EE 362: Signals and Systems

Time-Domain Models of Systems

Lecture 4

Convolution

The expression $\sum_{i=0}^{n} h[i]x[n-i]$ is called *convolution* of $h[n]$ and $x[n]$ and is denoted by the symbol "$*$"

$$h[n] * x[n] = \sum_{i=0}^{n} h[i]x[n-i]$$

Then,

$$y[n] = h[n] * x[n], \qquad n \geq 0$$

This convolution representation of the system is a time-domain model.

Convolution of Discrete-Time Signals

Given two discrete-time signals, $x[n]$ and $v[n]$, the convolution of these two signals is:

$$x[n] * v[n] = \sum_{i=-\infty}^{\infty} x[i]v[n-i]$$

The convolution operation $x[n] * v[n]$ is commutative; i.e., $x[n] * v[n] = v[n] * x[n]$

$$\sum_{i=-\infty}^{\infty} x[i]v[n-i] = \sum_{i=-\infty}^{\infty} v[i]x[n-i]$$

If $x[n]$ and $v[n]$ are zero for all integers $n < 0$, then $x[i] = 0$ for all integers $i < 0$ and $v[n-i] = 0$ for all integers $n < i$. Thus,

$$x[n] * v[n] = \begin{cases} 0, & n = -1, -2, \dots \\ \sum_{i=0}^{n} x[i]v[n-i] & n = 0, 1, 2, \dots \end{cases}$$

OR

$$x[n] * v[n] = \begin{cases} 0, & n = -1, -2, \dots \\ \sum_{i=0}^{n} v[i]x[n-i] & n = 0, 1, 2, \dots \end{cases}$$

Computation of the Convolution

1. Analytical form, 2. Array method, 3. MATLAB

Analytical form

If the signals $x[n]$ and $v[n]$ are given by simple mathematical expressions, the convolution $x[n] * v[n]$ can be calculated by inserting $x[n]$ and $v[n]$ into the convolution equation.

Example: Suppose that $x[n] = a^n u[n]$ and $v[n] = b^n u[n]$ where a and b are fixed nonzero real numbers.

$$x[n] * v[n] = \sum_{i=-\infty}^{\infty} a^i u[i] b^{n-i} u[n-i]$$

$u[i] = 1$ and $u[n-i] = 1$ for all integer values of i ranging from $i = 0$ to $i = n$. Thus,

$$x[n] * v[n] = \sum_{i=0}^{n} a^i b^{n-i} = b^n \sum_{i=0}^{n} \left(\frac{a}{b}\right)^i, n = 0,1, \ldots$$

If $a = b$, then for $n = 0, 1, 2, \ldots,$

$$\sum_{i=0}^{n} \left(\frac{a}{b}\right)^i = n + 1 \rightarrow x[n] * v[n] = b^n(n+1)$$

$$= a^n(n+1)$$

If $a \neq b$,

$$\sum_{i=0}^{n} \left(\frac{a}{b}\right)^i = \frac{1-(a/b)^{n+1}}{1-a/b}$$

Then,

$$x[n] * v[n] = b^n \frac{1-\left(\frac{a}{b}\right)^{n+1}}{1-\frac{a}{b}}, \qquad n = 0, 1, 2, \ldots$$

$$x[n] * v[n] = \frac{b^{n+1} - a^{n+1}}{b-a}, \qquad n = 0, 1, 2, \ldots$$

It is easy to generalize the convolution expressions to the case when $x[n]$ and $v[n]$ are not necessary zero for all integers $n < 0$. Suppose that $x[n] = 0$ for all $n < Q$ and $v[n] = 0$ for all $n < P$ where P and Q are positive or negative integer numbers.

$$x[n] * v[n] = \begin{cases} 0, & n < P + Q \\ \sum_{i=0}^{n-P} x[i]v[n-i] & n \geq P + Q \end{cases}$$

$$x[n] * v[n] = \begin{cases} 0, & n < P + Q \\ \sum_{i=0}^{n-Q} x[n-i]v[n] & n \geq P + Q \end{cases}$$

