Digital Signal Processing

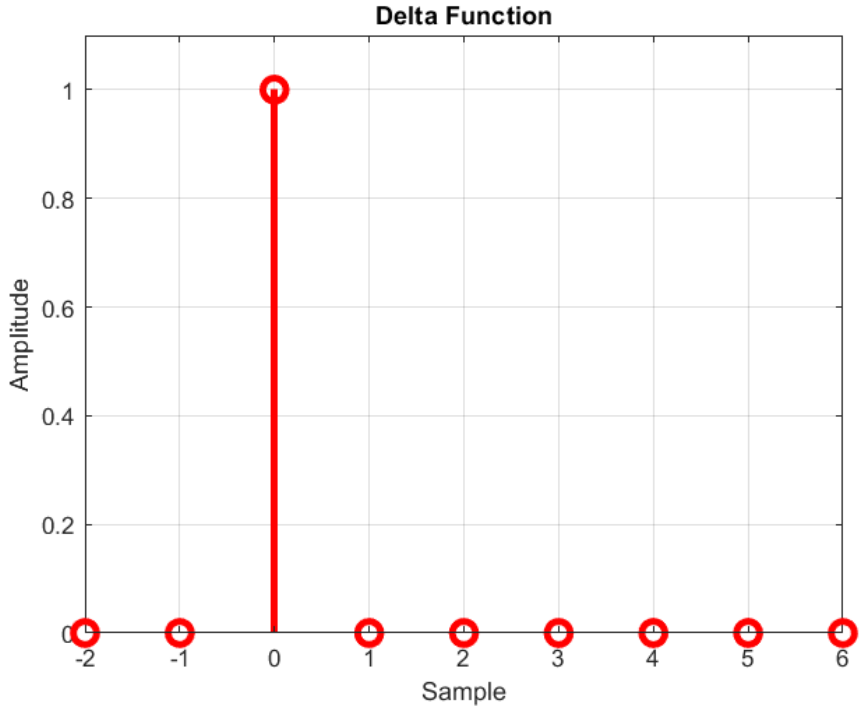
Convolution

Today's Topics

- Delta Function and Impulse response
- Convolution and Kernels
- Input Side Algorithm
- Output Side Algorithm

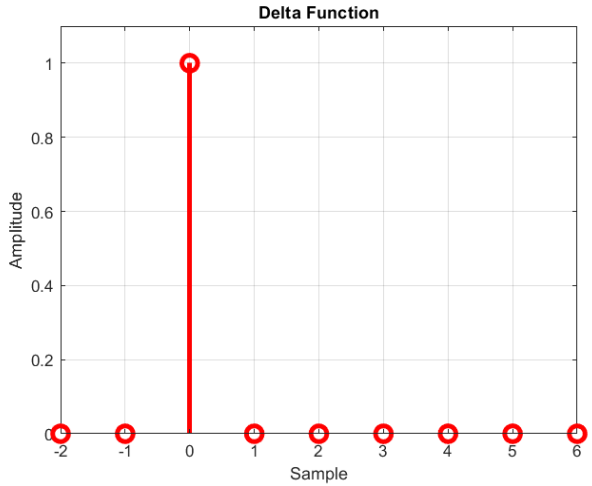
Delta Function

- Delta Function $\delta[0]$ – a unit impulse at zero. Amplitude normalized to 1.

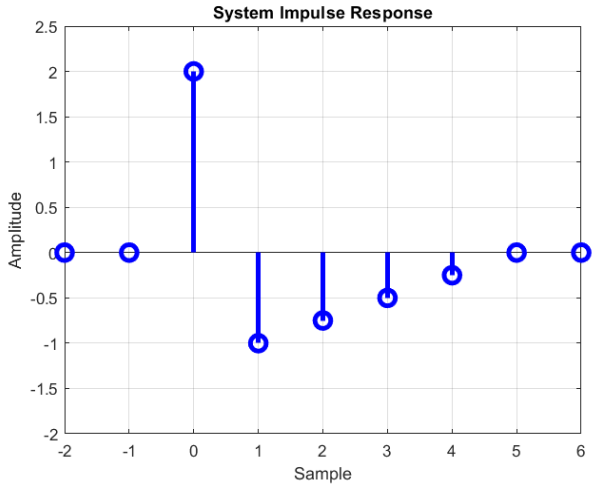


Impulse Response

- The impulse response is the signal that exits the system when the input to the system is a delta function.

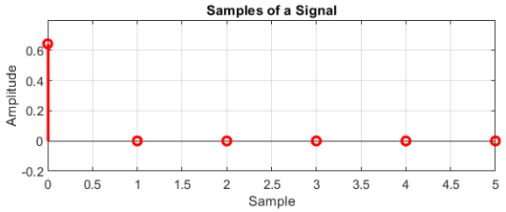


$x[n]$ → Linear System → $y[n]$

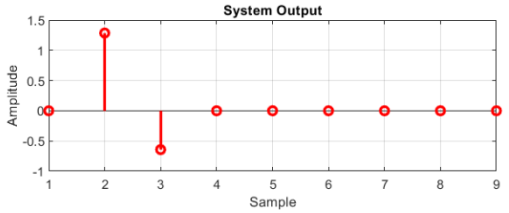


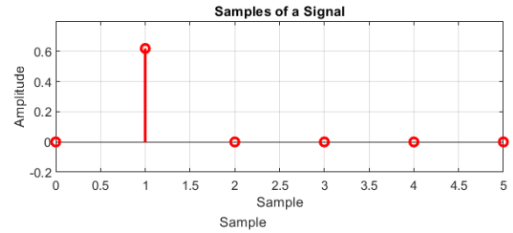
Impulse Response

- The input can be represented by a series of amplitude scaled and time shifted delta functions. (Homogeneity and Additivity plus shift invariance.)
- The output is amplitude scaled and time shifted responses to the input impulses.

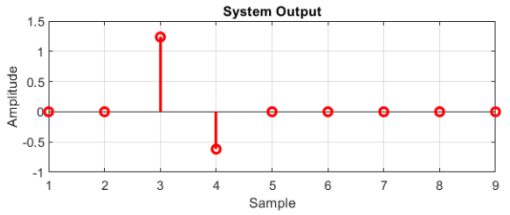


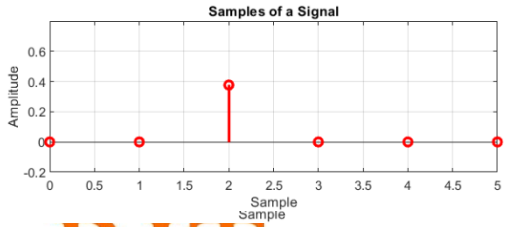
$x_0[n]$ → Linear System → $y_0[n]$



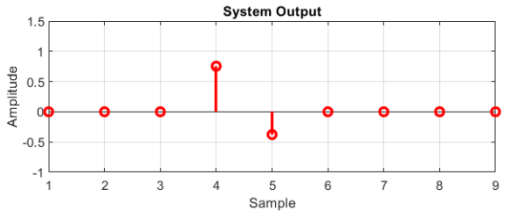


$x_1[n]$ → Linear System → $y_1[n]$



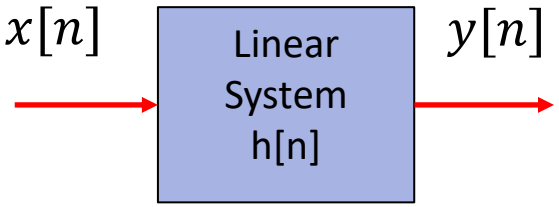


$x_2[n]$ → Linear System → $y_2[n]$



Convolution

- Convolution : Mathematical operation on input signal and system impulse response that results in the output signal



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

$*$ denotes convolution

- When the system is a filter, the impulse response is sometimes called a filter kernel, or just a kernel.

