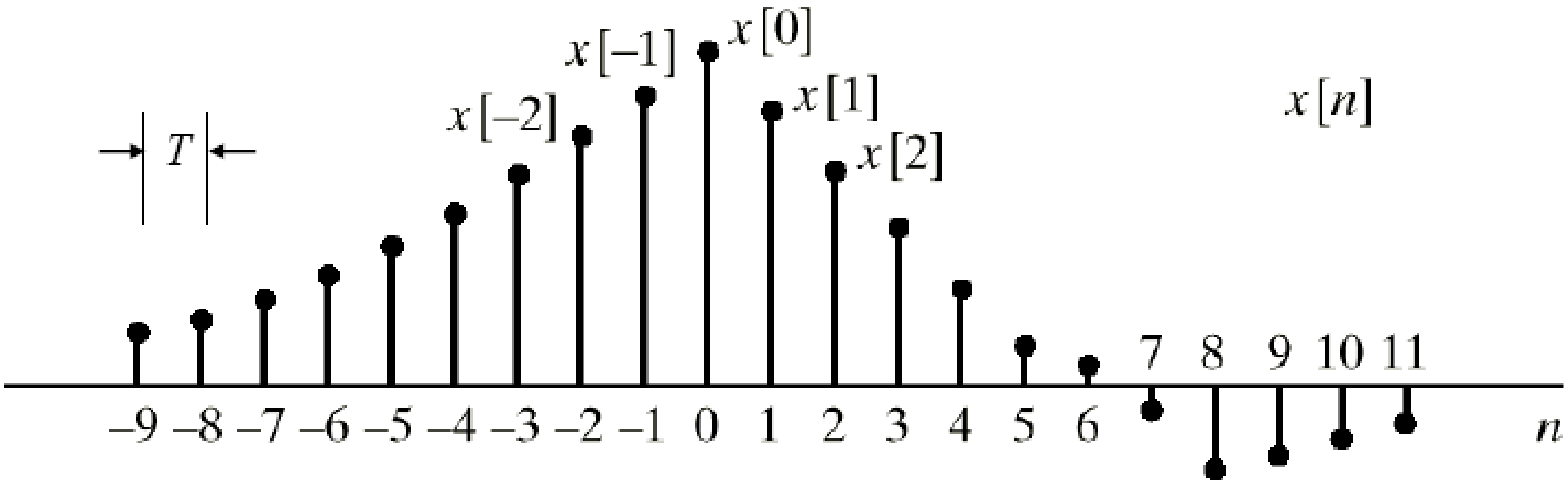
need to combine specifications of both LPFs and implement in a single stage of lowpass filtering

DEEE725 음성신호처리실습, 장길진

Time domain representation, LTI system, convolution

CHAPTER 2. FUNDAMENTALS OF DIGITAL SIGNAL PROCESSING, PART 2

Discrete-Time Sequences



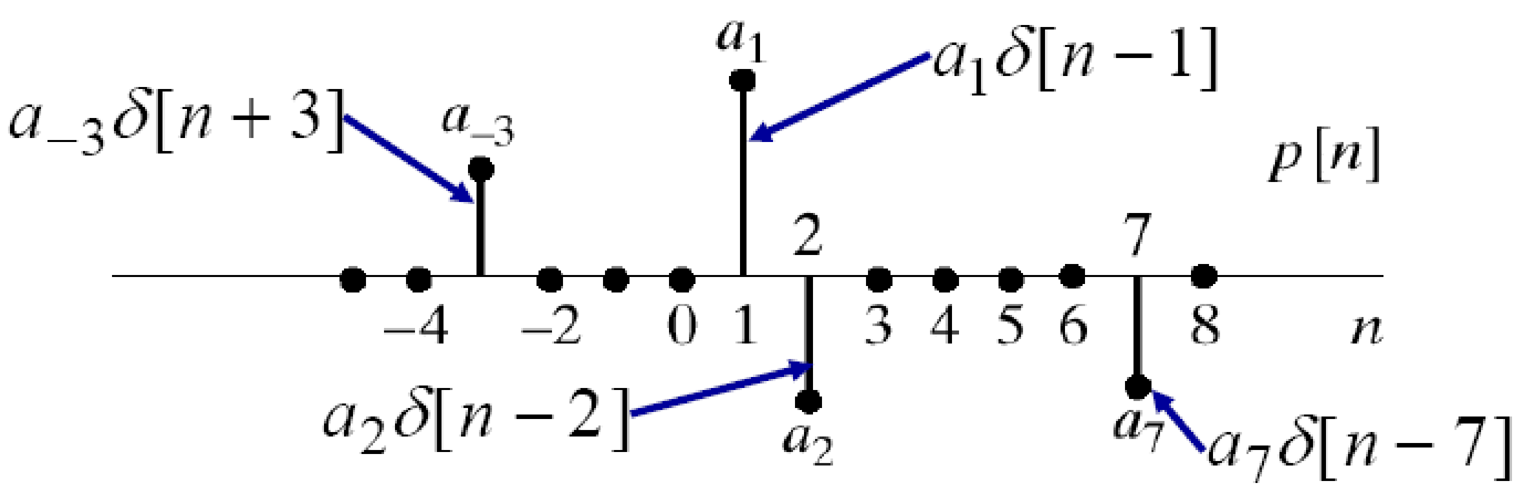
- $x[n]$ denotes the “sequence value at ‘time’ n”
- Sources of sequences:
  - Sampling a continuous-time signal
    $x[n] = x_c(nT) = x_c(t)|_{t=nT}$
  - Mathematical formulas – generative system
    e.g., $x[n] = 0.3 \cdot x[n\text{-}1] - 1$; $x[0] = 40$

Impulse Representation of Sequences

Value of the function at k

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

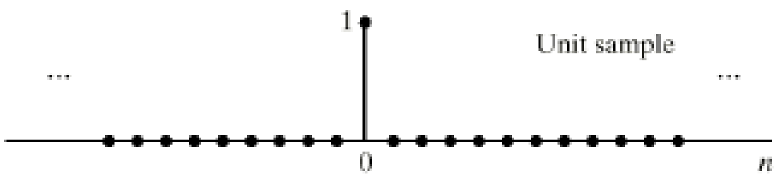
A sequence, a function



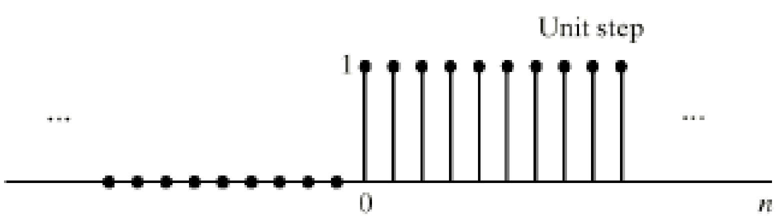
$$p[n] = a_{-3}\delta[n+3] + a_1\delta[n-1] + a_2\delta[n-2] + a_7\delta[n-7]$$

Some Useful Sequences

unit sample $\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$



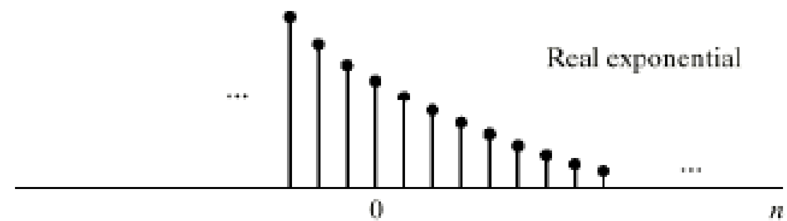
(a)



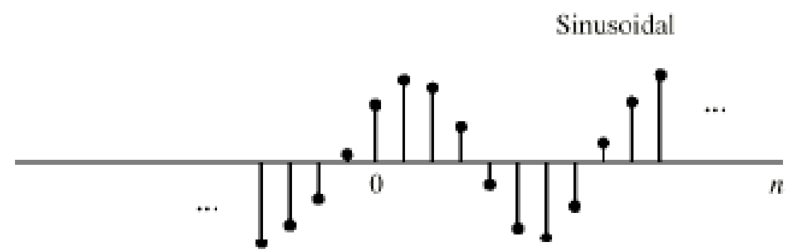
(b)

unit step $u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}$

real exponential $x[n] = \alpha^n$



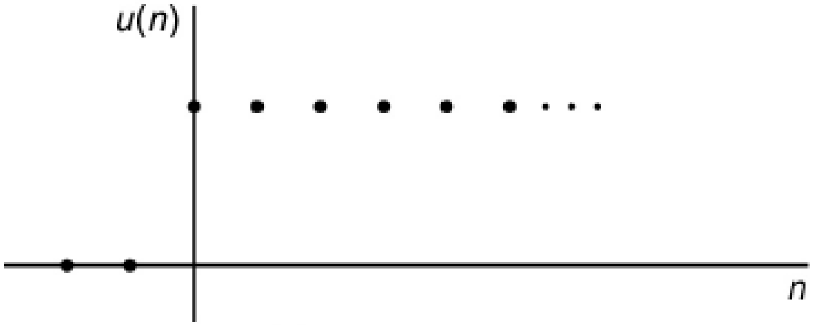
(c)



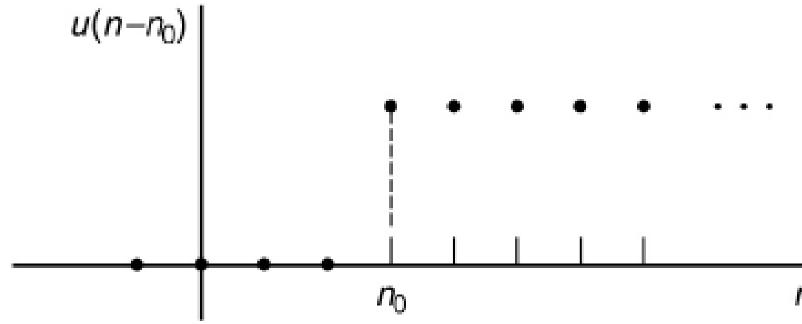
(d)

sine wave $x[n] = A\cos(\omega_0 n + \phi)$

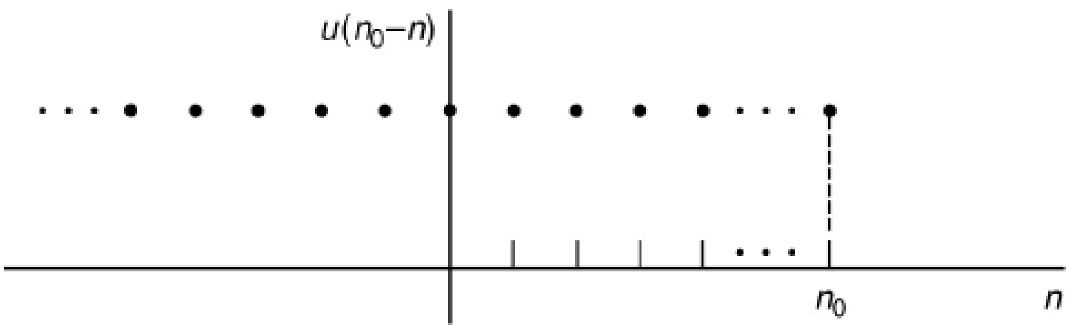
Variants on Discrete-Time Step Function



(a)



