Digital Signal Processing

Course Instructor
Dr. Ali J. Abboud

Lecture No. 4

Third Class
Department of Computer and Software Engineering

http://www.engineering.uodiyala.edu.iq/
https://www.facebook.com/Engineer.College1

Lecture Outline

- Analysis of DT-LTI System
- Difference Equations
- Recursive Systems
- Non-recursive Systems
- Describing Digital Signals with Impluse Function
- Describing Digital LTI Responses
  1) Impulse Response
  2) Step Response
- Discrete-Time Convolution
- Discrete-Time Circular Convolution
- Discrete-Time Correlation
- Deconvolution

Analysis of DT-LTI system

There are two basic methods for analyzing the behavior or response of a Linear system to a given input signal.

1. Method based on direct solution of input-output equation for the system.
2. Decomposition of the input signal into a sum of elementary signals (usually samples)
   - Elementary signals are selected so that the response of the system to each signal component is easily determined.

Difference Equations

- Difference eqs can be used to describe how a linear, time-invariant, causal digital system works.
- If present i/p & o/p ⟶ x[n] & y[n]

  then the preceding i/ps & o/ps

  x[n-1], x[n-2]….& y[n-1], y[n-2]…so on.
- Using these notations, the most general expression of the diff: eq: is

$$a_0y[n]+a_1y[n-1]+a_2y[n-2]+\ldots+a_N y[n-N] = b_0x[n]+b_1x[n-1]+b_2x[n-2]+\ldots+b_M x[n-M]$$

Weightings or Coefficients

Difference Equations

- The equation can be presented more compactly as

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

- If we make $a_0 = 1$, the equation can be written as

$$y[n] = -\sum_{k=1}^{N} a_k y[n-k] + \sum_{k=0}^{M} b_k x[n-k] \longrightarrow Eq:(2)$$

Past outputs

Past inputs

- The eq:(2) form shows how each new o/p from the system can be calculated using past o/ps, present i/ps & past i/ps.

Recursive Systems

- When a digital system relies on both i/ps and past o/ps, it is referred to as a Recursive system.
- Eq: (2) is the equation for Recursive systems.

Non-Recursive Systems

- When the digital system relies only on i/ps (present & past), and not on past o/ps, it is referred to as a non recursive system.
- The following eq: gives the general form for this kind of filter.

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

$$Y[n] = b_0x[n] + b_1x[n-1] + b_2x[n-2] + \ldots + b_M x[n-M]$$

Non-Recursive Systems

- Example 1: A system has the difference eq: y[n]= 0.5y[n-1]+x[n]
  1. Identify all coefficients a_k & b_k.
  2. Is this a Recursive or Non-recursive diff: eq:.
  3. If the i/p x[n] is as given in figure below, find the first 12 samples of the o/p, starting with n=0.

Solution:

1) Writing the o/ps on the left & i/ps on the right, we get

   y[n]-0.5y[n- 1]=x[n].

   So, $a_0=1$, $a_1=-0.5$, $b_0=1$.

2) Since the o/p y[n] depends on a past o/p y[n-1], the digital system is recursive.

Non-Recursive Systems

3) $Y[0]=0.5y[-1]+x[0] = 0.5(0.0)+1.0 = 1.0$

(since the sys: is considered causal, the o/p cant begin until the i/p first becomes nonzero, in this case at n=0. Hence y[-1] = 0.

$Y[1]=0.5y[0]+x[1] = 0.5(1.0)+1.0 = 1.5$

$Y[2]=0.5y[1]+x[2] = 0.5(1.5)+1.0 = 1.75$

$Y[3]=0.5y[2]+x[3] = 0.5(1.75)+1.0 = 1.875$

$Y[4]=0.5y[3]+x[4] = 0.5(1.875)+1.0 = 1.9375$

$Y[5]=0.5y[4]+x[5] = 0.5(1.9375)+1.0 = 1.9688$

$Y[6]=0.5y[5]+x[6] = 0.5(1.9688)+1.0 = 1.9844$

$Y[7]=0.5y[6]+x[7] = 0.5(1.9844)+1.0 = 1.9922$

$Y[8]=0.5y[7]+x[8] = 0.5(1.9922)+1.0 = 1.9961$

$Y[9]=0.5y[8]+x[9] = 0.5(1.9961)+1.0 = 1.9980$

$Y[10]=0.5y[9]+x[10] = 0.5(1.9980)+1.0 = 1.9990$

$Y[11]=0.5y[10]+x[11] = 0.5(1.9990)+1.0 = 1.9995$

Non-Recursive Systems

3) Because of the u[n] factor in the o/p, the values of the i/p before n=0 are zero.

n	-1	0	1	2	3	4	5
X[n]	0.0	0.0	0.643	0.985	0.866	0.342	-0.342
Y[n]	0.0	0.0	0.321	0.300	0.138	-0.089	-0.274
n	6	7	8	9	10	11	12
X[n]	-0.866	-0.985	-0.643	0.0	0.643	0.985	0.866
Y[n]	-0.330	-0.233	-0.026	0.193	0.321	0.300	0.138
n	13	14	15	16	17	18	19
X[n]	0.342	-0.342	-0.866	-0.985	-0.643	0.0	0.643
Y[n]	-0.089	-0.274	-0.330	-0.233	-0.026	0.193	0.321

Superposition in Diff: eq:

- In some instances, several i/ps may be applied to a system at the same time.
- When this happens, the system response to the sum of these inputs through superposition.
- Fortunately, when the sys: is linear, multiple inputs can be handled easily.

Superposition in Diff: eq:

- **Tutorial 1:** A system is described by the difference equation

$$Y[n]=x[n]+0.5x[n-1]$$

Two i/ps are $x_1[n]=2u[n]$

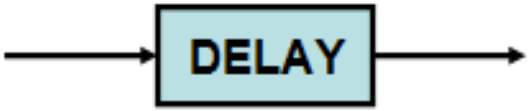
$$X_2[n]=\sin(n\pi/7)u[n]$$

Find and plot the first 20 samples of the o/p resulting from the combined effect of the 2i/ps.

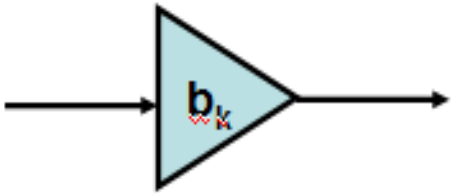
Difference eq: Diagrams

1. Non-recursive Diff: eqs: diagram

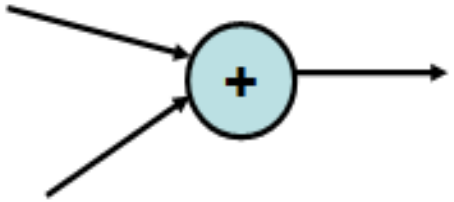
- The basic elements used in designing non-recursive diff: eq: diagrams are
  - Delay element
  - Coefficient multiplier
  - Summer



Delay element



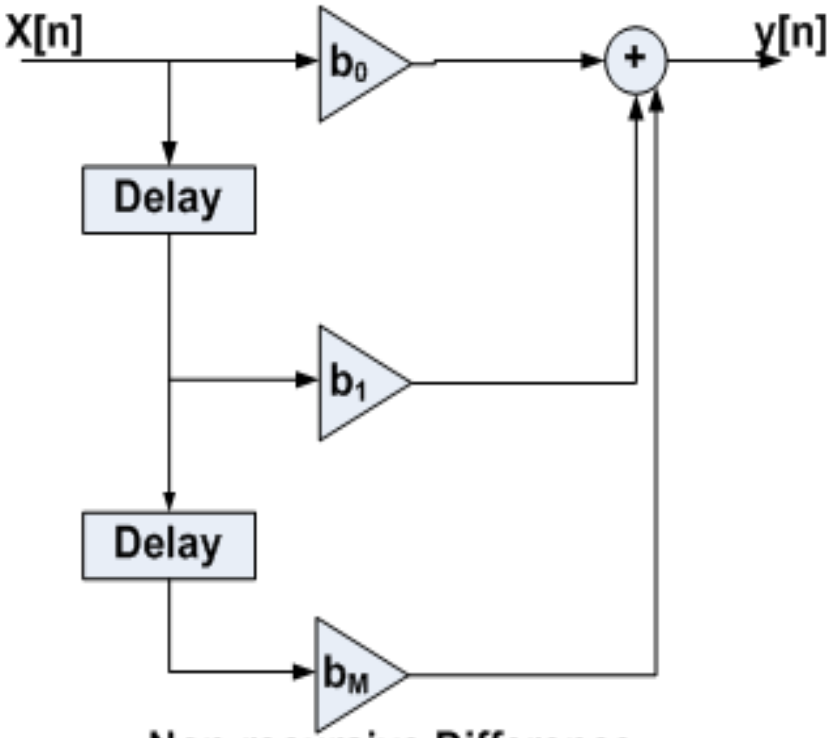
Coefficient multiplier



Summer

Difference eq: Diagrams

- A general non-recursive diff: eq: described previously can be presented schematically as below.