Convolution

- Mathematically
  - Continuous convolution

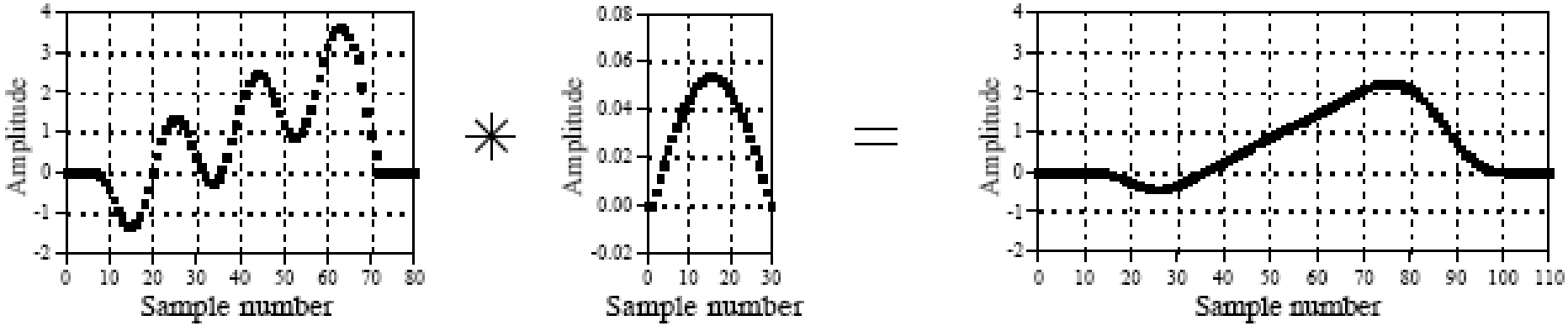
$$f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau) g(t-\tau) d\tau$$

  - Discrete Convolution

$$f[n] * g[n] = \sum_{-\infty}^{\infty} f[m] g[n-m]$$

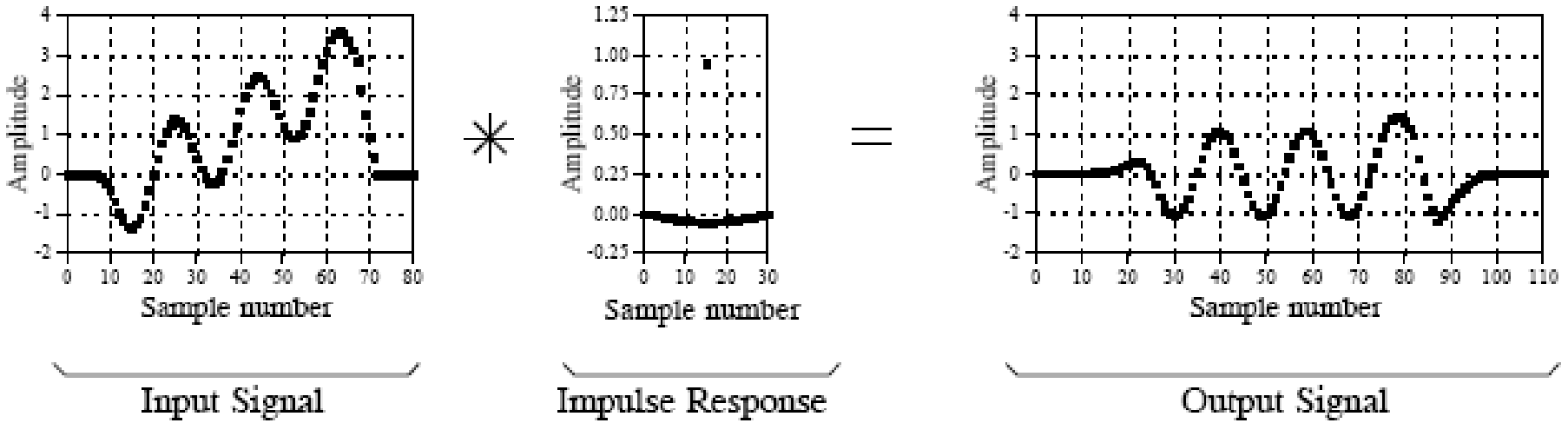
Example Kernels for Filters

- An input signal consists of a high frequency sinewave and a ramp
- A low pass kernel (impulse response) will filter out the higher frequency sine portion of the waveform and leave the ramp



Example Kernels for Filters

- A high pass kernel (impulse response) will remove the slower changing parts of the signal and leave the faster signals (higher frequency)



- The Ramp is removed leaving just the sine wave

Computing the Convolution

- There are two ways to look at computing the convolution
- Input side algorithm
- Output side algorithm

Input Side Algorithm

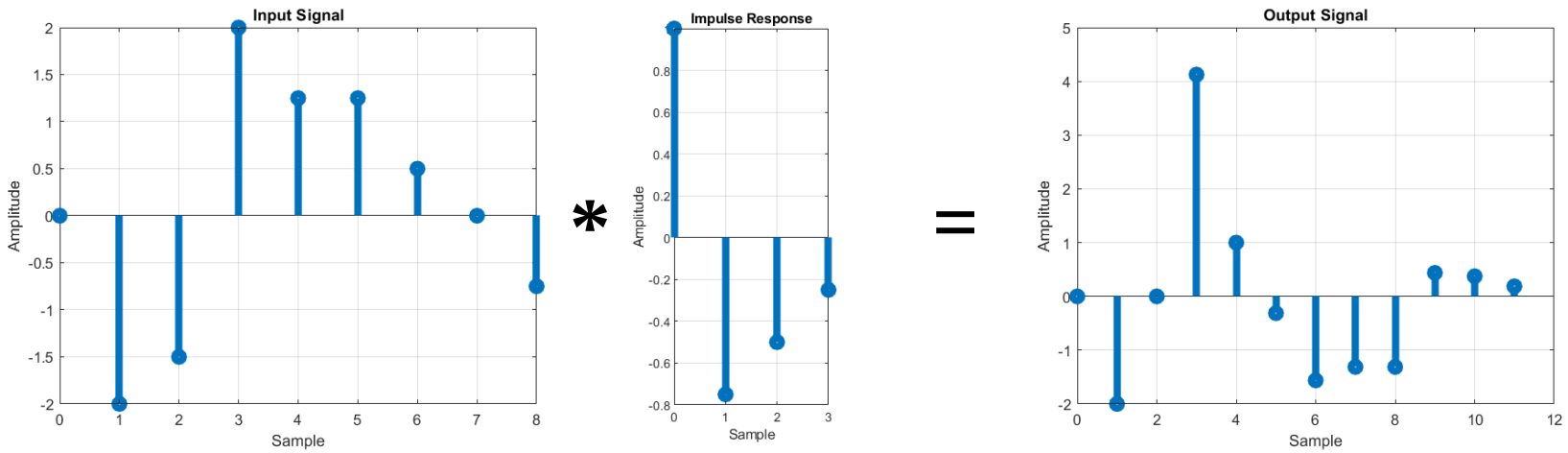
- Input Side Algorithm analyzes how each sample in the input contributes to many points in the output.
- Advantage -- Conceptually easier to understand
- Working from the input to the output

Output Side Algorithm

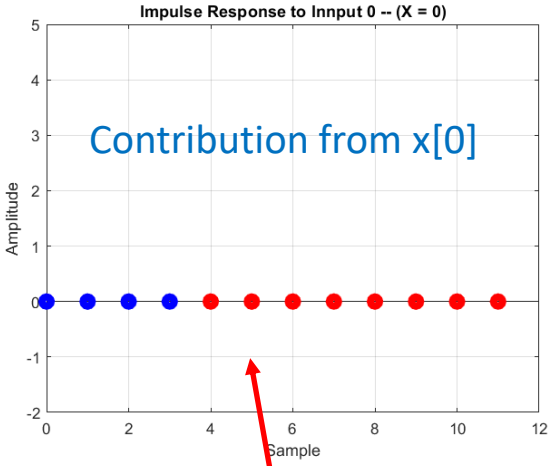
- Output Side Algorithm examines how each sample in the output receives information from many points in the input signal.
- More representative of how the algorithm is implemented in code.
- Working from the output to the input

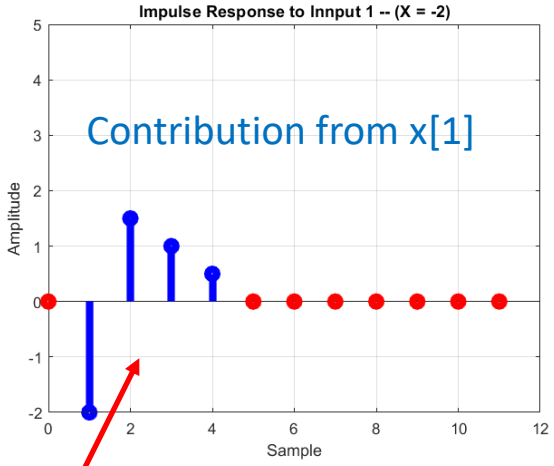
Input Side Algorithm

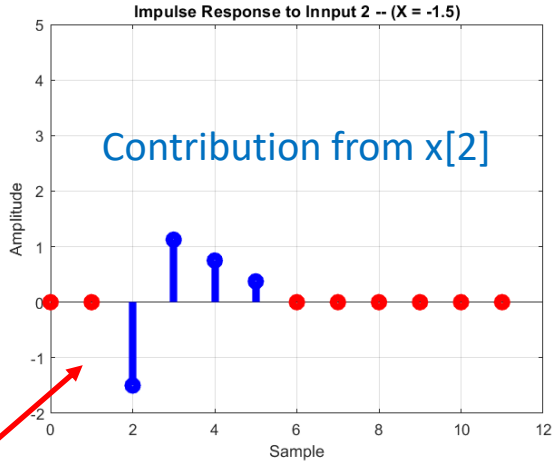
- Decompose the input, x[n] , into impulses.
- Pass each impulse through the system and generate scaled and time shifted copies of impulse response h[n]
- Synthesize the output (add them up)

