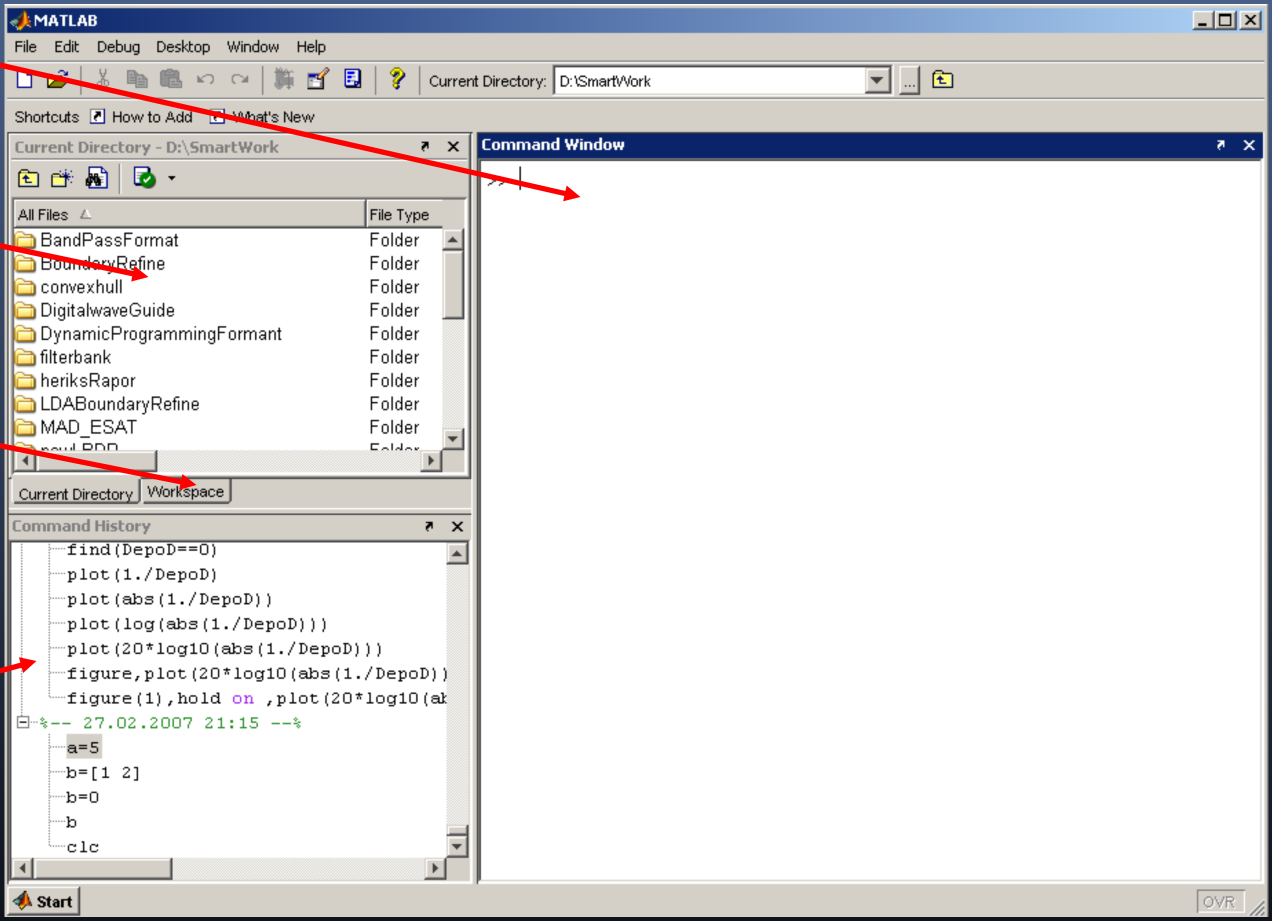
Matlab Screen

- Command Window
  - type commands
- Current Directory
  - View folders and m-files
- Workspace
  - View program variables
  - Double click on a variable to see it in the Array Editor
- Command History
  - view past commands
  - save a whole session using diary


MATLAB
File Edit Debug Desktop Window Help
Current Directory: D:\SmartWork
Shortcuts How to Add What's New
Current Directory - D:\SmartWork
Command Window
Current Directory
Workspace
Command History
Start
OVR

All Files	File Type
BandPassFormat	Folder
BoundaryRefine	Folder
convexhull	Folder
DigitalwaveGuide	Folder
DynamicProgrammingFormant	Folder
filterbank	Folder
heriksRapor	Folder
LDABoundaryRefine	Folder
MAD_ESAT	Folder
paul_RDD	Folder

```
find(DepoD==0)
plot(1./DepoD)
plot(abs(1./DepoD))
plot(log(abs(1./DepoD)))
plot(20*log10(abs(1./DepoD)))
figure,plot(20*log10(abs(1./DepoD))
figure(1),hold on ,plot(20*log10(ak
%-- 27.02.2007 21:15 --%
a=5
b=[1 2]
b=0
b
clc
```

```
>>
```

Variables

- No need for types. i.e.,

~~int a;~~
~~double b;~~
~~float c;~~

- All variables are created with double precision unless specified and they are matrices.

