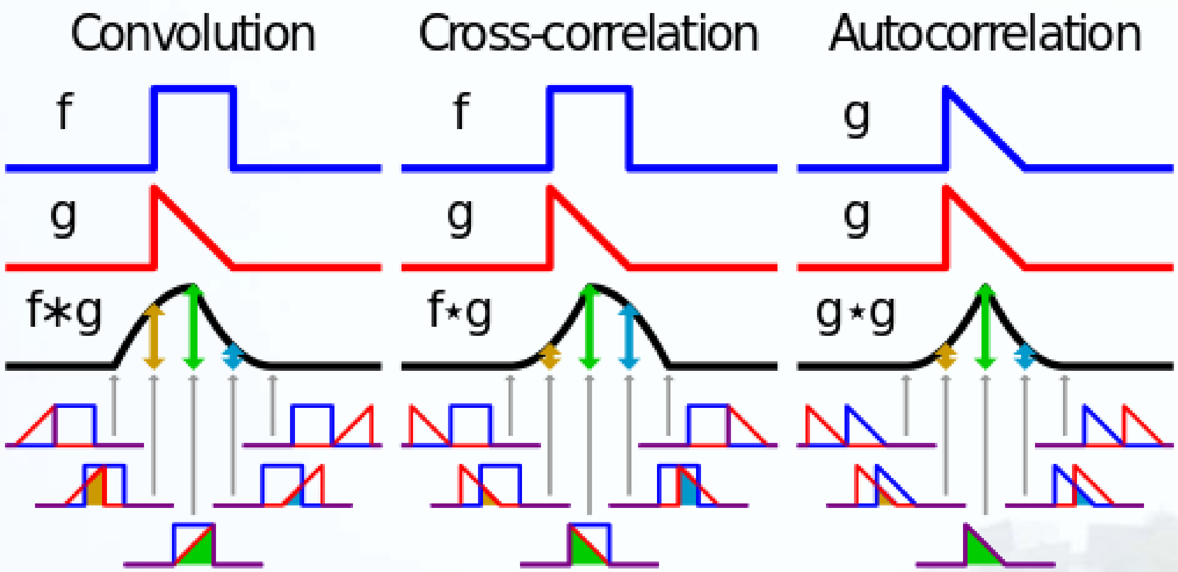
Pengolahan Sinyal Digital

Adhi Harmoko Saputro

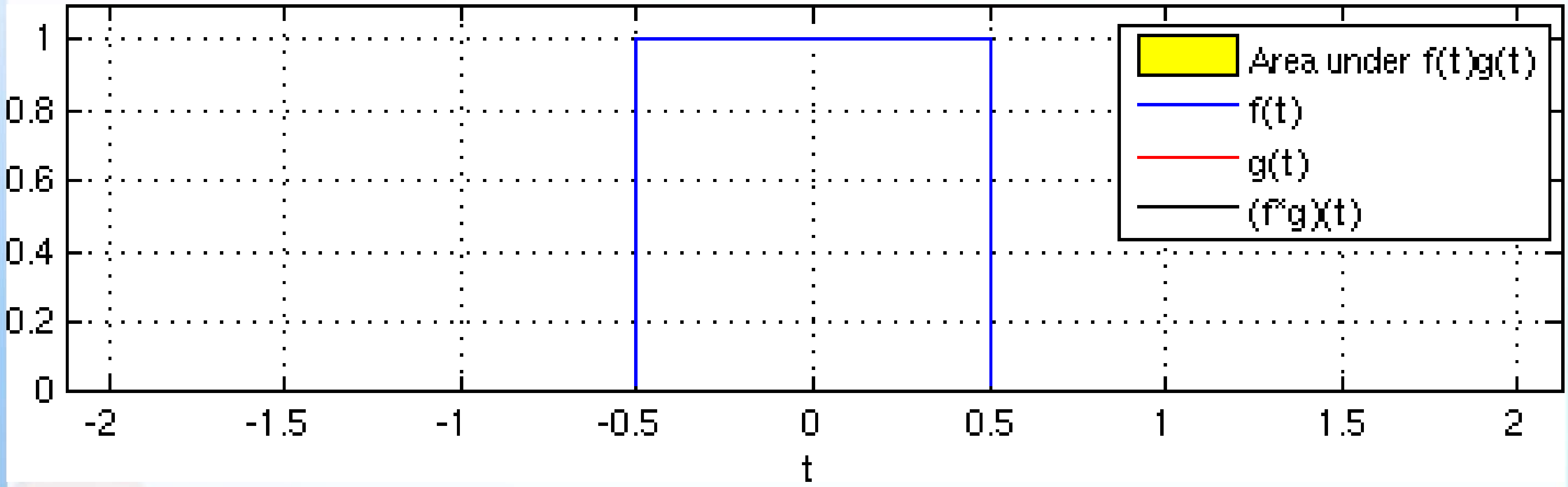
Convolution

Convolution

- Convolution is a mathematical operation on two functions (f and g)
- Produces a third function, that is typically viewed as a modified version of one of the original functions, giving the integral of the pointwise multiplication of the two functions as a function of the amount that one of the original functions is translated.



Convolution



Convolution

- The impulse response of an LTI system is given by $h[n]$

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

- The mathematical operation is called a *linear convolution sum* and is denoted by

$$y[n] \triangleq x[n] * h[n]$$

- If the sequences are mathematical functions (of finite or infinite duration), then we can analytically evaluate for all n to obtain a functional form of $y[n]$.

Convolution

- The "*n*" dependency of *y*[*n*] deserves some care: for each value of "*n*" the convolution sum must be computed ***separately*** over all values of a dummy variable "*m*". So, for each "*n*"
  1. Rename the independent variable as **m**. You now have **x**[**m**] and **h**[**m**]. Flip **h**[**m**] over the origin. This is **h**[**-m**]