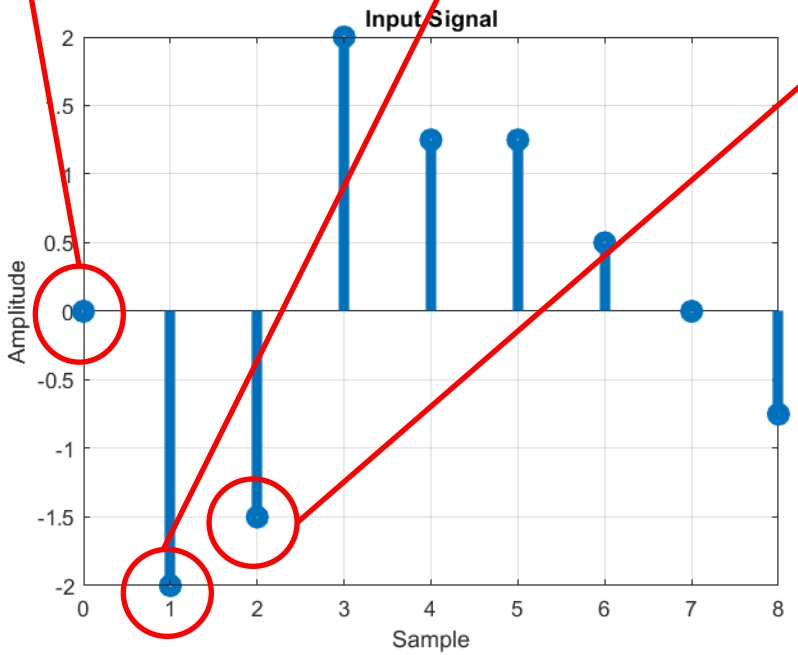


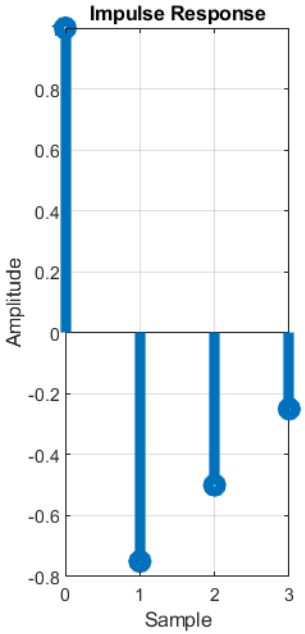
Input Side Algorithm



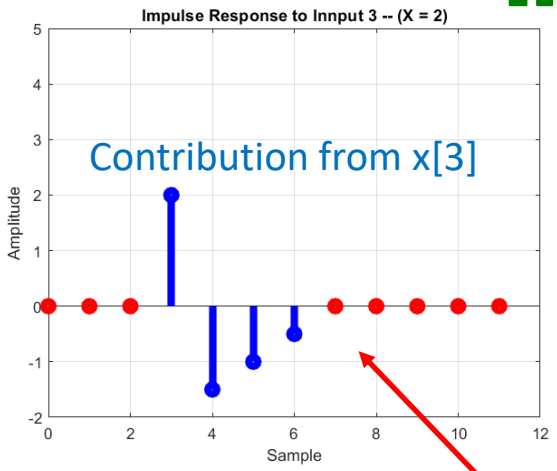


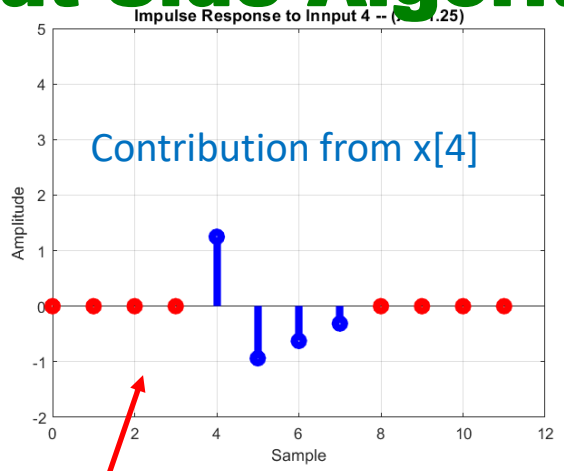


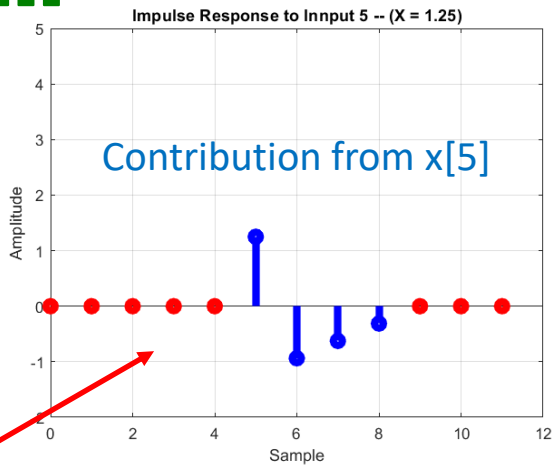


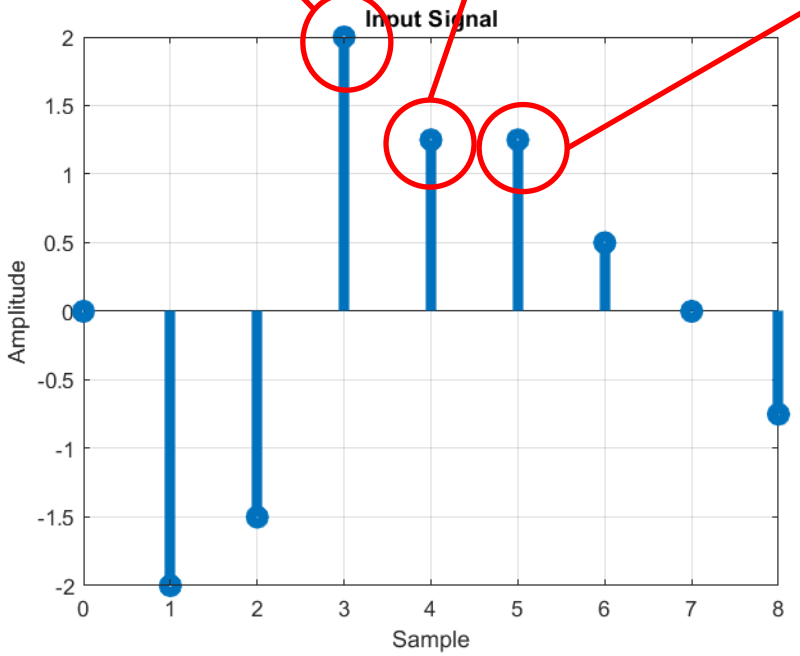


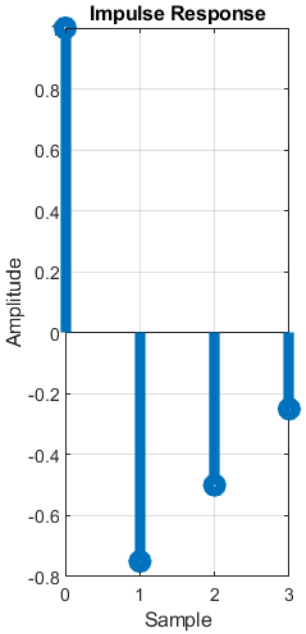
Input Side Algorithm



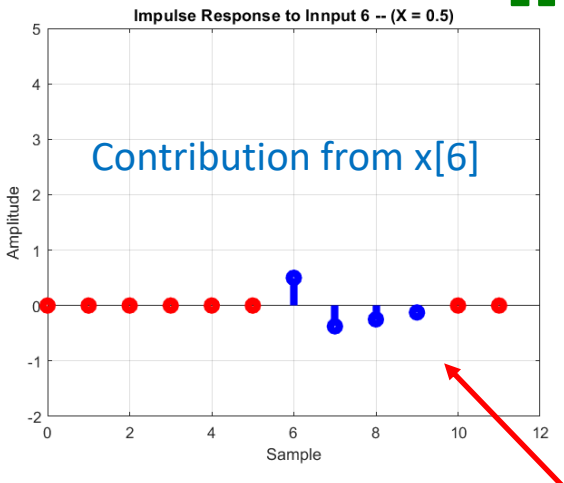


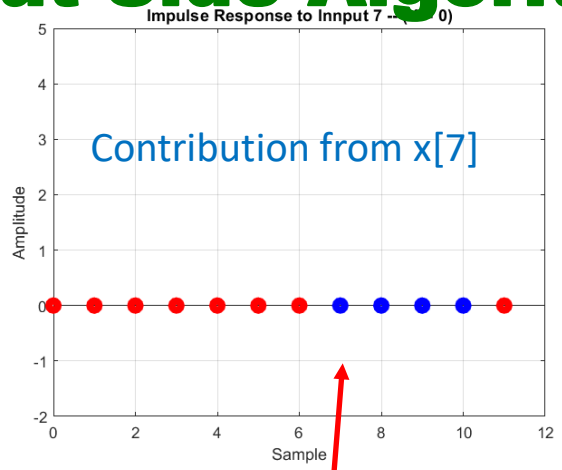


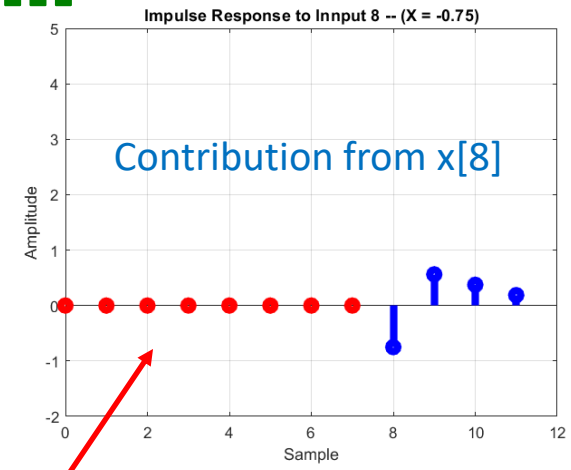