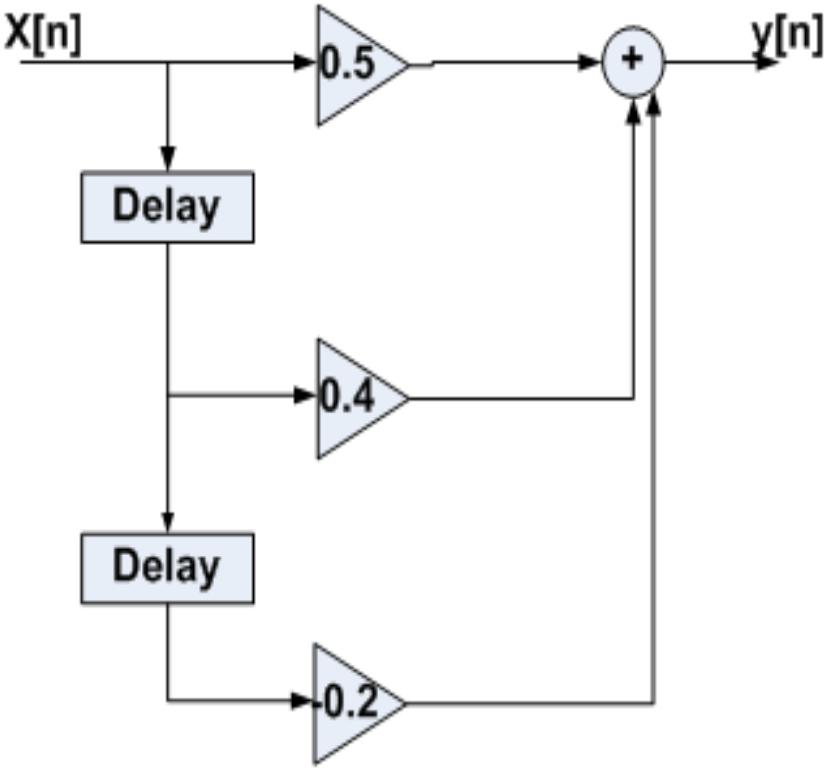


Non-recursive Difference equation Diagram

Difference eq: Diagrams

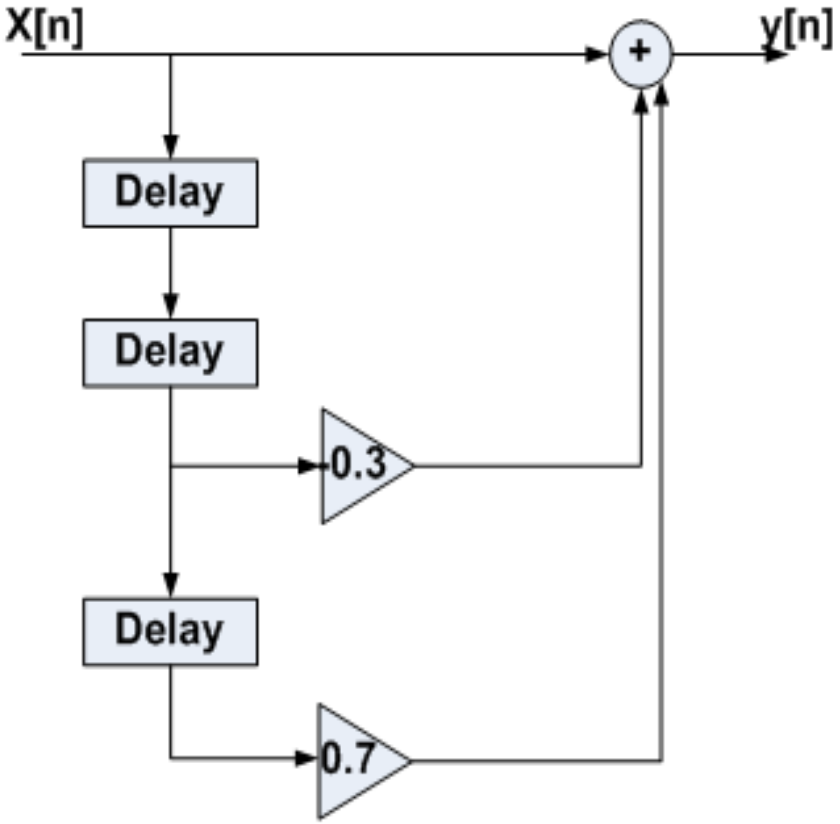
- **Example3:** Draw a diagram for the diff: eq: y[n]=0.5x[n]+0.4x[n-1]-0.2x[n-2]

Solution:



Difference eq: Diagrams

- **Tutorial 2:** Write the diff: eq: that corresponds to the diagram given below

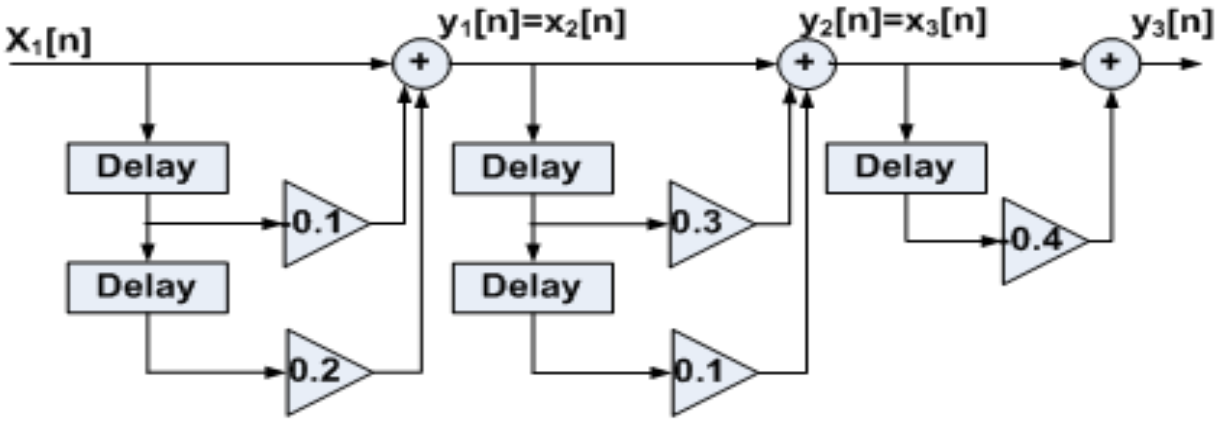


Difference eq: Diagrams

- Higher order system can be broken down into second order chunks, and cascaded together.
- When the order of the system is odd, a single first order section is added to the group of 2^{nd} order section.
- The following example illustrates this point.

Difference eq: Diagrams

- **Example4:** Find the difference eq: of the following cascaded diagram



Solution:

The first stage produces the diff: eq:$Y_1[n]=x_1[n]-0.1x_1[n-1]+0.2x_1[n-2]$

The 2nd stage produces the diff: eq:$Y_2[n]=x_2[n]+0.3x_2[n-1]+0.1x_2[n-2]$

The 3rd stage produces the diff: eq:$Y_3[n]=x_3[n]-0.4x_3[n-1]$

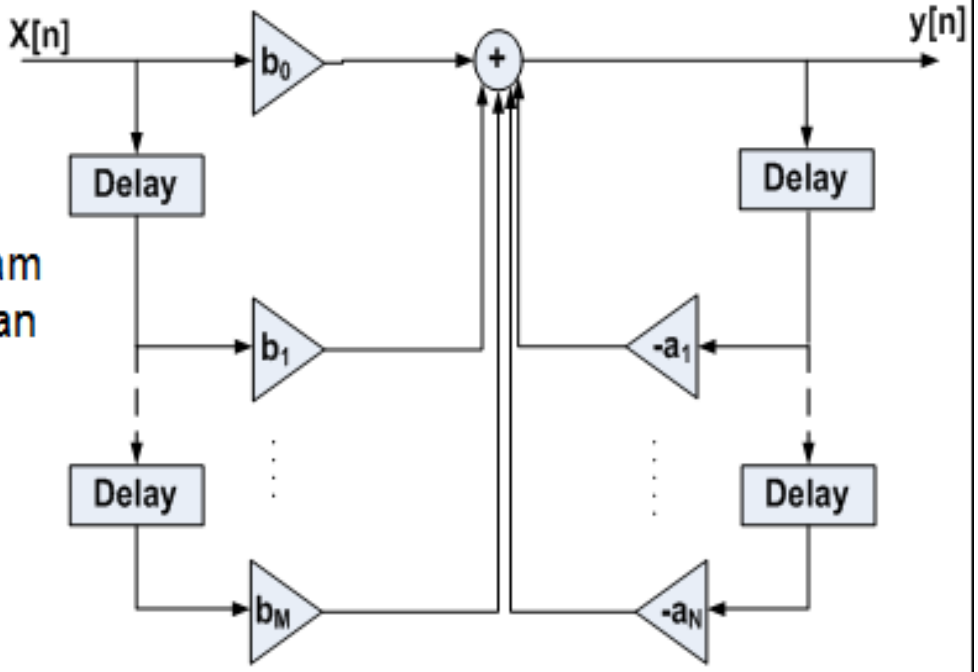
The final diff: eq: will become $y_3[n]=x_1[n]-0.2x_1[n-1]+0.19x_1[n-2]-0.058x_1[n-3]-0.008x_1[n-5]$

Difference eq: Diagrams

2. Recursive difference equations

1. Direct form 1 Realization

- In this form, the diagram of recursive diff: eq: can be made by using the previous diagram elements.
- The general recursive diff: eq: described previously cab be depicted as,



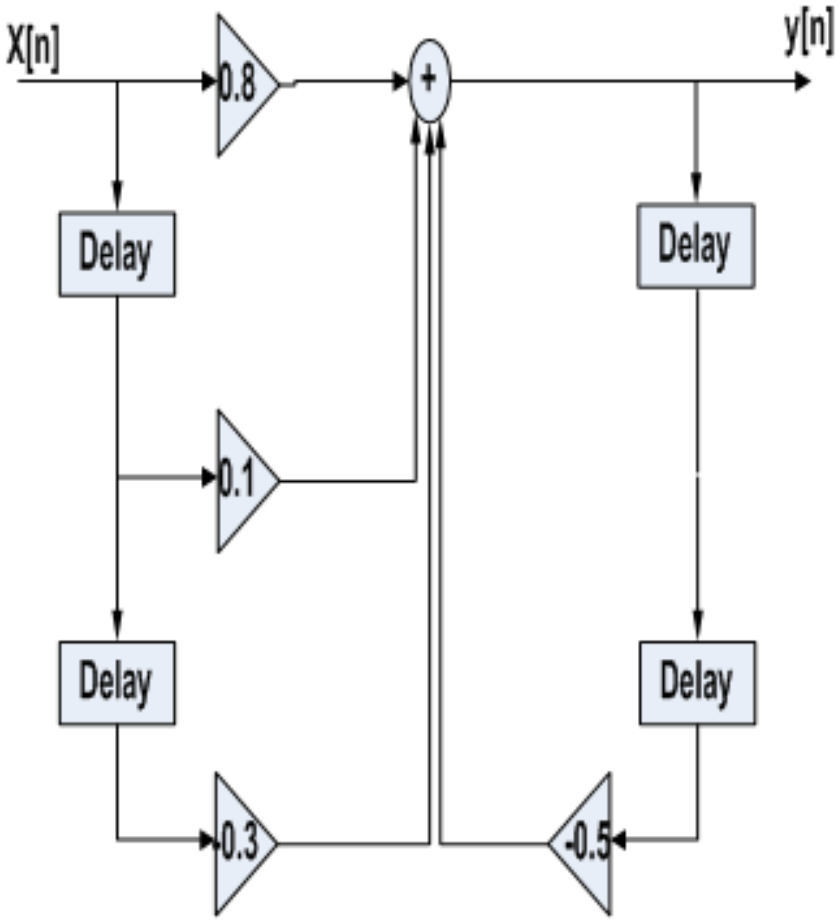
Recursive Direct Form 1 Realization
Difference equation Diagram

Difference eq: Diagrams

- **Example5:** Draw a direct form1 realization diff: eq: to describe the recursive system.
  y[n]+0.5y[n-2]=0.8x[n]+0.1x[n-1]-0.3x[n-2]

  Solution: Rearranging the eq: we get

  y[n]=-0.5y[n-2]+0.8x[n]+0.1x[n-1]-0.3x[n-2]



Difference eq: Diagrams

2. Direct for 2 Realization

- The Form 1 realization is not the most efficient to implement a recursive diff: eq:
- A much efficient way to implement recursive diff: eq: is direct form 2 realization.
- This realization requires the use of an intermediate signal w[n] that records salient information about the history of the system in place of past i/ps and past o/ps.
- The two eqs: that define DF2 realization are

$$w[n] = x[n] - \sum_{k=1}^{N} a_k w[n-k]$$

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

Difference Equation

A discrete-time signal $s(n)$ is **delayed** by n_0 samples when we write $s(n-n_0)$, with $n_0 > 0$. Choosing n_0 to be negative advances the signal along the integers. As opposed to analog delays (pg ??), discrete-time delays can *only* be integer valued. In the frequency domain, delaying a signal corresponds to a linear phase shift of the signal's discrete-time Fourier transform: $s(n-n_0) \leftrightarrow e^{-(j2\pi f n_0)} S\left(e^{j2\pi f}\right)$.