  2. Shift **h**[**-m**] as far left as possible to a point "**n**", where the two signals barely touch. This is **h**[**n-m**]
  3. Multiply the two signals and sum over all values of **m**. This is the convolution sum for the specific "**n**" picked above.
  4. Shift / move **h**[**-m**] to the right by one sample, and obtain a new **h**[**n-m**]. Multiply and sum over all **m**.
  5. Repeat 2~4 until **h**[**n-m**] no longer overlaps with **x**[**m**], i.e., shifted out of the **x**[**m**] zone.

Convolution

- The "*n*" dependency of *y*[*n*] deserves some care: for each value of "*n*" the convolution sum must be computed ***separately*** over all values of a dummy variable "*m*". So, for each "*n*"
  1. Rename the independent variable as **m**. You now have **x**[**m**] and **h**[**m**]. Flip **h**[**m**] over the origin. This is **h**[**-m**]
  2. Shift **h**[**-m**] as far left as possible to a point "**n**", where the two signals barely touch. This is **h**[**n-m**]
  3. Multiply the two signals and sum over all values of **m**. This is the convolution sum for the specific "**n**" picked above.
  4. Shift / move **h**[**-m**] to the right by one sample, and obtain a new **h**[**n-m**]. Multiply and sum over all **m**.
  5. Repeat 2~4 until **h**[**n-m**] no longer overlaps with **x**[**m**], i.e., shifted out of the **x**[**m**] zone.