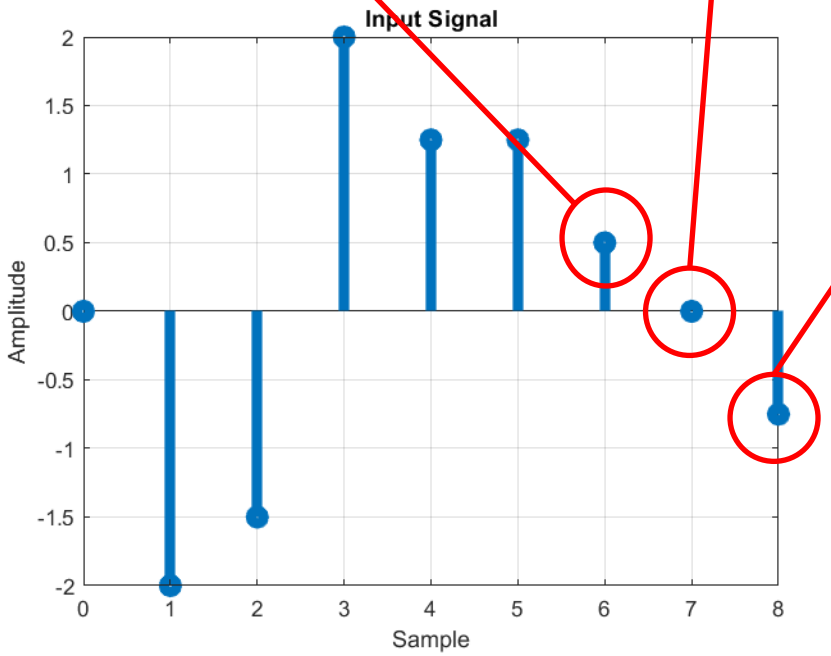


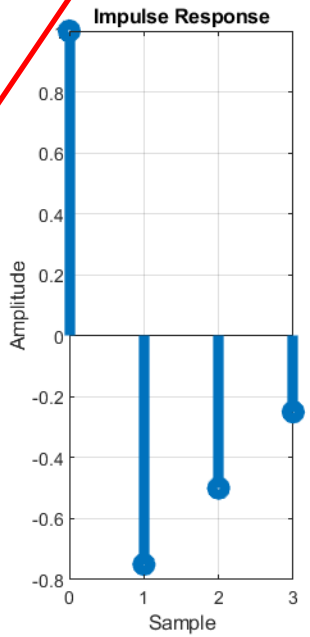
Input Side Algorithm



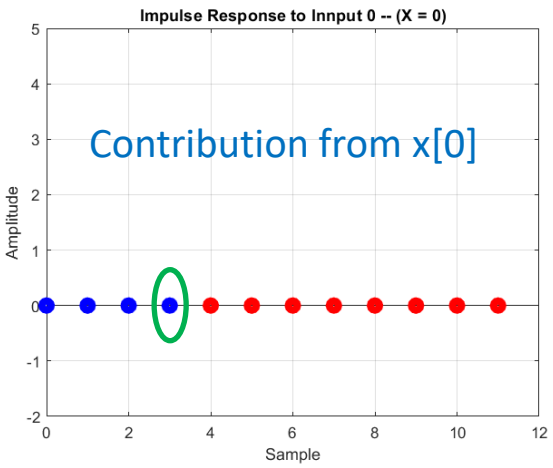


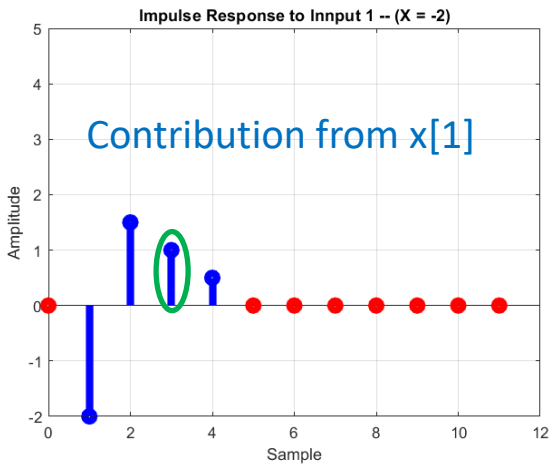


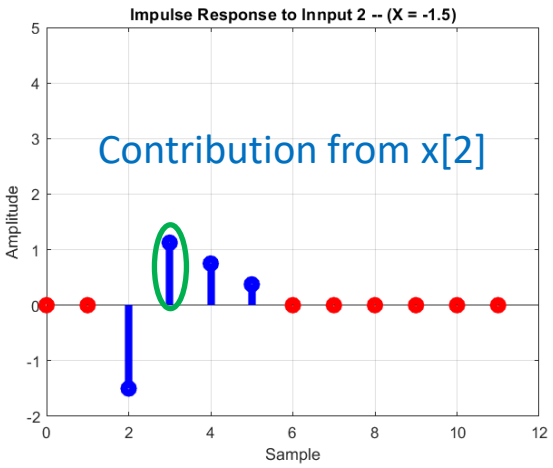


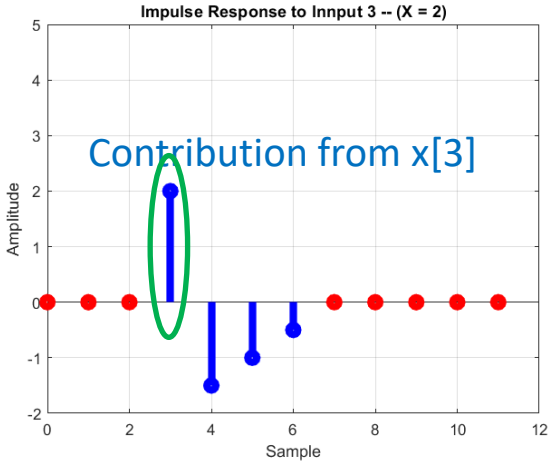


Input Side Algorithm









Using the input side algorithm, to get one output value I need to get the contributions from 4 input impulses to find one output value.

