

Signal & Linear system

Chapter 3 Time Domain Analysis of DT System

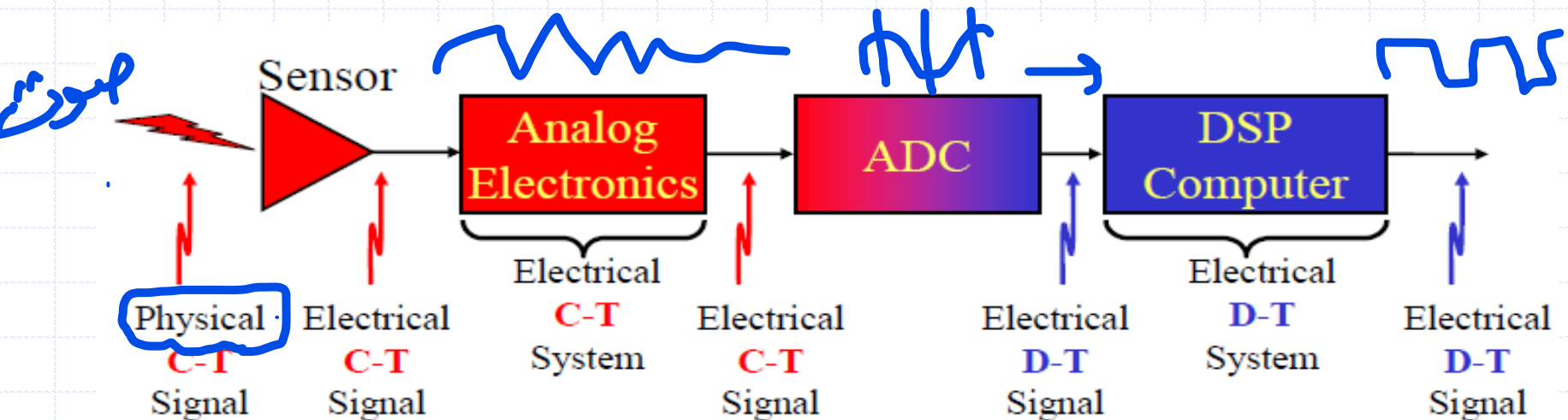
Basil Hamed

3.1 Introduction

Recall from Ch #1 that a common scenario in today's electronic systems is to do most of the processing of a signal using a computer.

DT

A computer can't directly process a C-T signal but instead needs a stream of numbers...which is a **D-T signal**.



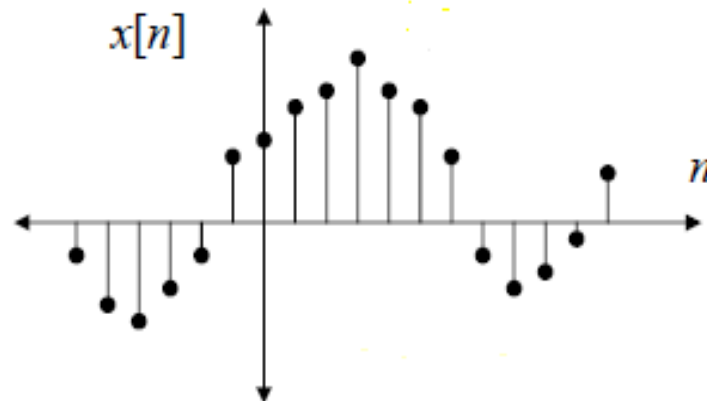
3.1 Introduction

What is a discrete-time (D-T) signal?

A discrete time signal is a sequence of numbers indexed by integers **Example:** $x[n] \rightarrow n = \dots, -3, -2, -1, 0, 1, 2, 3, \dots$

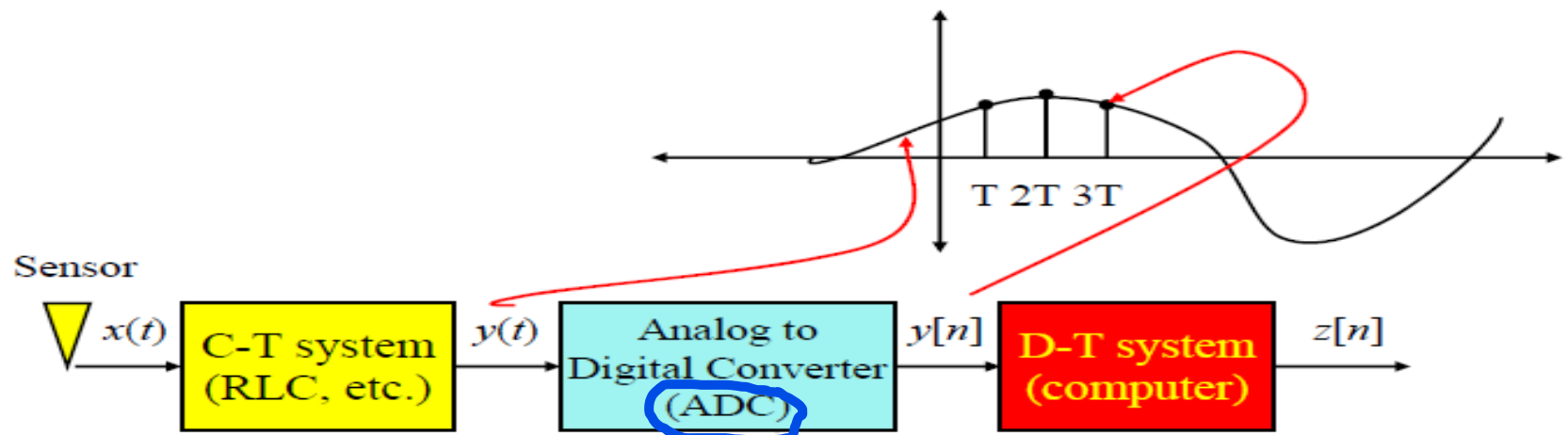
Brackets indicate discrete-time signal. Recall... we used parentheses to indicate a C-T signal.

A stem plot emphasizes that the signal does not exist in-between integer n values



3.1 Introduction

D-T systems allow us to process information in much more amazing ways than C-T systems!



“sampling” is how we typically get D-T signals

In this case the D-T signal $y[n]$ is related to the C-T signal $y(t)$ by:

$$y[n] = y(t) \big|_{t=nT} = y(nT)$$

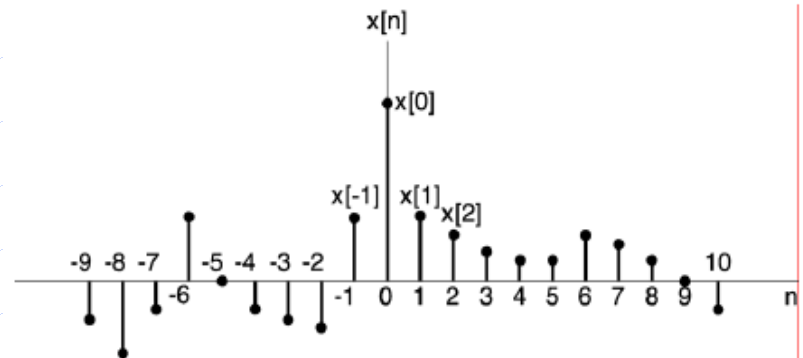
T is “sampling interval”

3.1 Introduction

- Discrete-time signal is basically a sequence of numbers. They may also arise as a result of sampling CT time signals.
- Systems whose inputs and outputs are DT signals are called digital system.
- $x[n]$, n —integer, time varies discretely

Examples of DT signals in nature:

- ✓ DNA base sequence
- ✓ Population of the n th generation of certain species



3.1 Introduction

- A function, e.g. $\sin(t)$ in continuous-time or $\sin(2\pi n/10)$ in discrete-time, useful in *analysis*
- A sequence of numbers, e.g. $\{1,2,3,2,1\}$ which is a sampled triangle function, useful in *simulation*

- A piecewise representation, e.g.

$$u(t) = \begin{cases} 1 & \text{for } t > 0 \\ \frac{1}{2} & \text{for } t = 0 \\ 0 & \text{for } t < 0 \end{cases}$$

$$u[n] = \begin{cases} 1 & \text{for } n \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Size of a discrete-time signal

Power and Energy of Signals

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

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

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

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

3.2 Useful Signal Operations

Three possible *time* transformations:

- Time Shifting
- Time Scaling
- Time Reversal

3.2 Useful Signal Operations

Time Shift

Delay or shift by integer k :

Definition: $y[n] = x[n - k]$

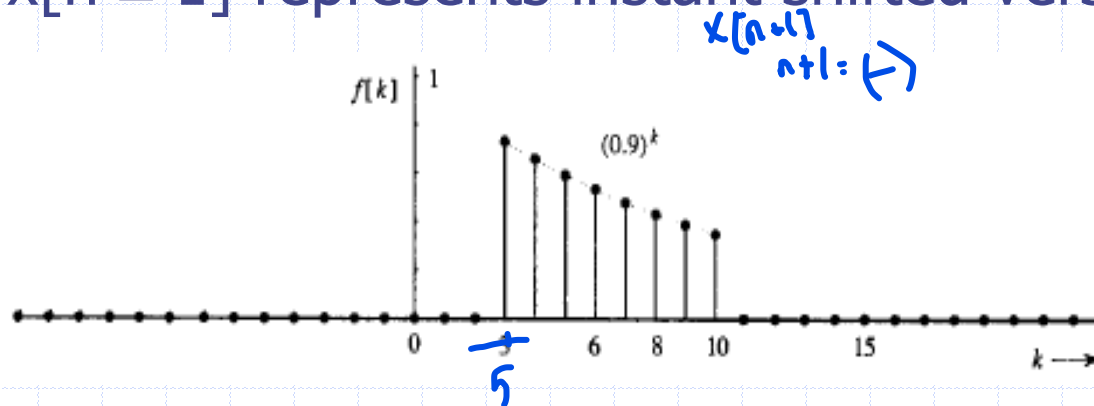
Interpretation:

- $k \geq 0 \rightarrow$ graph of $x[n]$ shifted by k units to the right
- $k < 0 \rightarrow$ graph of $x[n]$ shifted by k units to the left

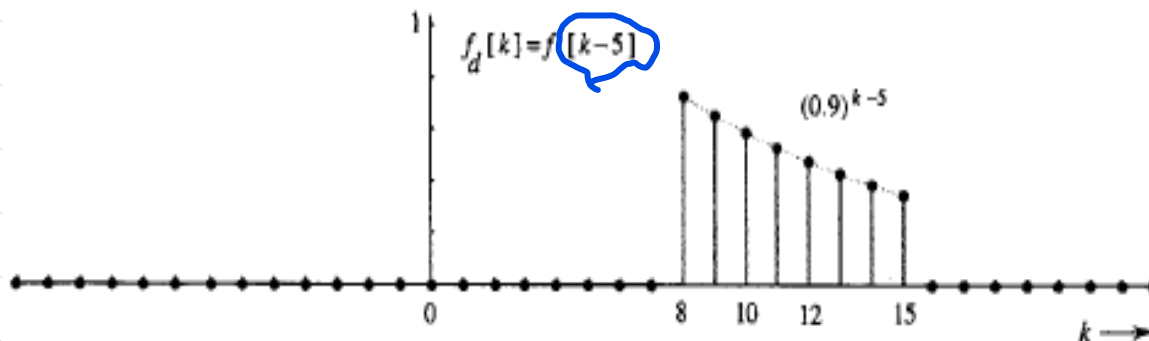
3.2 Useful Signal Operations

Time Shift

Signal $x[n \pm 1]$ represents instant shifted version of $x[n]$



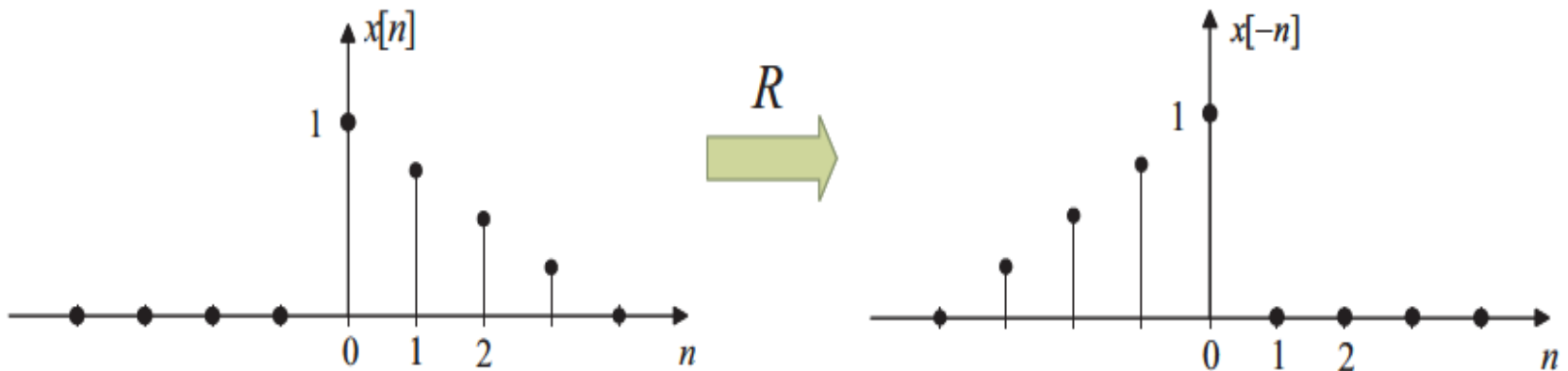
Find $f[k-5]$



3.2 Useful Signal Operations

Time- Reversal (Flip)

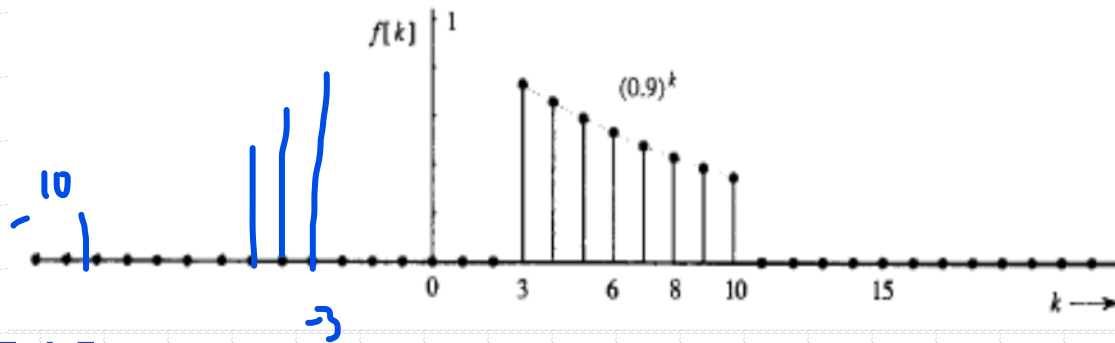
Graphical interpretation: mirror image about origin



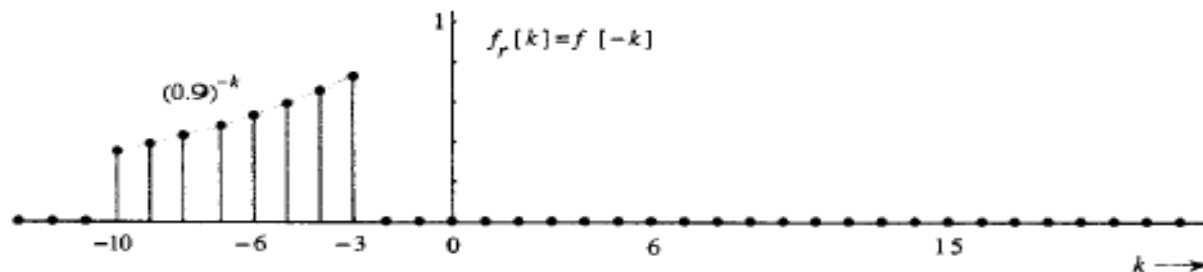
3.2 Useful Signal Operations

Time- Reversal (Flip)

Signal $x[-n]$ represents flip version of $x[n]$



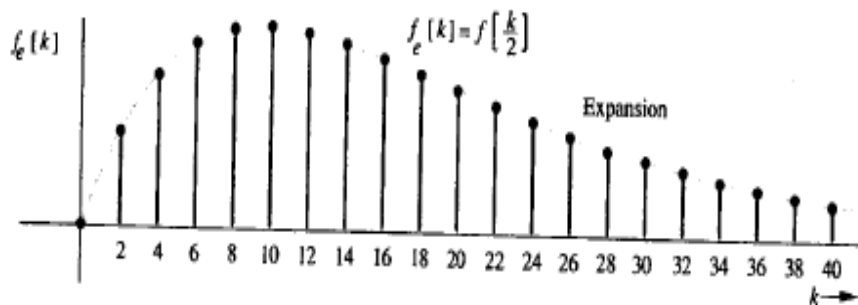
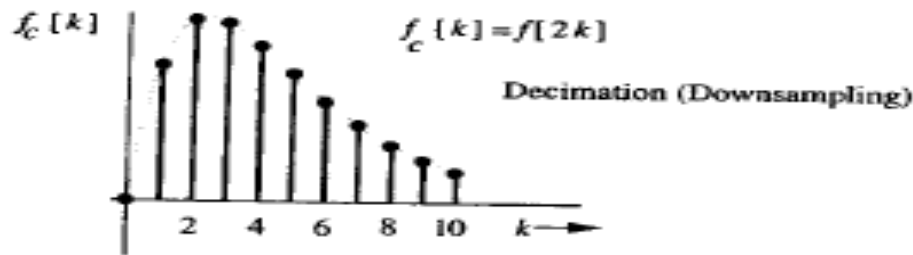
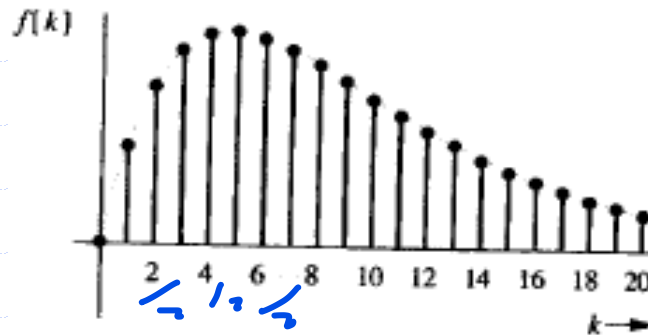
Find $f[-k]$