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

Useful Expressions

- The following expressions are often useful in calculating convolutions of analytical discrete signals

$$\sum_{n=0}^{\infty} a^n = \frac{1}{1-a}, \quad |a|<1$$

$$\sum_{n=k}^{\infty} a^n = \frac{a^k}{1-a}, \quad |a|<1$$

$$\sum_{n=m}^{N} a^n = \frac{a^m - a^{N+1}}{1-a}, \quad a \neq 1$$

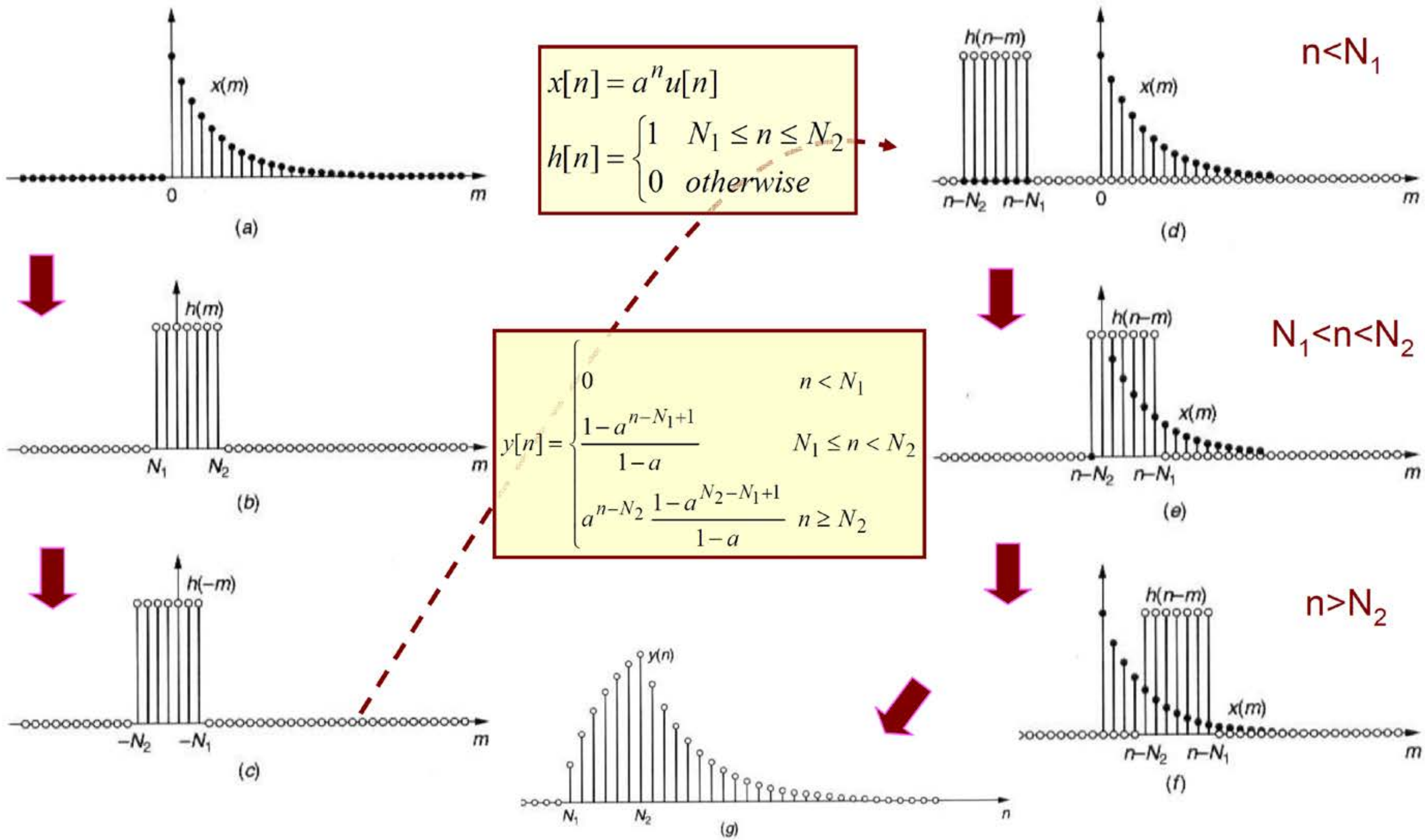
$$\sum_{n=0}^{N-1} a^n = \begin{cases} \dfrac{1-a^N}{1-a}, & |a| \neq 1 \\ N, & \text{a=1} \end{cases}$$

