PENGENALAN MATLAB
PENGOLAHAN SINYAL DIGITAL

PRODI SISTEM KOMPUTER
FAKULTAS ILMU KOMPUTER
IIB DARMAJAYA

MATLAB®

RETNO DWI HANDAYANI, S.Kom.,MTI

MathWorks®

1. Apa Matlab itu?

Matlab merupakan bahasa pemrograman dengan kemampuan tinggi dalam bidang komputasi. Matlab memiliki kemampuan mengintegrasikan komputasi, visualisasi, dan pemrograman. Oleh karenanya, matlab banyak digunakan dalam bidang riset-riset yang memerlukan komputasi numerik yang kompleks. Penggunaan Matlab meliputi bidang–bidang:

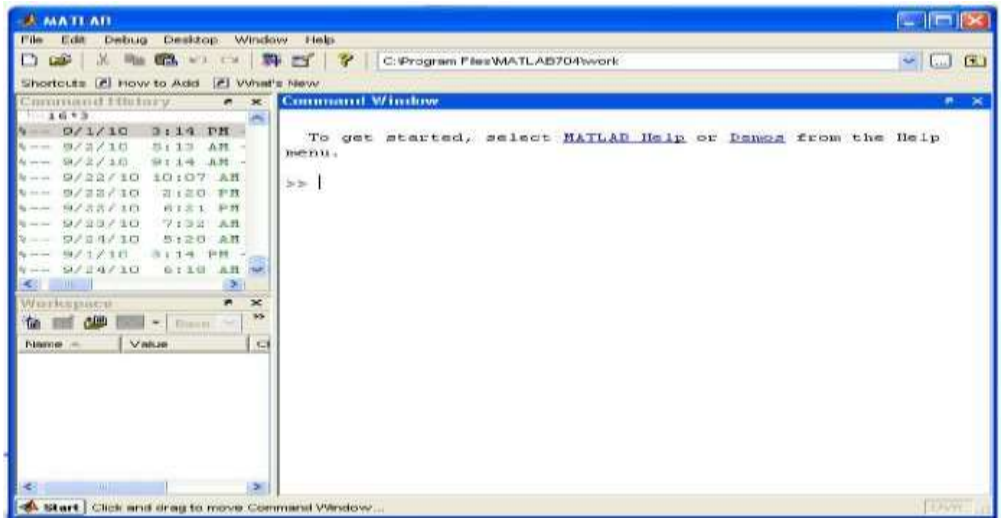
- ✓ Matematika dan Komputasi
- ✓ Pembentukan Algorithm
- ✓ Akusisi Data
- ✓ Pemodelan, simulasi, dan pembuatan prototype
- ✓ Analisa data, explorasi, dan visualisasi
- ✓ Grafik Keilmuan dan bidang Rekayasa

Matlab merupakan kepanjangan dari ***Matrix Laboratory***. Sesuai dengan namanya, struktur data yang terdapat dalam Matlab menggunakan matriks atau array berdimensi dua (double). Oleh karenanya penguasaan teori matriks mutlak diperlukan bagi pengguna pemula Matlab agar mudah dalam mempelajari dan memahami operasi-operasi yang ada di Matlab.

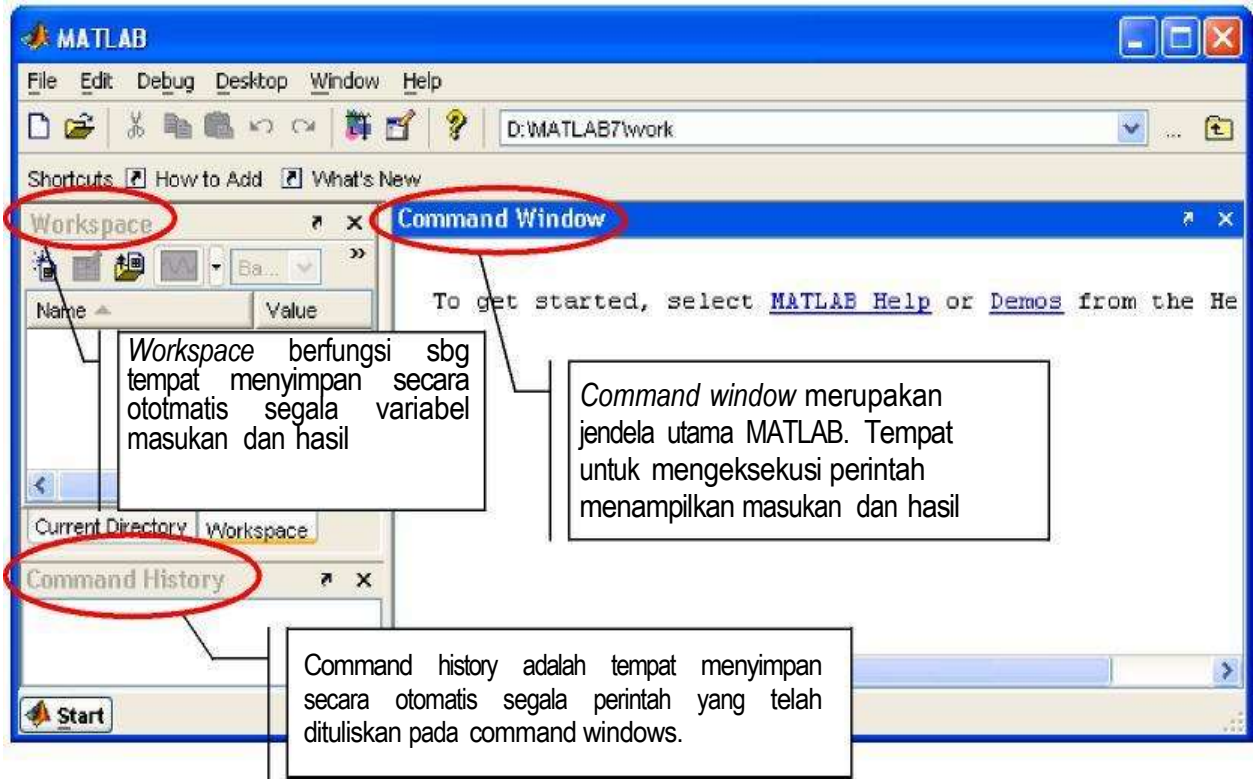
Kita dapat belajar Matlab melalui berbagai macam cara seperti dari buku maupun internet. Banyak situs di internet yang menyediakan tutorial tentang matlab. Seperti tutorial dasar, toolboxes, simulink, dan sebagainya. Kita dapat menggunakan situs http://www.mathworks.com. Untuk memperoleh informasi dan pengetahuan terkini tentang matlab.