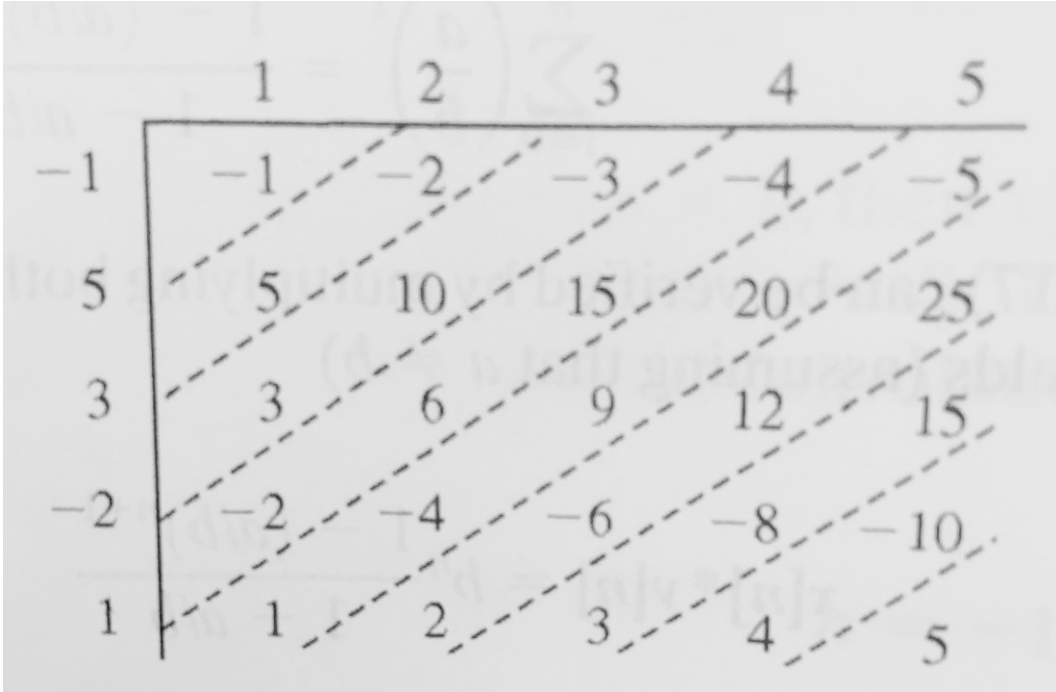
Array Method

$x[n] = 0$ for all $n < Q$ and $v[n] = 0$ for all $n < P$

	$x[Q]$	$x[Q+1]$	$x[Q+2]$	$x[Q+3]$
$v[P]$	$v[P]\,x[Q]$	$v[P]\,x[Q+1]$	$v[P]x[Q+2]$	$v[P]\,x[Q+3]$
$v[P+1]$	$v[P+1]\,x[Q]$	$v[P+1]\,x[Q+1]$	$v[P+1]x[Q+2]$	$v[P+1]\,x[Q+3]$
$v[P+2]$	$v[P+2]x[Q]$	$v[P+2]x[Q+1]$	$v[P+2]x[Q+2]$	$v[P+2]x[Q+3]$
$v[P+3]$	$v[P+3]x[Q]$	$v[P+3]x[Q+1]$	$v[P+3]x[Q+2]$	$v[P+3]x[Q+3]$

The values of $y[n]$ of the convolution $x[n] * v[n]$ are determined by the sum of the elements in the backward diagonals where $y[P+Q+i]$ is the sum of the diagonal $x[Q+i]$ to $v[P+i]$

Example: Suppose that $x[n] = 0$ for $n < -1$, $x[-1] = 1, x[0] = 2, x[1] = 3, x[2] = 4,$ $x[3] = 5$, …, and $v[n] = 0$ for $n < -2, v[-2] = -1, v[-1] = 5, v[0] = 3, v[1] = -2, v[2] = 1,$ … . In this case $Q = -1$ and $P = -2$ and the array is as follows