Example:

```
>>x=5;
>>x1=2;
```

- After these statements, the variables are 1x1 matrices with double precision

Vector, Matrix

- a vector `A = [1 2 5 1]`

```
A =
     1     2     5     1
```

- a matrix `B = [1 2 3; 5 1 4; 3 2 -1]`

```
B =
    1      2      3
     5      1      4
     3      2     -1
```

- transpose `C = A’`

```
C =
     1
     2
     5
     1
```

Matrix index

- The element in row i and column j of A is given by A(i,j).
- The matrix indices begin from 1 (not 0 (as in C))
- So to compute the sum of the elements in the 3rd column of B, we have: B(1,3)+B(2,3)+B(3,3)
  - Which produces: ans = 6
  - When you do not specify an output variable, MATLAB stores the results of a calculation in variable *ans*

The Colon Operator

- For example: a=[1:10]

  is a row vector containing the integers from 1 to 10:

  ```
  1   2   3   4   5   6   7   8   9   10
  ```

- To obtain non-unit spacing, specify an increment. For example: 100:-7:50 will give you

  ```
  100   93   86   79   72   65   58   51
  ```

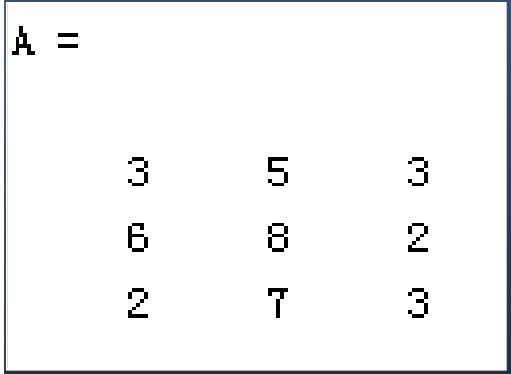
- B = [1:4; 5:8]

  ```
  B =
  1    2    3    4
  5    6    7    8
  ```

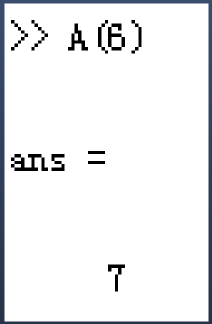
- Subscript expressions involving colons refer to portions of a matrix. For example: A(1:k,j) refers to the first k elements of the jth column of A.

Indexing, colon use

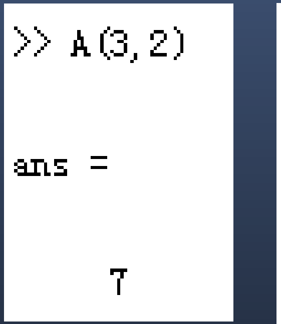
Given:



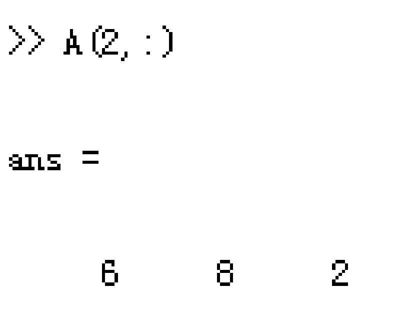
```
A =

     3     5     3
     6     8     2
     2     7     3
```



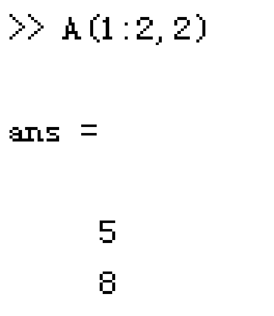
```
>> A(6)

ans =

     7
```



```
>> A(3,2)

ans =

     7
```



```
>> A(2,:)

ans =

     6     8     2
```



```
>> A(1:2,2)

ans =

     5
     8
```

A(-2), A(0)

Error: ??? Subscript indices must either be real positive integers or logicals.

A(4,2)

Error: ??? Index exceeds matrix dimensions.