Convolution Example

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

$$h[n] = \begin{cases} 1 & N_1 \le n \le N_2 \\ 0 & otherwise \end{cases}$$

$$y[n] = \begin{cases} 0 & n < N_1 \\ \dfrac{1-a^{n-N_1+1}}{1-a} & N_1 \le n < N_2 \\ a^{n-N_2}\dfrac{1-a^{N_2-N_1+1}}{1-a} & n \ge N_2 \end{cases}$$



Example

- Let the rectangular pulse $x(n) = u(n) - u(n - 10)$ be an input to an LTI system with impulse response

$$h(n) = (0.9)^n u(n)$$

- Determine the output $y(n)$.

Example

- From the convolution equation

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

$$y(n) = \sum_{k=0}^{9} (1)(0.9)^{(n-k)} u(n-k) = (0.9)^{n} \sum_{k=0}^{9} (0.9)^{-k} u(n-k)$$

- The sum in equation is almost a geometric series sum except that the term $u(n-k)$ takes different values depending on n and k. There are three possible conditions under which $u(n-k)$ can be evaluated.

Example

- Case 1
  - $n < 0$
  - Then $u(n - k) = 0$, $0 \leq k \leq 9$

$$y(n) = 0$$

Example

- Case 2
  - The nonzero values of $x(n)$ and $h(n)$ *do not overlap*.
  - $0 \le n < 9$: Then $u(n - k) = 1,\ 0 \le k \le n$.

$$y(n) = (0.9)^n \sum_{k=0}^{n} (0.9)^{-k} = (0.9)^n \sum_{k=0}^{n} \left[(0.9)^{-1} \right]^k$$

$$= (0.9)^n \frac{1 - (0.9)^{-(n+1)}}{1 - (0.9)^{-1}} = 10\left[1 - (0.9)^{(n+1)} \right]$$