3.2 Useful Signal Operations

Time-scale

Find $f[2k]$, $f[k/2]$



① بزیر وقت بنالاید
Sample
② $2 \leftarrow 2.5 = k/2$

3.3 Some Useful Discrete-time Signal Models

لومند
↑ پیدا صد الاصل

Combined Operations

Ex; $x[n] = \{-3, 1, 2, -2, 3, -1\}$

Find a) $x[2-n]$

b) $x[3n-4]$

-1 $x(3)$ -1

Solution

a) $x[2-n] = \{-1, 3, -2, 2, 1, -3\}$

b) $x[3n-4] = \{0, 1, 3\}$

3.3 Some Useful Discrete-time Signal Models

Much of what we learned about C-T signals carries over to D-T signals

Discrete-Time Impulse Function $\delta[n]$

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

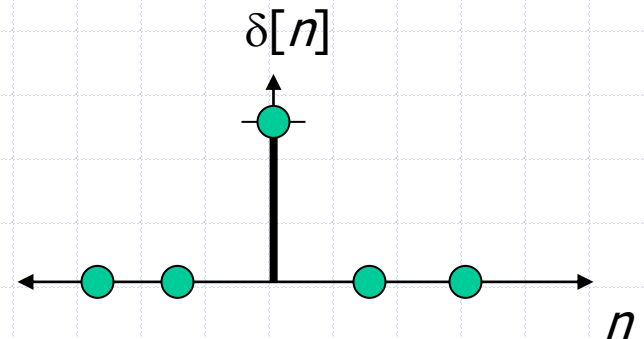
$$\delta[n] = u[n] - u[n-1]$$

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

$$x[n]\delta[n] = x[0]\delta[n]$$

$$x[n]\delta[n-k] = x[k]\delta[n-k]$$

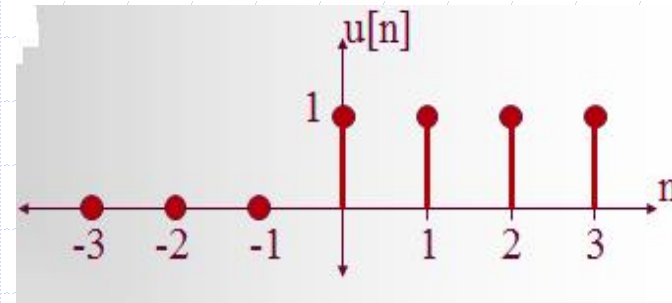
$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]$$



3.3 Some Useful Discrete-time Signal Models

Discrete-Time Unit Step Function $u[n]$

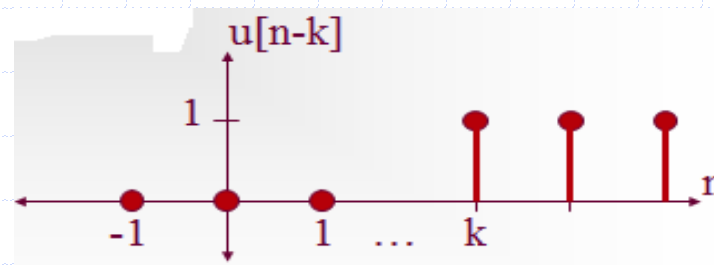
$$u[n] = \begin{cases} 0 & n < 0 \\ 1 & n \geq 0 \end{cases}$$



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

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

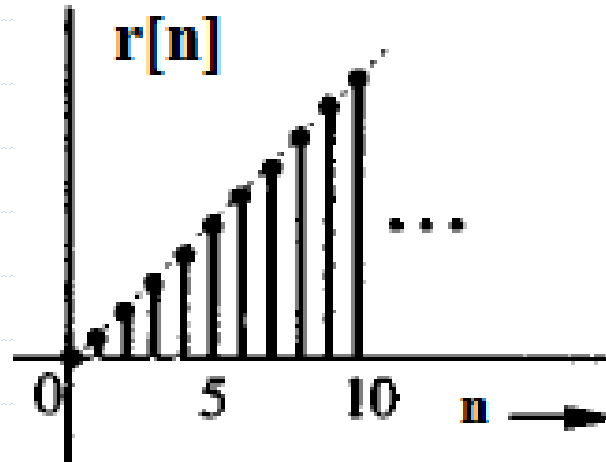
$$u[n-k] = \begin{cases} 1 & , n \geq k \\ 0 & , n < k \end{cases}$$



3.3 Some Useful Discrete-time Signal Models

Discrete-Time Unit ramp Function $r[n]$

$$r[n] = \begin{cases} n & , n \geq 0 \\ 0 & , n < 0 \end{cases}$$



3.3 Some Useful Discrete-time Signal Models

D-T Sinusoids

$$X[n] = A \cos(\Omega n + \theta)$$

Use “upper case omega” for frequency of D-T sinusoids

What is the unit for Ω ?

Anything inside a sine, cosine, or tangent must be in radians... NOT in degrees!!

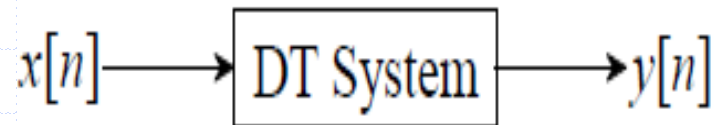
$\Omega n + \theta$ must be in radians $\Rightarrow \Omega n$ in radians

Ω is “how many radians jump for each sample”

Ω is in radians/sample

3.4 Classification of DT Systems

- **Linear Systems**
- **Time-invariance Systems**
- **Causal Systems**
- **Memory Systems**
- **Stable Systems**



Linear Systems:

A (DT) system is linear if it has the superposition property:

If $x_1[n] \rightarrow y_1[n]$ and $x_2[n] \rightarrow y_2[n]$

then $ax_1[n] + bx_2[n] \rightarrow ay_1[n] + by_2[n]$

- Example: Are the following system linear?

$$y[n] = nx[n]$$

3.4 Classification of DT Systems

$$\underline{\underline{y[n] = n \cdot x[n]}}$$

$$\underline{y_1[n] = n \cdot x_1[n]}$$

$$\underline{y_2[n] = n \cdot x_2[n]}$$

$$\boxed{x[n] = a x_1[n] + b x_2[n]} \xrightarrow{?} y[n] = a y_1[n] + b y_2[n]$$

$$y[n] = n \cdot \left[\underline{a x_1[n] + b x_2[n]} \right]$$

$$= a \cdot \underbrace{n \cdot x_1[n]}_{y_1[n]} + b \cdot \underbrace{n \cdot x_2[n]}_{y_2[n]}$$

$$= a y_1[n] + y_2[n]$$

System is linear
because
superposition holds

3.4 Classification of DT Systems

Time-Invariance

A system is time-invariant if a delay (or a time-shift) in the input signal causes the same amount of delay (or time-shift) in the output signal

If $x[n] \rightarrow y[n]$

then $x[n - n_0] \rightarrow y[n - n_0]$

$$x[n] = x_1[n - n_0] \rightarrow y[n] = y_1[n - n_0]$$

Ex. Check if the following system is time-invariant:

$$y[n] = nx[n]$$

3.4 Classification of DT Systems

$$\underline{y[n] = n \cdot x[n]}$$

$$x[n] \Rightarrow y[n] = n \cdot x[n]$$

$$x_1[n] = x[n - n_0] \stackrel{?}{\Rightarrow} y_1[n] = y[n - n_0]$$

$$y_1[n] = n \cdot x_1[n] = n \cdot x[n - n_0]$$

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

System is Time Varying

3.4 Classification of DT Systems

Causal System

A system is causal if the output does not anticipate future values of the input, i.e., if the output at any time depends only on values of the input up to that time.

A system $x[n] \rightarrow y[n]$ is causal if

When	$x_1[n] \rightarrow y_1[n]$	$x_2[n] \rightarrow y_2[n]$
And	$x_1[n] = x_2[n]$	for all $n \leq n_0$
Then	$y_1[n] = y_2[n]$	for all $n \leq n_0$

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

Ex. Check if the following system is Causal: $y[n] = nx[n]$

System is causal because it does not depend on future

$x[n-1]$
↓
Non-

3.4 Classification of DT Systems

- Memoryless (or static) Systems: System output $y[n]$ depends only on the input at instant n , i.e. $y[n]$ is a function of $x[n]$.
- Memory (or dynamic) Systems: System output $y[n]$ depends on input at past or future of the instant n

Ex. Check if the following systems are with memory :

- ◆ i. $y[n] = nx[n]$ ii. $y[n] = 1/2(x[n-1] + x[n])$
- i. Above system is memoryless because is instantaneous
- ii. System is with memory

3.5 DT System Equations:

Difference Equations:

- We saw that Differential Equations model C-T systems...
- D-T systems are “modeled” by Difference Equations.

A general Nth order Difference Equations looks like this:

$$y[n] + a_1 y[n-1] + \dots + a_N y[n-N] = b_0 x[n] + b_1 x[n-1] + \dots + b_M x[n-M]$$

Most “Advanced”
Output Sample

Least “Advanced”
Output Sample

The difference between these two index values is the “order” of the difference eq. Here we have: $n - (n - N) = N$

3.5 DT System Equations:

Difference equations can be written in two forms:

- The first form uses delay $y[n-1]$, $y[n-2]$, $x[n-1]$,.....

$$y[n] + a_1 y[n-1] + \dots + a_N y[n-N] = b_0 x[n] + \dots + b_M x[n-M]$$

Order is $\text{Max}(N, M)$

- The 2nd form uses advance $y[n+1]$, $y[n+2]$, $x[n+1]$,....

$$y[n+N] + a_1 y[n+N-1] + \dots + a_N y[n] = b_{N-M} x[n+m] + \dots + b_N x[n]$$

Order is $\text{Max}(N, M)$

3.5 DT System Equations:

- Sometimes Difference Equations will be presented as unit advances rather than delays

$$y[n+2] - 5 y[n+1] + 6 y[n] = 3 x[n+1] + 5 x[n]$$

- One can make a substitution that reindexes the equation so that it is in terms of delays

Substitute n with $n - 2$ to yield

$$y[n] - 5 y[n-1] + 6 y[n-2] = 3 x[n-1] + 5 x[n-2]$$

3.5 DT System Equations:

Solving Difference Equations

Although Difference Equations are quite different from Differential Equations, the methods for solving them are remarkably similar.

Here we'll look at a numerical way to solve Difference Equations. This method is called Recursion...and it is actually used to implement (or build) many D-T systems, which is the main advantage of the recursive method.

The disadvantage of the recursive method is that it doesn't provide a so-called "closed-form" solution...in other words, you don't get an equation that describes the output (you get a finite-duration sequence of numbers that shows part of the output).