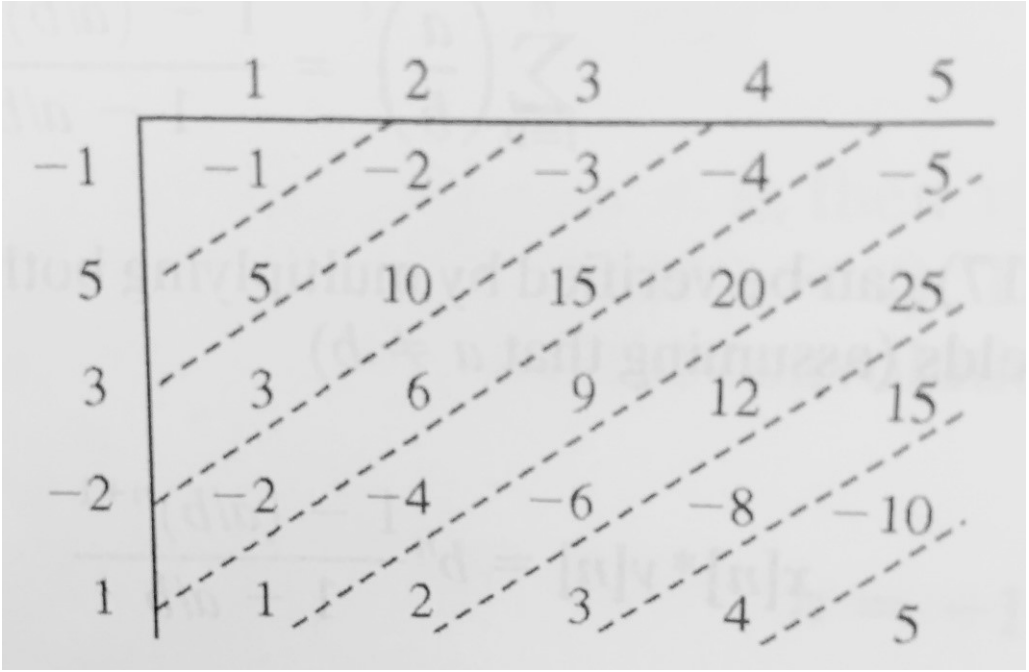


	1	2	3	4	5
−1	−1	−2	−3	−4	−5
5	5	10	15	20	25
3	3	6	9	12	15
−2	−2	−4	−6	−8	−10
1	1	2	3	4	5

Find $y[P + Q + i]$ for $i = 0, 1, 2, \ldots$

Find $y[-1+(-2)+i]$

for $i = 0, 1, 2, \ldots$



$y[-3] = -1$

$y[-2] = 5 - 2 = 3$

$y[-1] = 3 + 10 - 3 = 10$

$y[0] = -2 + 6 + 15 - 4 = 15$

$y[1] = 1 - 4 + 9 + 20 - 5 = 21$

and so on.