(b)

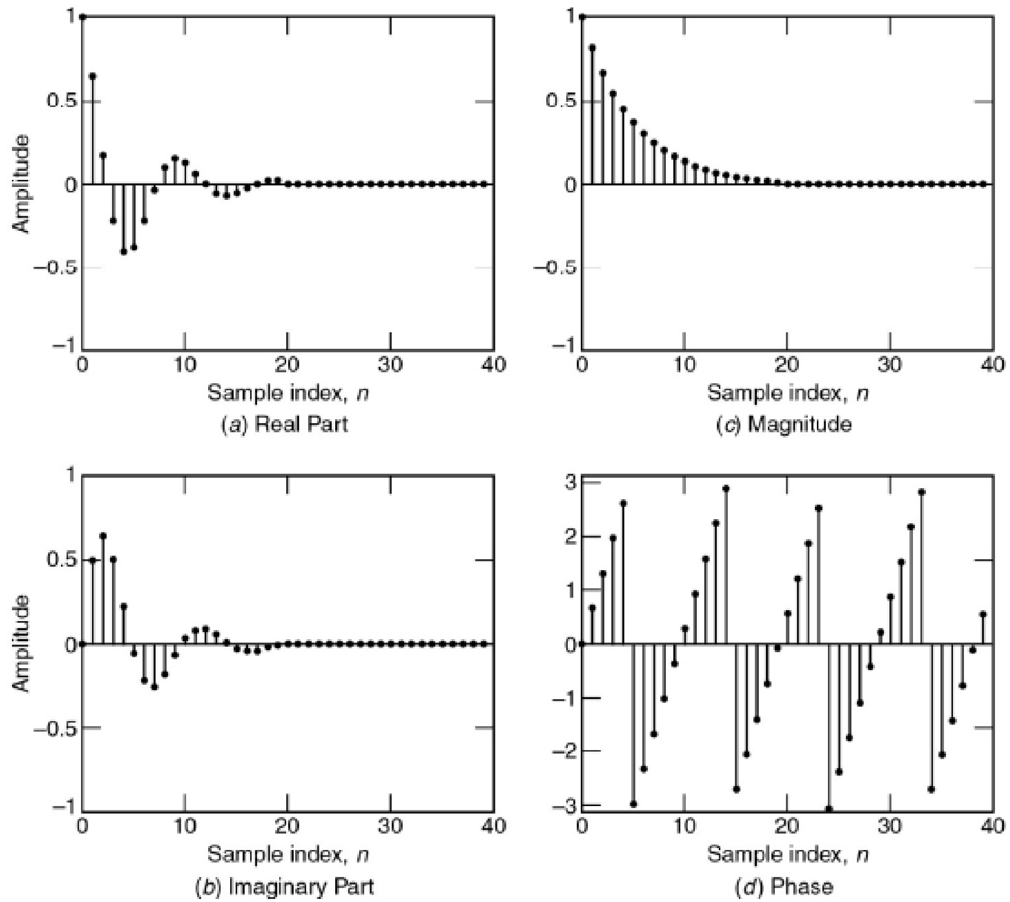


(c)

$n \rightarrow -n \;\Leftrightarrow\;$ signal flips around 0

Complex Signal

$$x[n] = (0.65 + 0.5j)^n \; u[n]$$

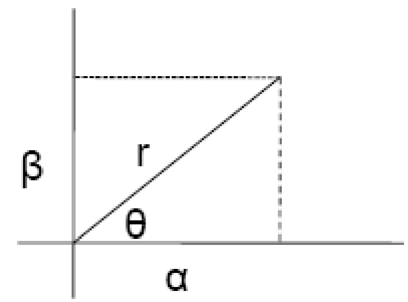


Complex Signal

$$x[n] = (\alpha + j\beta)^n u[n] = (re^{j\theta})^n u[n]$$

$$r = \sqrt{\alpha^2 + \beta^2}$$

$$\theta = \tan^{-1}(\beta / \alpha)$$



$$x[n] = r^n e^{j\theta n} u[n]$$

r^n is a dying exponential

$e^{j\theta n}$ is a linear phase term

Complex DT Sinusoid

$$x[n] = Ae^{j\omega n}$$

- Frequency ω is in radians (per sample), or just radians
  - not radians per second because “time” index n is dimensionless
  - once sampled, $x[n]$ is a sequence that **relates to time only through the sampling period** ***T***
- Important property: periodic in ω with period 2π:

$$Ae^{j\omega_0 n} = Ae^{j(\omega_0 + 2\pi r)n}$$

  - Only unique frequencies are 0 to 2π (or –π to +π)
  - Same applies to real sinusoids

Periodic DT Signals

- A signal is periodic with period N if $x[n] = x[n+N]$ for all n
- For the complex exponential this condition becomes

$$Ae^{j\omega_0 n} = Ae^{j(\omega_0 + \omega_0 N)n}$$

  - which requires $\omega_0 N = 2\pi k$ for some integer k
  - Thus, not all DT sinusoids are periodic!
- Consequence: there are N distinguishable frequencies with period N
  - e.g., $\omega_0 = 2\pi k/N$, $k = 0, 1, \ldots, N-1$

Signal Processing

- Transform digital signal into more desirable form

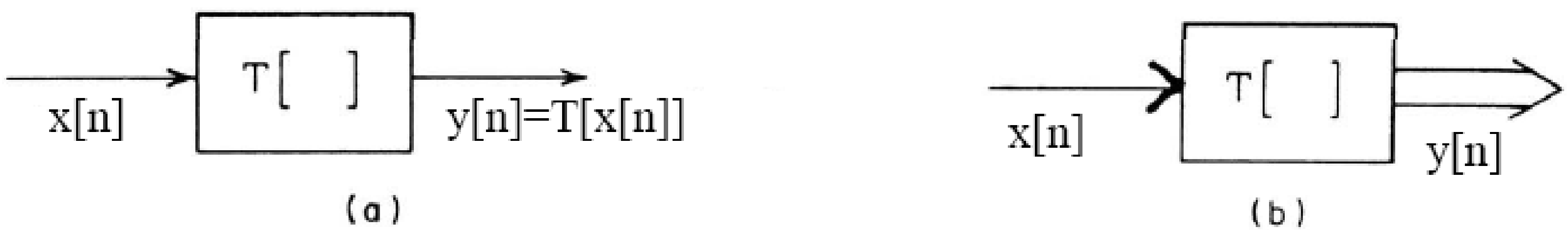
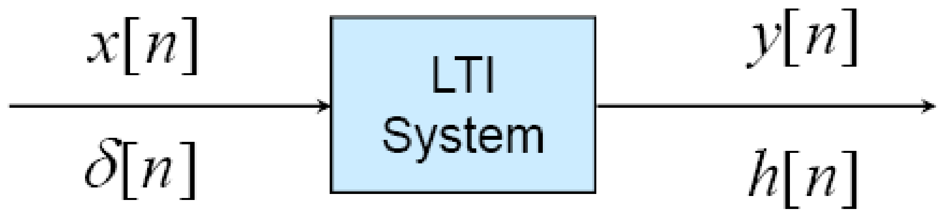


Fig. 2.3 Block diagram representations of: (a) single input/single output system; (b) single input/multiple output system.

single input—single output

single input—multiple output, e.g., filter bank analysis, sinusoidal sum analysis, etc.

LTI Discrete-Time Systems



- Linearity (superposition):

$$\mathrm{T}\{ax_1[n]+bx_2[n]\}=a\mathrm{T}\{x_1[n]\}+b\mathrm{T}\{x_2[n]\}$$

- Time-Invariance (shift-invariance):

$$x_1[n]=x[n-n_d] \quad \Rightarrow \quad y_1[n]=y[n-n_d]$$

- LTI implies discrete convolution:

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

LTI Discrete-Time Systems

Example:

Is system $y[n] = x[n] + 2x[n+1] + 3$ linear?

$$x_1[n] \leftrightarrow y_1[n] = x_1[n] + 2x_1[n+1] + 3$$

$$x_2[n] \leftrightarrow y_2[n] = x_2[n] + 2x_2[n+1] + 3$$

$$x_1[n] + x_2[n] \leftrightarrow y_3[n] = x_1[n] + x_2[n] + 2x_1[n+1] + 2x_2[n+1] + 3$$

$\neq y_1[n] + y_2[n] \Rightarrow$ Not a linear system!

Is system $y[n] = x[n] + 2x[n+1] + 3$ time/shift invariant?

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

$y[n-n_0] = x[n-n_0] + 2x[n-n_0+1] + 3 \Rightarrow$ System is time invariant!

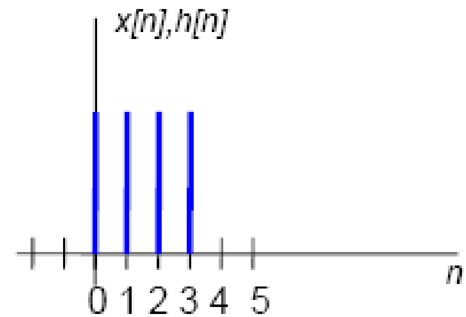
Is system $y[n] = x[n] + 2x[n+1] + 3$ causal?

$y[n]$ depends on $x[n+1]$, a sample in the future

$\Rightarrow$ System is not causal!