3.5 DT System Equations:

Solution by Recursion

We can re-write any linear, constant-coefficient difference equation in “recursive form”. Here is the form we’ve already seen for an N th order difference:

$$y[n] + a_1 y[n-1] + \dots + a_N y[n-N] = b_0 x[n] + b_1 x[n-1] + \dots + b_M x[n-M]$$

Re-Write As:

$$y[n] + \sum_{i=1}^N a_i y[n-i] = \sum_{i=0}^M b_i x[n-i]$$

3.5 DT System Equations:

Now...isolating the $y[n]$ term gives the “Recursive Form”:

$$y[n] = -\sum_{i=1}^N a_i y[n-i] + \sum_{i=0}^M b_i x[n-i]$$

“current”
Output value to
be computed

Some “past” output
values, with values
already known

current & past input
values already
“received”

The key to Recursive
Form is that you have
the current output $y[n]$
in terms of past
outputs $y[n-i]$

3.5 DT System Equations:

Note: sometimes it is necessary to re-index a difference equation using $n+k \rightarrow n$ to get this form...as shown below.

Here is a slightly different form...but it is still a difference equation:

$$y[n+2]-1.5y[n+1]+y[n]=2x[n]$$

If you isolate $y[n]$ here you will get the current output value in terms of future output values (Try It!)...We don't want that!

So...in general we start with the “Most Advanced” output sample...here it is $y[n+2]$...and re-index it to get only n (of course we also have to re-index everything else in the equation to maintain an equation):

3.5 DT System Equations:

So here we need to subtract 2 from each sample argument:

$$y[n]-1.5y[n-1]+y[n-2]=2x[n-2]$$

Now we can put this into recursive form as before.

Ex: Solve this difference equation recursively

$$y[n]-1.5y[n-1]+y[n-2]=2x[n-2]$$

For $x[n]=u[n]$ unit step

And ICs of: $\begin{cases} y[-2] = 2 \\ y[-1] = 1 \end{cases}$

Note: You need N “past” values as IC’s to solve an N^{th} order Difference Equation

3.5 DT System Equations:

Recursive Form:

$$y[n] = 1.5y[n-1] - y[n-2] + 2x[n-2]$$

n	$x[n]=u[n]$	$y[n]$
-2	0	2
-1	0	1
0	1	$1.5 \cdot 1 - 2 + 2 \cdot 0 = -0.5$
1	1	$1.5 \cdot (-0.5) - 1 + 2 \cdot 0 = -1.75$
2	1	-0.125
3	1	3.563

1st: Fill in Input (Unit Step here)

2nd: Put IC's Here

3rd: Compute $n=0$ Output
 $y[0] = 1.5y[-1] - y[-2] + 2x[-2]$

4th: Compute $n=1$ Output
 $y[1] = 1.5y[0] - y[-1] + 2x[-1]$

Etc.

3.5 DT System Equations:

Ex 3.9 P. 273

$$y[n+2]-y[n+1]+0.24y[n]=x[n+2]-2x[n+1]$$

$y[-1]=2$, $y[-2]=1$, and causal input $x[n]=n$

Solution

$$y[n]=y[n-1]-0.24y[n-2]+x[n]-2x[n-1]$$

$$y[0]=y[-1]-0.24y[-2]+x[0]-2x[-1]=2-0.24=1.76$$

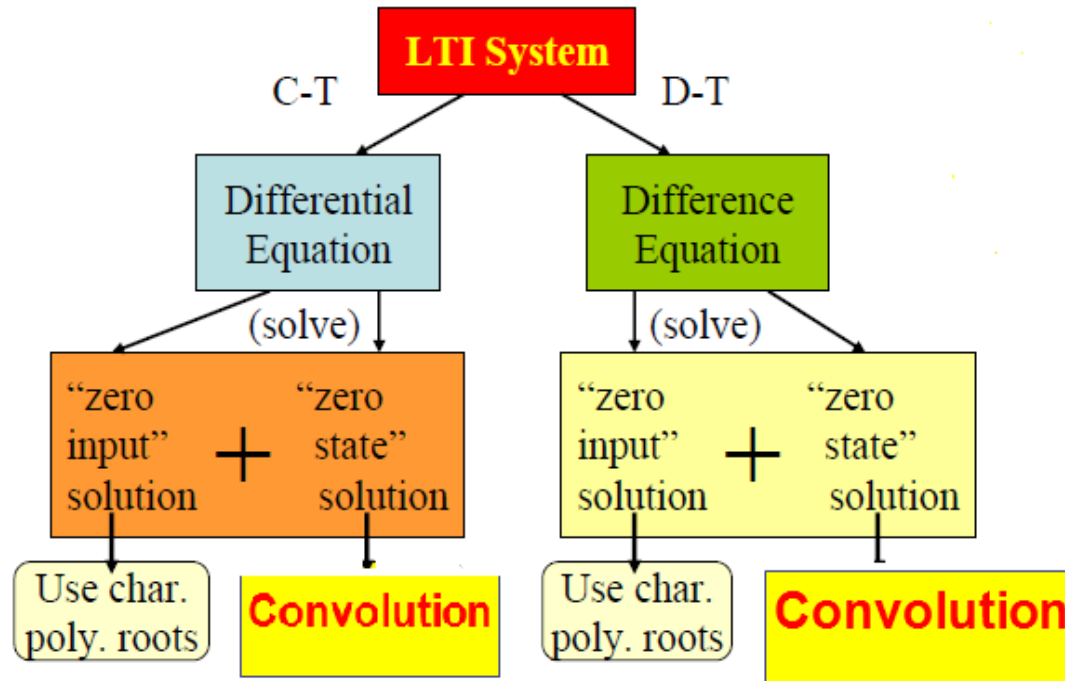
$$y[1]=y[0]-0.24y[-1]+x[1]-2x[0]=1.76-0.24(2)+1-0=2.28$$

:

:

Convolution

Our Interest: Finding the output of **LTI** systems (D-T & C-T cases)



Our focus in this chapter will be on finding the zero-state solution

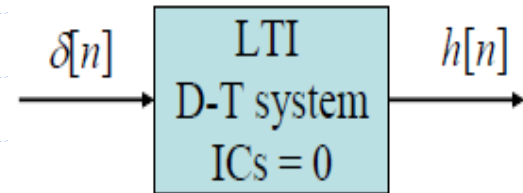
3.8 System Response to External Input: (Zero State Response)

Convolution:

For discrete case: $h[n] = H[\delta[n]]$

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

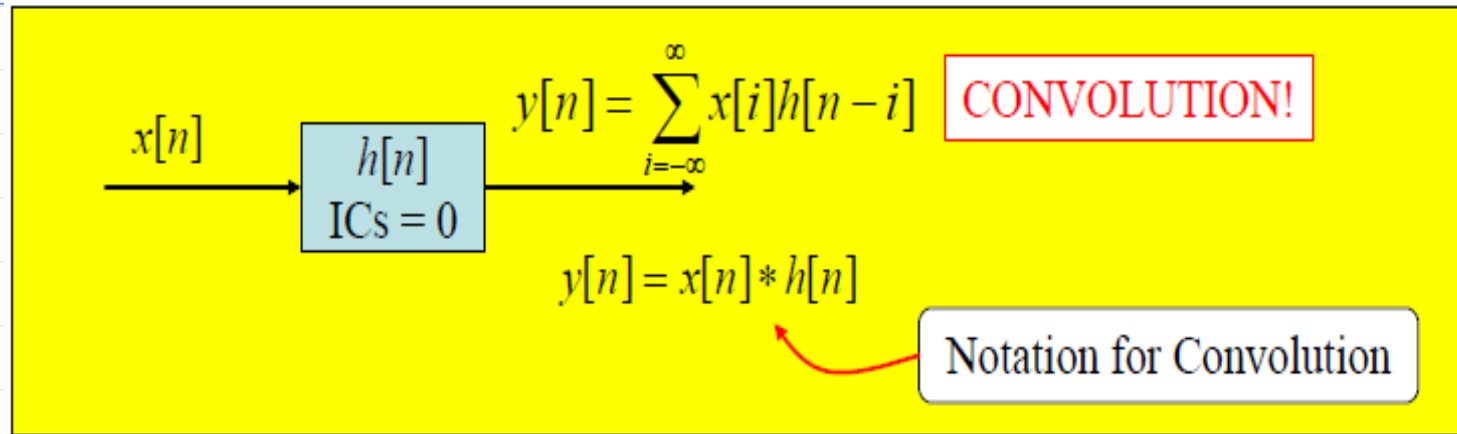
$$y[n] = x[n] * h[n]$$



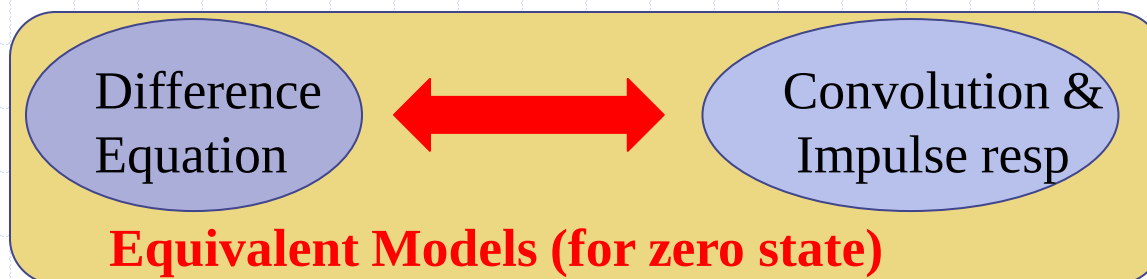
$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

- Notice that this is not multiplication of $x[n]$ and $h[n]$.
- Visualizing meaning of convolution:
 - Flip $h[k]$
 - By shifting $h[k]$ for all possible values of n , pass it through $x[n]$.