The 4 values highlighted are summed to create y[3]

Convolution Length

- The resulting length of the convolution is:

- Input Length -- M samples
- Impulse Response Length -- N samples
- Output Length -- M+N-1 samples

Input Side Algorithm

For each point in the input
Multiply and sum over the kernel

X[i]– input
H[j] – kernel
Y[i+j] -- Output

```
                                        'The input signal, 81 points
130 DIM H[30]                           'The impulse response, 31 poin
140 DIM Y[110]                          'The output signal, 111 points
150                                     '
160 GOSUB XXXX                          'Mythical subroutine to load X[
170                                     '
180 FOR I% = 0 TO 110                   'Zero the output array
190   Y(I%) = 0
200 NEXT I%
210                                     '
220 FOR I% = 0 TO 80                    'Loop for each point in X[ ]
230   FOR J% = 0 TO 30    'Loop for each point in H[ ]
240     Y[I%+J%] = Y[I%+J%] + X[I%]*H[J%]
250   NEXT J%
260 NEXT I%                             '(remember, * is multiplication
```

Loop over Input

Kernel

Convolution Example

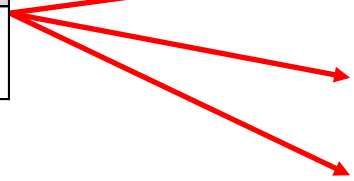
- Two signals, a[n] and b[n], are defined by:
  - $a[n]$: $2, 1, 0$
  - $b[n]$: $1, -1, -2, 1$

- Calculate $y[n] = a[n] * b[n]$
- Steps:
  - Decompose $a[n]$ using impulses
  - Convolve each impulse with $b[n]$ by scaling and time shifting
  - Synthesize the result by adding

Impulse Decomposition

- Decompose the input into individual impulses

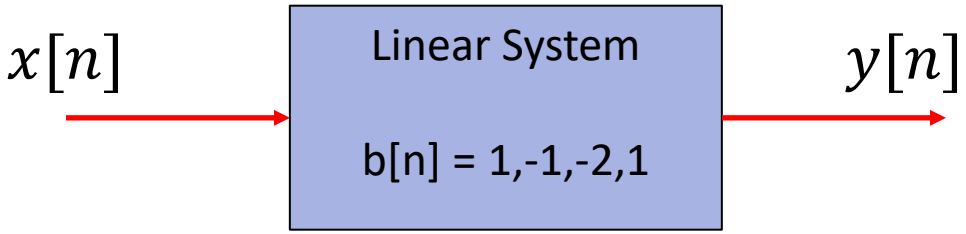
n	0	1	2
a[n]	2	1	0



n	0	1	2
a1[n]	2	0	0
a2[n]	0	1	0
a3[n]	0	0	0

Apply the System Impulse Response to Each Input

- Find the response of the system to each impulse



n	0	1	2
a1[n]	2	0	0
a2[n]	0	1	0
a3[n]	0	0	0

n	0	1	2	3	4	5
y1[n]	2	-2	-4	2	0	0
y2[n]	0	1	-1	-2	1	0
y3[n]	0	0	0	0	0	0

Sum Each Individual Response

- Sum the columns to get the result

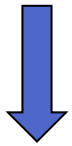
	n	0	1	2	3	4	5
	y1[n]	2	-2	-4	2	0	0
	y2[n]	0	1	-1	-2	1	0
	y3[n]	0	0	0	0	0	0
SUM	y[n]	2	-1	-5	0	1	0

Convolution Result

n	0	1	2
a[n]	2	1	0

*

n	0	1	2	3
b[n]	1	-1	-2	1



y[n]	2	-1	-5	0	1	0

- Note – $a[n]$ has length M = 3, $b[n]$ has length N = 4
- Note the length of the result is M+N-1 = 6

Convolution
In Class Problem

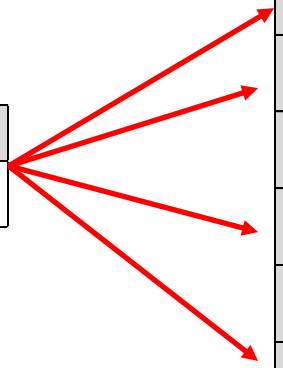
- Two signals, a[n] and b[n], are defined by:
  - $a[n]$: $1, 0, 0, 2, 1, 0$
  - $b[n]$: $0, -1, -2, 0, 0, 1$

- Calculate $y[n] = a[n] * b[n]$
- Decompose $a[n]$ using impulses
- Convolve each impulse with $b[n]$ by scaling and time shifting
- Synthesize the result by adding

Impulse Decomposition

- Decompose the input into individual impulses

n	0	1	2	3	4	5
a[n]	1	0	0	2	1	0

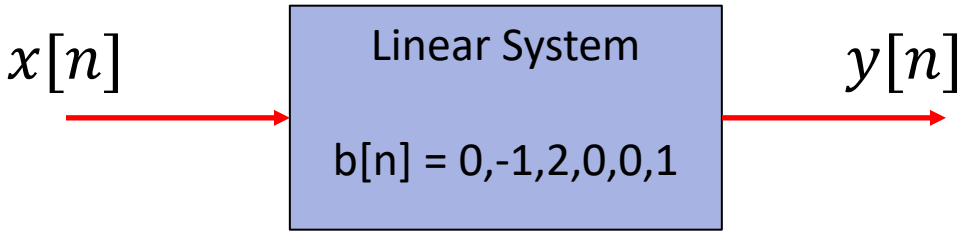


n	0	1	2	3	4	5
a1[n]	1	0	0	0	0	0
a2[n]	0	0	0	0	0	0
a3[n]	0	0	0	0	0	0
a4[n]	0	0	0	2	0	0
a5[n]	0	0	0	0	1	0
a6[n]	0	0	0	0	0	0

NOPRINT

Apply the System Impulse Response to Each Input

- Find the response of the system to each impulse



n	0	1	2	3	4	5
a1[n]	1	0	0	0	0	0
a2[n]	0	0	0	0	0	0
a3[n]	0	0	0	0	0	0
a4[n]	0	0	0	2	0	0
a5[n]	0	0	0	0	1	0
a6[n]	0	0	0	0	0	0

n	0	1	2	3	4	5	6	7	8	9	10
y1[n]	0	-1	-2	0	0	1	0	0	0	0	0
y2[n]	0	0	0	0	0	0	0	0	0	0	0
y3[n]	0	0	0	0	0	0	0	0	0	0	0
y4[n]	0	0	0	0	-2	-4	0	0	2	0	0
y5[n]	0	0	0	0	0	-1	-2	0	0	1	0
y6[n]	0	0	0	0	0	0	0	0	0	0	0

NOPRINT

Sum Each Individual Response

- Sum the columns to get the result

	n	0	1	2	3	4	5	6	7	8	9	10
	y1[n]	0	-1	-2	0	0	1	0	0	0	0	0
	y2[n]	0	0	0	0	0	0	0	0	0	0	0
	y3[n]	0	0	0	0	0	0	0	0	0	0	0
	y4[n]	0	0	0	0	-2	-4	0	0	2	0	0
	y5[n]	0	0	0	0	0	-1	-2	0	0	1	0
	y6[n]	0	0	0	0	0	0	0	0	0	0	0
SUM	y[n]	0	-1	-2	0	-2	-4	-2	0	2	1	0

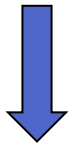
NOPRINT

Convolution Result

n	0	1	2	3	4	5
a[n]	1	0	0	2	1	0

*

n	0	1	2	3	4	5
b[n]	0	-1	-2	0	0	1



y[n]	0	-1	-2	0	-2	-4	-2	0	2	1	0

- Note – a has length M = 6, b has length N = 6
- Note the length of the result is M+N-1 = 11

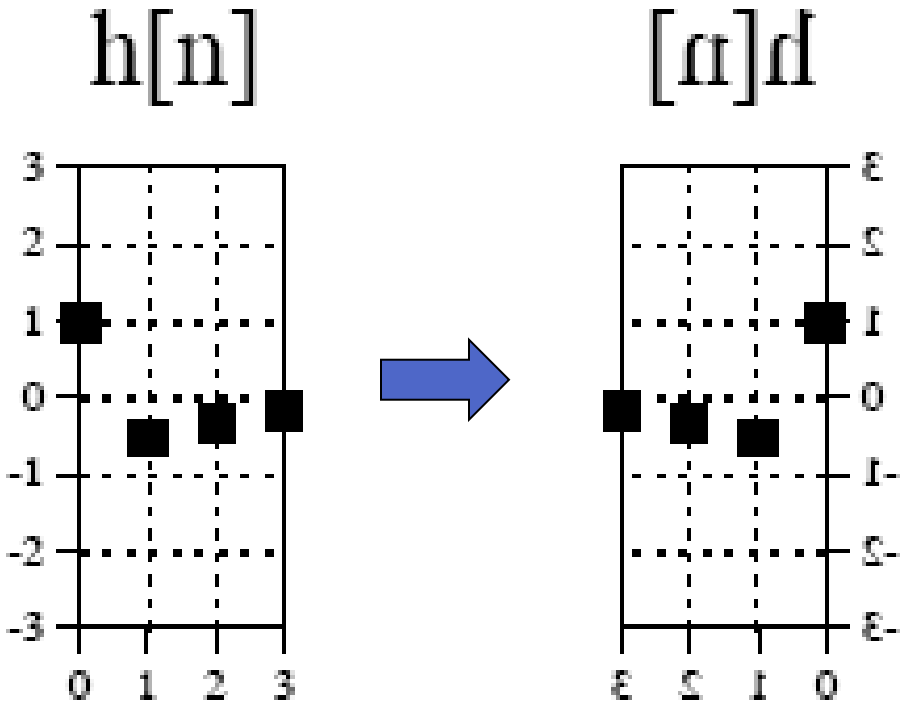
NOPRINT

Output Side Algorithm

- Computes a final value of the output after each iteration
  - $y[n]$ is a combination of input signals
- Can think of this as a machine to produce each sample of the output from the input sequence.