2. Memulai MATLAB

Perhatikan Dekstop pada layar monitor PC, mulailah MATLAB dengan melakukan *double-clicking* pada *shortcut icon* MATLAB. Selanjutnya akan muncul tampilan seperti pada Gambar berikut ini.



Gambar 1. Jendela Utama Matlab



Gambar 2. Lingkungan Kerja Matlab

Pada gambar diatas, terlihat beberapa jendela yang merupakan bagian penting di dalam Matab, antara lain:

a. Jendela perintah (Command Window)

Pada command window, semua perintah matlab dituliskan dan diekskusi. Kita dapat menuliskan perintah perhitungan sederhana, memanggil fungsi, mencari informasi tentang sebuah fungsi dengan aturan penulisannya (help), demo program, dan sebagainya.

Setiap penulisan perintah selalu diawali dengan prompt '>>'. Misal, mencari nilai sin 75^0, maka pada command window kita dapat mengetikkan:

```
>> sin(75)
ans =
     -0.38778
```

b. Jendela ruang kerja (Workspace)

Jendela ini berisi informasi pemakaian variabel di dalam memori matlab. Misalkan kita akan menjumlahkan dua buah bilangan, maka pada command window kita dapat mengetikkan:

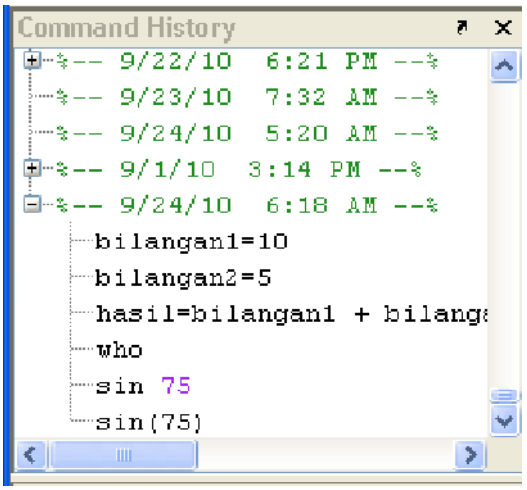
```
>> bilangan1  = 10
bilangan1=10
>> bilangan2  = 5
bilangan1=10
>> hasil= bilangan1  + bilangan2
hasil=15
```

Untuk melihat variabel yang aktif saat ini, kita dapat menggunakan perintah `who`.

```
>> who
Your variables are:
bilangan1     bilangan2  hasil
```

c. Jendela histori (Command History)

Jendela ini berisi informasi tentang perintah yang pernah dituliskan sebelumnya. Kita dapat mengambil kembali perintah dengan menekan tombol panah ke atas atau mengklik perintah pada jendela histori, kemudian melakukan copy-paste ke command window.



Gambar: Command History

3. Variabel dan Operator

3.1 Variabel

Seperti bahasa pemrograman lainnya, matlab pun memiliki variabel, tetapi dalam penulisannya, variabel di dalam matlab tidak perlu dideklarasikan, karena matlab mampu mengenali tipe data dari variable dari isi variabel tersebut. Aturan penulisan variabel pada matlab sama dengan aturan pada bahasa pemrograman lainnya, yaitu bersifa case sensitive, diawali dengan huruf dan selanjutanya boleh menggunakan gabungan huruf-angka atau tanda garis bawah. Matlab mampu mengenali sampai 31 karakter pertama, selanjutnya diabaikan.

Contoh:

```
>> var1=6.7
var1 =
       6.7
>> var_2=[2  3  4]
Var_2 =
       2     3     4
```

Semua tipe data di matlab memiliki bentuk yang sama, yaitu array. Array minimal berukuran 0x0 dan dapat bertambah menjadi array n x m dimensi dengan sebarang ukuran. Matlab mempunyai beberapa tipe data dasar (atau class), yaitu: logical, char, numeric, cell, structure, java classes, function handles.

3.2 Operator

Di dalam matlab, operator diklasifikasikan menjadi tiga bagian, yaitu:

a) Operator Arimatika

Operator aritmatika digunakan untuk mengerjakan komputasi numeric.

Operator	Arti
+	Penjumlahan
-	Pengurangan
*	Perkalian (aturan matriks)
.*	Perkalian masing-masing eleman yang bersesuaian (aturan
/	Pembagian kanan
./	Pembagian kanan (array)
\	Pembagian kiri (matriks)
.\	Pembagian kiri (array)
^	Perpangkatan (matriks)
.^	Perpangkatan (array)
:	langkah

b) Operator Relasional Operator relasional digunakan untuk membandingkan operand-operand secara kuantitatif

Operator	Arti
==	Sama dengan
~=	Tidak sama dengan
<	Kurang dari
>	Lebih dari
<=	Kurang dari sama dengan
>=	Lebih dari sama dengan

c) Operator Logika

Operator	Arti
&	Akan menghasilkan nilai 1 jika kedua elemen yang bersesuaian memiliki nilai true dan 0
|	Akan bernilai 1 jika salah satu elemennya true
~	Komplen dari elemen yang diinputkan
xor	Akan bernilai 1 jika salah satu dari kedua elemen memiliki nilai berbeda dan bernilai

4. Matriks

Matlab menggunakan matriks sebagai dasar komputasinya, maka pengetahuan tentang matriks sangatlah diperlukan bagi pengguna matlab. Secara garis besar matlab membagi matriks menjadi dua bagian.