$y[n] = 0$ for $n < -3$

Using MATLAB

Consider the convolution of the pulse $p[n]$ with itself, where $p[n]$ is defined by $p[n] = 1$ for $0 \leq n \leq 10$ and $p[n] = 0$ for all other n.

```
p=[0 ones(1,10) 0];

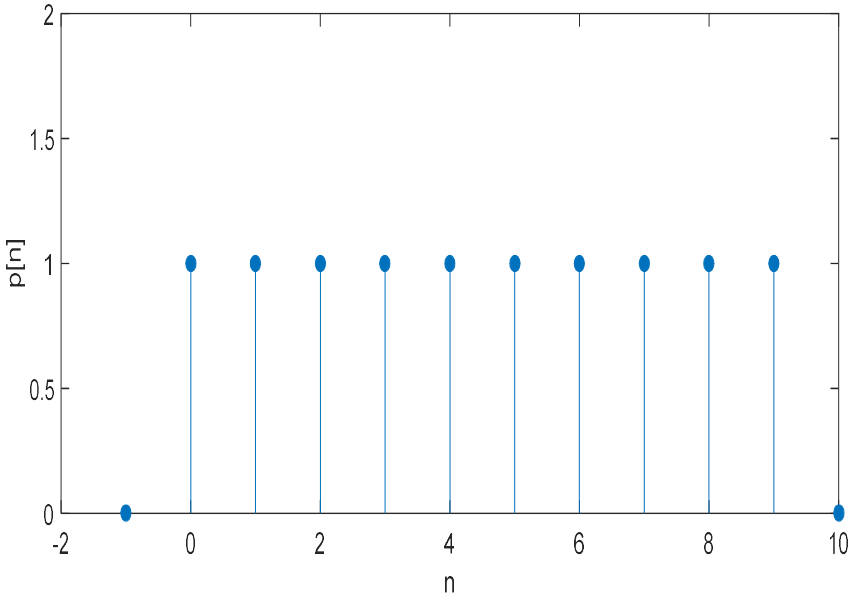
x=p; v=p; y=conv(x,v);

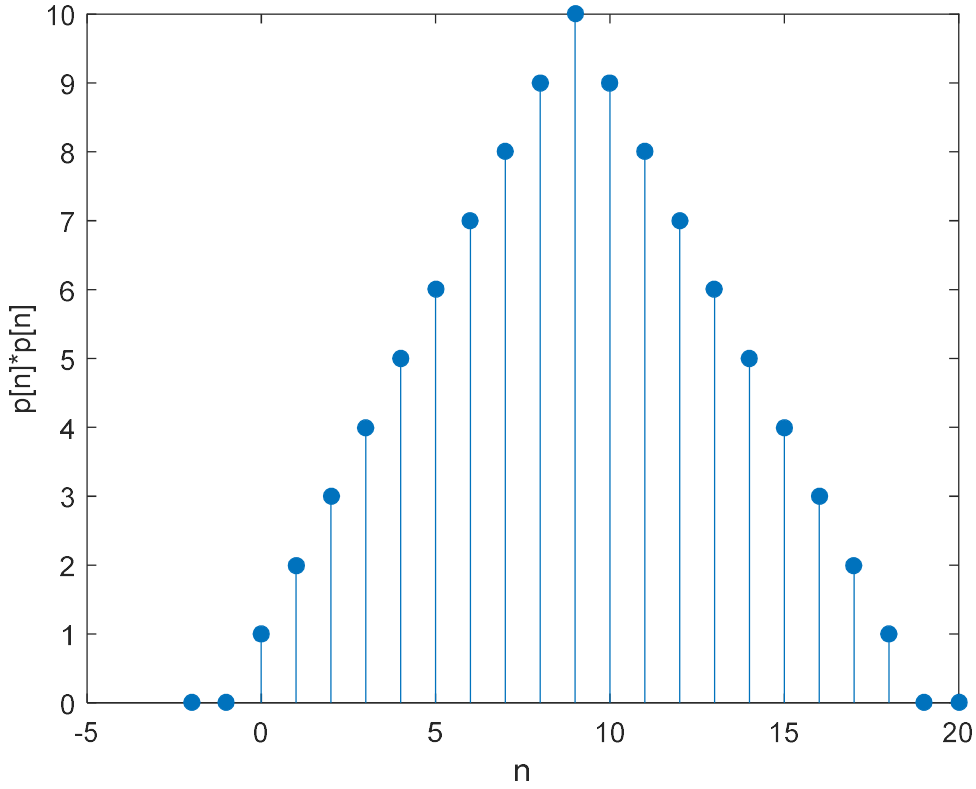
n=-2:20;

stem(n,y(1:length(n)), 'filled')

xlabel('n')

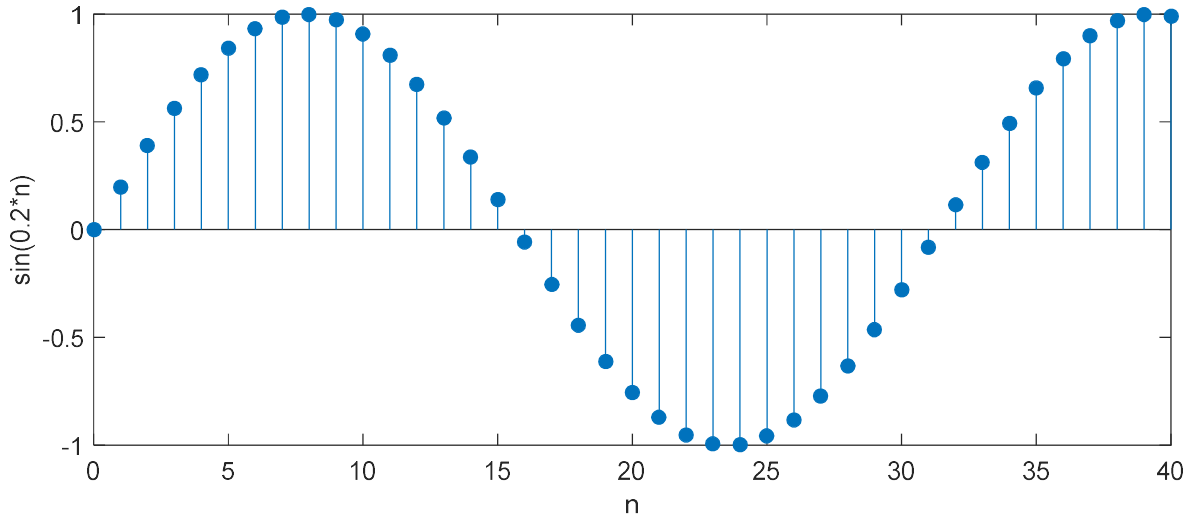
ylabel('p[n]*p[n]')
```