Linear discrete-time systems have the superposition property.

Superposition

$$S(a_1 x_1(n) + a_2 x_2(n)) = a_1 S(x_1(n)) + a_2 S(x_2(n))$$

A discrete-time system is called **shift-invariant** (analogous to time-invariant analog systems (pg ??)) if delaying the input delays the corresponding output.

Shift-Invariant

$$IfS(x(n)) = y(n), ThenS(x(n-n_0)) = y(n-n_0)$$

Difference Equation

We use the term shift-invariant to emphasize that delays can only have integer values in discrete-time, while in analog signals, delays can be arbitrarily valued.

We want to concentrate on systems that are both linear and shift-invariant. It will be these that allow us the full power of frequency-domain analysis and implementations. Because we have no physical constraints in "constructing" such systems, we need only a mathematical specification. In analog systems, the differential equation specifies the input-output relationship in the time-domain. The corresponding discrete-time specification is the **difference equation** .

The Difference Equation

$$y(n) = a_1 y(n-1) + \ldots + a_p y(n-p) + b_0 x(n) + b_1 x(n-1) + \ldots + b_q x(n-q)$$

Here, the output signal $y(n)$ is related to its *past* values $y(n-l)$, $l = \{1, \ldots, p\}$, and to the current and past values of the input signal $x(n)$. The system's characteristics are determined by the choices for the number of coefficients p and q and the coefficients' values $\{a_1, \ldots, a_p\}$ and $\{b_0, b_1, \ldots, b_q\}$.

Analysis of DT-LTI system

- The second method for analyzing the behavior of LTI system to a given i/p signal is first to decompose or resolve the input signal into a sum of elementary signals.
- The elementary signals are selected so that the response of the system to each signal component is easily determined.
- Then using the linearity property of the sys, the responses of the system to the elementary signals are added to obtain the total response of the sys to the given i/p signal.
- The elementary signal we choose to analyze LTI system is impulse signal.

Analysis of DT-LTI system

Suppose we have an arbitrary signal x[n] that we wish to resolve into a sum of unit sample sequence.

Consider the product of a signal x[n] and the impulse sequence δ[n], written as x[n]δ[n]

Since δ[n]=1 only at n=0,so we can write

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

If we were to repeat the multiplication of x[n] with δ[n-k], where δ [n-k] is time shifted impulse sequence, the result will be a sequence that is zero everywhere except at n=k,

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

This property allows us to express x[n] as the following weighted sum of time shifted impulses:

x[n] = ... + x[-2]δ[n+2]+ x[-1]δ[n+1]+ x[0]δ[n] + x[1]δ[n-1] + x[2]δ[n-2]+...

Or in concise form as:

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

Right hand side gives us the resolution of any arbitrary signal x[n] into weighted sum of shifted unit sample sequences.

Analysis of DT-LTI system

- The second method for analyzing the behavior of LTI system to a given i/p signal is first to decompose or resolve the input signal into a sum of elementary signals.
- The elementary signals are selected so that the response of the system to each signal component is easily determined.
- Then using the linearity property of the sys, the responses of the system to the elementary signals are added to obtain the total response of the sys to the given i/p signal.
- The elementary signal we choose to analyze LTI system is impulse signal.

Resolution of DT signals into Impulses

Suppose we have an arbitrary signal x[n] that we wish to resolve into a sum of unit sample sequence.

Consider the product of a signal x[n] and the impulse sequence δ[n], written as x[n]δ[n]

Since δ[n]=1 only at n=0,so we can write

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

If we were to repeat the multiplication of x[n] with δ[n-k], where δ [n-k] is time shifted impulse sequence, the result will be a sequence that is zero everywhere except at n=k,

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

This property allows us to express x[n] as the following weighted sum of time shifted impulses:

$$x[n] = \ldots + x[-2]\delta[n+2] + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + x[2]\delta[n-2] + \ldots$$

Or in concise form as:

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

Right hand side gives us the resolution of any arbitrary signal x[n] into weighted sum of shifted unit sample sequences.

Impulse Response of LTI System

The **impulse response** is exactly what its name implies - the response of an LTI system, such as a filter, when the system's input is the unit impulse (or unit sample). A system can be completed describe by its impulse response due to the idea mentioned above that all signals can be represented by a superposition of signals. An impulse response gives an equivalent description of a system as a transfer fucntion, since they are Laplace Transforms of each other.

NOTATION: Most texts use $\delta(t)$ and $\delta[n]$ to denote the continuous-time and discrte-time impulse response, respectively.

Impulse Response of LTI System

The response of a system to the unit input $\delta[n]$ is called the *Impulse Response*, normally written as $h[n]$.

In the case of Linear-Time-Invariant (LTI) systems it conpletely characterizes their behavior. This is because every input sequence can be described as a linear combination of delayed copies of the unit sequence, and using linearity and time-invariance, the reponse can be built as a superposition of delayed impulse reponses. Such superposition can be written as the sum:

Impulse Response of LTI System

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

called *convolution* of the input and the impulse response.

The convolution has interesting properties, such as *commutativity* ($x*y = y*x$), *associativity* ($(x*y)*z = x*(y*z)$) and distributivity ($x*(y+z) = x*y+x*z$). Furthermore, properties of LTI systems are simply described by $h[n]$:

- Stability: A system is stable if and only if the impulse response is absolutely summable ($\sum_{-\infty}^{\infty} |x[n]| < \infty$)
- Causality: A system is causal if and only if its impluse response is a causal signal.

Finally, simple interconnection schemes of systems result in simple composition of the impulse response:

- *Cascade connection*: The impulse response is the convolution of the responses ($h_{tot} = h_1 * h_2$). Important consequence is the fact that order is not important in cascade connections. Stability, passivity and losslessness are preserved.
- *Parallel connection*: The response is the sume of the responses ($h_{tot} = h_1 + h_2$). Stability is preserved.

Impulse Response of LTI System

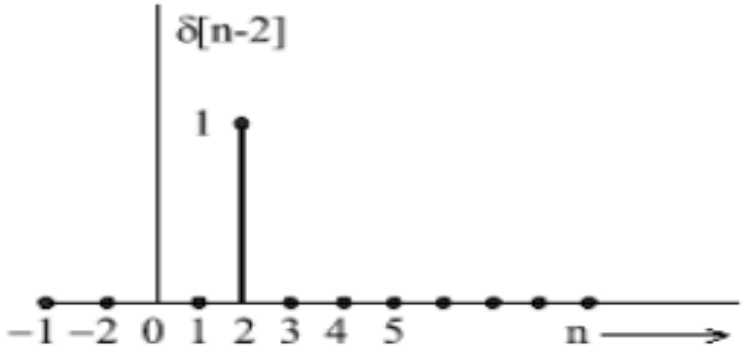
The unit impulse was described above as:

$$\delta[n] = 0, \quad n \neq 0$$
$$\delta[n] = 1, \quad n = 0$$

This is also sometimes known as the Kronecker ***delta function***

This can be tabulated

n		-2	-1	0	1	2	3	4	5	6	
δ[n]	0	0	0	**1**	0	0	0	0	0	0	0
δ[n-2]	0	0	0	0	0	**1**	0	0	0	0	0



Shifted impulse sequence, δ[n – 2]

Impulse Response of LTI System

The third row of table 1 gives the values of the shifted impulse $\delta[n-2]$.
Now consider the following signal:

$$x[n] = 2\delta[n] + 4\delta[n-1] + 6\delta[n-2] + 4\delta[n-3] + 2\delta[n-4]$$

Table 2 shows the individual sequences and their sum.

n		-2	-1	0	1	2	3	4	5	6	
2δ[n]	0	0	0	**2**	0	0	0	0	0	0	0
4δ[n-1]	0	0	0	0	**4**	0	0	0	0	0	0
6δ[n-2]	0	0	0	0		**6**	0	0	0	0	0
4δ[n-3]	0	0	0	0	0	0	**4**	0	0	0	0
2δ[n-4]	0	0	0	0	0	0	0	**2**	0	0	0
x[n]	0	0	0	**2**	**4**	**6**	**4**	**2**	0	0	0

Table 2

Hence any sequence can be represented by the equation:

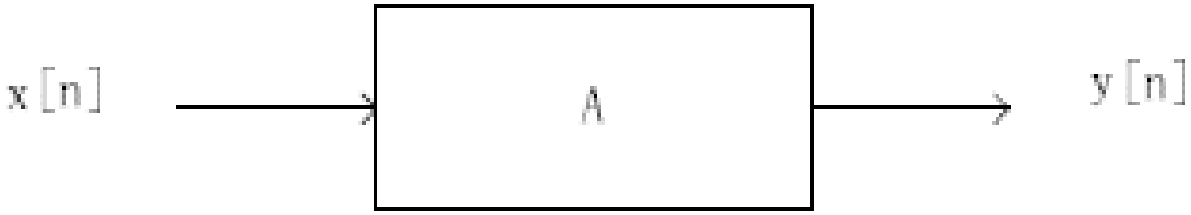
$$x[n] = \sum_k x[k]\delta[n-k]$$

$$= + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + x[2]\delta[n-2] + \ldots\ldots$$

When the input to an FIR filter is a unit impulse sequence, $x[n] = \delta[n]$, the output is known as the **unit impulse response**, which is normally donated as $h[n]$.