Convolution Example

$$x[n] = \begin{cases} 1 & 0 \le n \le 3 \\ 0 & \text{otherwise} \end{cases} \qquad h[n] = \begin{cases} 1 & 0 \le n \le 3 \\ 0 & \text{otherwise} \end{cases}$$

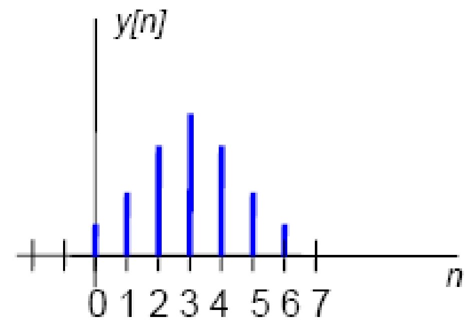


What is $y[n]$ for this system?

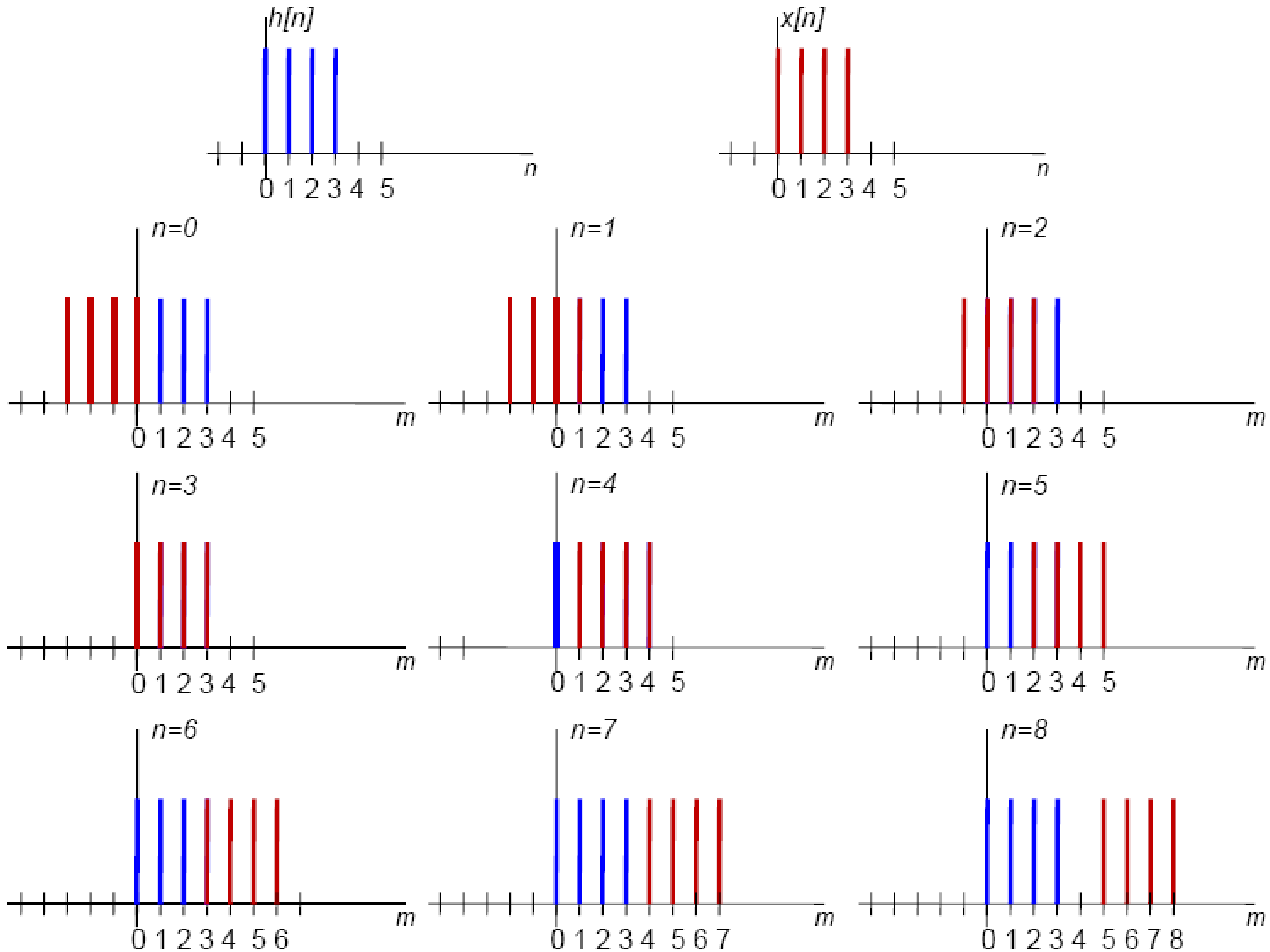
Solution :

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

$$= \begin{cases} \displaystyle\sum_{m=0}^{n} 1 \cdot 1 = (n+1) & 0 \le n \le 3 \\ \displaystyle\sum_{m=n-3}^{3} 1 \cdot 1 = (7-n) & 4 \le n \le 6 \\ 0 & n \le 0,\ n \ge 7 \end{cases}$$



Convolution Example



Convolution Example

The impulse response of an LTI system is of the form:

$$h[n] = a^n \; u[n] \qquad |a| < 1$$

and the input to the system is of the form:

$$x[n] = b^n \; u[n] \qquad |b| < 1, \, b \neq a$$

Determine the output of the system using the formula for discrete convolution.

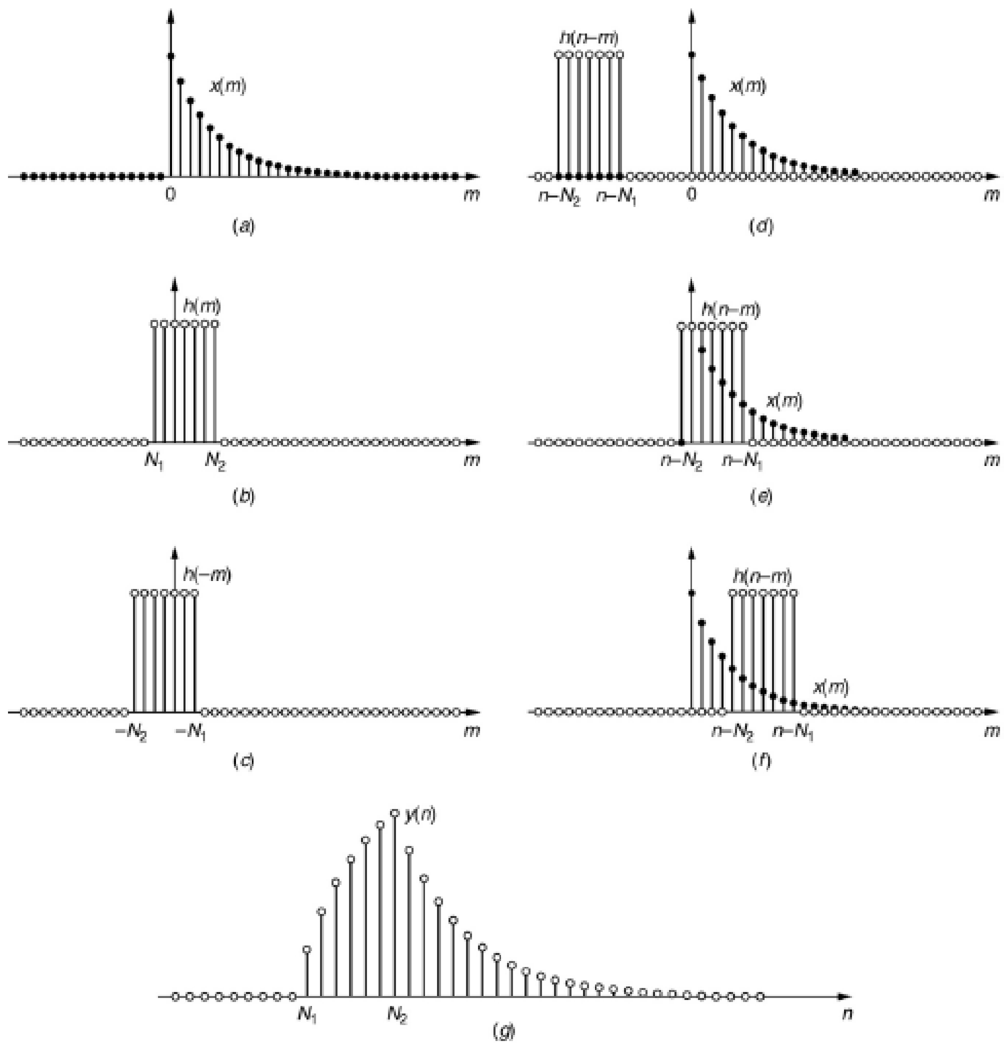
SOLUTION:

$$y[n] = \sum_{m=-\infty}^{\infty} a^m \, u[m] \, b^{n-m} \, u[n-m]$$

$$= b^n \sum_{m=0}^{n} a^m \, b^{-m} \, u[n] = b^n \sum_{m=0}^{n} (a/b)^m \, u[n]$$

$$= b^n \left[\frac{1-(a/b)^{n+1}}{1-(a/b)} \right] = \left[\frac{b^{n+1} - a^{n+1}}{b-a} \right] u[n]$$

Convolution Example



Convolution Example

Consider a digital system with input $x[n]=1$ for $n=0,1,2,3$ and 0 everywhere else, and with impulse response $h[n]=a^n u[n],\ |a|<1$. Determine the response $y[n]$ of this linear system.