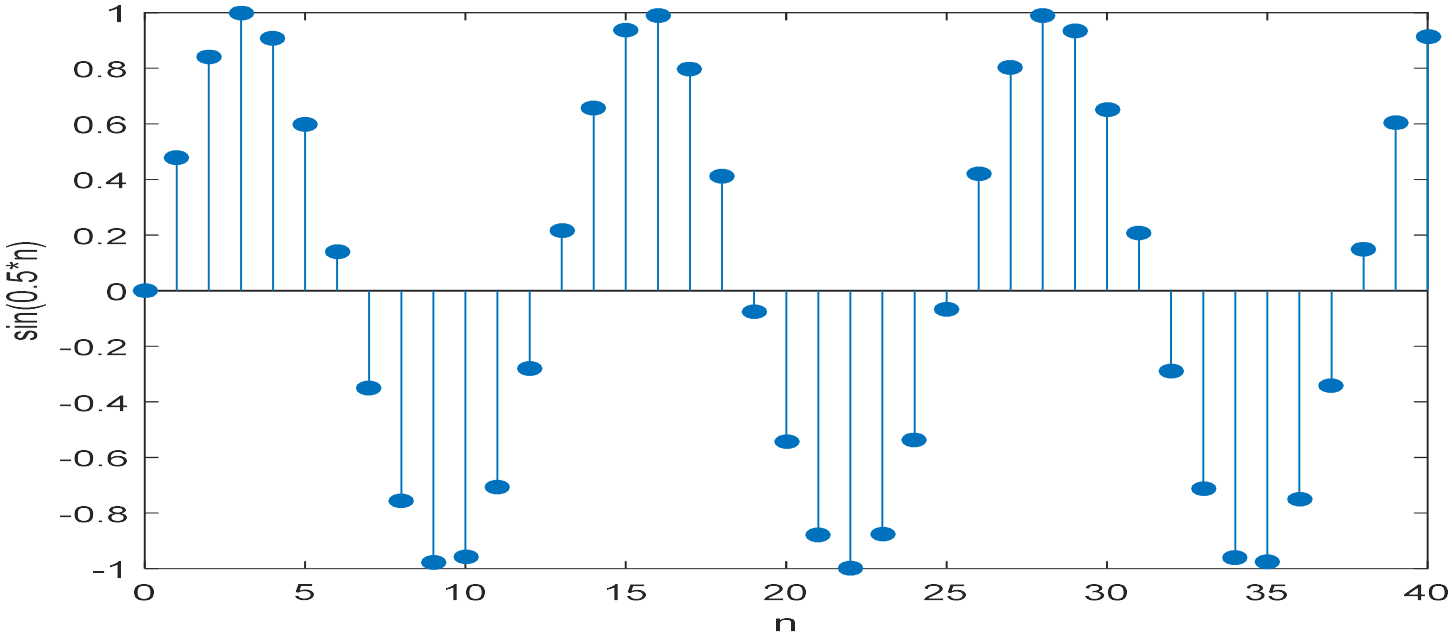

2
1.5
1
0.5
0
p[n]
-2
0
2
4
6
8
10
n

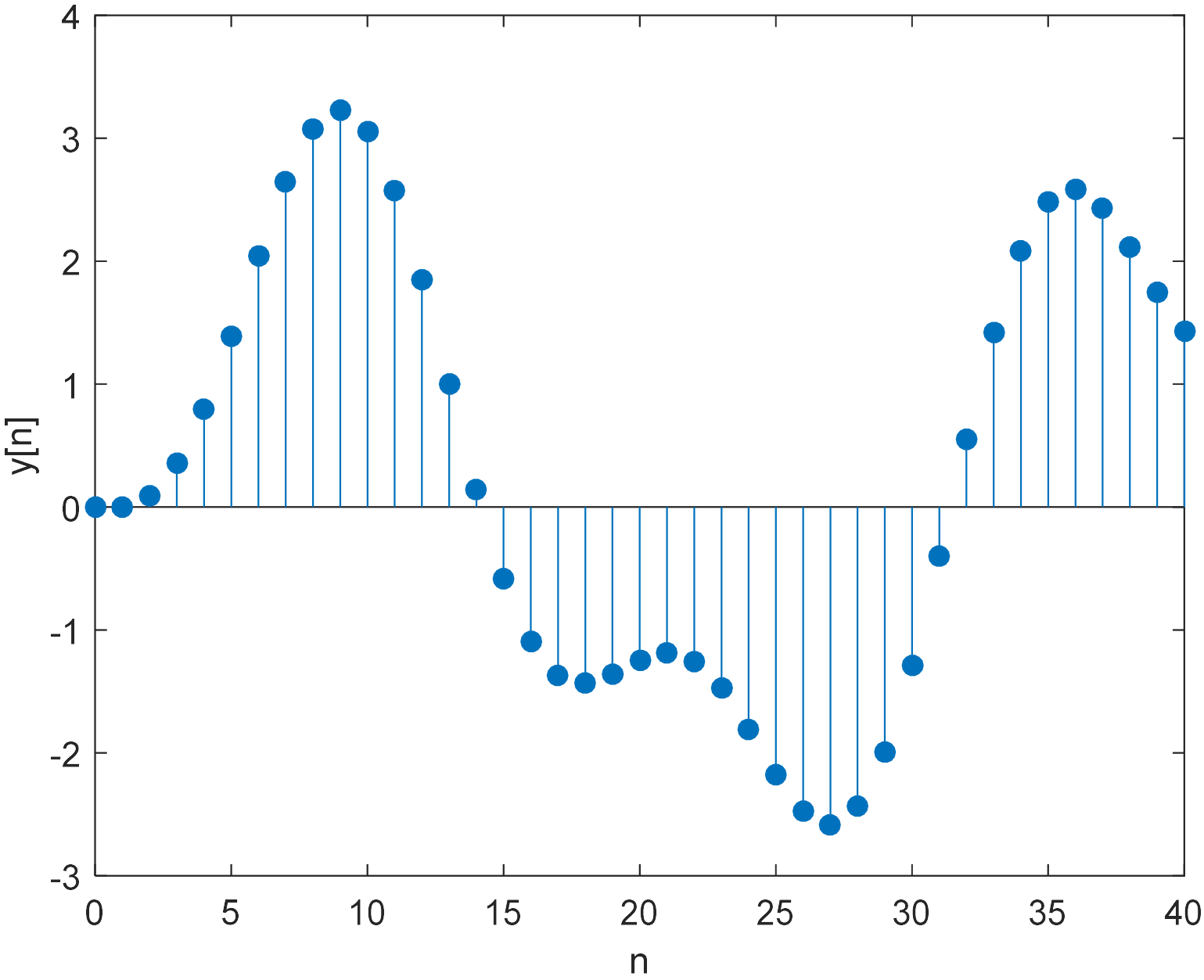

10
9
8
7
6
5
4
3
2
1
0
p[n]*p[n]
-5
0
5
10
15
20
n

Example: Suppose that the unit-pulse response $h[n]$ is equal to $\sin(0.5n)$ for $n \geq 0$ and the input $x[n] = \sin(0.2n)$ for $n \geq 0$. Compute the response $y[n]$ for $n = 0,1,2, \ldots, 40$.

```
n=0:40;
x=sin(0.2*n);
h=sin(0.5*n);
y=conv(x,h);
stem(n,x(1:length(n)), 'filled'), xlabel('n'), ylabel('sin(0.2*n)')
figure, stem(n,h(1:length(n)), 'filled'), xlabel('n'), ylabel('sin(0.5*n)')
figure, stem(n,y(1:length(n)), 'filled'), xlabel('n'), ylabel(‘y[n]')
```


sin(0.2*n)
n
1
0.5
0
-0.5
-1
0
5
10
15
20
25
30
35
40


sin(0.5*n)
n
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
5
10
15
20
25
30
35
40


4
3
2
1
0
-1
-2
-3
y[n]
0
5
10
15
20
25
30
35
40
n

Using MATLAB

Examples are available at:

http://users.ece.gatech.edu/~bonnie/book3/

Difference and Differential Equation Models

- **Difference Equation Models**

In many applications, a **causal linear time-invariant** discrete-time system is given by an input/output difference equation instead of an input/output convolution model.