Flip the Impulse Response

- The impulse response is “flipped” in time



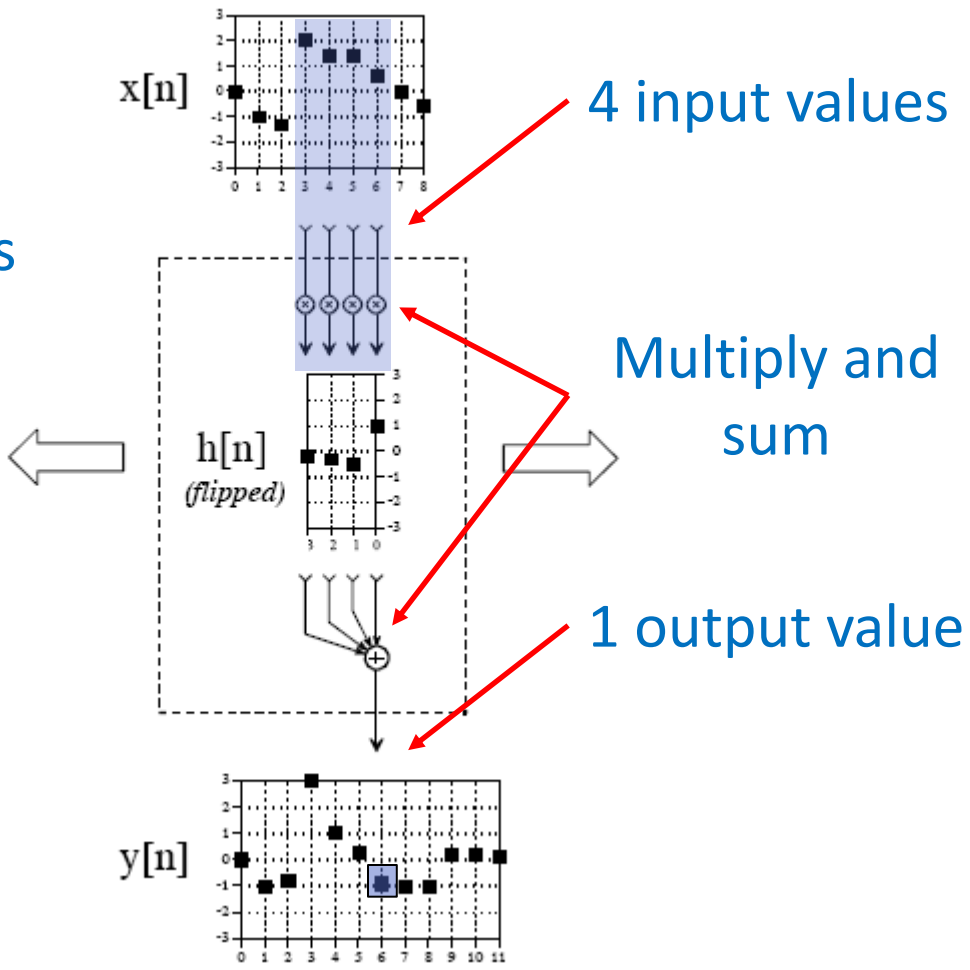
Output Side Algorithm

- $h[n]$ is "flipped" in time

The convolution "machine" takes four values of the input

Multiplies them by the "flipped" $h[n]$ then sums them

Result is one output value in time



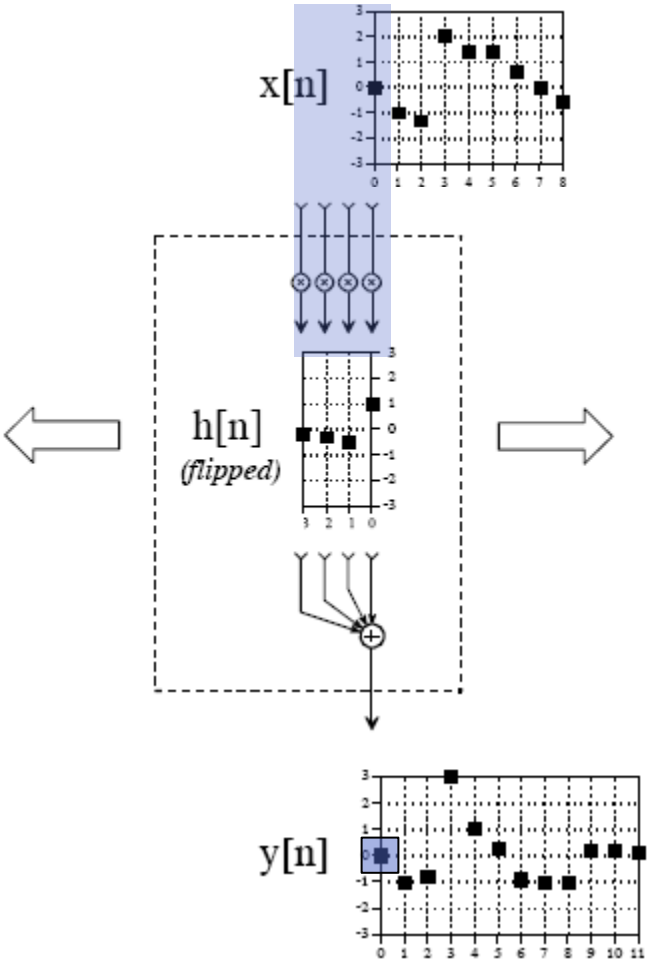
Computing $y[0]$

The convolution "machine" output is aligned with $y[0]$

There are no input values for some values of the kernel

Set these input values to 0

Multiply x[0] h[0]