SOLUTION:

We recognize that $x[n]$ can be written as the difference between two step functions, i.e., $x[n]=u[n]-u[n-4]$. Hence we can solve for $y[n]$ as the difference between the output of the linear system with a step input and the output of the linear system with a delayed step input. Thus we solve for the response to a unit step as:

$$y_1[n]=\sum_{m=-\infty}^{\infty} u[m]a^{n-m}u[n-m]=\left[\frac{a^n-a^{-1}}{1-a^{-1}}\right]u[n]$$

$$y[n]=y_1[n]-y_1[n-4]$$

Linear Time-Invariant Systems

- easiest to **understand**
- easiest to **manipulate**
- **powerful** processing capabilities
- **characterized completely** by their response to unit sample, *h(n)*, via **convolution relationship**

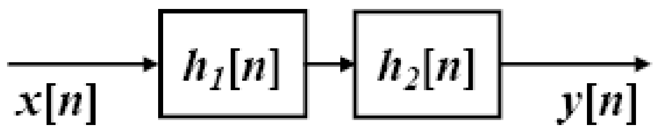
$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]\,h[n-k] = \sum_{k=-\infty}^{\infty} h[k]\,x[n-k]$$

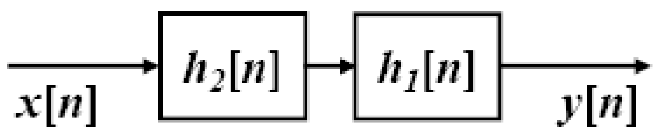
$$y[n] = h[n] * x[n], \quad \text{where } * \text{ denotes discrete convolution}$$

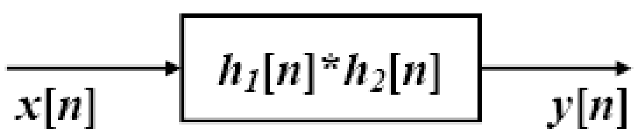
$$x[n] \longrightarrow \boxed{h[n]} \longrightarrow y[n]=x[n] * h[n]$$

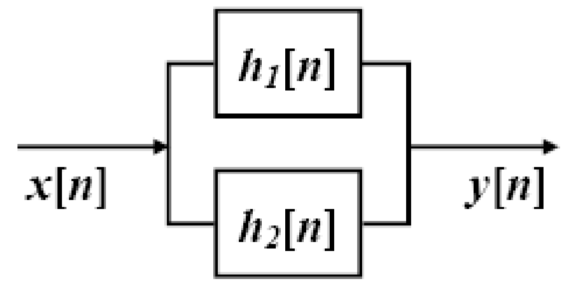
- basis for **linear filtering**
- used as **models for speech production** (source convolved with system)

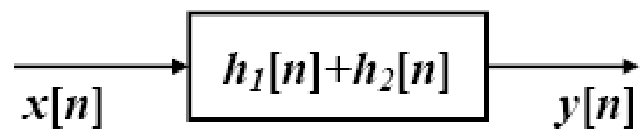
Equivalent LTI Systems







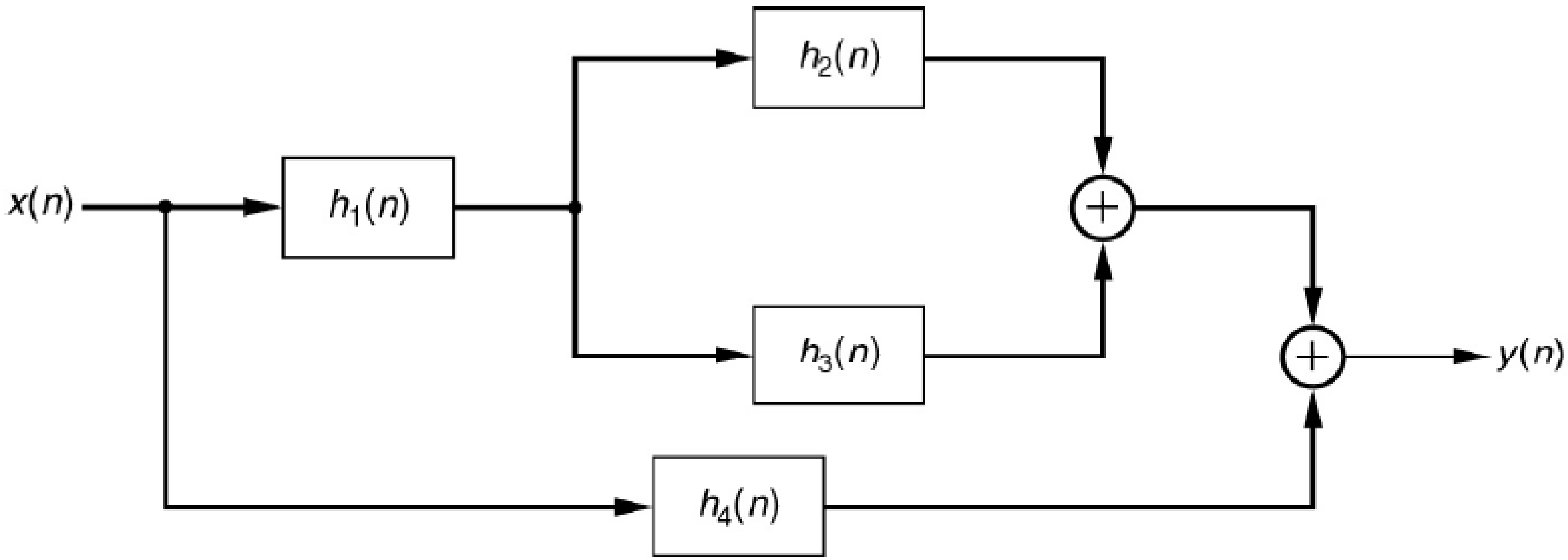




$$h_1[n]*h_2[n] = h_2[n]*h_1[n]$$

$$h_1[n]+h_2[n] = h_2[n]+h_1[n]$$

More Complex Filter Interconnections



$$y[n] = x[n] * h_c[n]$$

$$h_c[n] = h_1[n] * (h_2[n] + h_3[n]) + h_4[n]$$

Example 1. Identity Transform

- The identity system is defined by

  T: $y[n] = x[n]$, $-\infty < n < \infty$

- Find $h[n]$ that describes the system T, such that

  – $y[n]$ =$h[n] * x[n]$, $-\infty < n < \infty$

$$\begin{aligned} h[n] &= \delta[n] \\ &= u[n] - u[n-1] \end{aligned}$$

Example 2. Ideal Delay System

- The ideal delay system is defined by

  T: $y[n] = x[n - n_d]$, $-\infty < n < \infty$

- Find $h[n]$ that describes the system T, such that
  - $y[n] = h[n] * x[n]$, $-\infty < n < \infty$

$$h[n] = \begin{cases} 1 & n = n_d \\ 0 & otherwise \end{cases}$$
$$= \delta[n - n_d]$$

Example 3. Moving Average

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

$$= \frac{1}{M_1 + M_2 + 1} \{x[n+M_1] + \ldots + x[n-M_2]\}$$

$$= h[n] * x[n]$$

$$h[n] = \frac{1}{M_1 + M_2 + 1} \sum_{k=-M_1}^{M_2} \delta[n-k]$$

Example 4. Accumulator

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

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

- How to obtain the impulse response of the system → replace $x[n]$ with unit impulse $\delta[n]$, then the output $y[n]$ becomes $h[n]$
- So, $h[n]$ is called ***impulse response function*** of a system

Example 5. Difference Functions

- forward difference system

  T: $y[n] = x[n+1] - x[n]$

  IRF: $h[n] = \delta[n+1] - \delta[n]$

- backward difference system

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

  IRF: $h[n] = \delta[n] - \delta[n-1]$

Example 6. Compressor

$$y[n] = x[Mn], \quad -\infty < n < \infty$$

$$x_1[n] = x[n - n_0]$$
$$y_1[n] = x_1[Mn]$$
$$= x[Mn - n_0]$$

$$y[n - n_0] = x[M(n - n_0)]$$
$$\neq y_1[n]$$

- Also called ***decimator*** or ***down-sampler***
- Is this **LTI**?
- No, except when M=1

Other Characteristics

- Causality
  - the impulse response is causal only if it depends on the past sequence of the input
- Stability
  - an LTI system is stable only if its impulse response is absolutely summable
  - also called finite-duration impulse response (FIR) system

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

DSP Reference

- Discrete-Time Signal Processing (now 3rd Edition); Alan V. Oppenheim and Ronald W. Schafer; Prentice Hall; 2009

DEEE725 음성신호처리실습, 장길진

z-transform, Fourier transform

CHAPTER 2. FUNDAMENTALS OF DIGITAL SIGNAL PROCESSING, PART 3

z-Transform Representations

- Definition: infinite power series in z^{-1}, with $x[n]$ as coefficients of term in z^{-n}
  - z is a complex variable
- $X(z)$ is finite and converges only for certin values of z:
  - sufficient condition for convergence
  - region of convergence:

$$x[n] \leftrightarrow X(z)$$

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

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

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

$$R_1 < |z| < R_2$$

Examples of Convergence Regions

1. $x[n] = \delta[n - n_0]$ – delayed impulse
$X(z) = z^{-n_0}$ – converges for
$|z| > 1, n_0 > 0;\ |z| < 1, n_0 < 0;\ \forall z < \infty, n_0 < 0;$

2. $x[n] = u[n] - u[n - N]$ – box pulse
$$X(z) = \sum_{n=0}^{N-1} (1) z^{-n} = \frac{1 - z^{-N}}{1 - z^{-1}}$$ – converges for $0 < |z| < \infty$
– <u>all finite length sequences converge in the region $0 < |z| < \infty$</u>

3. $x[n] = a^n u[n] (a < 1)$
$$X(z) = \sum_{n=0}^{\infty} a^n z^{-n} = \frac{1}{1 - a z^{-1}}$$ – converges for $|z| > |a|$
– <u>all infinite duration sequences which are non-zero for $n \geq 0$ converge in a region $|z| > R_1$</u>

Examples of Convergence Regions

4. $x[n] = -b^n u[-n-1]$

$X(z) = \sum_{n=-\infty}^{-1} -b^n z^{-n} = \frac{1}{1-bz^{-1}}$ − converges for $|z| < |b|$

− <u>all infinite duration sequences which are non-zero for $n < 0$ converge in a region $|z| < R_2$</u>