Consider repayment of a bank loan. Let $y[n]$ denote the balance of the loan after the nth month and $x[n]$ denote the amount of the loan payment in the nth month.

Thus, the balance after the nth month is:

$$y[n] = \left(1 + \frac{I}{12}\right) y[n-1] - x[n]$$

where I is the yearly interest rate in decimal form.

We can solve this equation recursively as follows:

$$n = 1 \rightarrow y[1] = \left(1 + \frac{I}{12}\right) y[0] - x[1]$$

$$n = 2 \rightarrow y[2] = \left(1 + \frac{I}{12}\right) y[1] - x[2]$$

$$n = 3 \rightarrow y[3] = \left(1 + \frac{I}{12}\right) y[2] - x[3], \text{so on}$$

The next value is computed recursively from the present value of the output plus an input term. This is called a first-order ***recursive*** problem.

MATLAB:

```
y0=input('Amount of loan = ');
c=input('Monthly loan payment = ');
I=input('Yearly interest rate, in decimal = ');
y=[];
y(1)=(1+I/12)*y0-c;
n=1;
while y(n)>0
n=n+1;
y(n)=(1+I/12)*y(n-1)-c;
end
y
```

Nth-Order Input/Output Difference Equation

The first-order input/output difference equation can be easily generalized to the Nth-order case.

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

The coefficients a_i and b_i are constants.

The output $y[n]$ can be computed recursively for some finite range of integer values of n.