Impulse Response of LTI System

Suppose that a signal x[n] is given as input to a linear system



First, let us look at $x[n]\delta[n-k]$ as a function of $n \in \mathbb{Z}$, where k is fixed.

$$x[n]\delta[n-k] = \begin{cases} x[k] & \text{if } n = k \\ 0 & \text{if } n \neq k \end{cases}$$

Impulse Response of LTI System

This holds for any fixed k,

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

This is the *sifting property*.

The system is linear. If the response of the system to $\delta[n-k]$ (where k is fixed and $n \in \mathbb{Z}$) is $h_k[n]$, then

Impulse Response of LTI System

the output of the system for $x[n]$ is given by

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

If in addition, the system is time-invariant(LTI), then if we let $h_0[n] = h[n]$ to be the response to $\delta[n]$, then $h_k[n] = h[n-k]$, so we have

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

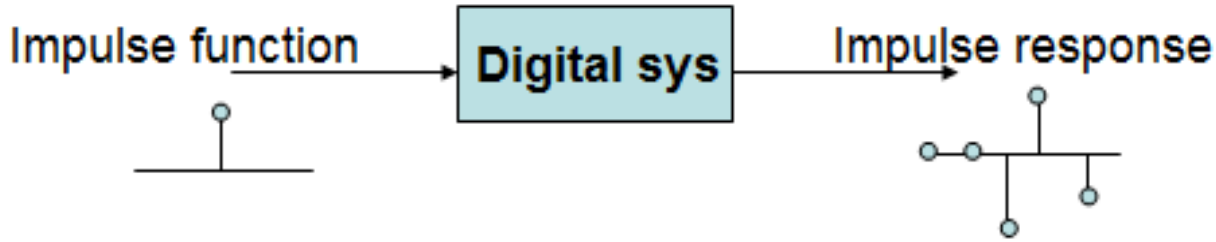
Impulse Response of LTI System

Summary

$h[n]$ is the response of an LTI system to $\delta[n]$ and is called the impulse response of an LTI system. Then $y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$ is the response of the system to $x[n]$.

Impulse Response & Difference Eq:

- When i/p to a system is a unit impulse function the o/p from the sys: is the unit impulse response as shown in figure.



- The diff: eq: for a sys can be used to calculate the impulse response for the system.
- Just replace x[n] by $\delta[n]$ and y[n] by h[n] an further steps are usual.

Impulse Response & Difference Eq:

- **Example7:** Find the first 6 samples of the impulse for the different equation.

  Y[n]-0.4y[n-1]=x[n]-x[n-1]

Solution:

First, replace x[n] with $\delta[n]$ and y[n] with h[n] to give

h[n]-0.4h[n-1]= $\delta[n]$ - $\delta[n-1]$

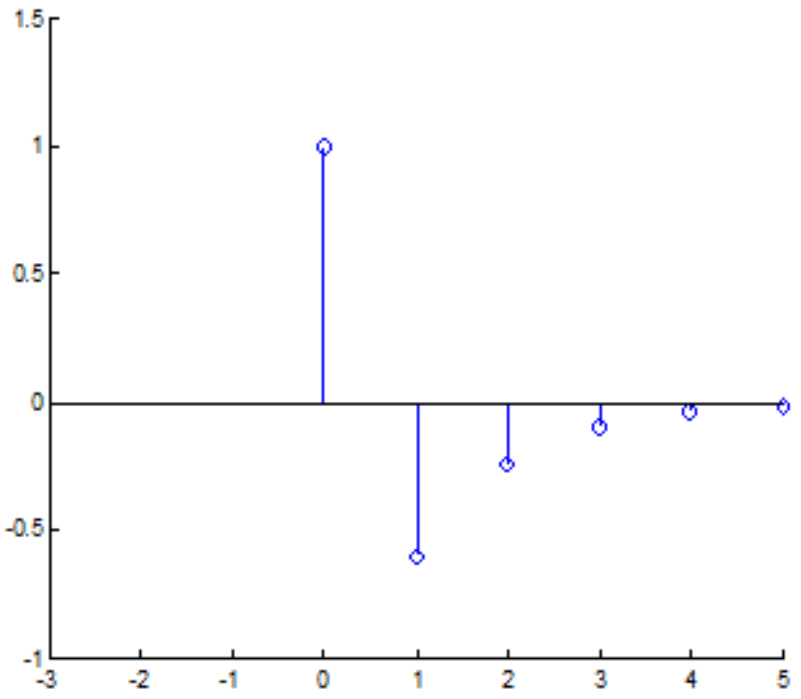
Starting with n=0:

h[0]=0.4h[-1]+δ[0]-δ[-1]

h[0]=0.4(0.0)+1.0-0.0=1.0

And further… h[1]=-0.6

h[2]=-0.24; h[3]=-0.096; h[4]=-0.0384; h[5]=-0.01536



Infinite Impulse Response (IIR)

- In previous example, the impulse response never dies away. Reason is the new o/ps depends on old o/ps.
- The impulse response that never dies away or tends to infinite is called IIR and is typical for recursive diff: eq:.

Infinite Impulse Response (IIR)

- **Example8:** Find & plot first 6 samples in the impulse response for the system

$$y[n]=0.25(x[n]+x[n-1]+x[n-2]+x[n-3])$$

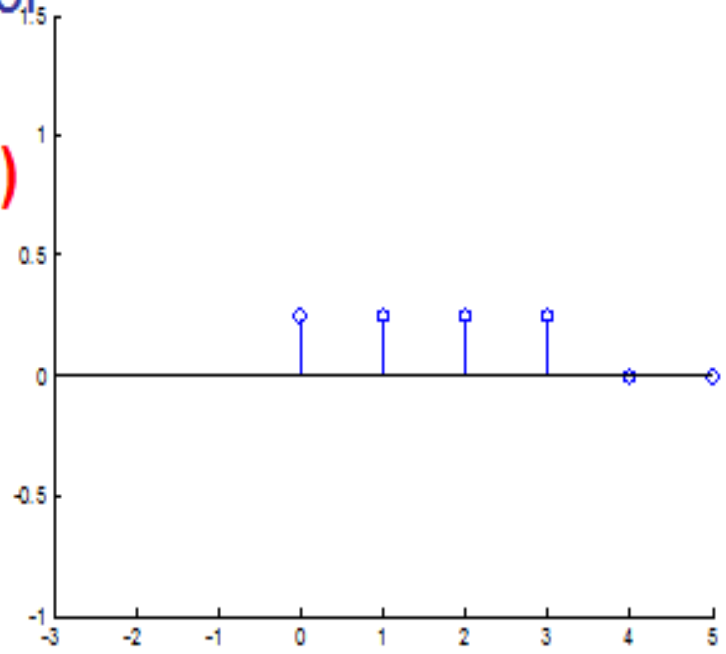
Solution:

Substituting symbols for impulse response we get

$$h[n]=0.25(\delta[n]+\delta[n-1]+\delta[n-2]+\delta[n-3])$$

So we get,

$$h[0]=h[1]=h[2]=h[3]=0.25;\ h[4]=h[5]=0.0$$



Finite Impulse Response (FIR)

- In example#8, note that the impulse response drops to zero after a finite no of nonzero samples.
- If the impulse response drops to zero after a finite no: of nonzero samples the response is said to as FIR and is typical for non-recursive diff: eqs:.

Step Response & Difference Equations

- It is a response for a system to a unit step function.
- Step function (i/p) is u [n]
- Step response (o/p) is s [n]
- There are two simple ways to find the step response for a sys:
  1. Use of diff: eq: with u[n] as i/p
  2. Determine impulse response and sum it

Tutorial#3:

Find & plot the step response for the system

$$y[n]-0.2y[n-2]=0.5x[n]+0.3x[n-1]$$

by the following methods:

1. Use of diff: eq: with u[n] as i/p
2. Determine impulse response and sum it

Step Response of an LTI System

The step response of an LTI system is simply the response of the system to a unit step. It conveys a lot of information about the system. For a discrete-time system with impulse response $h[n]$, the step response is $s[n] = u[n] * h[n]$. However, based on the commutative property of convolution, $s[n] = h[n] * u[n]$, and therefore, $s[n]$ can be viewed as the response to input $h[n]$ of a discrete-time LTI system with unit impulse response. We know that $u[n]$ is the unit impulse response of the accumulator. Therefore,

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

Step Response of an LTI System

From this equation, $h[n]$ can be recovered from $s[n]$ using the relation

$$h[n] = s[n] - s[n-1].$$

It can be seen the step response of a discrete-time LTI system is the *running sum of its impulse* response. Conversely, the impulse response of a discrete-time LTI system is the *first difference* of its step response.

Step Response of an LTI System

Similarly, in continuous time, the step response of an LTI system is the running integral of its impulse response,

$$s(t) = \int_{-\infty}^{t} h(\tau)d\tau\,,$$

and the unit impulse response is the first derivative of the unit step response,

$$h(t) = \frac{ds(t)}{dt} = s'(t)\,.$$

Therefore, in both continuous and discrete time, the unit step response can also be used to characterize an LTI system.

Step Response of an LTI System

The unit impulse response can be derived from the unit step response as

$$h(t) = \frac{ds(t)}{dt} = s'(t)$$

In discrete time

$$s[n] = u[n] * h[n] = \sum_{k=-\infty}^{n} h[k]$$

$$h[n] = s[n] - s[n-1]$$

Discrete-Time Convolution

- If $h(n)$ is the system impulse response, then the input-output relationship is a convolution.
- It is used for designing filter or a system.
- Definition of convolution:

$$y(n) = h(n) * x(n) = \sum_{\lambda=-\infty}^{\infty} h(\lambda)x(n-\lambda)$$

$$y(n) = x(n) * h(n) = \sum_{\lambda=-\infty}^{\infty} x(\lambda)h(n-\lambda)$$

Discrete-Time Convolution

The simplest example of convolution is the multiplication of two polynomials. e.g.

$$y = (4x^2 - 3x + 9)(3x^2 + 4x + 4)$$

This is calculated by:

$$y = \left((4x^2 \times 3x^2) + (4x^2 \times 4x) + (4x^2 \times 4)\right) + \left((-3x \times 3x^2) + (-3x \times 4x) + (-3x \times 4)\right) + \left((9 \times 3x^2) + (9 \times 4x) + (9 \times 4)\right)$$

$$y = 12x^4 + 16x^3 + 16x^2 - 9x^3 - 12x^2 - 12x + 27x^2 + 36x + 36$$

$$y = 12x^4 + 7x^3 + 31x^2 + 24x + 36$$

Convolution is a weighted moving average with one signal *flipped back to front*:

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

Discrete-Time Convolution

A tabulated version of convolution

n	n<0	0	1	2	3	4	5	6	7	n<7
x[n]	0	2	4	6	4	2	0	0	0	0
h[n]	0	3	-1	2	1					
h[0]x[n]	0	6	12	18	12	6	0	0	0	0
h[1]x[n-1]	0	0	-2	-4	-6	-4	-2	0	0	0
h[2]x[n-2]	0	0	0	4	8	12	8	4	0	0
h[3]x[n-3]	0	0	0	0	2	4	6	4	2	0
y[n]	0	6	10	18	16	18	12	8	2	0

$$h[0]x[n] = x[0] * h[0] + x[1] * h[0] + x[2] * h[0] + x[3] * h[0] + x[4] * h[0]$$

$$h[0]x[n] = 2 * 3 + 4 * 3 + 6 * 3 + 4 * 3 + 2 * 3$$

$$h[0]x[n] = 6 + 12 + 18 + 12 + 6$$

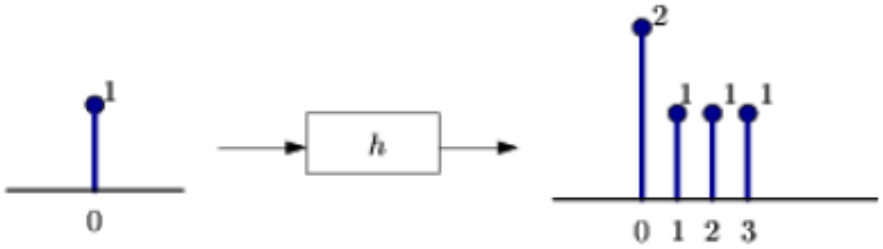
$$h[1]x[n-1] = x[0] * h[1] + x[1] * h[1] + x[2] * h[1] + x[3] * h[1] + x[4] * h[1]$$

$$h[1]x[n-1] = 2 * -1 + 4 * -1 + 6 * -1 + 4 * -1 + 2 * -1$$

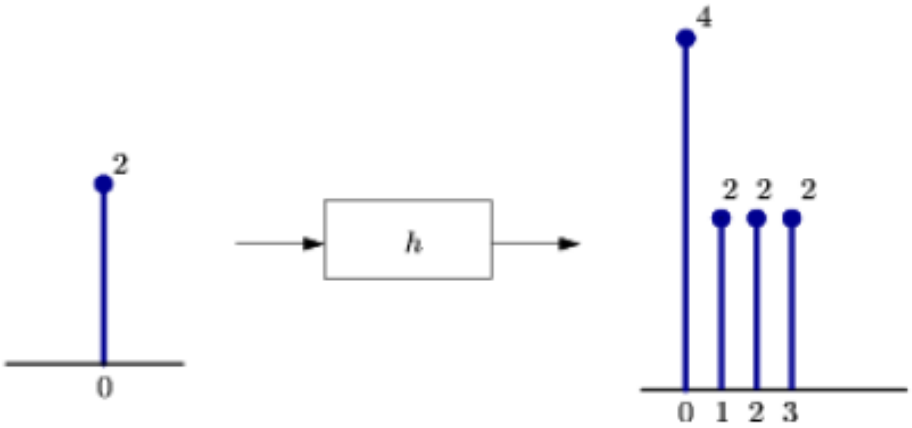
$$h[1]x[n-1] = -2 + -4 + -6 + -4 + -2$$

Discrete-Time Convolution

The diagrams below show how convolution works.

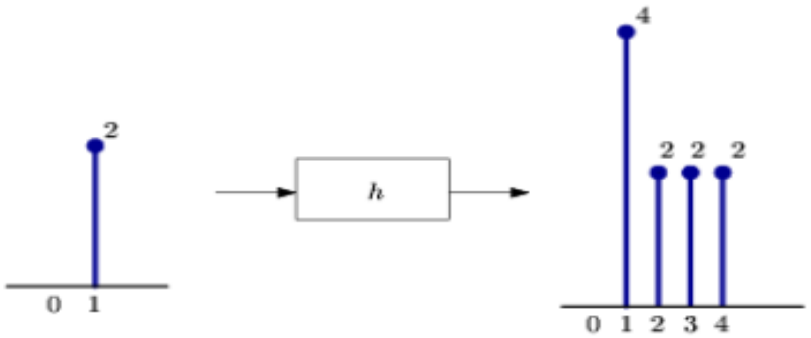


A single impulse input yields the system's impulse response

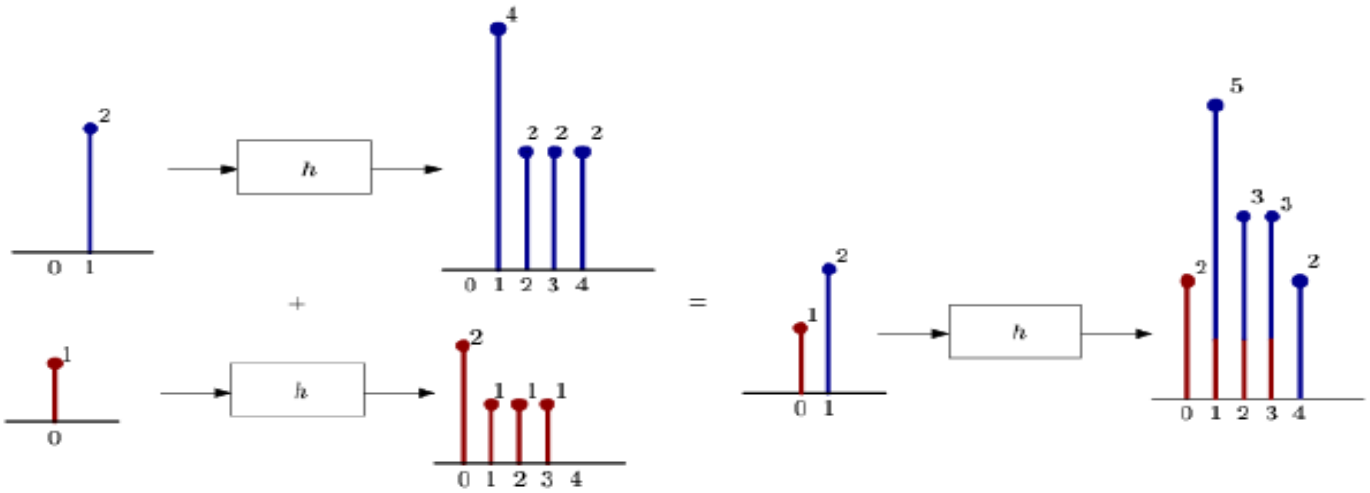


A scaled impulse input yields a scaled response, due to the ***scaling property*** of the system's linearity.

Discrete-Time Convolution



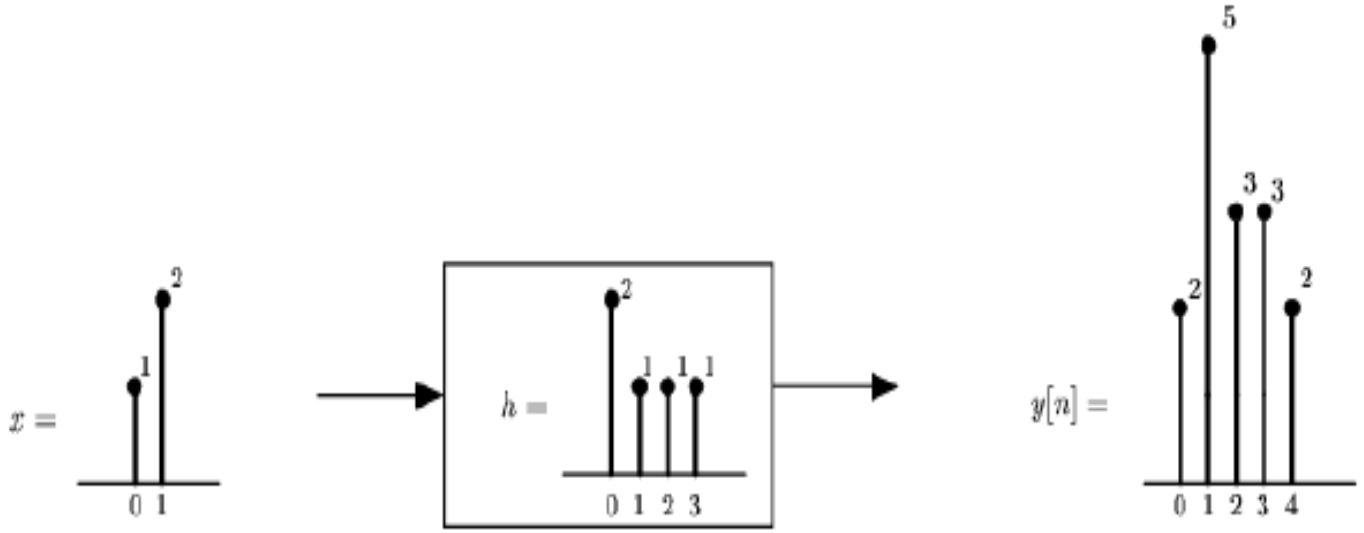
This demonstrates the use the *time-invariance property* of the system to show that a delayed input results in an output of the same shape, only delayed by the same amount as the input



Discrete-Time Convolution

This now demonstrates the additivity portion of the linearity property of the system to complete the picture. Since any discrete-time signal is just a sum of scaled and shifted discrete-time impulses, we can find the output from knowing the input and the impulse response

No if we convolve x(n) with h(n) as shown in Figure 9 we will get the output y(n)

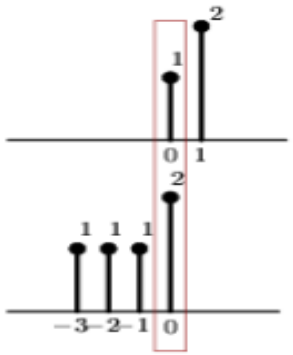


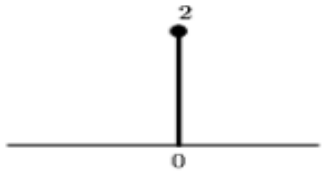
This is the end result that we are looking to find

Discrete-Time Convolution

The following diagrams are a breakdown of how the y(n) output is achieved.

$n = 0$

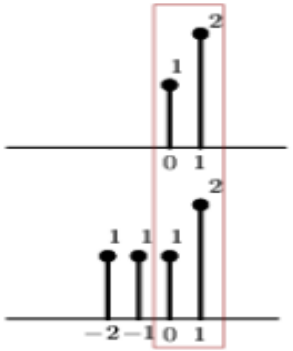


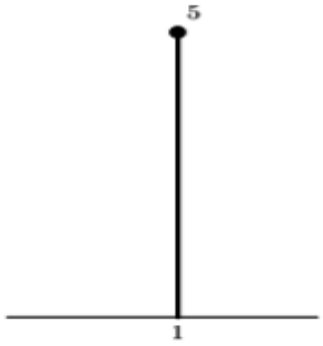


$x[0] \times h[-0] = 1 \times 2 = 2$

The impulse response, h , is reversed and begin its traverse at time 0.