4.1 Matriks Khusus

Matriks khusus merupakan matriks yang didefiniskan oleh matlab, sehingga kita

tinggal menggunakannya. Contoh: matriks nol, matriks diagonal, matriks identitas, dan sebagainya.

a) Matriks nol

Matriks yang elemenya bilangan nol
Bentuk umum:

```
>> zeros(n,m)
```

Contoh :

```
>> zeros(2,3)
ans =
     0     0     0
     0     0     0
```

b) Matriks satu

Matriks yang elemenya bilangan nol Bentuk umum:

```
>> ones(n,m)
```

Contoh :

```
>> ones(3,3)
ans =
     1     1     1
     1     1     1
     1     1     1
```

c) Matriks identitas

Bentuk umum:

```
>> eye(n)
```

Contoh :

```
>> eye(3)
ans =
     1     0     0
     0     1     0
     0     0     1
```

d) Matriks bujur sangkar ajaib

Matriks yang memiliki jumlahan yang sama pada tiap baris, kolom maupun diagonalnya Bentuk umum:

```
>>magic(n)
```

Contoh :

```
>> magic(4)
ans =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

e) Matriks acak

Matriks isinya bernilai acak berdasarkan distribusi

statistic. Bentuk umum:

>>rand(n,m)

Contoh :

```
>>
rand(4,4)
ans =
      0.61543       0.17627       0.41027       0.81317
      0.79194       0.40571       0.89365     0.0098613
      0.92181       0.93547      0.057891       0.13889
      0.73821        0.9169       0.35287       0.20277
```

4.2 Matriks yang didefiniskan oleh pengguna

Selain bentuk khusus, matlab juga menyediakan bentuk matriks yang disefinisikan oleh pengguna, yaitu menggunakan tanda kurung siku.

Contoh:

```
>> A=[ 1 2 3; 3 4 5]
A =
     1     2     3
     3     4     5
```

Tanda semicolon ';' digunakan untuk memisahkan baris satu dengan yang lain.

Perhatian !

Coba Anda bangkitkan dua buah matriks dengan sembarang ukuran, kemudian lakukan operasi aritmatika dan juga cari tahu determinan dan invers dari suatu matriks. Perhatikan cara kerjanya!

5. Operasi Aljabar Matrik

5.1 Penjumlahan & Pengurangan

Operasi penjumlahan dan pengurangan pada matriksm hanya dapat dilakukan jika matrik-matrik yang akan dijumlahkan memiliki orde sama

$$\begin{bmatrix}2 & 3 & 1 & 6\\ 1 & 4 & 5 & 2\end{bmatrix}+\begin{bmatrix}2 & 3 & 1 & 6\\ 1 & 4 & 5 & 2\end{bmatrix}=\begin{bmatrix}4 & 6 & 2 & 12\\ 2 & 8 & 10 & 4\end{bmatrix}$$

$$\begin{bmatrix}2 & 3 & 1 & 6\\ 1 & 4 & 5 & 2\end{bmatrix}-\begin{bmatrix}2 & 3 & 1 & 6\\ 1 & 4 & 5 & 2\end{bmatrix}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$$

Simulasi menggunakan matlab :

>> A=[2 3 1 6;1 4 5 2]

A =

2 3 1 6

1 4 5 2

```
>> A+A

ans =

    4    6    2   12

    2    8   10    4

>> A-A

ans =

    0    0    0    0

    0    0    0    0
```

5.2 Perkalian Matrik

AB syarat → jumlah kolom A = jumlah kolom baris B

$AB \neq BA$

Misal,

$$A = [1\ 2\ 3] \qquad B = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$AB = [1\ 2\ 3] \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 1+4+9 = 14$$

(1x3)(3x1)=(1x1)

$$BA = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} [1\ 2\ 3] = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix}$$

Operasi perkalian matrik dalam MATLAB dilakukan dengan simbol *

Simulasi menggunakan matlab :

```
>> A = [1 2 3]

A =

    1    2    3

>> B = [1;2;3]

B =

    1

    2

    3
```

```
>> A*B

ans =

   14

>> B*A

ans =

    1    2    3

    2    4    6

    3    6    9
```

6. Membuat Grafik

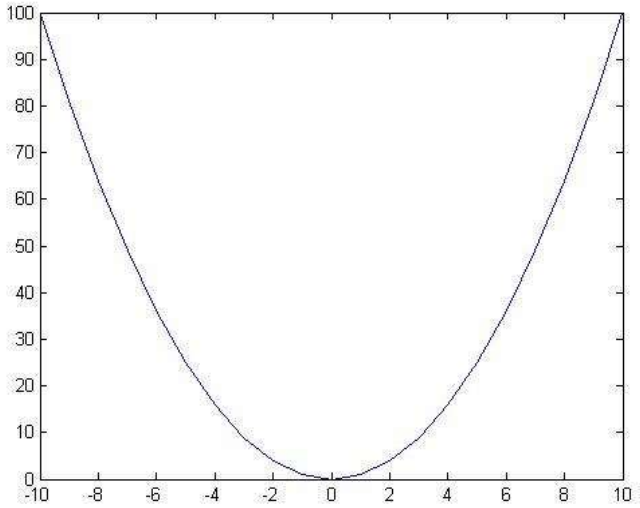
Matlab mempunyai bermacam-macam fungsi untuk menampilkan grafik, dimana setiap fungsi memiliki perbedaan dalam menskalakan garis sumbu. Setiap menerima inputan dalam bentuk vector atau matriks, matlab akan menskalakan secara otomatis.

6.1 Plot

Fungsi plot digunakan untuk menggambar grafik 2D dengan skala linear pada kedua sumbunya. Contoh:

```
>> x=-10:10;
>> y=x.^2;
>> plot(x,y)
```

Hasilnya akan tampak sebagai berikut:

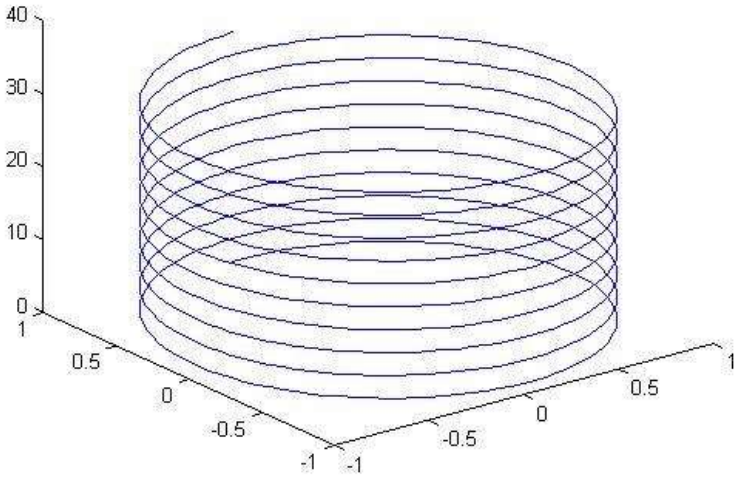


6.2 Plot3

Fungsi plot3 digunakan untuk menampilkan grafik 3 dimensi. Plot3 memerlukan 3 argumen dengan bentuk plot3(x,y,z), dimana x, y, z merupakan 3 bagian vector yang sama panjang.

Contoh:
>> t=0:pi/100:10*pi;
>> plot3(sin(2*t), cos(2*t), t)

Hasilnya akan tampak sebagai berikut:

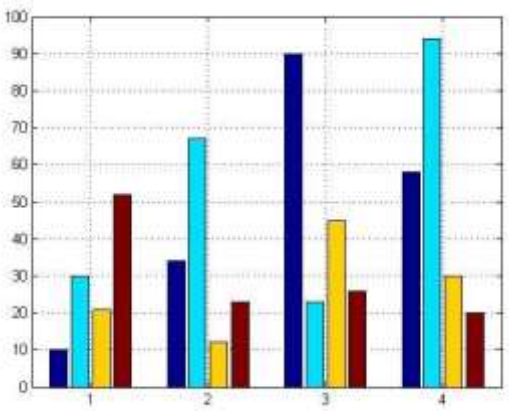


6.3 Bar

Fungsi bar digunakan untuk menampilkan data yang berbentuk vector maupun matriks. grafIk bar digunakan untuk menampilkan sekumpulan data selama kurun waktu terentu dan cocok untuk menampilkan data dalam bentuk diskrit.

Contoh :
>> t=[10 30 21 52; 34 67 12 23; 90, 23, 45, 26; 58 94 30 20];
>> bar(t)
>>grid on

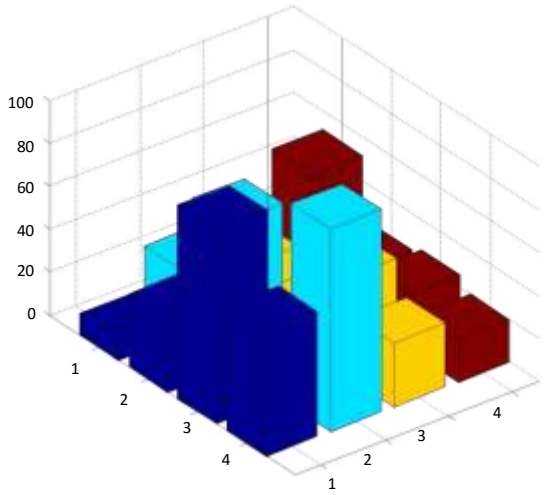
Hasilnya akan tampak sebagai berikut:



Matlab juga menyediakan dalam bentuk 3 dimensi, yaitu bar3. Missal grafik diatas disajikan dalam bar3, maka kita ketik perintah berikut:

>>bar3

Hasilnya akan tampak sebagai berikut:

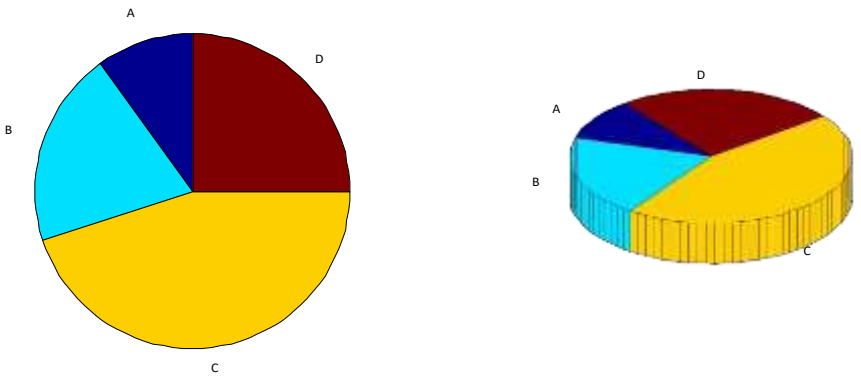


6.4 Pie

Fungsi pie digunakan untuk menampilkan data secara prosentase, dimana setiap elemen data akan dibandingkan dengan penjumlahan seluruh data yang ada. Grafik pie dapat disajikan dalam bentuk 2 dimensi maupun 3 dimensi.
Contoh:

>> x=[10 20 45 25];
>> pie(x, {'A', 'B', 'C', 'D') → pie 2 dimensi
>> pie3(x, {'A', 'B', 'C', 'D') → pie 3 dimensi

Perintah diatas akan menghasilkan gambar sebagai berikut:



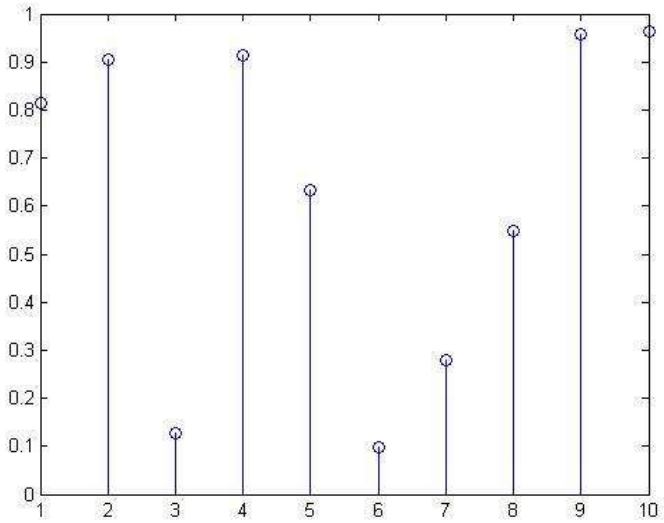
6.5 Stem

Fungsi stem cocok digunakan untuk menampilkan data dalam bentuk diskrit.

Contoh:

```
>> x=1:10;
>> y=rand(1,10);
>> stem(x,y)
```

Perintah diatas akan menghasilkan gambar sebagai berikut:



6.6 Grafik 3D

Perintah menggambar grafik 3D

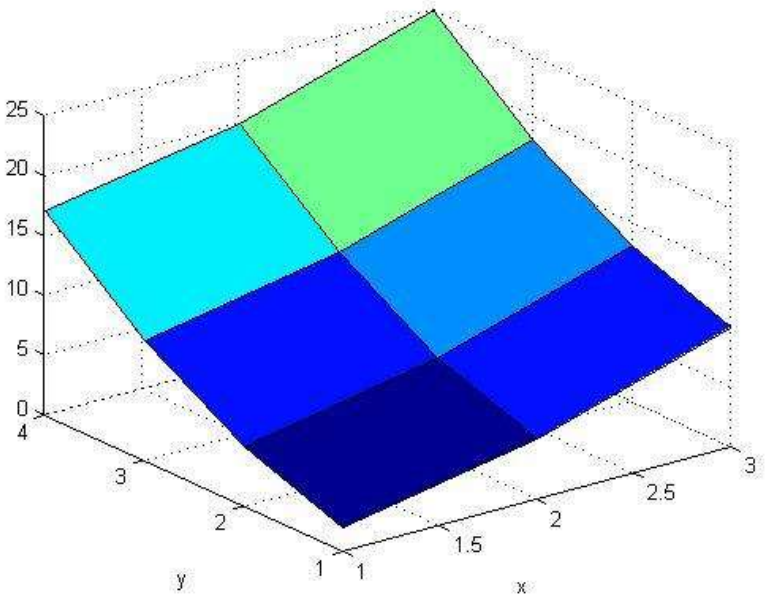
surf (x,y,z)

Misalkan :

x	y	Z(x=1)	Z(x=2)	Z(x=3)
1	1	2	5	10
2	2	5	8	13
3	3	10	13	18
	4	17	20	25

```
>> x = [1 2 3]

x =

    1    2    3
```

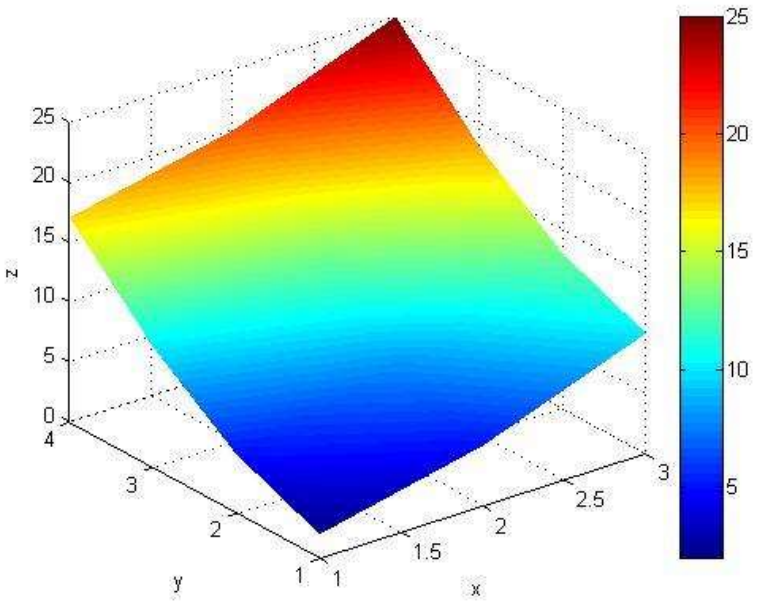
```
>> y = [1 2 3 4]

y =

    1    2    3    4

>> z = [2 5 10;5 8 13;10 13 18;17 20 25]

z =

     2     5    10
     5     8    13
    10    13    18
    17    20    25

>> surf (x,y,z)
>> xlabel ('x')
>> ylabel ('y')
>> zlabel ('z')
```



Untuk memperindah tampilan dan memudahkan penafsiran grafik, bisa dilakukan dengan menambahkan legenda warna, ketikkan perintah berikut :

```
>> shading interp
>> colorbar
```

Maka gambar yang dihasilkan akan memiliki tampilan sebagai berikut :

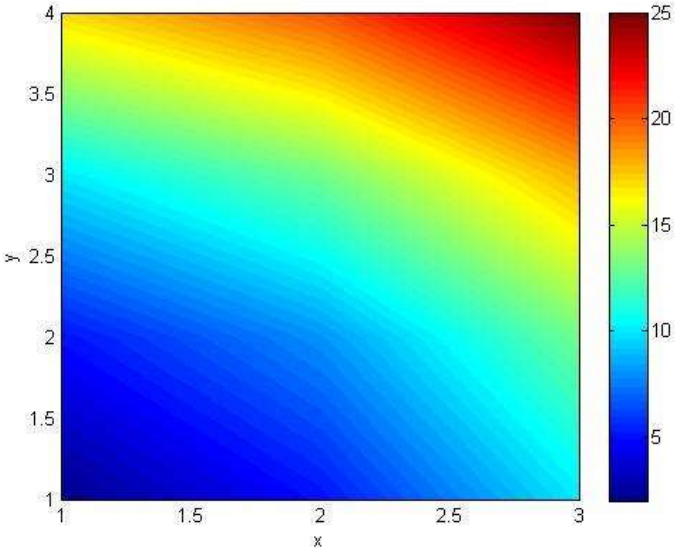


6.7 Grafik 3D semu

Jika penafsiran grafik 3D seperti digambarkan pada poin 6.6 masih dirasakan sulit, MATLAB telah menyediakan perintah untuk mengubah grafik 3D menjadi grafik 2D. Caranya dengan mengetikkan perintah berikut :