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

$$y[0] = -a_1 y[-1] - a_2 y[-2] - \cdots - a_N y[-N] + b_0 x[0] + b_1 x[-1] + \cdots + b_M x[-M]$$

$$y[1] = -a_1 y[0] - a_2 y[-1] - \cdots - a_N y[-N+1] + b_0 x[1] + b_1 x[0] + \cdots + b_M x[-M+1]$$

and so on,

It is clear that the next value of the output is a linear combination of the N past values of the output and M+1 values of the input.

If all a_i are zero, the output at any fixed time point depends only on $x[n]$ and the output is not computed recursively.

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

This is nonrecursive system such as MA and EWMA

- Unlike the convolution model, the difference equation model needs initial conditions for the output and the input.
- Unit-pulse response, $h[n]$, of the difference equation model: set $x[n] = \delta[n]$ with zero initial conditions.

Example: Compute the unit-pulse response of the following system.

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

Arrange:

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

To compute the unit-pulse response, $h[n]$, of the system, set $x[n] = \delta[n]$ with initial conditions $y[-1]$ and $y[-2]$ both equal to zero. This gives $h[0] = 0$, $h[1] = 0, h[2] = 2\delta[0] = 2, h[3] = 1.5h[2] = 3$, $h[4] = 1.5h[3] - h[2] = 2.5$ and so on.

Now, suppose that the input to the above system is a discrete-time unit-step function $u[n]$ and that the initial output values are $y[-2] = 2$ and $y[-1] = 1$.

$$y[0] = 1.5y[-1] - y[-2] + 2x[-2]$$

$$y[0] = 1.5(1) - 2 + 2(0) = -0.5$$

$$y[1] = 1.5y[0] - y[-1] + 2x[-1]$$

$$y[1] = 1.5(-0.5) - 1 + 2(0) = -1.75$$

$$y[2] = 1.5y[1] - y[0] + 2x[0]$$

$$y[2] = 1.5(-1.75) - (-0.5) + 2(1) = -0.125$$

$$y[3] = 1.5y[2] - y[1] + 2x[1]$$

$$y[3] = 1.5(-0.125) + 1.75 + 2(1) = 3.5625$$

and so on

In recursively solving the difference equation model, the process of computing $y[n]$ can start at any desired time period. If the desired value is the output $y[q]$ at time q, the initial values of the output that are required are $y[q-N]$, $y[q-N+1]$, …, $y[q-1]$ where N is the order of the difference equation.

The N-order difference equation can be solved using MATLAB (please see the textbook)

Complete Solution of the Difference Equation

By solving the difference equation recursively, it is possible to generate a complete solution.

Consider the following first-order linear difference equation

$$y[n] = -ay[n-1] + bx[n], \qquad n = 1,2, \ldots$$

Then

$$y[1] = -ay[0] + bx[1]$$

$$y[2] = -ay[1] + bx[2] = a^2 y[0] - abx[1] + bx[2]$$

$$\begin{aligned} y[3] &= -ay[2] + bx[3] \\ &= -a^3 y[0] + a^2 bx[1] - abx[2] + bx[3] \end{aligned}$$

$$\therefore y[n] = (-a)^n y[0] + \sum_{i=1}^{n} (-a)^{n-i} bx[i]$$

Example: Inventory Level

Consider a manufacturer that produces a specific product. Let y[n] denote the number of the product in inventory at the end of the nth day, p[n] denote the number of the product whose manufacturing is completed during the nth day, and d[n] denote the number of the product that is delivered to customers during the nth day. Thus the number of the product in the inventory at the end of the nth day can be expressed as follows:

$$y[n] = y[n-1] + p[n] - d[n], for\ n = 1, 2, \ldots$$

$y[0]$ is the initial number of the product in the inventory

Define $x[n] = p[n] - d[n]$, then

$$y[n] = y[0] + \sum_{i=1}^{n} x[i]\,, for\ n = 1, 2, \ldots$$

$$y[n] = y[0] + \sum_{i=1}^{n} p[n] - d[n]\,, for\ n = 1, 2, \ldots$$

If the manufacturing delivery rates are known, the initial value of the product can be set so that depletion of the inventory can be avoided.