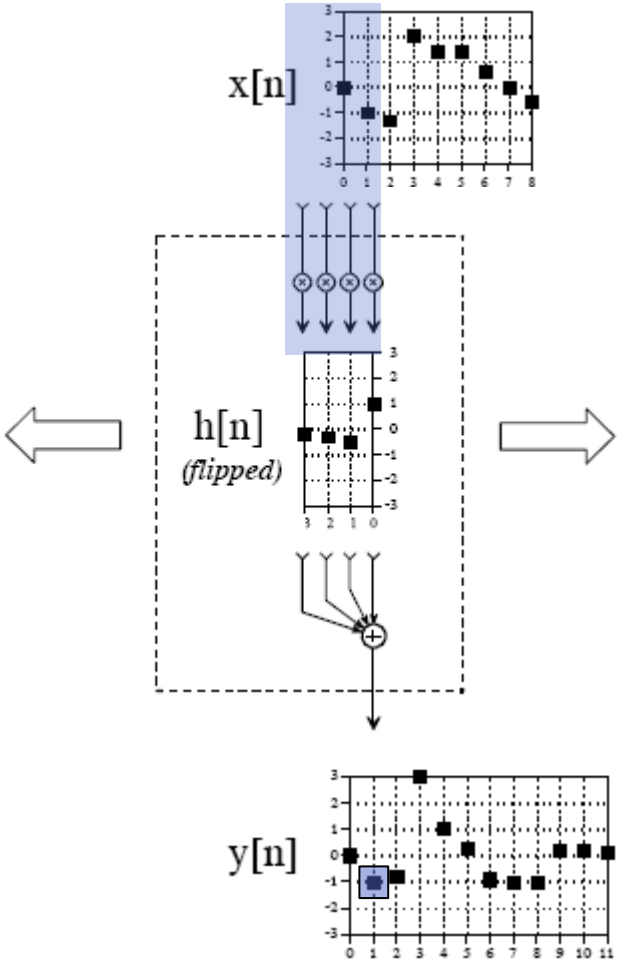
Computing $y[1]$

The convolution "machine" moves one place to the right aligned with $y[1]$

There are no input values for some values of the kernel

Set these input values to 0

Multiply x[1] h[0] and x[0]h[1] and sum

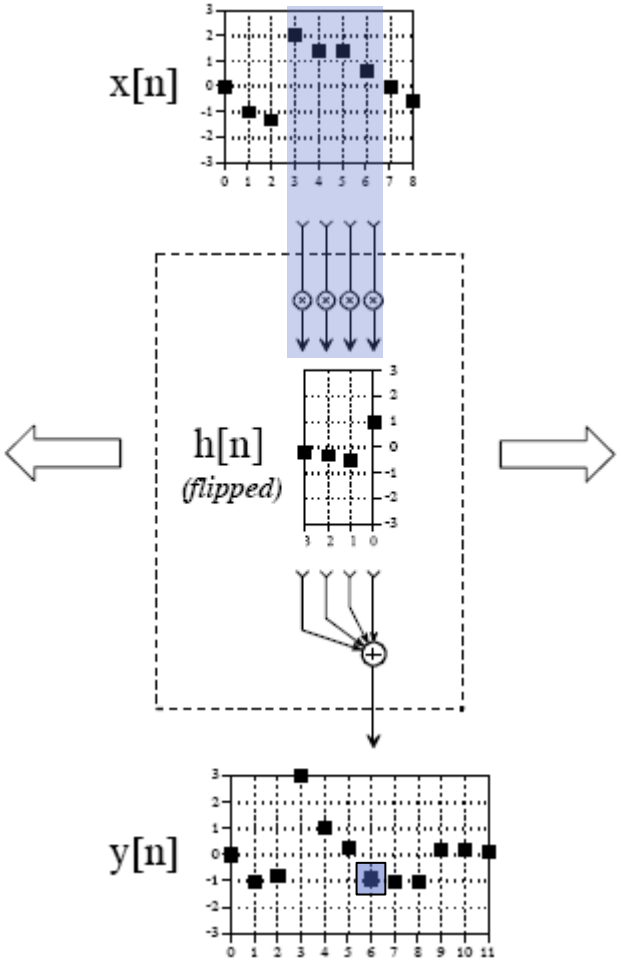


Computing $y[6]$

The convolution "machine" moves so that the output is aligned with $y[6]$

Multiply x[6]h[0], x[5]h[1], x[4]h[2], x[3]h[1]

Sum the values for the output



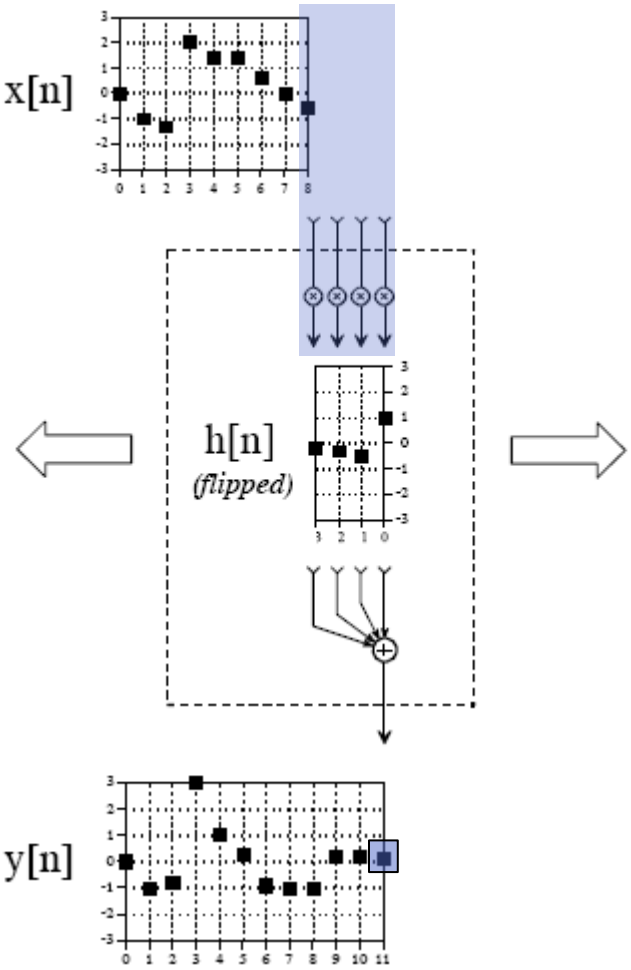
Computing $y[11]$

The convolution "machine" moves so that the output is aligned with $y[11]$

Some values of the input don't exist so replace with 0's

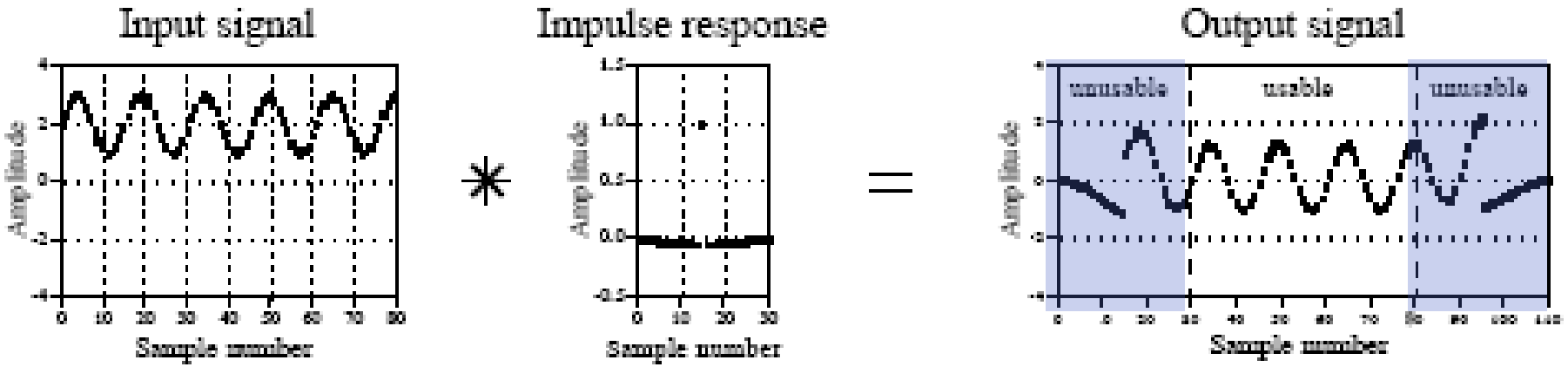
Multiply x[8]h[3]

Sum the values for the output



End Effects

- The effect of “padding” the input and the output with zero values can cause some of the first and last outputs to be “odd”
- These are referred to as “end effects”
- Can be unusable outputs



Output Side Algorithm

For each point in the output
Multiply and sum the kernel over the input

$$y[i] = \sum_{j=0}^{M-1} h[j]x[i-j]$$

X[i]– input
H[j] – kernel
Y[i+j] -- Output

Loop over Output

Kernel

```
120 DIM X[80]                    'The input signal, 81 poir
130 DIM H[30]                    'The impulse response, 3
140 DIM Y[110]                   'The output signal, 111 p
150                              '
160 GOSUB XXXX                   'Mythical subroutine to lc
170                              '
180 FOR I% = 0 TO 110            'Loop for each point in Y|
190   Y[I%] = 0                  'Zero the sample in the o
200   FOR J% = 0 TO 30           'Loop for each point in H|
210     IF (I%-J% < 0)           THEN GOTO 240
220     IF (I%-J% > 80)          THEN GOTO 240
230     Y(I%) = Y(I%) + H(J%) * X(I%-J%)
240   NEXT J%
250 NEXT I%
```

In Class Example

- A system has an impulse response
- $\mathrm{h[n]} = 3, -2, 2, 1$

- Calculate the output of the system in response to the input signal.