3.8 System Response to External Input: (Zero State Response)



For a LTI D-T system in zero state we no longer need the difference equation model...-Instead we need the impulse response $h[n]$ & convolution



3.8 System Response to External Input: (Zero State Response)

Properties of DT Convolution:

Same as CT Convolution

Ex: 3.13 P.289



$x[n] = (0.8)^n u[n]$, $h[n] = (0.3)^n u[n]$ Find $y[n]$

Solution

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

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

$$x[k] = (0.8)^k u[k]$$

$$h[n-k] = (0.3)^{n-k} u[n-k]$$

$$u[k]u[n-k] = 1 \\ = 0$$

$$0 < k < n \\ k < 0$$

3.8 System Response to External Input: (Zero State Response)

$$y[n] = \sum_{k=0}^n (0.8)^k (0.3)^{n-k} = (0.3)^n \sum_{k=0}^n \left(\frac{0.8}{0.3}\right)^k \quad n \geq 0$$

From Section B7-4 P49 $\sum_{k=m}^n r^k = \frac{r^{n+1} - r^m}{r-1} \quad r \neq 1$

OR

“Geometric Sum”

$$\sum_{i=0}^{N-1} r^i = \frac{1 - r^N}{1 - r}, \quad r \neq 1$$

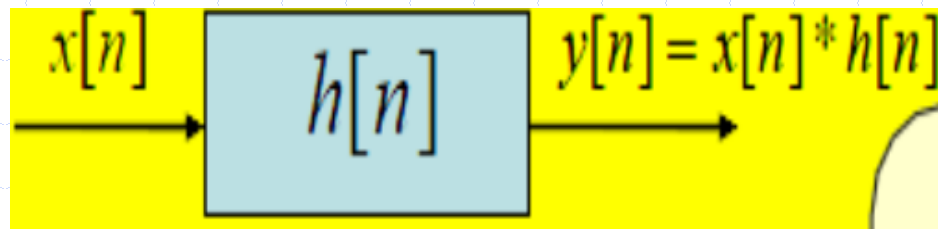
$$y[n] = (0.3)^n \frac{(0.8)^{n+1} - (0.3)^{n+1}}{(0.3)^n (0.8 - 0.3)} u[n]$$

$$y[n] = 2[(0.8)^{n+1} - (0.3)^{n+1}]u[n]$$

3.8 System Response to External Input: (Zero State Response)

Example

Determine $y(n)$ as the convolution of $h(n)$ and $x(n)$, where



$$x(n) = \left(\frac{1}{4}\right)^n u(n - 3)$$

$$h(n) = \left(\frac{1}{3}\right)^n u(n - 5)$$

3.8 System Response to External Input: (Zero State Response)

$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} \left(\frac{1}{4}\right)^k u(k-3) \left(\frac{1}{3}\right)^{n-k} u(n-k-5)$$

$$u(k-3) u(n-k-5) = \begin{cases} 1 & 3 \leq k \leq n-5 \\ 0 & \text{otherwise} \end{cases}$$

$$\therefore n-5 \geq 3 \quad \therefore n \geq 8$$

$$\boxed{y(n) = 0 \quad \text{for } n < 8}$$

3.8 System Response to External Input: (Zero State Response)

$y[n]$ for $n > 8$:

$$y[n] = \sum_{k=3}^{n-5} \left(\frac{1}{4}\right)^k \left(\frac{1}{3}\right)^{n-k} = \left(\frac{1}{3}\right)^n \sum_{k=3}^{n-5} \left(\frac{3}{4}\right)^k$$

From Section B7-4 P49 $\sum_{k=m}^n r^k = \frac{r^{n+1} - r^m}{r-1} \quad r \neq 1$

$$y[n] = \left(\frac{27}{16}\right) \left(\frac{1}{3}\right)^n \left(1 - \left(\frac{3}{4}\right)^{n-7}\right) \text{ for } n > 8$$

3.8 System Response to External Input: (Zero State Response)

$$y[n] = \begin{pmatrix} 0 & n < 8 \\ (\frac{27}{16})(\frac{1}{3})^n (1 - (\frac{3}{4})^{n-7}) & n \geq 8 \end{pmatrix}$$

$$y[n] = (\frac{27}{16})(\frac{1}{3})^n (1 - (\frac{3}{4})^{n-7}) \quad u[n - 8]$$

3.8 System Response to External Input: (Zero State Response)

Graphical procedure for the convolution:

Step 1: Write both as functions of k : $x[k]$ & $h[k]$

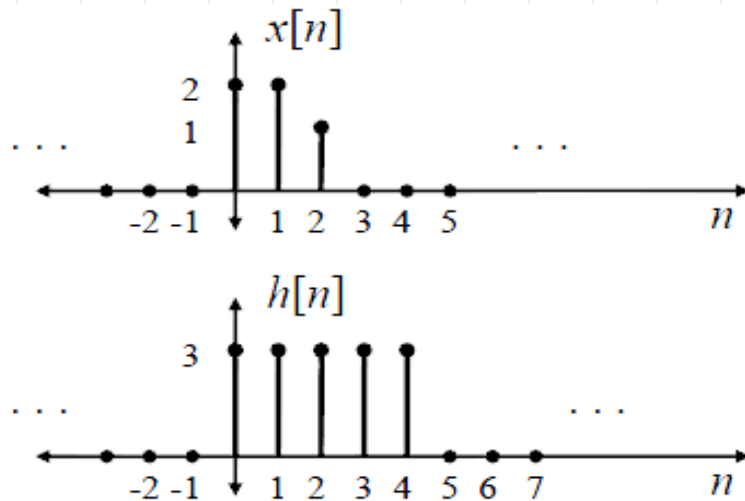
Step 2: Flip $h[k]$ to get $h[-k]$

Step 3: For each output index n value of interest, shift by n to get $h[n-k]$ (Note: positive n gives right shift!!!!)

Step 4: Form product $x[k]h[n-k]$ and sum its elements to get the number $y[n]$

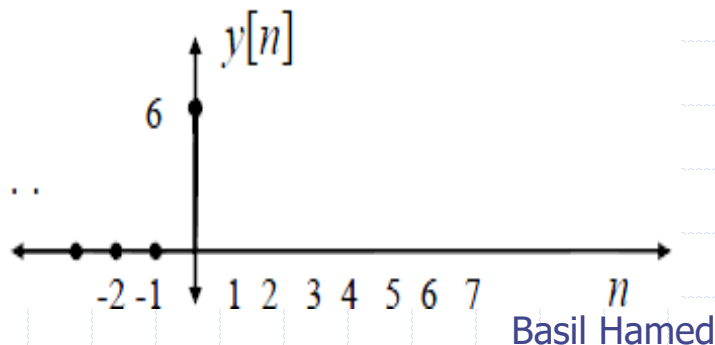
3.8 System Response to External Input: (Zero State Response)

Example of Graphical Convolution



Find $y[n] = x[n] * h[n]$
for all integer values of n

So..what we know so far is that: $y[n]$ = starts ($0 + 0 = 0$) ends ($2 + 4 = 6$)



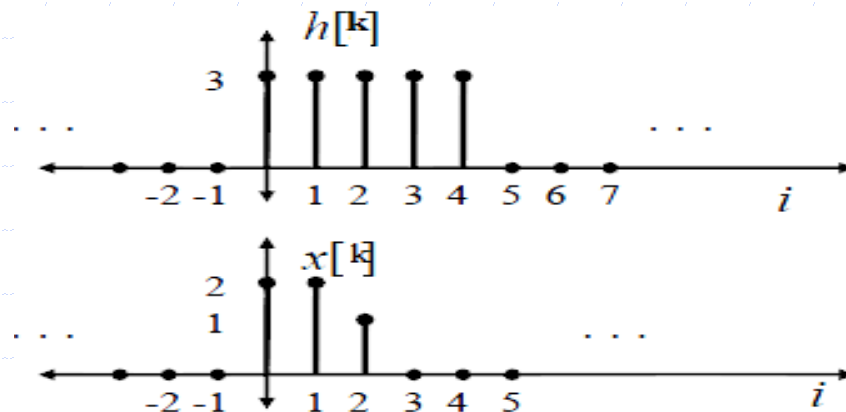
$y[n]$ starts at 0 ends at 6

3.8 System Response to External Input: (Zero State Response)

Solution

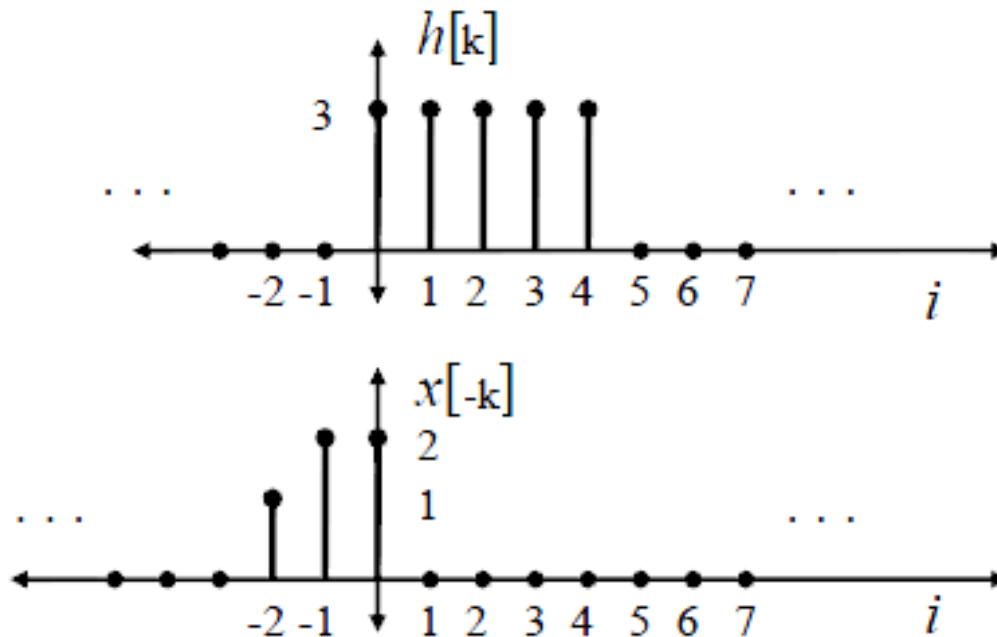
- For this problem I choose to flip $x[n]$
- My personal preference is to flip the shorter signal although I sometimes don't follow that “rule”...only through lots of practice can you learn how to best choose which one to flip.