5. $x[n]$ non-zero for $-\infty < n < \infty$

viewed as a combination of 3 and 4

$\Rightarrow$ giving a convergence region $R_1 < |z| < R_2$

− sub-sequence for $n \geq 0 \rightarrow |z| > R_1$

− sub-sequence for $n < 0 \rightarrow |z| < R_2$

− total sequence $\rightarrow R_1 < |z| < R_2$

Some z-Transforms

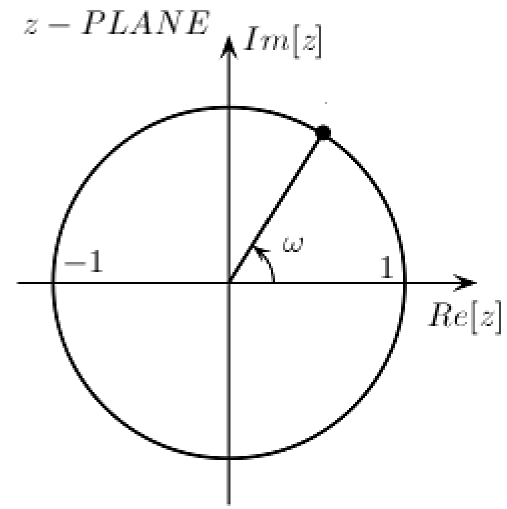
Property	Sequence	z-Transform
Linearity	$ax_1[n] + bx_2[n]$	$aX_1(z) + bX_2(z)$
Shift	$x[n+n_0]$	$z^{n_0}X(z)$
Exponential	$a^n x[n]$	$X(a^{-1}z)$
Linear Weighting	$nx[n]$	$-z\frac{dX(z)}{dz}$
Time reversal	$\boxed{x[-n]}$ 1	$X(z^{-1})$
Convolution	$x[n] * h[n]$	$X(z)H(z)$
Multiplication	$x[n]w[n]$	$\frac{1}{2\pi j}\overset{2}{\boxed{\oint_C}} X(v)W(\frac{z}{v})v^{-1}dv$

*1: non-causal, need $x[N_0-n]$ to be causal for finite length sequence

*2: circular convolution in the frequency domain

Discrete-Time Fourier Transform

The discrete-time Fourier transform (DTFT) is defined by an evaluation of *X*(*z*) on the unit circle in the z-plane



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

$$z = e^{j\omega} \Leftrightarrow |z| = 1, \arg(z) = j\omega$$

sufficient condition for existence of Fourier transform:
$$\sum n = -\infty^{\infty} |x[n]||z^{-n}| = \sum n = -\infty^{\infty} |x[n]| < \infty, \because |z| = 1$$

Properties – periodic; period of 2π corresponds to once around unit circle in the z-plane

$$X(e^{j\omega}) = X(e^{j(\omega+2\pi n)})$$

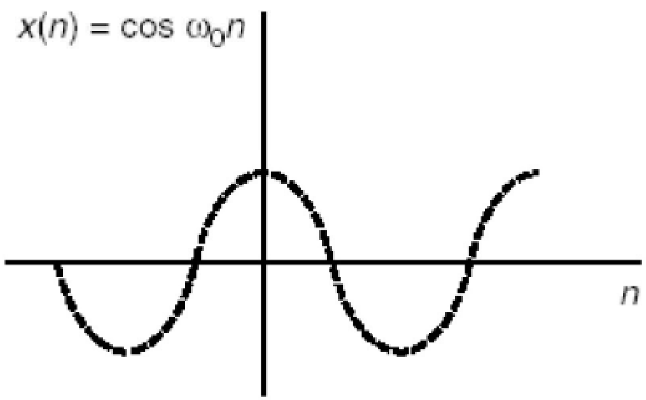
Simple DTFTs

Impulse	$x[n] = \delta[n],$	$X(e^{j\omega}) = 1$
Delayed impulse	$x[n] = \delta[n - n_0],$	$X(e^{j\omega}) = e^{-j\omega n_0}$
Step function	$x[n] = u[n],$	$X(e^{j\omega}) = \dfrac{1}{1 - e^{-j\omega}}$
Rectangular window	$x[n] = u[n] - u[n - N],$	$X(e^{j\omega}) = \dfrac{1 - e^{-j\omega N}}{1 - e^{-j\omega}}$
Exponential	$x[n] = a^n \; u[n],$	$X(e^{j\omega}) = \dfrac{1}{1 - ae^{-j\omega}},\; a < 1$
Backward exponential	$x[n] = -b^n \; u[-n - 1],$	$X(e^{j\omega}) = \dfrac{1}{1 - be^{-j\omega}},\; b > 1$

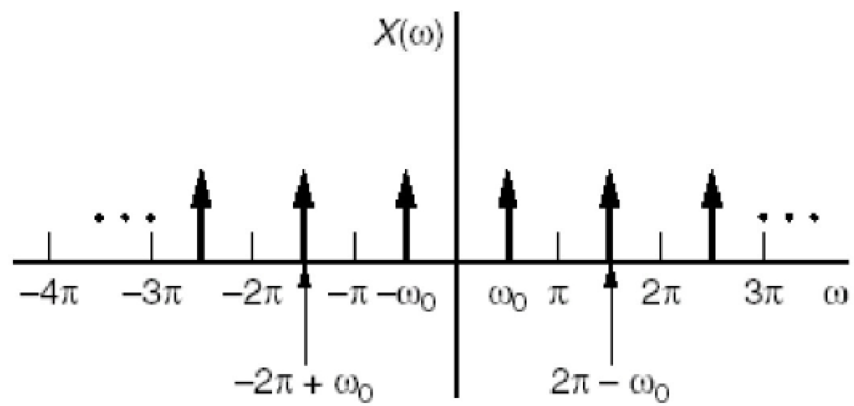
DTFT of a Cosine Signal

$x[n] = cos(\omega_0 n),\ -\infty < n < \infty$
$X(e^{j\omega}) = \sum_{k=-\infty}^{\infty} [\pi\delta\,(\omega - \omega_0 + 2\pi k) + \pi\delta\,(\omega + \omega_0 + 2\pi k)]$

Within interval $-\pi < \omega < \pi$,
$X(e^{j\omega})$ is comprised of a pair of impulses at $\pm\omega_0$



(a)



(b)

DFT – discrete Fourier transform

Discrete Fourier Transform

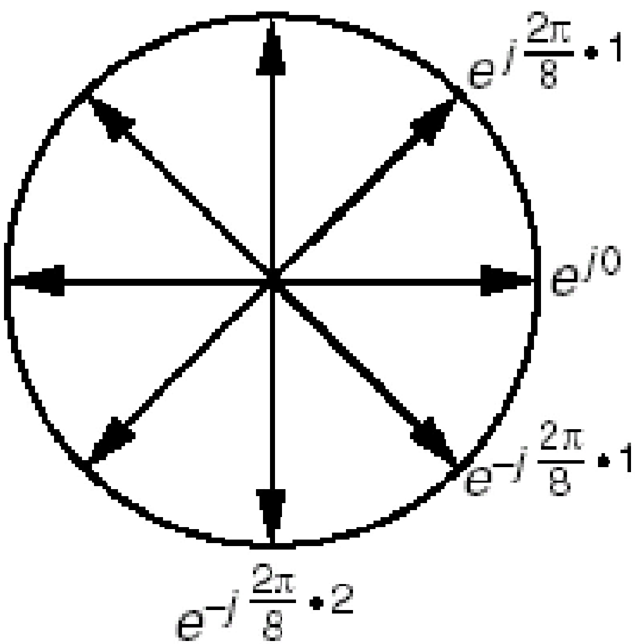
- consider a periodic signal with period *N* (samples), such that
  - $x[n] = x[n+N], -\infty < n < \infty$
- $x[n]$ can be represented exactly by a discrete sum of sinusoids – **<u>exact</u>** representation of the discrete periodic sequence
  - $x[n]$: *N* sequence values
  - $X[k]$: *N* DFT coefficients

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

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

DFT can be viewed as computing correlation of input signal with sinusoids (sin and cosine functions)

Sampling the DTFT



$$k=0;\ e^{-j2\pi k/8}=1$$

$$k=1;\ e^{-j2\pi k/8}=\frac{\sqrt{2}}{2}(1-j)$$

$$k=2;\ e^{-j2\pi k/8}=-j$$

$$k=3;\ e^{-j2\pi k/8}=\frac{\sqrt{2}}{2}(-1-j)$$

$$k=4;\ e^{-j2\pi k/8}=-1$$

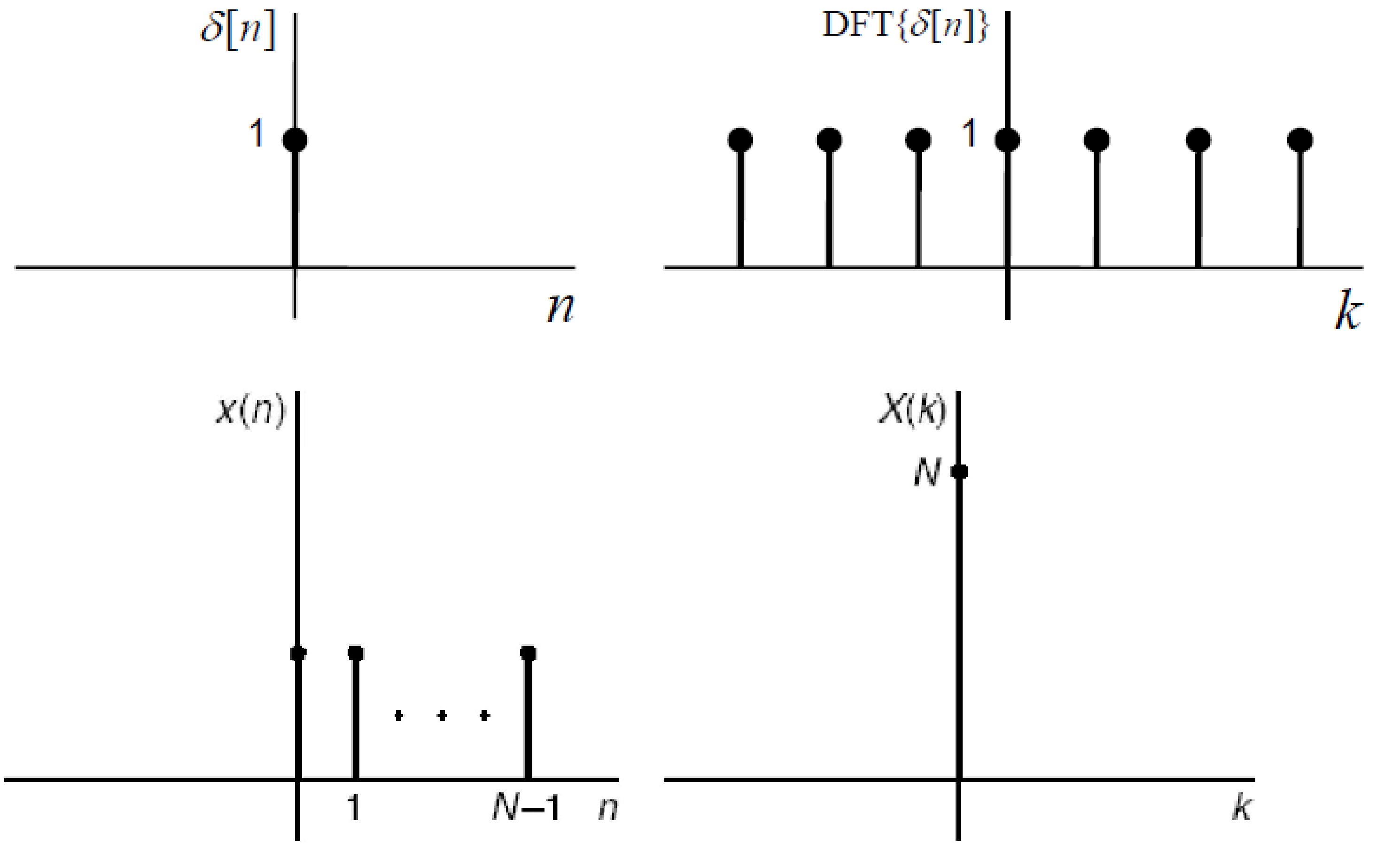
$$k=5;\ e^{-j2\pi k/8}=\frac{\sqrt{2}}{2}(-1+j)$$