```
pcolor (x,y,z)
>> xlabel ('x')
>> ylabel ('y')
>> zlabel ('z')
>> shading interp
>> colorbar
```

Maka grafik yang dihasilkan akan menjadi seperti berikut :



7. Sinyal-Sinyal Dasar Dalam Pengolahan Sinyal Digital

Pada bagian ini akan dicontohkan cara menghasilkan bentuk sinyal-sinyal dasar, diantaranya adalah :

1. Impulse Signal
2. Step Signal
3. Ramp Signal
4. Square Wave
5. Sine Wave
6. Cosine Wave
7. Circle
8. Sawtooth wave
9. Triangular wave
10. Exponentially Growing Signal
11. Exponentially decaying Signal

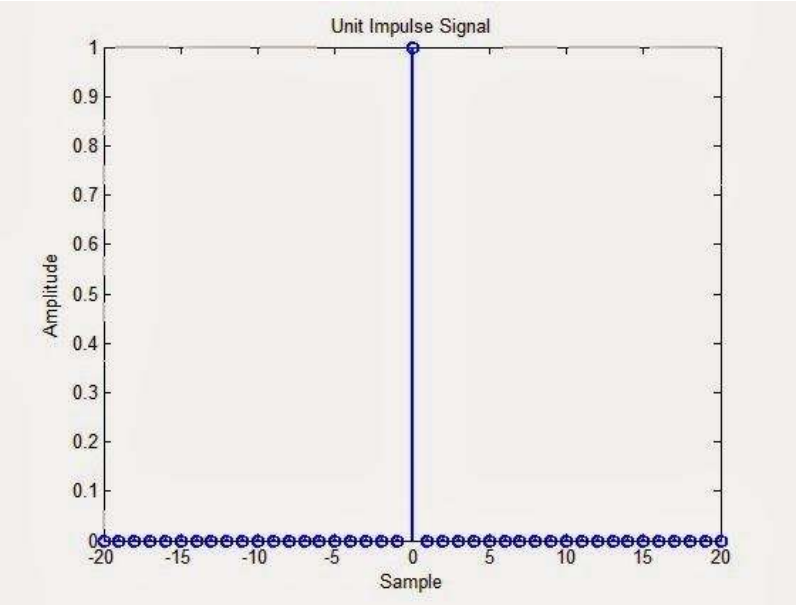
Point untuk diperhatikan :

- Pada contoh ini digunakan kedua jenis sinyal baik itu diskrit maupun kontinu.
- ***stem*** digunakan untuk plot sinyal diskrit

- ***plot*** digunakan untuk plot sinyal kontinu
- ***filled*** digunakan untuk mengisi lingkaran kecil dengan warna
- ***Linewidth*** digunakan untuk menjelaskan ketebalan garis yang digunakan dalam plot

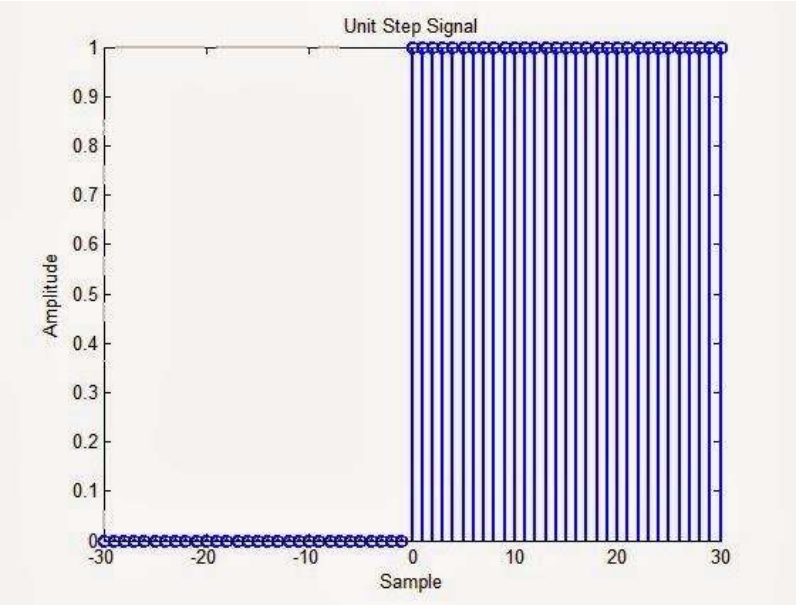
7.1 Impulse Signal

```
clc;
clear all;
close all;
n=-20:20; % specify index n
delta=(n==0);          % define the delta sequence
stem(n,delta,'LineWidth',2) % plot the delta sequence
xlabel('Sample');
ylabel('Amplitude');
title('Unit Impulse Signal');
```



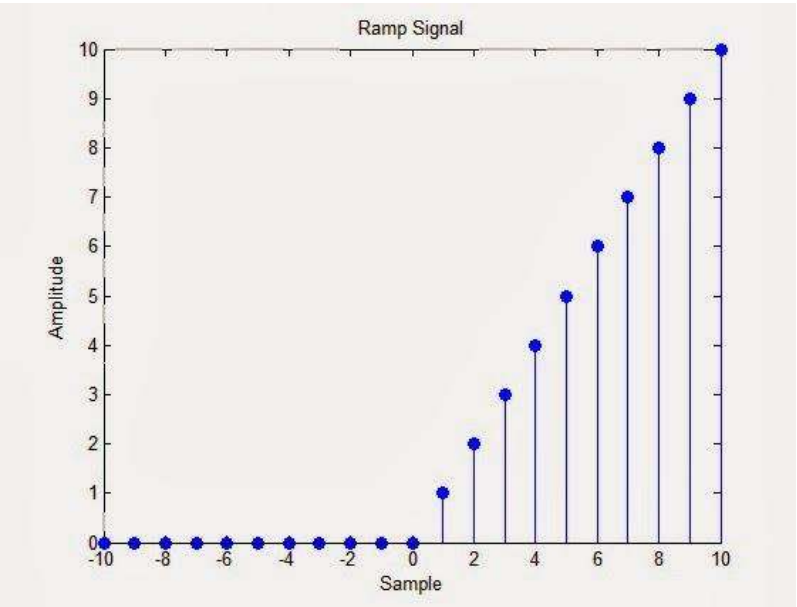
7.2 Step Signal

```
clc;
clear all;
close all;
n=-30:30; % specify index n
step_sig=(n>=0); % define the unit step sequence
stem(n, step_sig,'LineWidth',2) % plot the unit step sequence
xlabel('Sample');
ylabel('Amplitude');
title('Unit Step Signal');
```



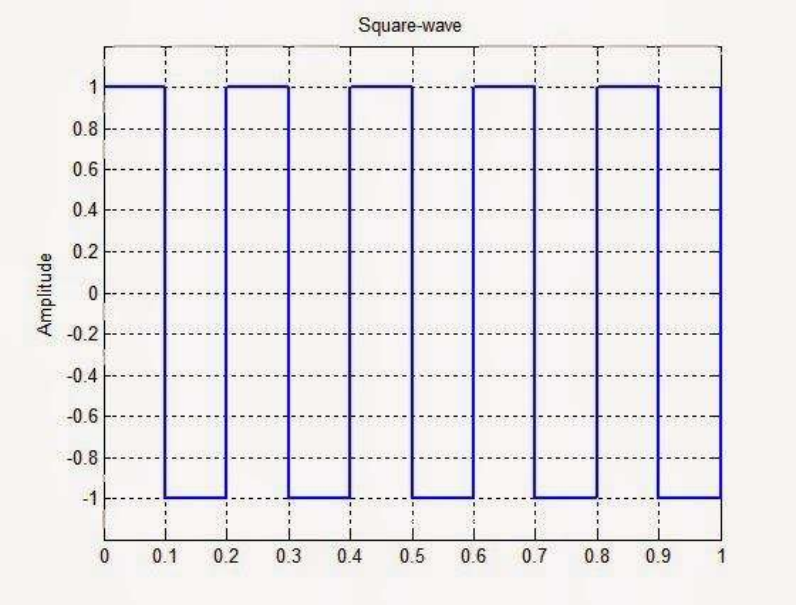
7.3 Ramp Signal