Step 1: Write both as functions of k : $x[k]$ & $h[k]$



3.8 System Response to External Input: (Zero State Response)

Step 2: Flip $x[k]$ to get $x[-k]$



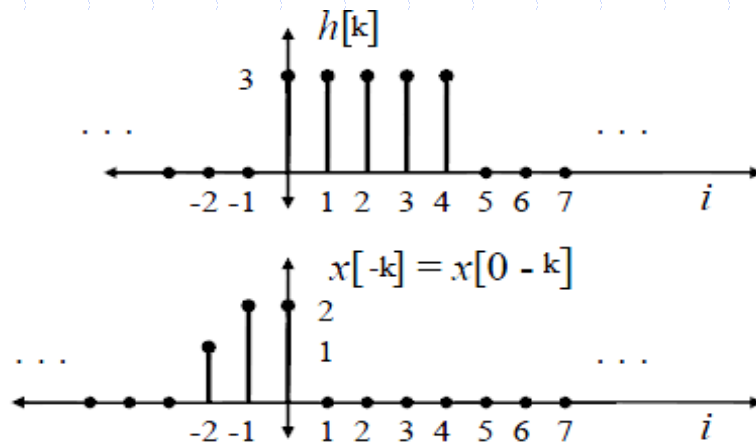
“Commutativity” says
we can flip either $x[k]$
or $h[k]$ and get the same
answer...

Here I flipped $x[k]$

3.8 System Response to External Input: (Zero State Response)

We want a solution for $n = \dots -2, -1, 0, 1, 2, \dots$ so must do Steps 3&4 for all n . But...let's first do: **Steps 3&4 for $n=0$** and then proceed from there.

Step 3: For $n=0$, shift by n to get $x[n-k]$



For $n=0$ case there is no shift!
 $x[0-k] = x[-k]$

Step 4: For $n=0$, Form the product $x[k]h[n-k]$ and sum its elements to give $y[n]$

Sum over $k \Rightarrow$

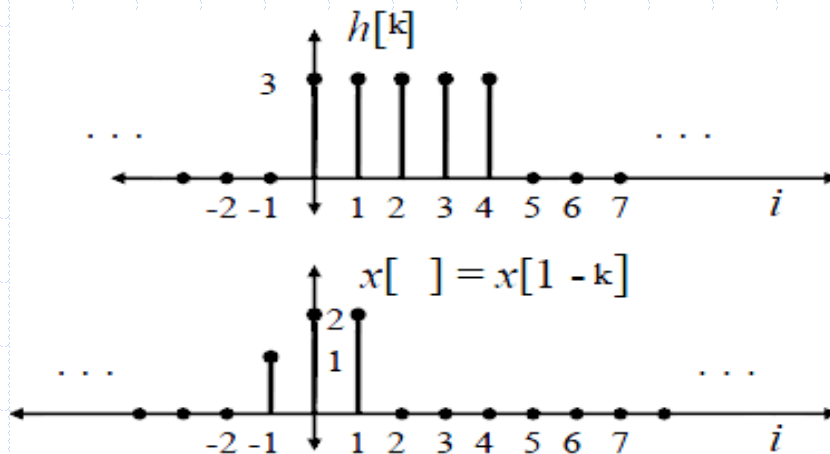
$$\underline{y[0]=6}$$

Hamed

3.8 System Response to External Input: (Zero State Response)

Steps 3&4 for $n=1$

Step 3: For $n=1$, shift by n to get $x[n-k]$



Step 4: For $n=1$, Form the product $x[k]h[n-k]$ and sum its elements to give $y[n]$

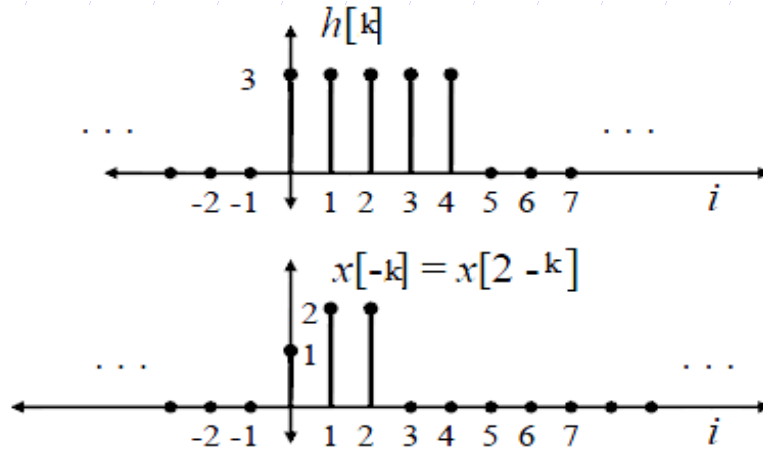
Sum over $k \Rightarrow$

$$y[1]=6+6=12$$

3.8 System Response to External Input: (Zero State Response)

Steps 3&4 for $n=2$

Step 3: For $n=2$, shift by n to get $x[n-k]$



Step 4: For $n=2$, Form the product $x[k]h[n-k]$ and sum its elements to give $y[n]$

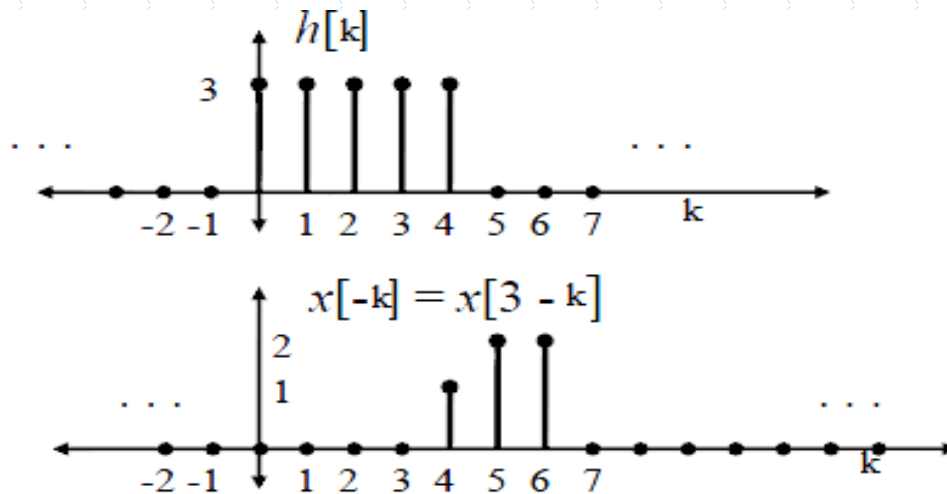
Sum over $k \Rightarrow$

$$y[2]=3+6+6=15$$

3.8 System Response to External Input: (Zero State Response)

Steps 3&4 for $n=6$

Step 3: For $n=6$, shift by n to get $x[n-k]$



Step 4: For $n=6$, Form the product $x[k]h[n-k]$ and sum its elements to give $y[n]$

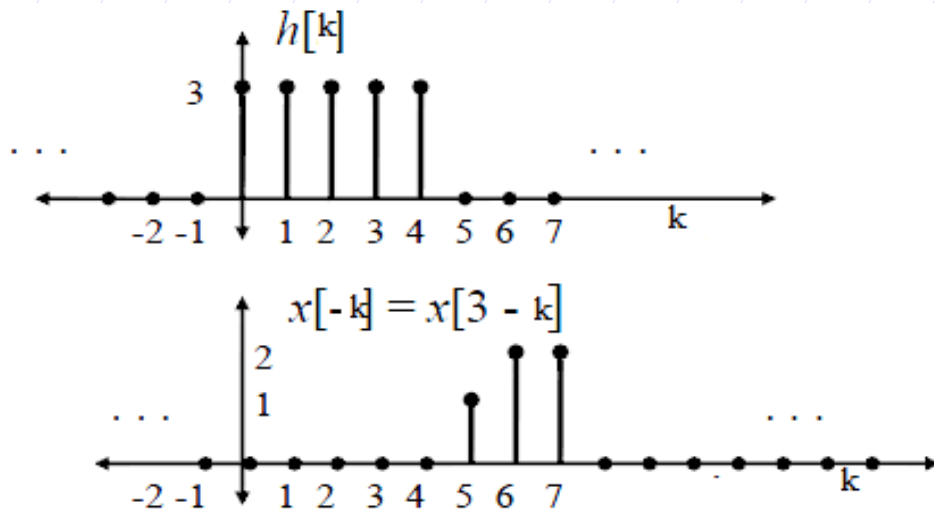
Sum over $k \Rightarrow$

$$y[6]=3$$

3.8 System Response to External Input: (Zero State Response)

Steps 3&4 for all $n > 6$

Step 3: For $n > 6$, shift by n to get $x[n-k]$



Step 4: For $n > 6$, Form the product $x[k]h[n-k]$ and sum its elements to give $y[n]$