$n = 1$



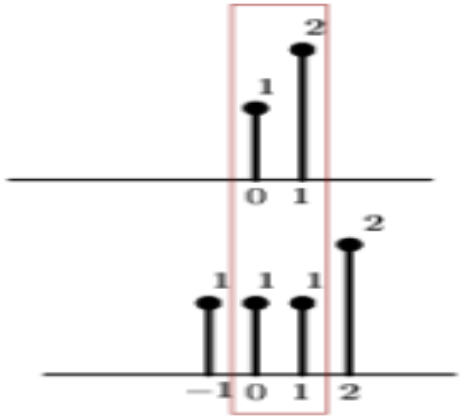


$(x[0] \times h[1]) + (x[1] \times h[0]) = 1 + 4 = 5$

Continuing the traverse. At time 1 , the two elements of the input signal are multiplied by two elements of the impulse response.

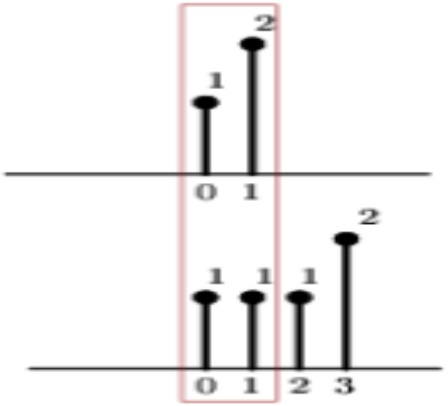
Discrete-Time Convolution

$n = 2$



$(x[0] \times h[2]) + (x[1] \times h[1]) = 1 + 2 = 3$

$n = 3$



$(x[0] \times h[3]) + (x[1] \times h[2]) = 1 + 2 = 3$

Discrete Time Convolution

Overview

Convolution is a concept that extends to all systems that are both **linear and time-invariant** (**LTI**). The idea of **discrete-time convolution** is exactly the same as that of continuous-time convolution . For this reason, it may be useful to look at both versions to help your understanding of this extremely important concept. Recall that convolution is a very powerful tool in determining a system's output from knowledge of an arbitrary input and the system's impulse response. It will also be helpful to see convolution graphically with your own eyes and to play around with it some, so experiment with the applets available on the internet. These resources will offer different approaches to this crucial concept.

Discrete Time Convolution

As mentioned above, the convolution sum provides a concise, mathematical way to express the output of an LTI system based on an arbitrary discrete-time input signal and the system's response. The **convolution sum** is expressed as

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

As with continuous-time, convolution is represented by the symbol *, and can be written as

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

By making a simple change of variables into the convolution sum, $k = n - k$, we can easily show that convolution is **commutative**:

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

Discrete-Time Convolution

Let us call $x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$ the convolution of $x[n]$ and $h[n]$.

- The response of an LTI system to $x[n]$ is given by $y[n] = x[n] * h[n]$, where $h[n]$ is the impulse response of the system.

Discrete-Time Convolution

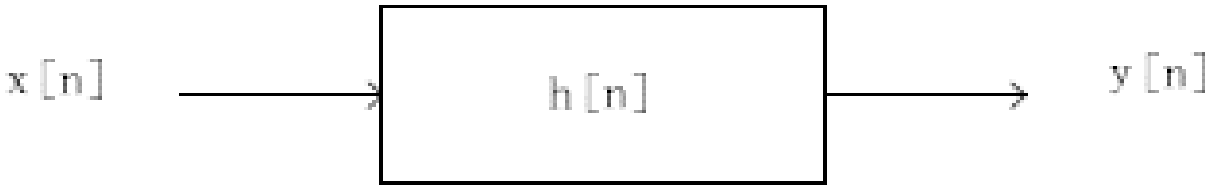
Let us consider $x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$

Let $n - k = m$, then $k = n - m$

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

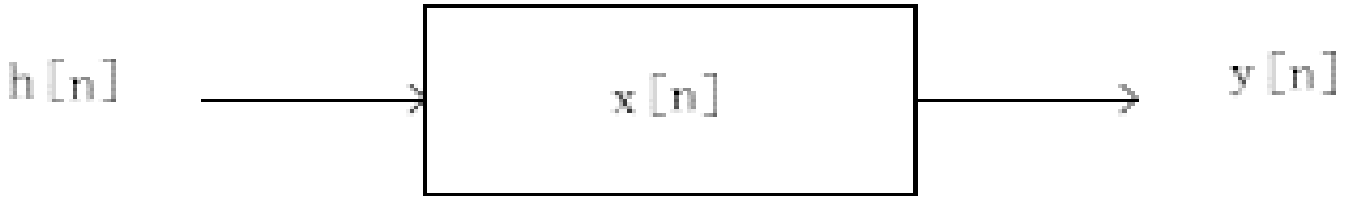
so convolution is commutative.

Notation:



Discrete-Time Convolution

This means that the box is an LTI system with impulse response $h[n]$. In the other case:

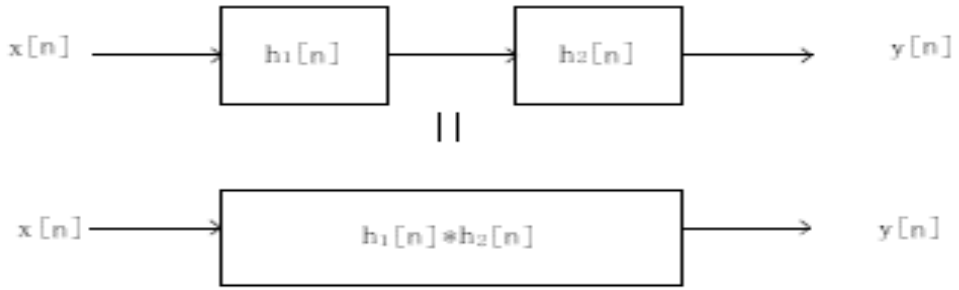


It can be shown that the convolution operation is associative and distributive.

Associativity:

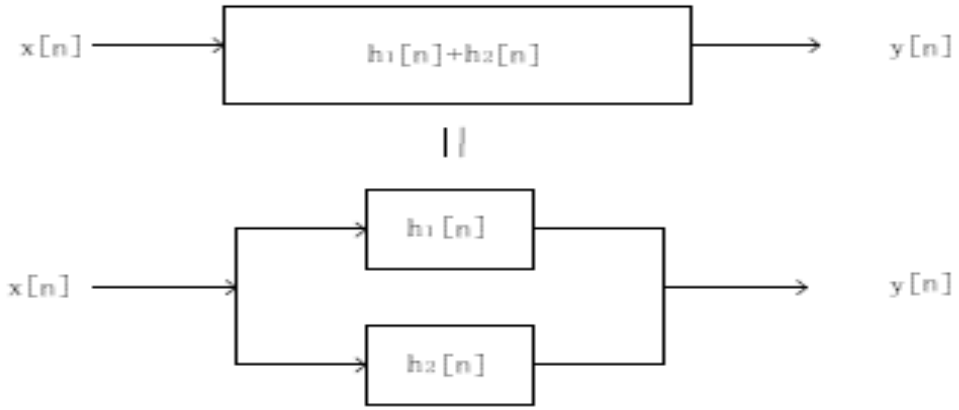
$$(x[n] * h_1[n]) * h_2[n] = x[n] * (h_1[n] * h_2[n])$$

Discrete-Time Convolution



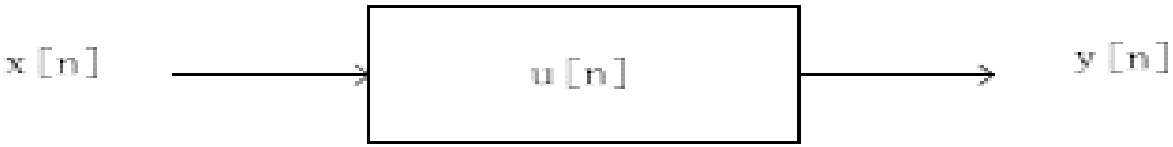
Distributivity:

$$x[n] * (h_1[n] + h_2[n]) = x[n] * h_1[n] + x[n] * h_2[n]$$



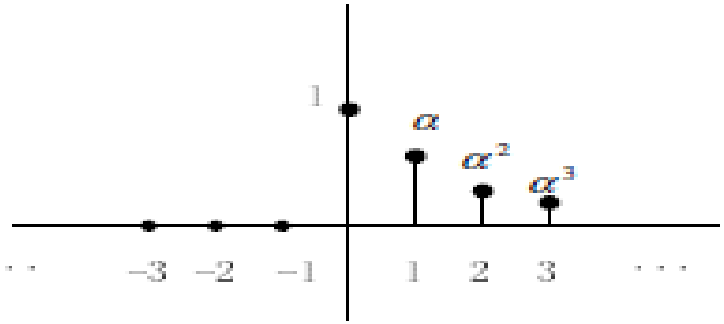
Discrete-Time Convolution: Example 1

Example:

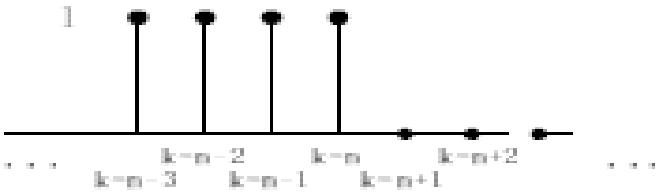


$h[n] = u[n]$, and $x[n] = \alpha^n u[n]$, $|\alpha| < 1$, what is $y[n]$?