$$k=6;\ e^{-j2\pi k/8}=j$$

$$k=7;\ e^{-j2\pi k/8}=\frac{\sqrt{2}}{2}(1+j)$$

$X(e^{j\omega})$ is evaluated (**sampled**) at N equally spaced normalized frequencies $\omega_k = (\frac{2\pi k}{N})$, for $k = 0, 1, \ldots, N-1$

DFT Examples



DFT Properties

- The DFT, $X[k]$, can be viewed as a sampled version of the DTFT of a finite-length sequence
- The DFT has properties very similar to many of the useful ones of z-transform and DTFT
- The N values of $X[k]$ can be computed very efficiently, in $O(N \log N)$, by a set of computational algorithms known as the *fast Fourier transform* (FFT)

Property	Sequence	N-point DFT
Linearity	$ax_1[n] + bx_2[n]$	$aX_1[k] + bX_2[k]$
Shift	$x[((n - n_0))_N]$	$e^{-j\frac{2\pi k}{N}n_0}X[k]$
Modulation	$x[n]e^{j\frac{2\pi k_0}{N}n}$	$X[((k - k_0))_N]$
Time reversal	$x[((-n))_N]$	$X[((-k))_N] = X^*[k]$
Convolution	$\sum_{m=0}^{N-1} x[m]h[((n - m))_N]$	$X[k]H[k]$
Multiplication	$x[n]w[n]$	$\frac{1}{N}\sum_{r=0}^{N-1} X[r]W[((k - r))_N]$
Parseval's Theorem	$\sum_{n=0}^{N-1} \lvert x[n]\rvert^2 = \frac{1}{N}\sum_{n=0}^{N-1} \lvert X[k]\rvert^2$	

DEEE725 음성신호처리실습, 장길진

Digital filters and MATLAB examples

CHAPTER 2. FUNDAMENTALS OF DIGITAL SIGNAL PROCESSING, PART 4

Digital Filters

- digital filter is a discrete-time linear, shift invariant system with input-output relation:

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

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

- $H(z)$ is the system function with $H(e^{j\omega})$ as the complex frequency response

$$H(e^{j\omega}) = H_r(e^{j\omega}) + jH_i(e^{j\omega}) \quad \text{real, imaginary representation}$$

$$H(e^{j\omega}) = |H(e^{j\omega})| \cdot e^{j \arg H(e^{j\omega})} \quad \text{magnitude, phase representation}$$

$$\log H(e^{j\omega}) = \log |H(e^{j\omega})| + j \arg H(e^{j\omega})$$

$$|H(e^{j\omega})|^2 = H_r^2(e^{j\omega}) + H_i^2(e^{j\omega})$$

Causality and Stability

- causal linear shift-invariant ➔ $h[n]=0$ for $n<0$
- stable system ➔ every bounded input produces a bounded output
- a necessary and sufficient condition for stability and for the existence of $H(e^{j\omega})$

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

Digital Filter Implementation

- input and output satisfy linear difference equation of the form:

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

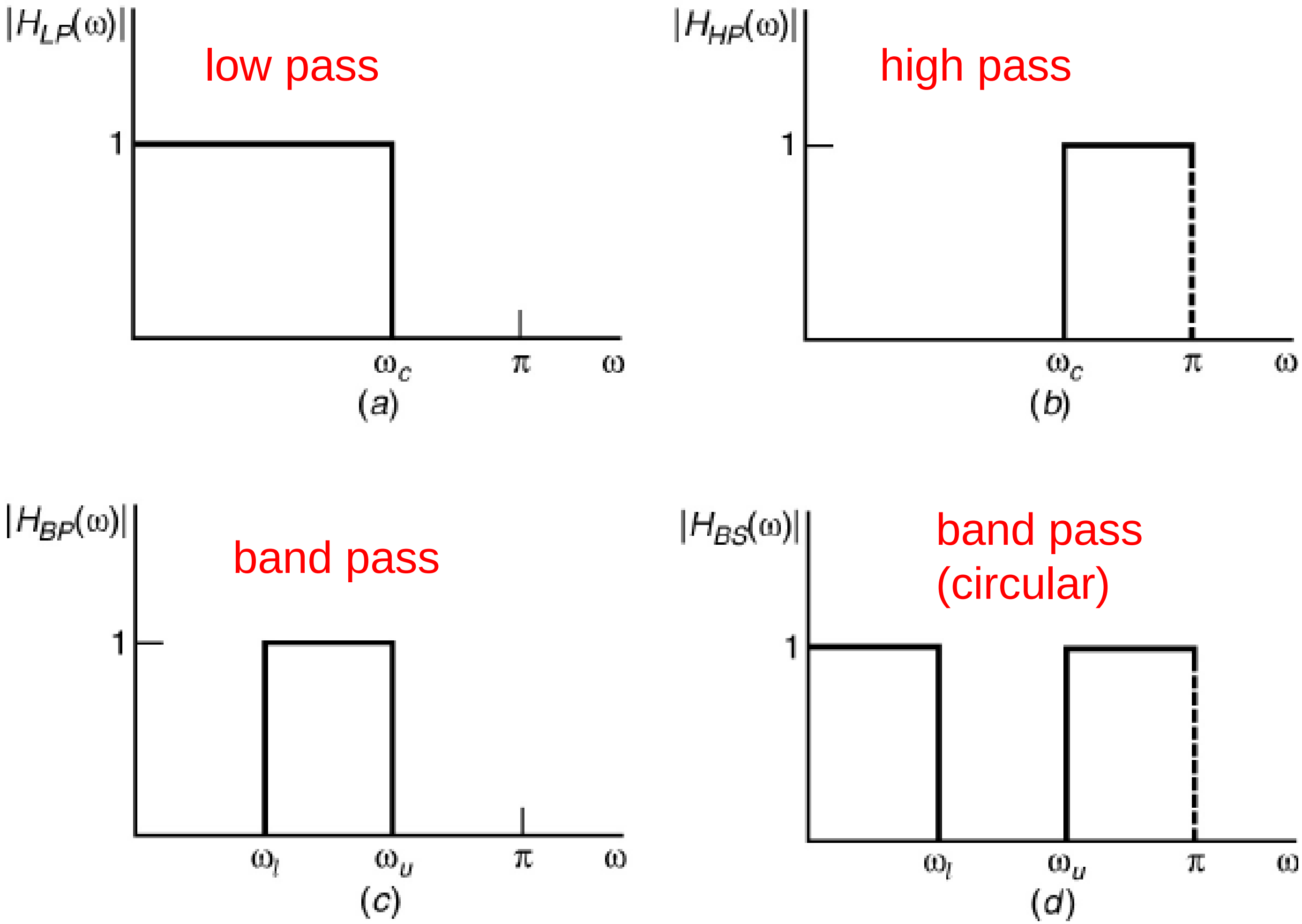
- evaluating z-transforms of both sides gives:
  - a rational function in z^{-1}
  - M zeros, N poles
    - zero makes denominator zero, while pole makes numerator zero

$$Y(z)-\sum_{k=1}^{N}a_k z^{-k}Y(z)=\sum_{r=0}^{M}b_r z^{-r}X(z)$$

$$Y(z)\left(1-\sum_{k=1}^{N}a_k z^{-k}\right)=X(z)\sum_{r=0}^{M}b_r z^{-r}$$

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

Ideal Filter Responses



FIR Systems

current output sample depends on past input only, i.e., all a_k's are zero

$$y[n] = \sum_{r=0}^{M} b_r x[n-r] = b_0 x[n] + b_1 x[n-1] + \ldots + b_M x[n-M]$$

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

$$H(z) = \sum_{r=0}^{M} b_r z^{-r}$$

M zeros, no pole

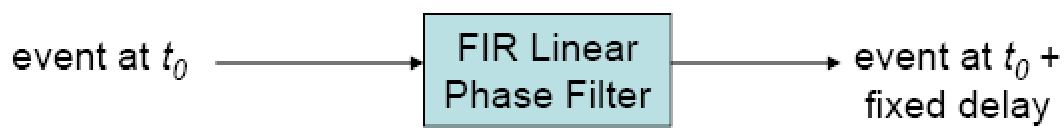
Linear Phase Filter

- An FIR filter *h*[*n*] is called ***linear phase*** if it satisfies either of the relations

$$h[n] = h[M-n] \ \ \text{: symmetric} \Leftrightarrow A\left(e^{j\omega}\right) = \text{purely real}$$

$$h[n] = -h[M-n] \ \ \text{: anti-symmetric} \Leftrightarrow A\left(e^{j\omega}\right) = \text{purely imaginary}$$

  - a function is called linear phase if the phase response (output) of the filter is a linear function of frequency