Sum over $k \Rightarrow$

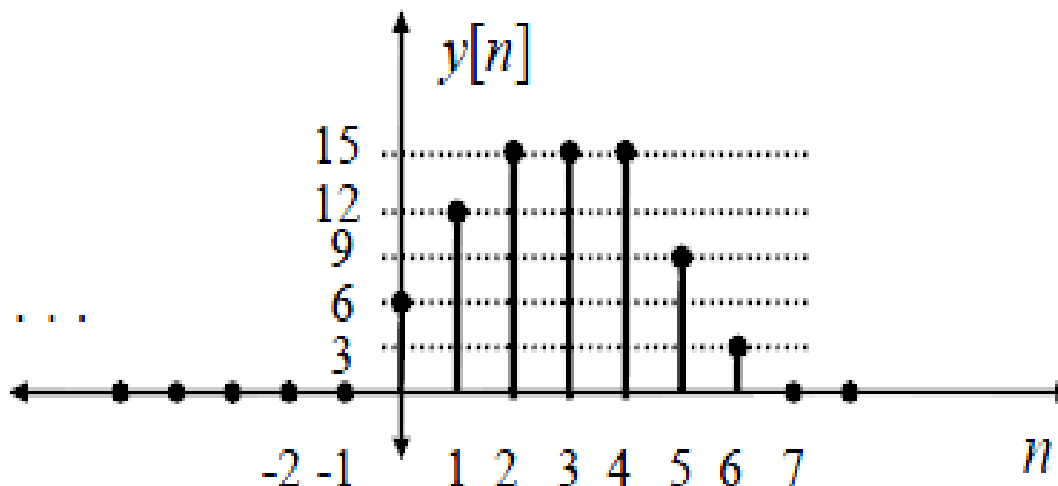
$$y[n] = 0 \quad n > 6$$

3.8 System Response to External Input: (Zero State Response)

So...now we know the values of $y[n]$ for all values of n

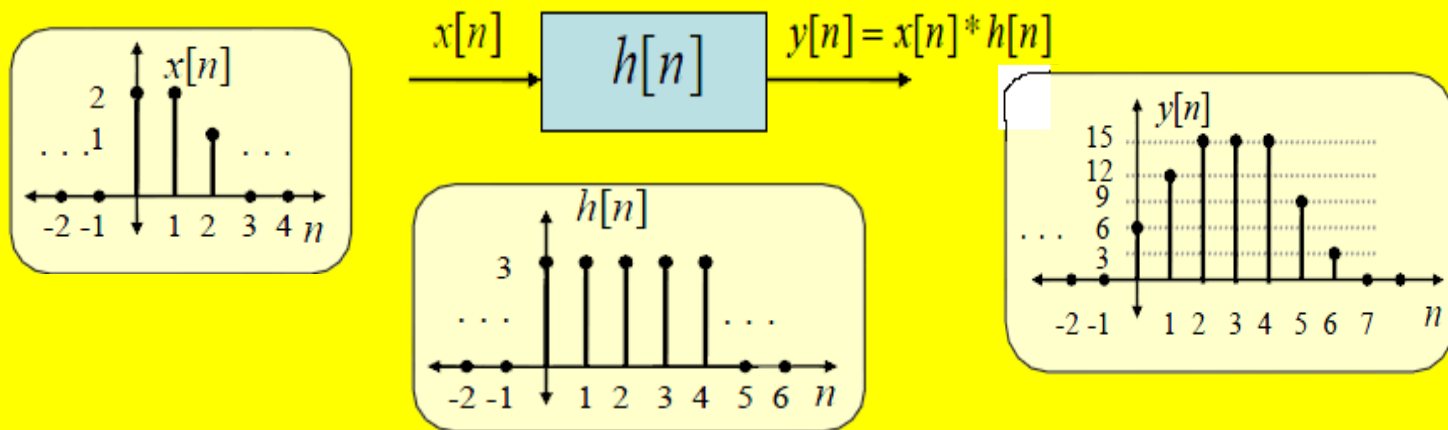
We just need to put it all together as a function...

Here it is easiest to just plot it...you could also list it as a table



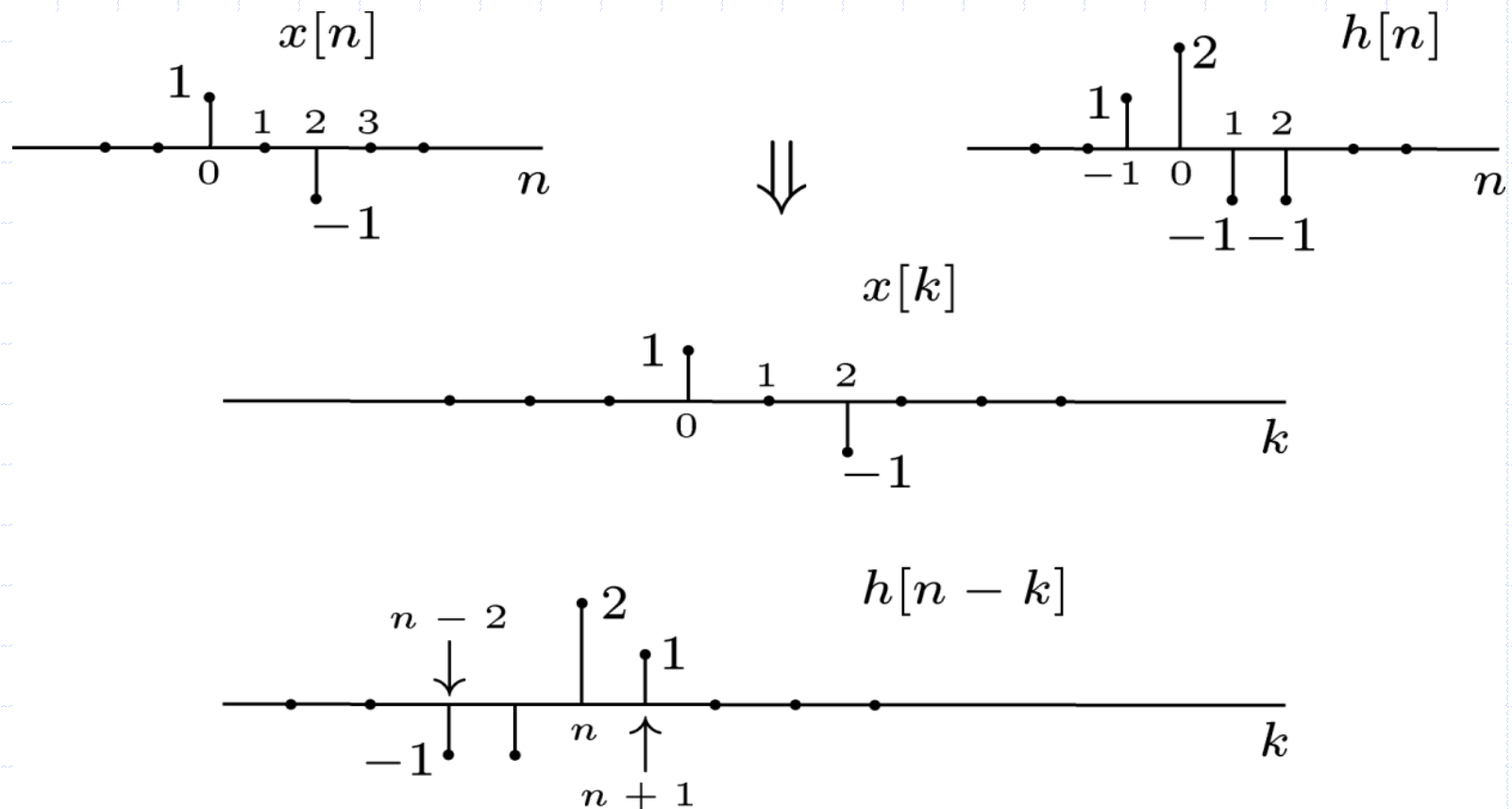
3.8 System Response to External Input: (Zero State Response)

BIG PICTURE: So...what we have just done is found the zero-state output of a system having an impulse response given by this $h[n]$ when the input is given by this $x[n]$:

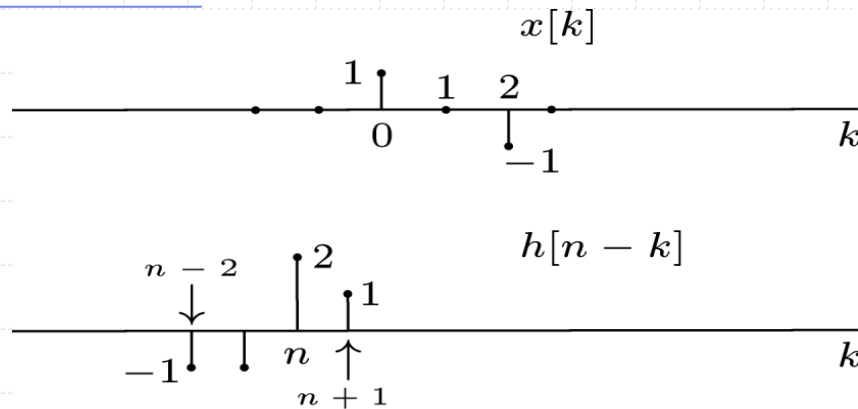


3.8 System Response to External Input: (Zero State Response)

EX: given $x[n]$, and $h[n]$, find $y[n]$



3.8 System Response to External Input: (Zero State Response)



$$y[n] = 0 \quad \text{for } n < -1$$

$$y[-1] = 1 \times 1 = 1$$

$$y[0] = 0 \times 1 + 1 \times 2 = 2$$

$$y[1] = (-1) \times 1 + 0 \times 2 + 1 \times (-1) = -2$$

$$y[2] = (-1) \times 2 + 0 \times (-1) + 1 \times (-1) = -3$$

$$y[3] = (-1) \times (-1) + 0 \times (-1) = 1$$

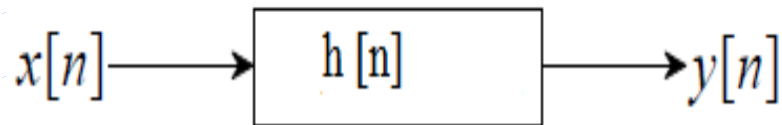
$$y[4] = (-1) \times (-1) = 1$$

$$y[n] = 0 \quad \text{for } n > 4$$

$$y[n] = \{1, 2, -2, -3, 1, 1\}$$

3.8 System Response to External Input: (Zero State Response)

Exercises : given the following systems Find $y[n]$



i. $x[n]=\{-2,-1,0,1,2\}$, $h[n]=\{-1,0,1,2\}$

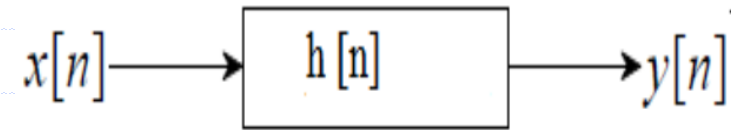
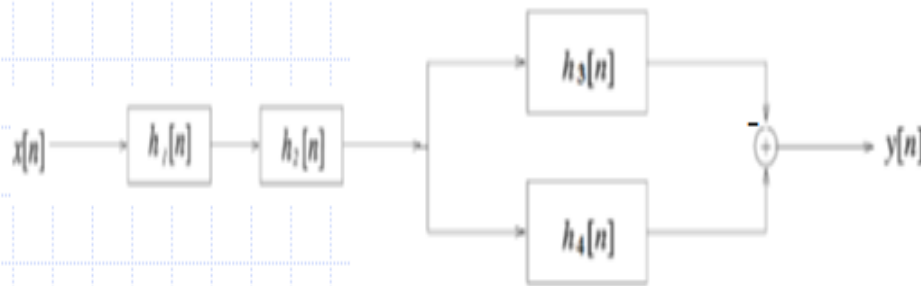
ii. $x[n]=\{-1,3,-1,-2\}$, $h[n]=\{-2,2,0,-1,1\}$

Solution:

i. $y[n]=\{2,1,-2,-6,-4,1,4,4\}$

ii. $y[n]=x[n]*h[n]=\{2,-8,8,3,-8,4,1,-2\}$

3.8-2 Interconnected Systems



Example Find $h[n]$ given:

$$h_1[n] = \delta[n] - a\delta[n-1], \quad h_2[n] = a^n u[n]$$

$$h_3[n] = [n-1]u[n], \quad h_4[n] = \delta[n] + nu[n-1] + \delta[n-2]$$

Solution:

$$h_4[n] - h_3[n] = \delta[n] + nu[n-1] + \delta[n-2] - [n-1]u[n]$$

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

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

$$h_1[n] * h_2[n] = (\delta[n] - a\delta[n-1]) * a^n u[n]$$

3.8-2 Interconnected Systems

$$\begin{aligned} h_1[n] * h_2[n] &= a^n u[n] - a^{n+1} u[n] \uparrow_{n \rightarrow n-1} \\ &= a^n u[n] - a^n u[n-1] = a^n (u[n] - u[n-1]) \\ &= a^n \delta[n] = \delta[n] \end{aligned}$$

$\delta[n]$

$$h[n] = h_1[n] * h_2[n] * (h_4[n] - h_3[n])$$

$$h[n] = \delta[n] * (\delta[n] + \delta[n-2] + u[n])$$

$$h[n] = \delta[n] + \delta[n-2] + u[n]$$

Comparison of Discrete convolution and Difference Eq.

1. Difference Eq. require less computation than convolution
 2. Difference Eq. require less memory
 3. Convolutions describe only zero-state responses. ($IC=0$)
- Since difference Eq have many advantages over convolutions, we use mainly difference Eq. in studying LTI lumped systems.
 - For distributed system, we have no choice but to use convolution.
 - Convolution can be used to describe LTI distributed and lumped systems. Where as difference Eq describes only lumped systems.

3.10 System Stability

A discrete-time system is BIBO stable if every bounded input sequence $x[n]$ produces a bounded output sequence.

LTID with $h[n]$ is BIBO stable if:

$$S = \sum_{k=-\infty}^{\infty} |h[k]| \text{ is finite}$$

Ex Find if the following systems are stable;

i. $h_1[n] = a^n u[n]$

ii. $h_2[n] = \begin{cases} \left(\frac{1}{2}\right)^n & n \geq 0 \\ \left(\frac{1}{3}\right)^n & n < 0 \end{cases}$

iii. $h_3[n] = \begin{cases} 3^n & 0 \leq n \leq 100 \\ 0 & \text{otherwise} \end{cases}$

3.10 System Stability

I. $h_1 = \sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=0}^{\infty} |a^n| = \frac{1}{1-|a|}$ which is finite if $|a| < 1$, System is stable

II. $h_2 = \sum_{n=-\infty}^{\infty} |h[n]| = \sum_{n=-\infty}^{-1} \left(\frac{1}{3}\right)^n + \sum_{n=0}^{\infty} \left(\frac{1}{2}\right)^n$ is not bounded System is BIUBO unstable.

III. $h_3 = \sum_{n=0}^{100} (3)^n = \frac{1}{2} (3^{101} - 1) < \infty$ system is bounded
System is stable

3.10 System Stability

Example

$$y[n] = e^{x[n]}.$$

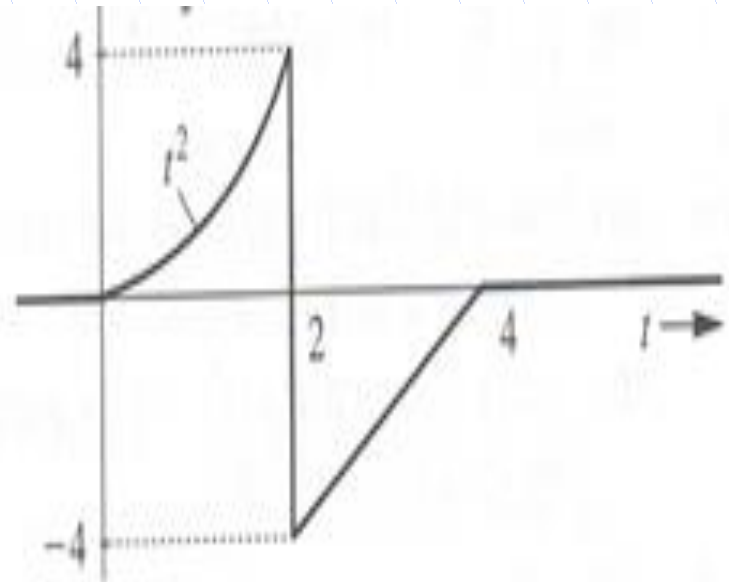
Let $x[n]$ an arbitrary signal bounded by B , i.e.,

$$|x[n]| < B \text{ or } -B < x[n] < B \text{ for all } n.$$

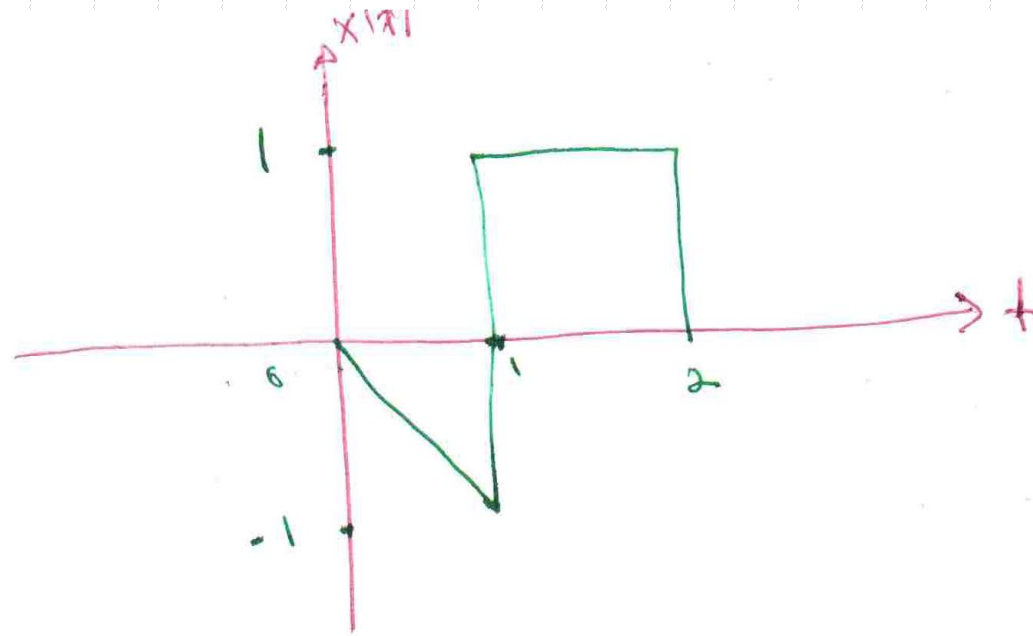
In that case $e^{-B} < e^{x[n]} < e^B$, i.e., it is guaranteed that if the input is bounded by a positive number B , the output is bounded by a positive number e^B . Thus, the system is stable.

Quiz

- i. Sketch the continuous-time signal;
$$x(t) = (t+1) u(t-1) - t u(t) - u(t-2)$$
- ii. Express the signal shown in terms of elementary functions



i.



ii.

$$t^2 [u(t) - u(t-2)] + (2t - 8) [u(t-2) - u(t-4)]$$

OR

$$2t(t-2) - 4u(t-2) - 2t(t-4) + t^2 [u(t) - u(t-2)]$$