Example

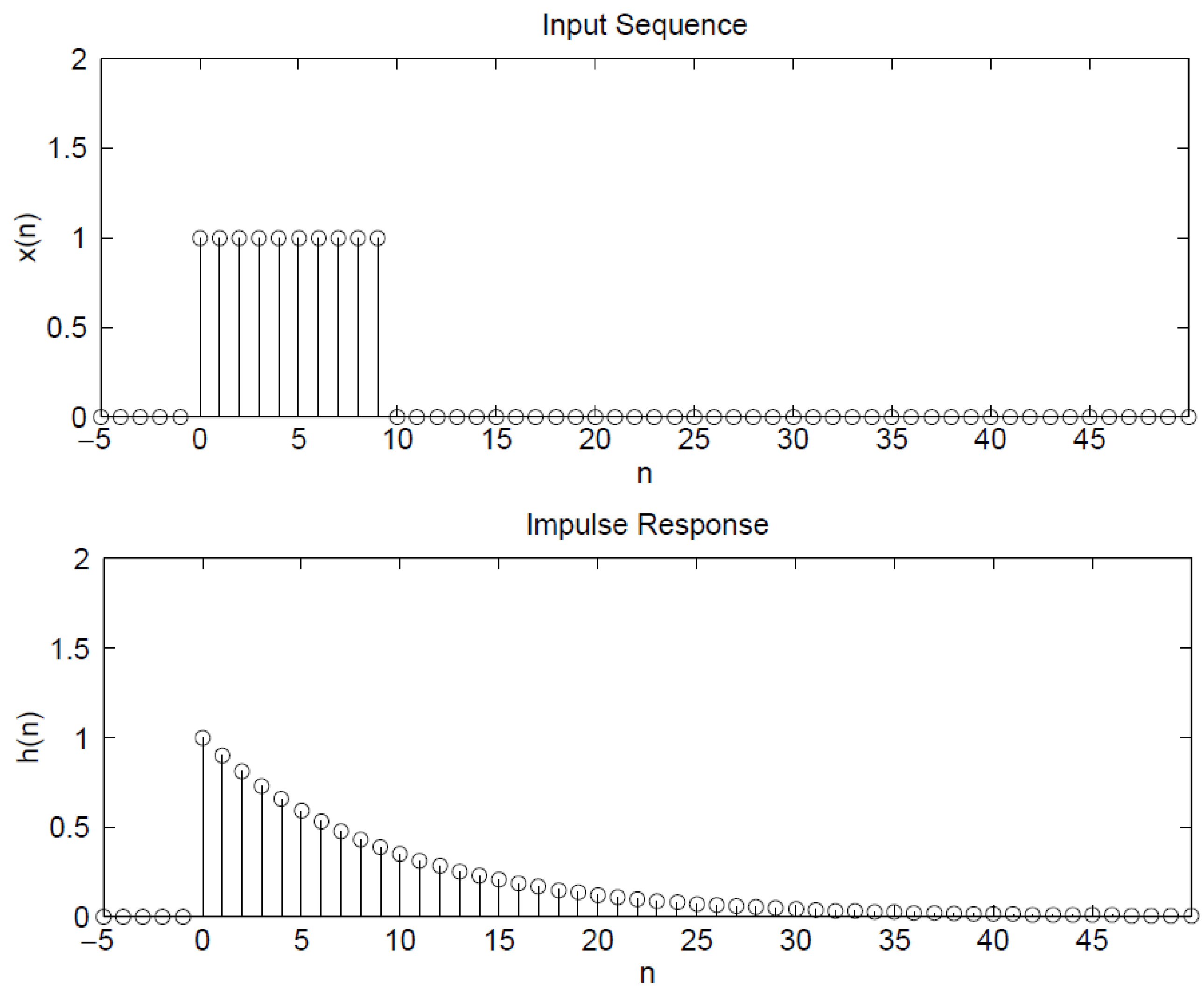
- Case 3
  - The impulse response $h(n)$ *partially overlaps* the input $x(n)$.
  - $n \geq 9$: Then $u(n-k) = 1,\ 0 \leq k \leq 9$

$$y(n) = (0.9)^n \sum_{k=0}^{9} (0.9)^{-k}$$

$$= (0.9)^n \frac{1-(0.9)^{-10}}{1-(0.9)^{-1}} = 10(0.9)^{n-9}\left[1-(0.9)^{10}\right]$$