  - symmetric linear phase filters are very common, such as Wiener filters
  - an LTI system is ***minimum-phase*** if the system and its inverse are causal and stable (roots within a unit circle)

FIR Filter Design Methods

- cost of linear phase filter designs
  - any response can ne theoretically approximated to any degree of accuracy
  - it requires longer filters than non-linear phase designs
- FIR filter design methods
  - **<u>window</u>** approximation
    - analytical, closed form method
  - frequency sampling
    - iterative optimization method
  - Optimal (minimax error) approximation
    - iterative optimization method

Windowing

- The exact frequency response of a system requires infinite sequence ➔ practically impossible
- Assume the finite length sequence to be multiplication of a finite window and infinite sequence ➔ the frequency response becomes the convolution of frequency responses of the system and the window

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]$$

$$w[n] = \begin{cases} w_n & 0 \le n \le M \\ 0 & otherwise \end{cases}$$

$$\tilde{h}[n] = \sum_{n=-\infty}^{\infty} w[n]h[n] = \sum_{n=0}^{M} w_n h[n]$$

$$\tilde{H}(z) = W(z) * H(z)$$

➔ Design the window so that its frequency response should be as close as an impulse function

Common Windows

1. Rectangular $w[n] = \begin{cases} 1 & 0 \le n \le M \\ 0 & otherwise \end{cases}$

2. Bartlett $w[n] = 1 - \dfrac{2\,|\,n - M/2\,|}{M}$

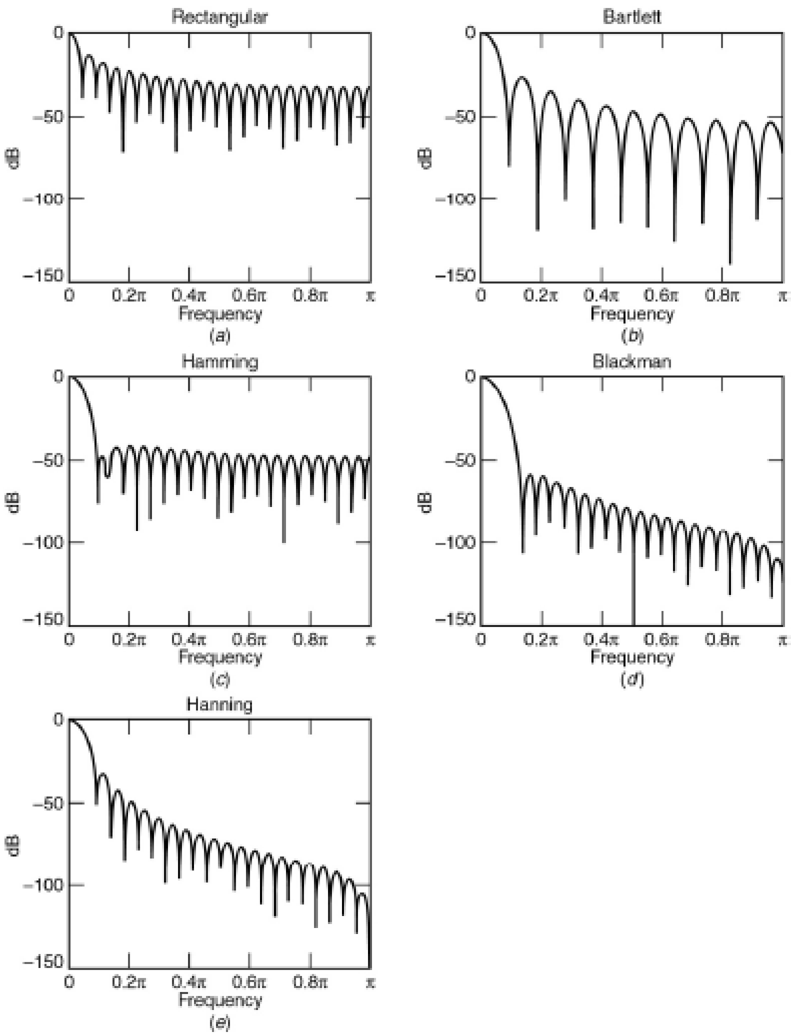
3. Blackman $w[n] = 0.42 - 0.5\cos\left(\dfrac{2\pi n}{M}\right) + 0.08\cos\left(\dfrac{4\pi n}{M}\right)$

4. Hamming $w[n] = 0.54 - 0.46\cos\left(\dfrac{2\pi n}{M}\right)$

5. Hanning $w[n] = 0.5 - 0.5\cos\left(\dfrac{2\pi n}{M}\right)$

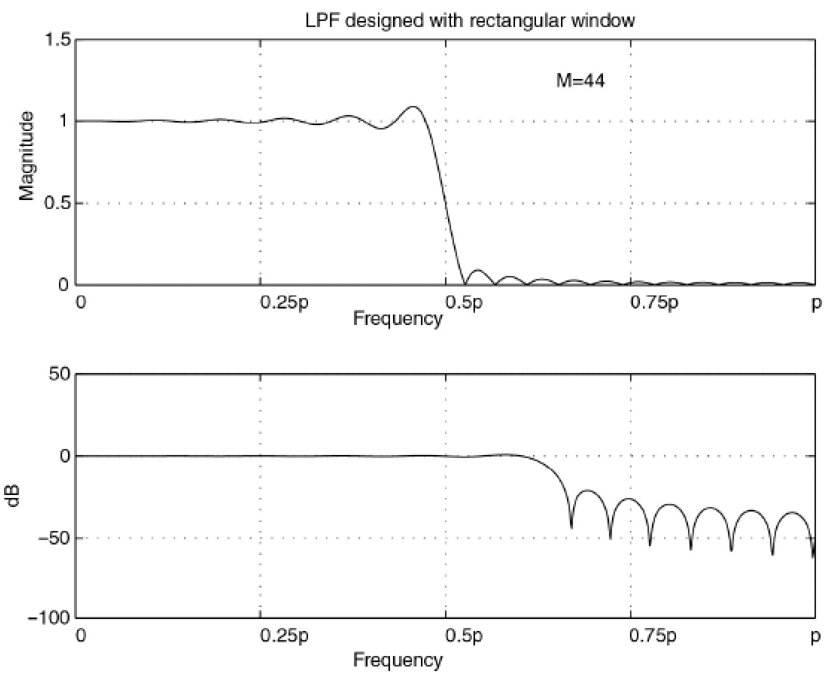
6. Kaiser $w[n] = \dfrac{I_0\left\{\beta\sqrt{1 - ((n - M/2)/(M/2))^2}\right\}}{I_0\{\beta\}}$

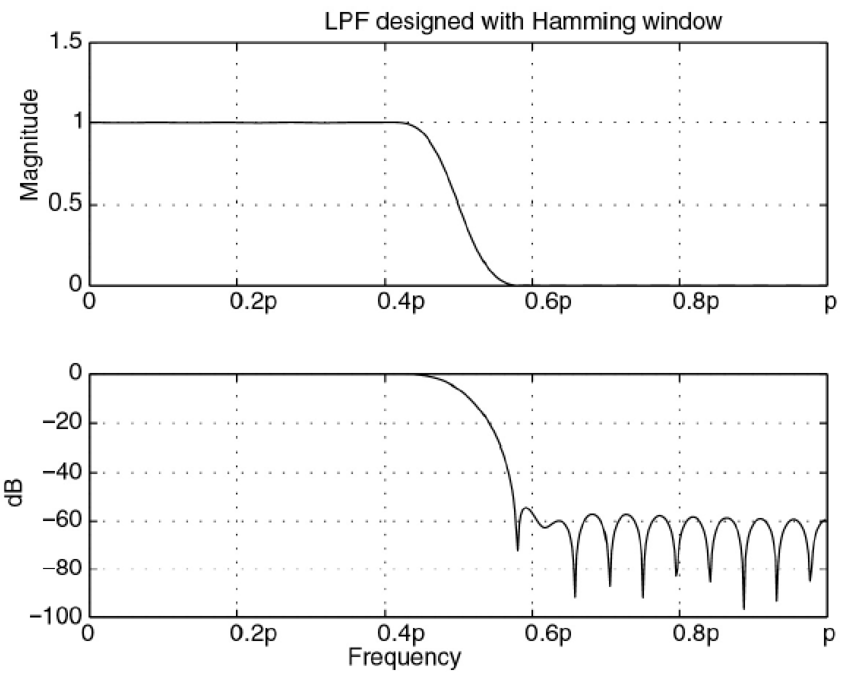
Frequency Responses of Common Windows



Window	Mainlobe Width	Side lobe Attenuation
Rectangular	4π/M	-13 dB
Bartlett	8π/M	-27 dB
Hamming	8π/M	-43 dB
Blackman	12π/M	-58 dB
Hanning	8π/M	-32 dB

Low Pass Filter Examples



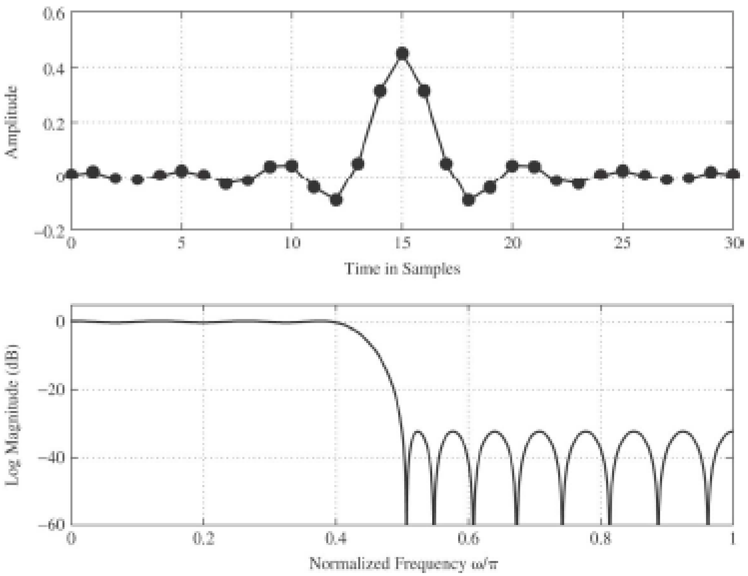


MATLAB FIR Design

1. Use **firpm** to design FIR filters
   >> B=firpm(N,F,A)
   – N+1 point linear phase, FIR design
   – B=filter coefficients (numerator polynomial)
   – F=ideal frequency response band edges (in pairs) (normalized to 1.0)
   – A=ideal amplitude response values (in pairs)
2. Use **freqz** to convert to frequency response (complex)
   >> [H,W]=freqz(B,den,NF)
   – H=complex frequency response
   – W=set of radian frequencies at which FR is evaluated (0 to pi)
   – B=numerator polynomial=set of FIR filter coefficients
   – den=denominator polynomial=[1] for FIR filter
   – NF=number of frequencies at which FR is evaluated
3. Use plot to evaluate log magnitude response
   >> plot(W/pi, 20log10(abs(H)))