```
n=-10:10; % define index n
ramp=n.*(n>=0); % define a ramp
stem(n,ramp, 'filled')
xlabel('Sample');
ylabel('Amplitude');
title('Ramp Signal');
```



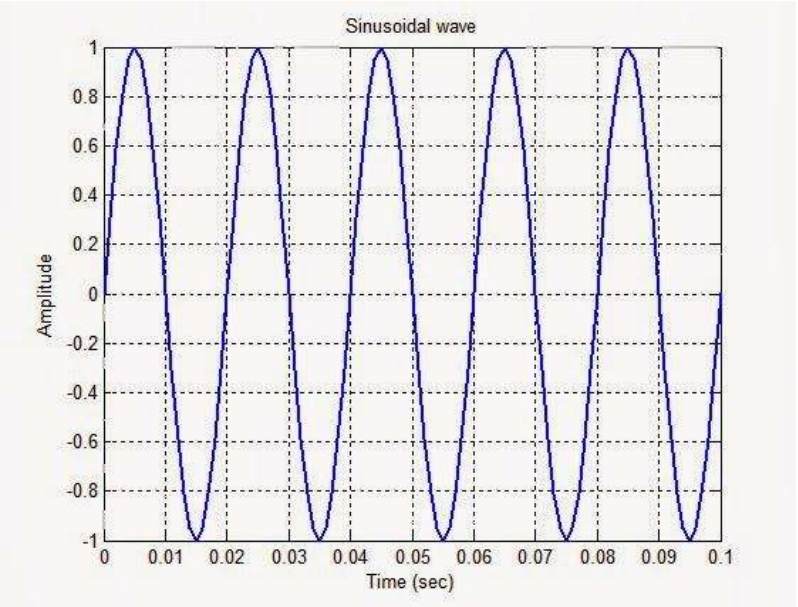
7.4 Square Wave

```
t=(0:0.001:1) % time base
x=square(2*pi*5*t); % squarewave generator
plot(t,x,'LineWidth',2);grid % plot squarewave
axis([0 1 -1.2 1.2]); % scale axes
title('Square-wave') % add title
ylabel('Amplitude'); % label
```



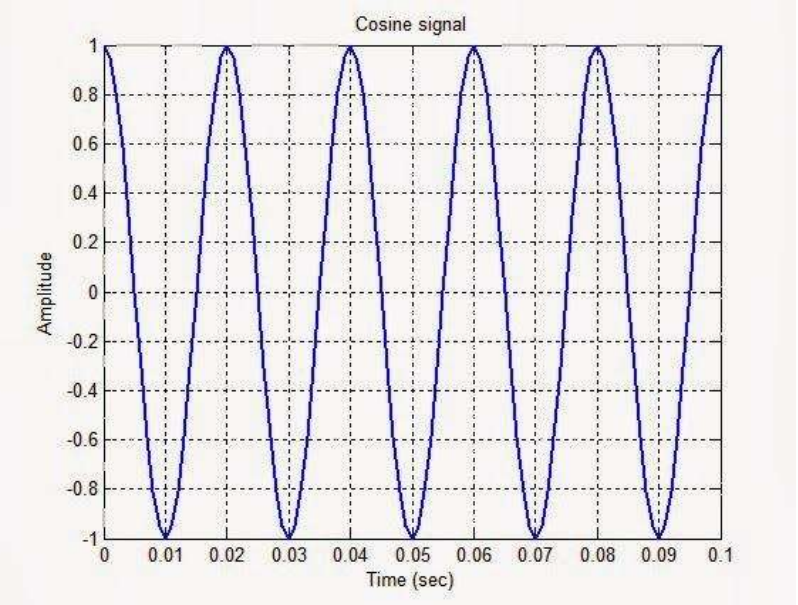
7.5 Sine Wave