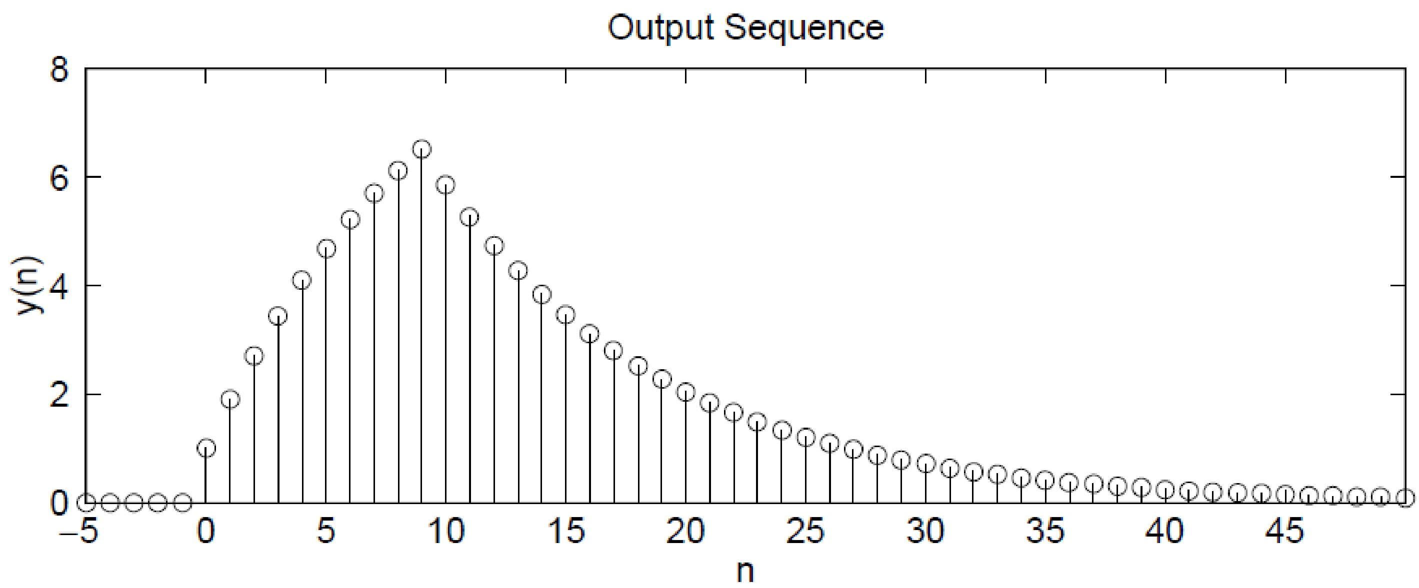
Example

- The input sequence and the impulse response



Example

- The output sequence



Example

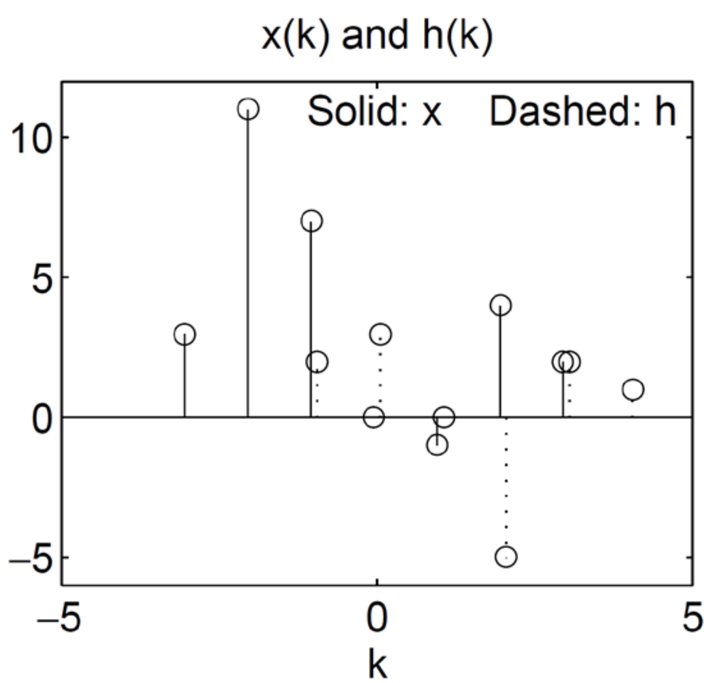
- Given the following two sequences

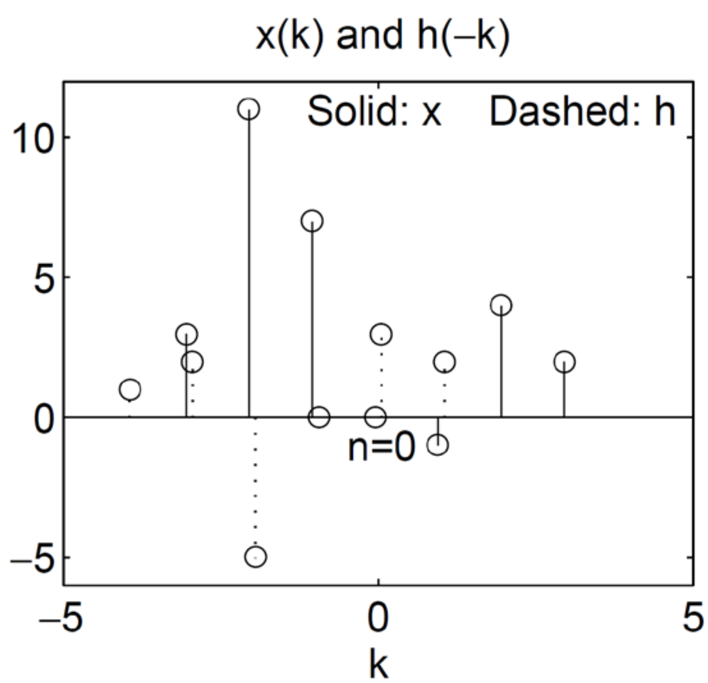
$$x(n) = \left[3, 11, 7, \underset{\uparrow}{0}, -1, 4, 2\right], \quad -3 \le n \le 3$$