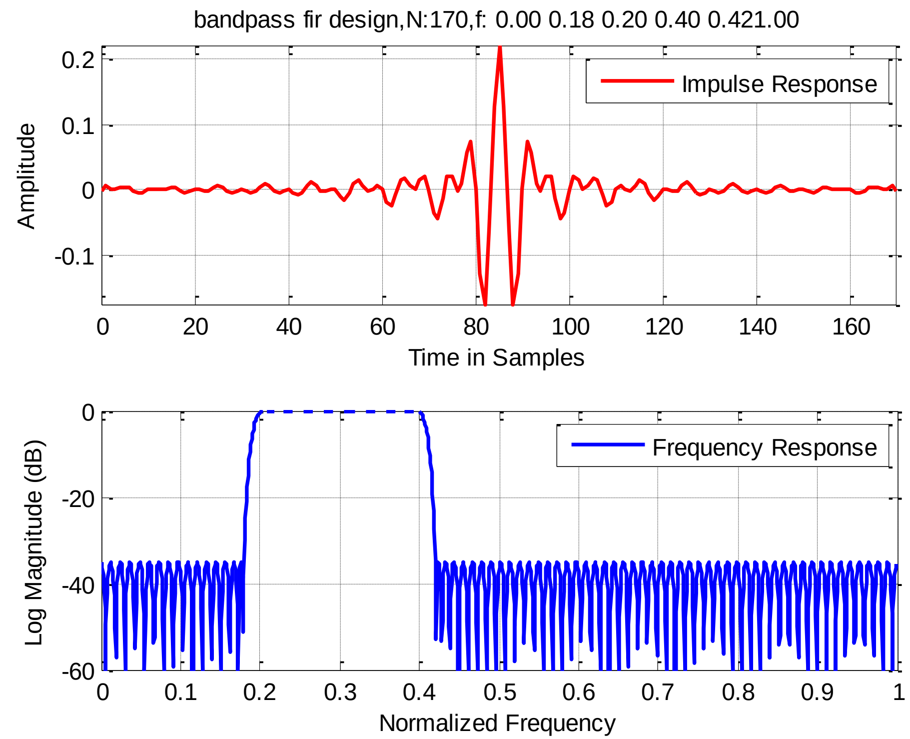
Lowpass Filter Design

```
N=30;
F=[0 0.4 0.5 1];
A=[1 1 0 0];
B=firpm(N,F,A);

NF=512;
[H,W]=freqz(B,1,NF);
plot(W/pi,20log10(abs(H)));
```



Bandpass Filter Design

```
% bandpass_filter_design
N=170;
F=[0 0.18 .2 .4 .42 1];
A=[0 0 1 1 0 0];
B=firpm(N,F,A);

NF=1024;
[H,W]=freqz(B,1,NF);
figure,orient landscape;
stitle=sprintf('bandpass fir design, N:%d’,N);
n=0:N;
subplot(211),plot(n,B,'r','LineWidth',2);
axis tight,grid on,title(stitle);
xlabel('Time in Samples'),ylabel('Amplitude');
legend('Impulse Response');
subplot(212),plot(W/pi,20*log10(abs(H)),'b','LineWidth',2);
axis ([0 1 - 60 0]); grid on;
xlabel('Normalized Frequency');
ylabel('Log Magnitude (dB)');
legend('Frequency Response');
```



IIR Systems

general filters, y[n] depends on y[n-1], y[n-2],…, y[n-N] as well as x[n-1], x[n-2],…, x[n-M]

$$\boxed{y[n] = \sum_{k=0}^{N} a_k y[n-k] + \sum_{r=0}^{M} b_r x[n-r]}$$

partial fraction expansion is possible, with $A_0 = 0$ for M<N:

$$H(z) = \frac{\sum_{r=0}^{M} b_r z^{-r}}{1 - \sum_{k=1}^{N} a_k z^{-k}} = A_0 + \sum_{k=1}^{N} \frac{A_k}{1 - d_k z^{-1}}$$

for a causal system, it is shown that

$$h[n] = A_0 \delta[n] + \sum_{k=1}^{N} A_k (d_k)^n u[n]$$

h[n] has infinite duration due to u[n] – infinite impulse response (IIR)

IIR Filter Design

- IIR filter issues:
  - efficient implementations in terms of computations
  - can approximate any desired magnitude response with **arbitrarily small error**
  - **non-linear phase ➔** time dispersion of waveform
- IIR filter design methods
  - **Butterworth** designs-maximally flat amplitude
  - **Bessel** designs-maximally flat group delay
  - **Chebyshev** designs-equi-ripple in either passband or stopband
  - **Elliptic** designs-equi-ripple in both passband and stopband

Matlab Elliptic Filter Design

- use **ellip** to design elliptic filter

```
>> [B,A]=ellip(N,Rp,Rs,Wn)
```

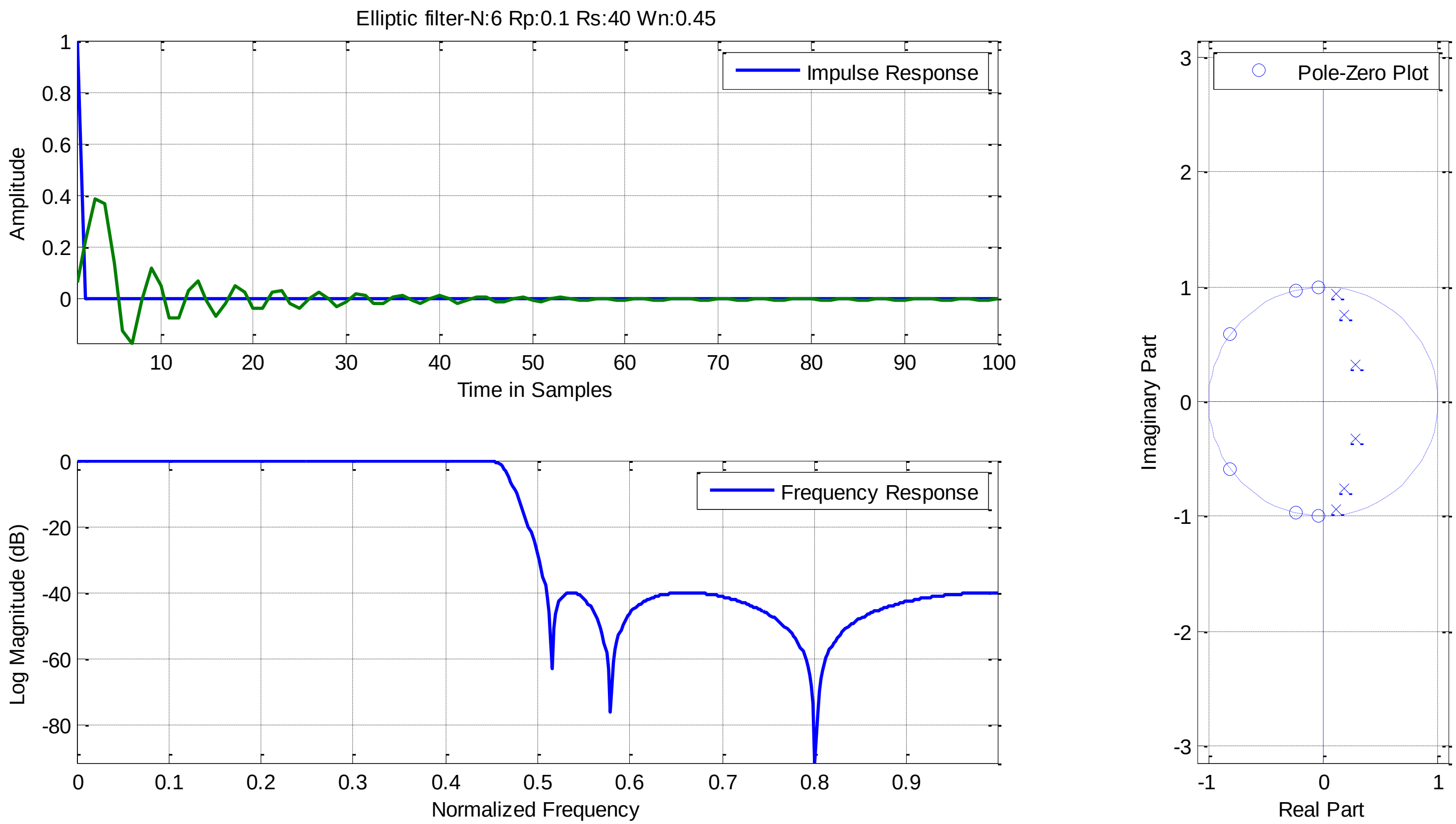
  - B=numerator polynomial—N+1 coefficients
  - A=denominator polynomial—N+1 coefficients
  - N=order of polynomial for both numerator and denominator
  - Rp=maximum in-band (passband) approximation error (dB)
  - Rs=out-of-band (stopband) ripple (dB)
  - Wp=end of passband (normalized radian frequency)
- use filter to generate impulse response

```
>> y=filter(B,A,x)
```

  - y=filter impulse response
  - x=filter input (impulse)
- use zplane to generate pole-zero plot

```
>> zplane(B,A)
```

Matlab Elliptic Lowpass Filter




```
[b,a]=ellip(6,0.1,40,0.45);
[h,w]=freqz(b,a,512);
zplane(b,a);
```

DEEE725 음성신호처리실습, 장길진

END OF CHAPTER 2. FUNDAMENTALS OF DIGITAL SIGNAL PROCESSING