```
Fs=1000; % sampling frequency
Ts=1/Fs; % sampling interval
t=0:Ts:0.1; % sampling instants
x=sin(2*pi*50*t); % signal vector
plot(t,x,'LineWidth',2);grid % plot the signal
xlabel('Time (sec)') % add label to the horizontal axis
ylabel('Amplitude') % add label to the vertical axis
title('Sinusoidal wave') % add title to the plot
```

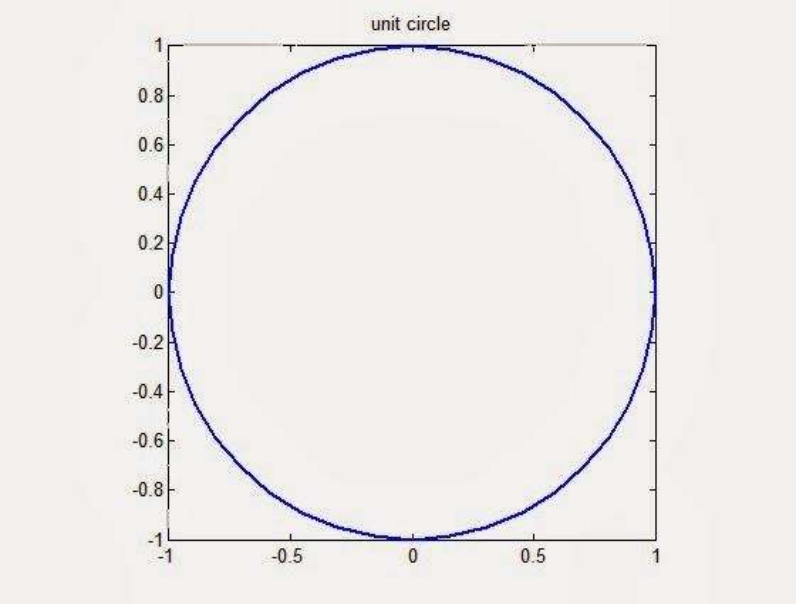


7.6 Cosine Wave

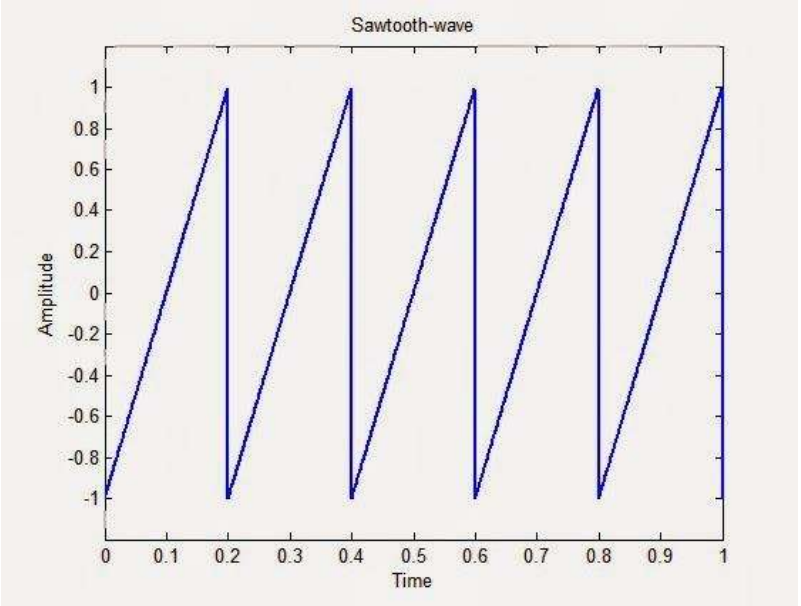
```
Fs=1000; % sampling frequency
Ts=1/Fs; % sampling interval
t=0:Ts:0.1; % sampling instants
x=cos(2*pi*50*t); % signal vector
plot(t,x,'LineWidth',2);grid % plot the signal
xlabel('Time (sec)') % add label to the horizontal axis
ylabel('Amplitude') % add label to the vertical axis
title('Cosine signal') % add title to the plot
```



7.7 Circle t=0:pi/20:2*pi;
plot(sin(t),cos(t),'lineWidth',2)
axis square
title('unit circle')

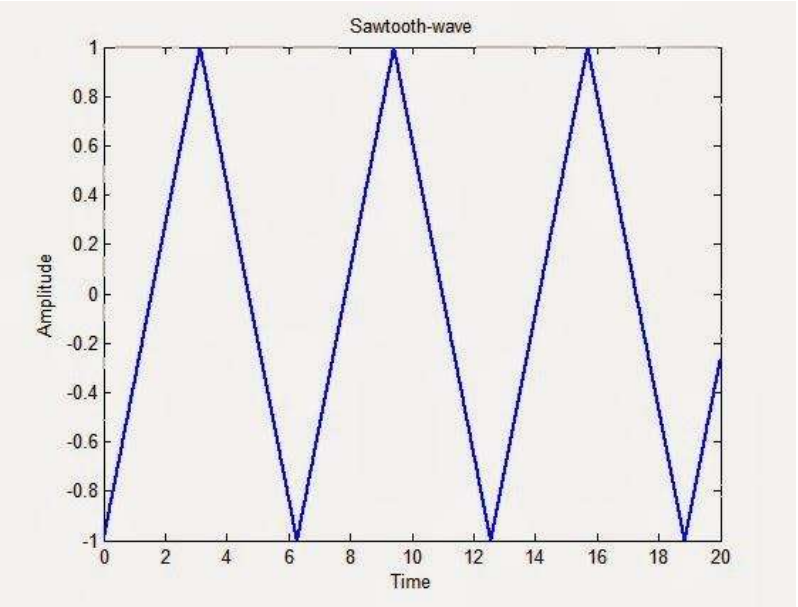


7.8 Sawtooth wave
t=(0:0.001:1) % time
z= sawtooth(2*pi*5*t); % sawtooth wave
plot(t,z,'LineWidth',2);% plot sawtooth wave
title('Sawtooth-wave') % add title
ylabel('Amplitude') % label the vertical axis
xlabel('Time') % label the horizontal axis
axis([0 1 -1.2 1.2])



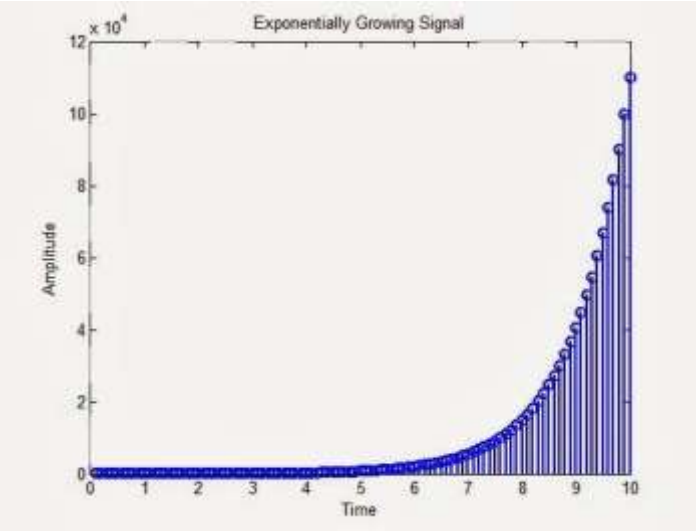
7.9 Triangular wave

```
clc;clear all;
N=20; % number of samples
x=0:0.1/N:20;
tri=sawtooth(x,0.5); % width=0.5 for Triangular signal
plot(x,tri,'LineWidth',2);% plot sawtooth wave
title('Sawtooth-wave') % add title
ylabel('Amplitude') % label the vertical axis
xlabel('Time') % label the horizontal axis
```



7.10 Exponentially Growing Signal

```
N=10; %Number of samples
A=5;  %Maximum Amplitude
t=0:0.1:N;
x=A*exp(t);
figure,stem(t,x,'LineWidth',2);
xlabel('Time');
ylabel('Amplitude');
title('Exponentially Growing Signal');
```



7.11 Exponentially decaying Signal

```
N=10; %Number of samples
A=5;  %Maximum Amplitude
t=0:0.1:N;
x=A*exp(-t);
figure,stem(t,x,'LineWidth',2);
xlabel('Time');
ylabel('Amplitude');
title('Exponentially Decaying Signal');
```