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



$h[n-k] = u[n-k]$



Discrete-Time Convolution: Example 1

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

If $n < 0$, then $x[k]h[n-k] = 0$, so $x[n] * h[n] = 0$

$$\text{If } n \geq 0, \text{ then } x[k]h[n-k] = \begin{cases} \alpha^k, & 0 \leq k \leq n \\ 0, & otherwise \end{cases}$$

$$\text{so } x[n] * h[n] = \sum_{k=0}^{n} \alpha^k = \frac{1-\alpha^{n+1}}{1-\alpha}$$

Discrete-Time Convolution: Example 1

Mathematically,

$$
\begin{aligned}
x[n] * h[n] &= \sum_{k=-\infty}^{\infty} x[k]h[n-k] \\
&= \sum_{k=-\infty}^{\infty} \alpha^k u[k]h[n-k] \\
&= \sum_{k=0}^{n} \alpha^k u[k]h[n-k] \\
&= \begin{cases} \sum_{k=0}^{n} \alpha^k & n \geq 0 \\ 0 & n < 0 \end{cases}
\end{aligned}
$$

Discrete-Time Convolution: Example 2

Consider a system with an impulse response of

$$h(n) = [\,1\ 1\ 1\ 1\,]$$

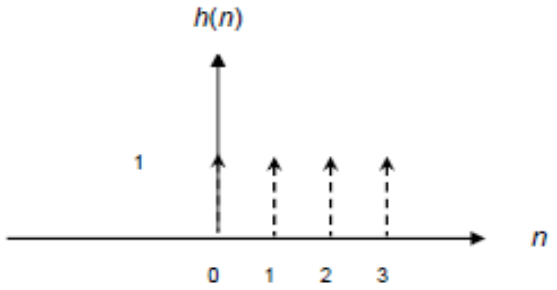
If the input to the signal is

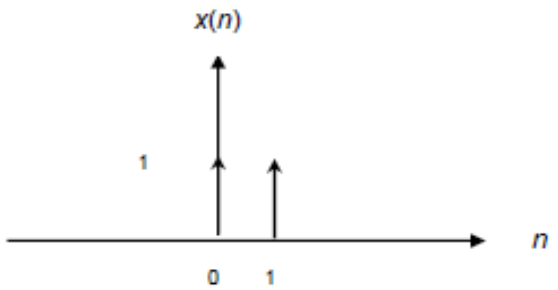
$$x(n) = [\,1\ 1\,]$$

- Thus, the output of the system is $$y(n) = \sum_{\lambda=-\infty}^{\infty} h(\lambda)x(n-\lambda)$$

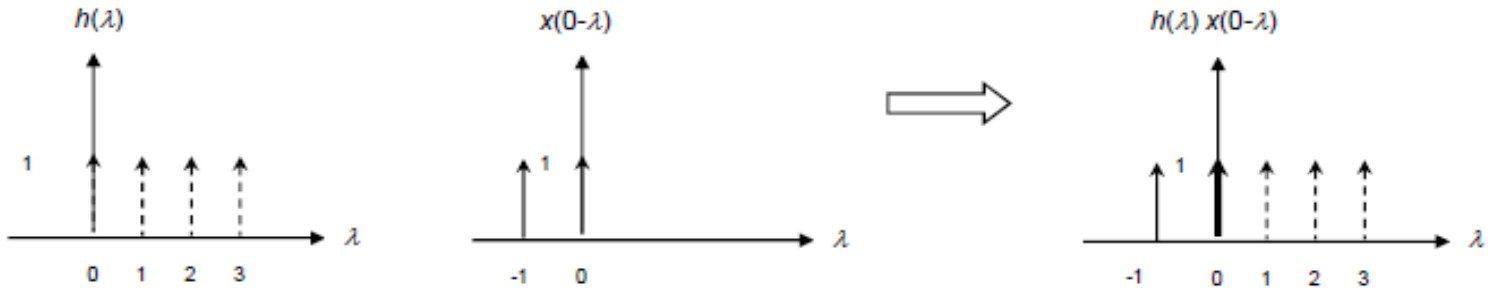
- The result of the convolution procedure in its graphical form is :

Discrete-Time Convolution: Example 2





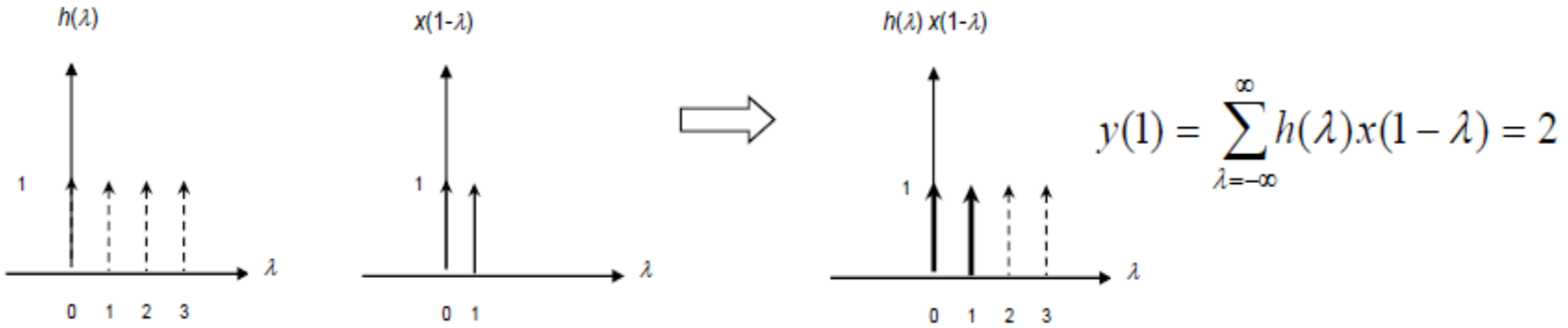
i) The definition of the system impulse response $h(n)$ and the input signal $x(n)$



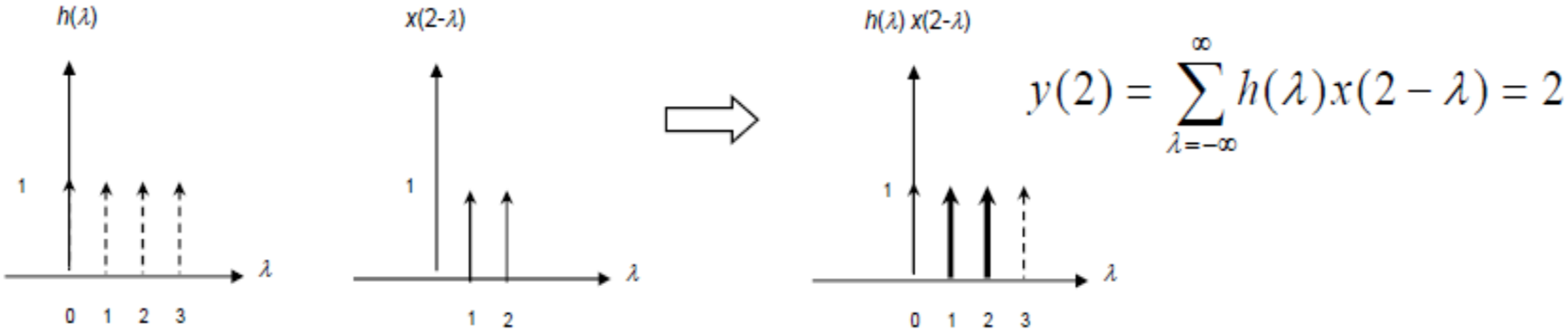
$$y(0) = \sum_{\lambda=-\infty}^{\infty} h(\lambda)x(0-\lambda) = 1$$

Discrete-Time Convolution: Example 2

ii) The result at n=1.

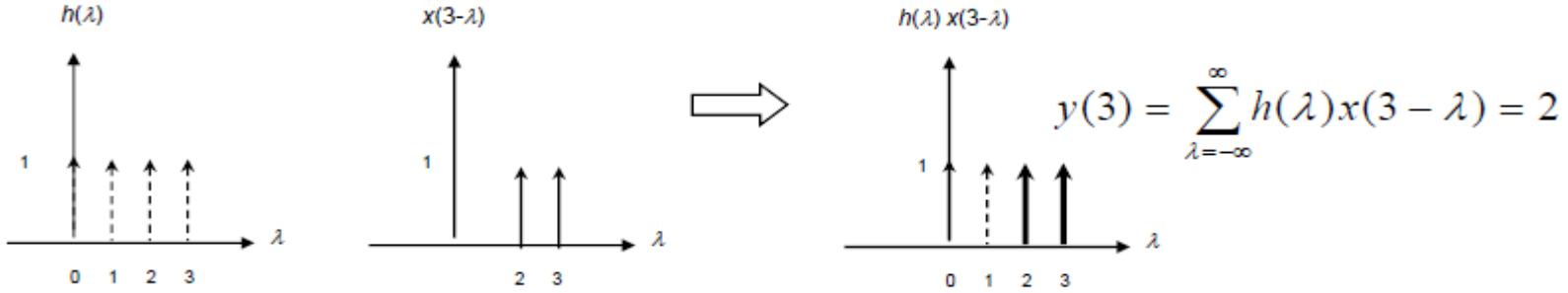


iii) The result at n=2.

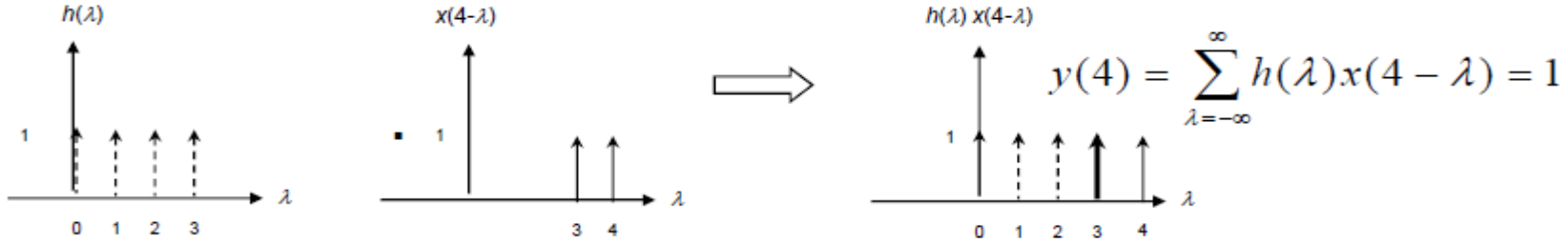


Discrete-Time Convolution: Example 2

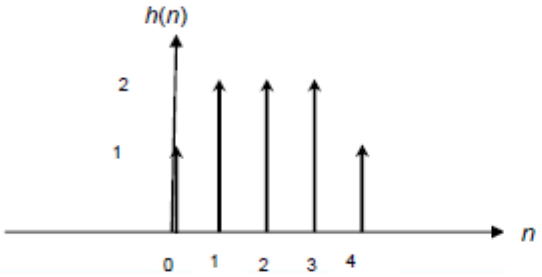
iV) At n=3



v) At n=4



Finally



Performance of Convolution

- Convolution can be performed in numerous ways. Some of those are:
  - Direct-evaluation
  - Graphical method
  - Slide-rule method
  - Fourier transform, and
  - Z-transform

Deconvolution

- Unwanted convolution is an inherent problem in transferring analog information. For instance, all of the following can be modelled as a convolution: image blurring in a shaky camera, echoes in long distance telephone calls, the finite bandwidth of analog sensors and electronics, etc. Deconvolution is the process of filtering a signal to compensate for an undesired convolution.