$$h(n) = \left[2, \underset{\uparrow}{3}, 0, -5, 2, 1\right], \quad -1 \le n \le 4$$

- Determine the convolution $y(n) = x(n) * h(n)$.

Example

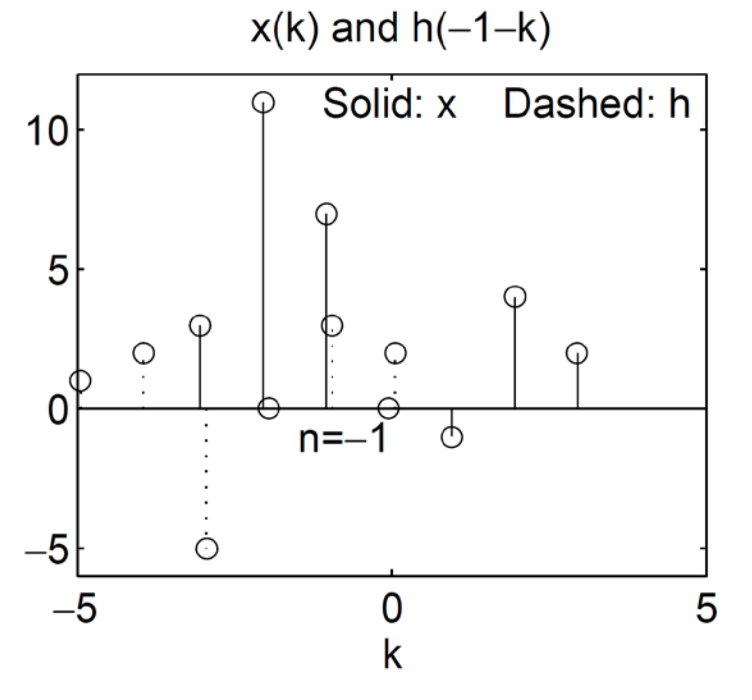




The original sequences

x(k) and h(−k), the folded version of h(k)