- $y[n] = 4, 1, -2, -1, 0, 1, 3, 4, 3, 2$

- Use the output side algorithm

Flip the Impulse Response

- The impulse response is

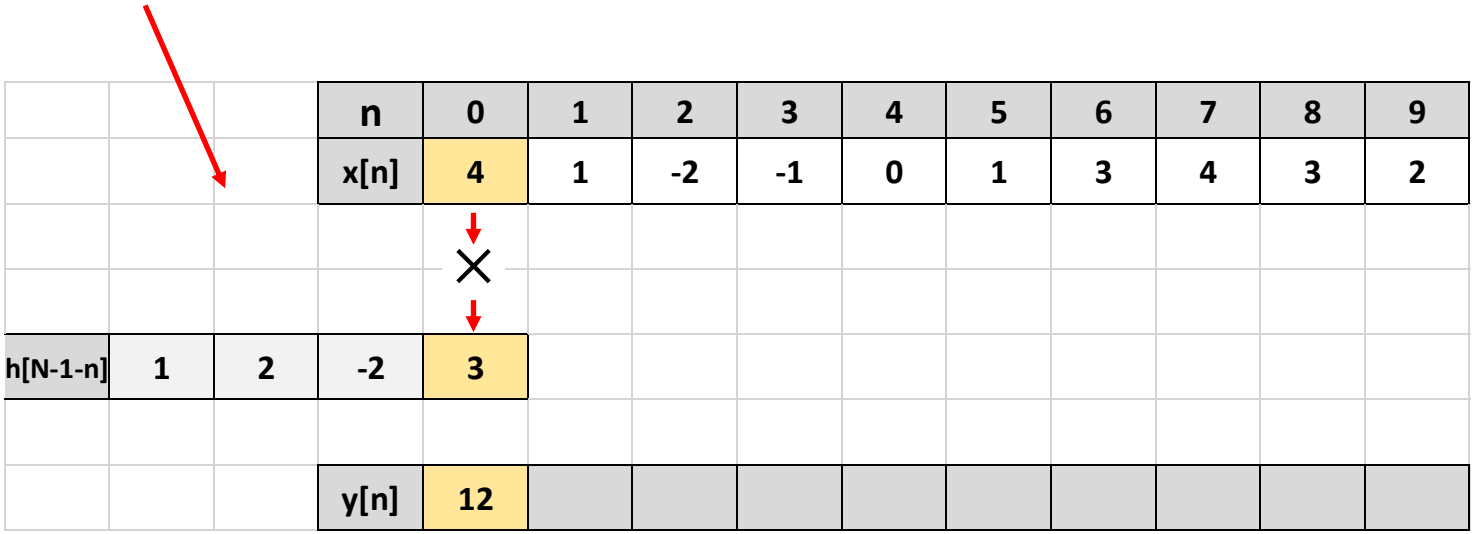
$$h[n] = [3, -2, 2, 1]$$

- The flipped impulse response is

$$h[N-1-n] = [1, 2, -2, 3]$$

Place the "machine" to Compute $y[0]$

The input sequence is padded with 0's



			n	0	1	2	3	4	5	6	7	8	9
			x[n]	4	1	-2	-1	0	1	3	4	3	2
				×									
h[N-1-n]	1	2	-2	3									
			y[n]	12									

Compute the output by multiplying the values in columns and sum them together

$$y[0] = 4 \times 3 = 12$$

$y[1]$

		n	0	1	2	3	4	5	6	7	8	9
		x[n]	4	1	-2	-1	0	1	3	4	3	2
h[N-1-n]	1	2	-2	3								
		y[n]	12	-5								

Move the impulse response 1 sample to the right

$$y[1] = (4)(-2) + (1)(3) = -5$$

The input sequence is padded with 0's

$y[2]$ *and* $y[3]$

	n	0	1	2	3	4	5	6	7	8	9
	x[n]	4	1	-2	-1	0	1	3	4	3	2
		×	×	×							
h[N-1-n]	1	2	-2	3							
	y[n]	12	-5	0							

The input sequence is padded with a 0

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

	n	0	1	2	3	4	5	6	7	8	9
	x[n]	4	1	-2	-1	0	1	3	4	3	2
		×	×	×	×						
h[N-1-n]	1	2	-2	3							
	y[n]	12	-5	0	7						

$$y[3] = (4)(1) + (1)(2) + (-2)(-2) + (-1)(3) = 7$$

$y[4]$ *and* $y[5]$

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
		×	×	×	×					
h[N-1-n]		1	2	-2	3					
y[n]	12	-5	0	7	-1					

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

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
			×	×	×	×				
h[N-1-n]			1	2	-2	3				
y[n]	12	-5	0	7	-1	-1				

$$y[5] = (-2)(1) + (-1)(2) + (0)(-2) + (1)(3) = -1$$

$y[6]$ *and* $y[7]$

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
h[N-1-n]				1	2	-2	3			
y[n]	12	-5	0	1	2	-1	6			

$$y[6] = (-1)(1) + (0)(2) + (1)(-2) + (3)(3) = 6$$

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
h[N-1-n]					1	2	-2	3		
y[n]	12	-5	0	1	2	-1	6	8		

$$y[6] = (0)(1) + (1)(2) + (3)(-2) + (4)(3) = 8$$

$y[8]$ *and* $y[9]$

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
						×	×	×	×	
h[N-1-n]						1	2	-2	3	
y[n]	12	-5	0	1	2	-1	6	8	8	

$$y[8] = (1)(1) + (3)(2) + (4)(-2) + (3)(3) = 8$$

n	0	1	2	3	4	5	6	7	8	9
x[n]	4	1	-2	-1	0	1	3	4	3	2
							×	×	×	×
h[N-1-n]							1	2	-2	3
y[n]	12	-5	0	1	2	-1	6	8	7	11

$$y[9] = (3)(1) + (4)(2) + (3)(-2) + (2)(3) = 11$$

$y[10]$ *and* $y[11]$

n	0	1	2	3	4	5	6	7	8	9	
x[n]	4	1	-2	-1	0	1	3	4	3	2	0
								×	×	×	×
h[n]								1	2	-2	3
y[n]	12	-5	0	1	2	-1	6	8	7	11	6

The input sequence is padded with 0's

$$y[10] = (4)(1) + (3)(2) + (2)(-2) + (0)(3) = 6$$

n	0	1	2	3	4	5	6	7	8	9		
x[n]	4	1	-2	-1	0	1	3	4	3	2	0	0
									×	×	×	×
h[N-1-n]									1	2	-2	3
y[n]	12	-5	0	1	2	-1	6	8	7	11	6	7

$$y[11] = (3)(1) + (2)(2) + (0)(-2) + (0)(3) = 7$$

$y[12]$

The input sequence is padded with 0's

n	0	1	2	3	4	5	6	7	8	9			
x[n]	4	1	-2	-1	0	1	3	4	3	2	0	0	0
										×	×	×	×
h[N-1-n]										1	2	-2	3
y[n]	12	-5	0	1	2	-1	6	8	7	11	6	7	2

$$y[12] = (2)(1) + (0)(2) + (0)(-2) + (0)(3) = 2$$

The length of the output sequence is 10 + 4 – 1 = 13 samples

In Class Problem

- A system has an impulse response
- $\mathrm{h[n]} = 2, -1, 3$

- Calculate the output of the system in response to the input signal.

- $y[n] = 3, 2, -1, -2, 0, 1$

- Use the output side algorithm

Flip the Impulse Response

- The impulse response is

$$h[n] = [2, -1, 3]$$

- The flipped impulse response is

$$h[N-1-n] = [3, -1, 2]$$

$y[0]$ *and* $y[1]$

Pad input with 0's

		n	0	1	2	3	4	5
		x[n]	3	2	-1	-2	0	1
h[N-1-n]	3	-1	2					
		y[n]	6					

	n	0	1	2	3	4	5
	x[n]	3	2	-1	-2	0	1
h[N-1-n]	3	-1	2				
	y[n]	6	1				

$y[2]$ *and* $y[3]$

n	0	1	2	3	4	5
x[n]	3	2	-1	-2	0	1
h[N-1-n]	3	-1	2			
y[n]	6	1	5			

n	0	1	2	3	4	5
x[n]	3	2	-1	-2	0	1
	h[N-1-n]	3	-1	2		
y[n]	6	1	5	3		

$y[4]$ *and* $y[5]$

n	0	1	2	3	4	5
x[n]	3	2	-1	-2	0	1
		h[N-1-n]	3	-1	2	
y[n]	6	1	5	3	-1	

n	0	1	2	3	4	5
x[n]	3	2	-1	-2	0	1
			h[N-1-n]	3	-1	2
y[n]	6	1	5	3	-1	-4

$y[6]$ *and* $y[7]$

Pad input with 0's

n	0	1	2	3	4	5	
x[n]	3	2	-1	-2	0	1	
				h[N-1-n]	3	-1	2
y[n]	6	1	5	3	-1	-4	-1

n	0	1	2	3	4	5		
x[n]	3	2	-1	-2	0	1		
					h[N-1-n]	3	-1	2
y[n]	6	1	5	3	-1	-4	-1	3

Topic Summary

- Delta Function and Impulse response
  - The output of a system with a delta function input is its impulse response
- Convolution and Kernels
  - The output of a system to an input will be the input convolved with the system impulse response
- Input Side Algorithm
  - The input side algorithm computes the convolution based on the input values

Topic Summary

- Output Side Algorithm
  - The output side algorithm computes a single output value by combining inputs. Uses a “flipped” impulse response