- The goal of deconvolution is to recreate the signal as it existed *before* the convolution took place. This usually requires the characteristics of the convolution (i.e., the impulse or frequency response) to be known. This can be distinguished from **blind deconvolution**, where the characteristics of the parasitic convolution are *not* known. Blind deconvolution is a much more difficult problem that has no general solution, and the approach must be tailored to the particular application.

Deconvolution

Deconvolution is nearly impossible to understand in the *time domain*, but quite straightforward in the *frequency domain*. Each sinusoid that composes the original signal can be changed in amplitude and/or phase as it passes through the undesired convolution. To extract the original signal, the deconvolution filter must *undo* these amplitude and phase changes.

For example, if the convolution changes a sinusoid's amplitude by 0.5 with a 30 degree phase shift, the deconvolution filter must amplify the sinusoid by 2.0 with a -30 degree phase change.

Deconvolution

Reverse of Convolution

$$x_t = w_t * e_t \quad \rightarrow \quad e_t = x_t * w^{-1}_t$$

=> Inverse Filtering

- Aim of Deconvolution
  1. Theoretical: Reconstruction of the Reflectivity function
  2. Practical:
     - Shorting of the Signal
     - Suppression of Noise
     - Suppression of Multiples

Deconvolution

In mathematics, **deconvolution** is an algorithm-based process used to reverse the effects of convolution on recorded data. The concept of deconvolution is widely used in the techniques of signal processing and image processing. Because these techniques are in turn widely used in many scientific and engineering disciplines, deconvolution finds many applications.

In general, the object of deconvolution is to find the solution of a convolution equation of the form:

$$f * g = h$$

Usually, h is some recorded signal, and f is some signal that we wish to recover, but has been convolved with some other signal g before we recorded it. The function g might represent the transfer function of an instrument or a driving force that was applied to a physical system. If we know g, or at least know the form of g, then we can perform deterministic deconvolution. However, if we do not know g in advance, then we need to estimate it. This is most often done using methods of statistical estimation.

In physical measurements, the situation is usually closer to

$$(f * g) + \varepsilon = h$$

In this case ε is noise that has entered our recorded signal. If we assume that a noisy signal or image is noiseless when we try to make a statistical estimate of g, our estimate will be incorrect. In turn, our estimate of f will also be incorrect. The lower the signal-to-noise ratio, the worse our estimate of the deconvolved signal will be. That is the reason why inverse filtering the signal is usually not a good solution. However, if we have at least some knowledge of the type of noise in the data (for example, white noise), we may be able to improve the estimate of f through techniques such as Wiener deconvolution.

The foundations for deconvolution and time-series analysis were largely laid by Norbert Wiener of the Massachusetts Institute of Technology in his book *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (1949).[2] The book was based on work Wiener had done during World War II but that had been classified at the time. Some of the early attempts to apply these theories were in the fields of weather forecasting and economics.

Deconvolution

Deconvolution is a key area in signal and image processing. It is used for objectives in signal and image processing that include the following:

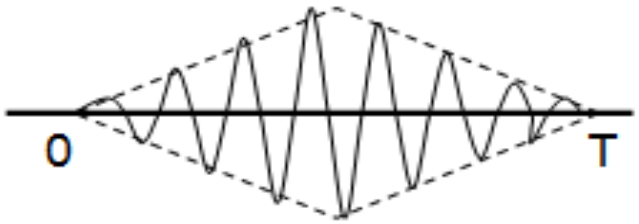
1. deblurring,
2. removal of atmospheric seeing degradation,
3. correction of mirror spherical aberration,
4. image sharpening,
5. mapping detector response characteristics to those of another,
6. image or signal zooming, and
7. optimizing display.

Correlation

Transmitted Signal, x(n)

Reflected Signal,

y(n) = x(n-D) + w(n)



Correlation

- A mathematical operation that closely resembles convolution is correlation.
- Just like in convolution, two signal sequences are involved in correlation.
- Correlation is a measure of the similarity between two signals as a function of time shift between them.
- Correlation is maximum when two signals are similar in shape, and are in phase (or 'unshifted' with respect to each other).
- Correlation is often encountered in Radar, Sonar, Digital communications etc
- Correlating two different signals is called Cross-correlation.
- Correlating a signal with itself is called autocorrelation.

Cross-Correlation

- Cross correlation can be used to identify a signal by comparison with a library of known reference signals.

Definition: Crosscorrelation

The crosscorrelation between two signals x[n] and y[n] is given by:

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x[n]y[n-l] \qquad (1)$$

where the time shift l is called the lag.

OR

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x[n+l]y[n] \qquad (2)$$

Cross-Correlation

- If we reverse the roles of x[n] and y[n] in (1) and (2) and hence reverse the order of indices xy, we obtain the cross correlation sequence

$$r_{yx}(l) = \sum_{n=-\infty}^{\infty} y[n]x[n-l] \qquad (3)$$

Or, equivalently

$$r_{yx}(l) = \sum_{n=-\infty}^{\infty} y[n+1]x[n] \qquad (4)$$

By comparing (1) and (4) or (2) and (3), we conclude that (5)

$$r_{xy}(l) = r_{yx}(-l)$$

Therefore, $r_{yx}[l]$ is simply the folded version of $r_{xy}[l]$.

Cross-Correlation

- **Example:** Determine the crosscorrelation sequence of the sequences

  $x[n] = \{\ldots, 0, 0, 2, -1, 3, 7, \underline{1}, 2, -3, 0, 0, \ldots\}$

  $y[n] = \{\ldots, 0, 0, 1, -1, 2, -2, \underline{4}, 1, -2, 5, 0, 0, \ldots\}$

Solution: The only difference in convolution and crosscorrelation is that in crosscorrelation we don't need to fold the sequence. Otherwise all steps are same.

Cross-Correlation

X[n]				2	-1	3	7	1	2	-3					
y[n+3]	1	-1	2	-2	4	1	-2	5							$R_{xy}(-3)=-14$
y[n+2]		1	-1	2	-2	4	1	-2	5						$R_{xy}(-2)=33$
Y[n+1]			1	-1	2	-2	4	1	-2	5					$R_{xy}(-1)=0$
Y[n]				**1**	**-1**	**2**	**-2**	**4**	**1**	**-2**	**5**				$R_{xy}(0)=7$
Y[n-1]					1	-1	2	-2	4	1	-2	5			$R_{xy}(1)=13$
Y[n-2]						1	-1	2	-2	4	1	-2	5		$R_{xy}(2)=-18$
Y[n-3]							1	-1	2	-2	4	1	-2	5	$R_{xy}(3)=16$

$$R_{xy}(l)=\{10, -9, 19, 36, -14, 33, 0, \underline{7}, 13, -18, 16, -7, 5, -3, 0\}$$

Auto-Correlation

- Autocorrelation is the special case of crosscorrelation, in which one signal is compared with its time shifted version.

Definition: Autocorrelation

The autocorrelation of a real signal x[n] is given by:

$$r_{xx}(l) = \sum_{n=-\infty}^{\infty} x[n]x[n-l]$$

where the time shift m is called the lag.

Or equivalently as, $r_{xx}(l) = \sum_{n=-\infty}^{\infty} x[n+l]x[n]$

Properties of Auto-correlation and Cross-correlation Sequences

Let us assume that we have two sequences x[n] and y[n] with finite energy from which we form the linear combination

a x[n] + b y[n-l]

where a and b are arbitrary constants and k is some time shift. The energy in this signal is

$$\sum_{n=-\infty}^{\infty}\left[ax[n]+by[n-l]\right]^2 = a^2\sum_{n=-\infty}^{\infty}x^2[n]+b^2\sum_{n=-\infty}^{\infty}y^2[n-l]$$

$$+2ab\sum_{n=-\infty}^{\infty}x[n]y[n-l] = a^2r_{xx}(0)+b^2r_{yy}(0)+2abr_{xy}(l)$$

Note that $r_{xx}(0) = E_x$ and $r_{yy}(0) = E_y$, the energies of x[n] and y[n] respectively.

It is obvious that

$$a^2r_{xx}(0)+b^2r_{yy}(0)+2abr_{xy}(k)\geq 0$$

Properties of Auto-correlation and Cross-correlation Sequences

Now, assuming that b ≠ 0, we divide the above equation by b^2 to obtain

$$r_{xx}(0)\left(\frac{a}{b}\right)^2 + 2r_{xy}[l]\left(\frac{a}{b}\right) + r_{yy}(0) \geq 0 \qquad (3)$$

This is a quadratic equation. Since the quadratic is non-negative, its discriminant must be non-positive. That is,

$$4\left[r_{xy}^2[l] - r_{xx}(0)\,r_{yy}(0)\right] \leq 0 \qquad (4)$$

Therefore, the crosscorrelation sequence satisfies the condition that

$$\left|r_{xy}(l)\right| \leq \sqrt{r_{xx}(0)\,r_{yy}(0)} = \sqrt{E_x E_y} \qquad (5)$$

In the special case i-e in Autocorrelation where y[n] = x[n], (5) reduces to

$$\left|r_{xx}(l)\right| \leq r_{xx}(0) = E_x \qquad (6)$$

Properties of Auto-correlation and Cross-correlation Sequences

- This means that the autocorrelation sequence of a signal attains its maximum value at zero lag.
- If any one or both of the signals involved are scaled, the shape of the cross correlation sequence does not change; only the amplitudes of the crosscorrelation sequence are scaled accordingly.

It is often desirable in practice to normalize the autocorrelation and crosscorrelation sequences to the range from -1 to 1. The normalized autocorrelation sequence is defined as,

$$\rho_{xx}(l) = \frac{r_{xx}(l)}{r_{xx}(0)} \qquad (7)$$

Similarly, we define the normalized crosscorrelation sequence

$$\rho_{xy}(l) = \frac{r_{xy}(l)}{\sqrt{r_{xx}(0) r_{yy}(0)}} \qquad (8)$$

- One other important property of autocorrelation is that its an even function.
  - We know that $r_{xy}(l)=r_{yx}(-l)$, so
  - If we make x[n]=y[n] (autocorrelation) the condition becomes $r_{xx}(l)=r_{xx}(-l)$, Hence the autocorr: function is an even function.

Difference between Convolution and Correlation

- Convolution is usually between a signal and a filter; we think of it as a system with a single input and stored coefficients.
- Crosscorrelation is usually between two signals; we think of a system with two inputs and no stored coefficients.