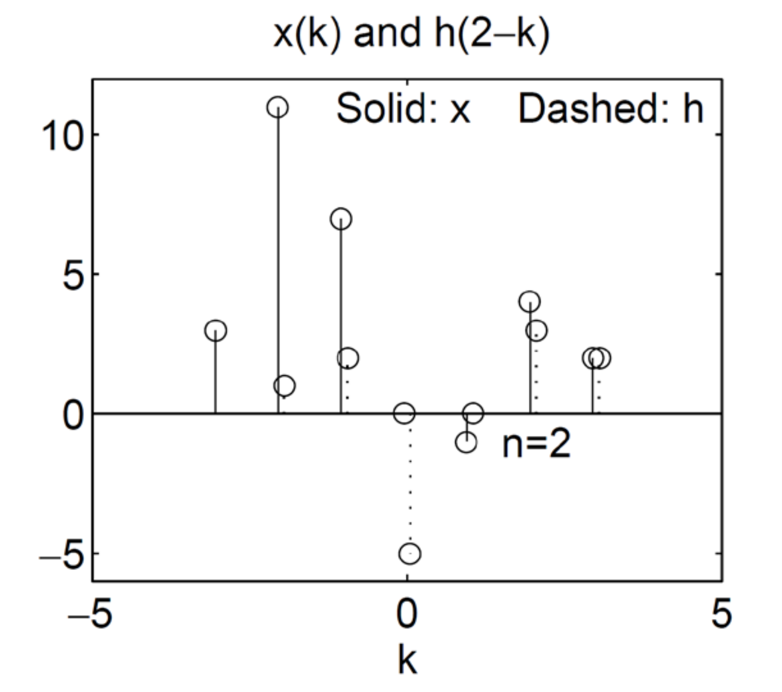
Example



x(k) and h(−1−k), the folded-and-shifted by −1 version of h(k).

$$\sum_k x(k)h(-1-k) = 3\times(-5)+11\times 0+7\times 3+0\times 2 = 6 = y(-1)$$

Example



x(k) and h(2 − k), the folded-and-shifted-by-2 version of h(k)

$$\sum_k x(k)h(2-k) = 11\times 1 + 7\times 2 + 0\times(-5) + (-1)\times 0 + 4\times 3 + 2\times 2 = 41 = y(2)$$

Example

- Similar graphical calculations can be done for other remaining values of $y(n)$.
- The beginning point (first nonzero sample) of $y(n)$ is given by

$$n = -3 + (-1) = -4$$

- The end point (the last nonzero sample) is given by

$$n = 3 + 4 = 7$$

- The complete output is given by

$$y(n) = \{6, 31, 47, 6, -\underset{\uparrow}{5}1, -5, 41, 18, -22, -3, 8, 2\}$$

Matlab Convolution

- If arbitrary sequences are of infinite duration, then MATLAB cannot be used *directly* to compute the convolution.
- MATLAB does provide a built-in function called conv that computes the convolution between two finite-duration sequences. The conv function assumes that the two sequences begin at $n = 0$ and is invoked by
- `>> y = conv(x,h);`

Example

- Let the rectangular pulse $x(n) = u(n) - u(n - 10)$ be an input to an LTI system with impulse response

$$h(n) = (0.9)^n u(n)$$

- Determine the output $y(n)$.

```
>> x = [3, 11, 7, 0, -1, 4, 2];
>> h = [2, 3, 0, -5, 2, 1];
>> y = conv(x, h)
y =
6 31 47 6 -51 -5 41 18 -22 -3 8 2
```

Matlab Convolution

- A simple modification of the conv function, which performs the convolution of arbitrary support sequences

```
function [y,ny] = conv_m(x,nx,h,nh)
% Modified convolution routine for signal processing
% ---------------------------------------------------
% [y,ny] = conv_m(x,nx,h,nh)
% [y,ny] = convolution result
% [x,nx] = first signal
% [h,nh] = second signal
%
nyb = nx(1)+nh(1); nye = nx(length(x)) + …
nh(length(h));
ny = [nyb:nye]; y = conv(x,h);
```

Example

- Given the following two sequences

$$x(n) = \left[3, 11, 7, \underset{\uparrow}{0}, -1, 4, 2\right], \quad -3 \le n \le 3$$

$$h(n) = \left[2, \underset{\uparrow}{3}, 0, -5, 2, 1\right], \quad -1 \le n \le 4$$

- Determine the convolution $y(n) = x(n) * h(n)$.

Example

```
>> x = [3, 11, 7, 0, -1, 4, 2]; nx = [-3:3];
>> h = [2, 3, 0, -5, 2, 1]; ny = [-1:4];
```

```
>> [y,ny] = conv_m(x,nx,h,nh)
y =
6 31 47 6 -51 -5 41 18 -22 -3 8 2
ny =
-4 -3 -2 -1 0 1 2 3 4 5 6 7
```

$$y(n) = \{6, 31, 47, 6, -\underset{\uparrow}{5}1, -5, 41, 18, -22, -3, 8, 2\}$$

Terima Kasih