Generating Vectors from functions

- zeros(M,N) MxN matrix of zeros

```
x = zeros(1,3)
x =
  0     0     0
```

- ones(M,N) MxN matrix of ones

```
x = ones(1,3)
x =
  1     1     1
```

- rand(M,N) MxN matrix of uniformly distributed random numbers on (0,1)

```
x = rand(1,3)
x =
0.9501   0.2311 0.6068
```

Concatenation of Matrices

- x = [1 2], y = [4 5], z=[0 0]

```
A = [ x y]

    1    2    4    5

 B = [x ; y]

       1  2
       4  5
```

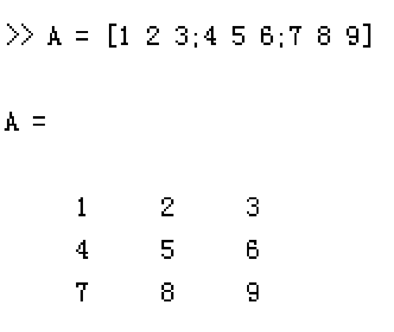
```
    C = [x y ;z]
Error:
??? Error using ==> vertcat CAT arguments dimensions are not consistent.
```

Operators (arithmetic)

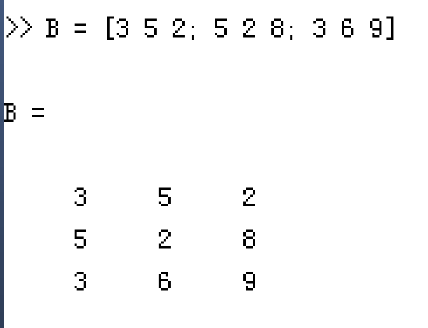
+	addition
-	subtraction
*	multiplication
/	division
^	power
`	complex conjugate transpose

Matrix Operations

Given A and B:

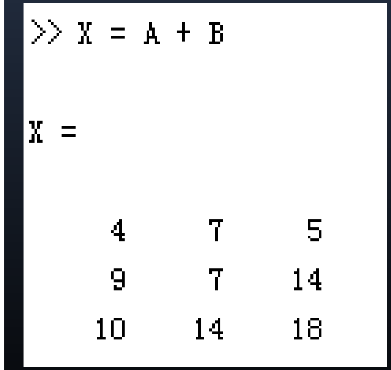


```
>> A = [1 2 3;4 5 6;7 8 9]

A =

     1     2     3
     4     5     6
     7     8     9
```



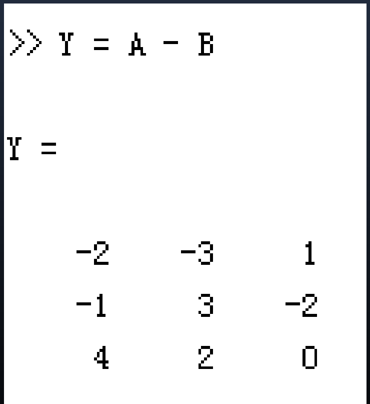
```
>> B = [3 5 2; 5 2 8; 3 6 9]

B =

     3     5     2
     5     2     8
     3     6     9
```

Addition



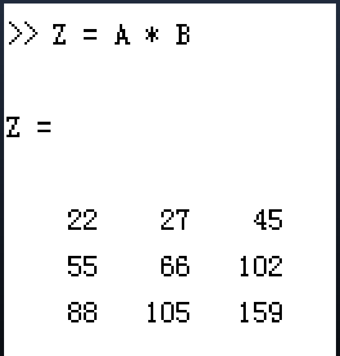
```
>> X = A + B

X =

     4     7     5
     9     7    14
    10    14    18
```

Subtraction



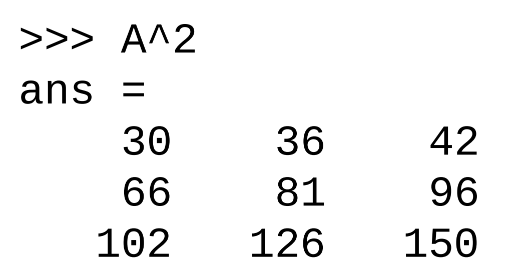
```
>> Y = A - B

Y =

    -2    -3     1
    -1     3    -2
     4     2     0
```

Product



```
>> Z = A * B

Z =

    22    27    45
    55    66   102
    88   105   159
```

A*A



```
>>> A^2
ans =
     30     36     42
     66     81     96
    102    126    150
```

…More matrix operations

- **Performing operations to every entry in a matrix**

```
>>> A=[1 2 3;4 5 6;7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9

>>> A-2
ans =
    -1     0     1
     2     3     4
     5     6     7

>>> A.^2
ans =
     1     4     9
    16    25    36
    49    64    81
```

```
>>> A+3
ans =
     4     5     6
     7     8     9
    10    11    12
```

```
>>> A*2
ans =
     2     4     6
     8    10    12
    14    16    18
```

```
>>> A/3
ans =
    0.3333    0.6667    1.0000
    1.3333    1.6667    2.0000
    2.3333    2.6667    3.0000
```

Useful Matrix Functions

`sum(v)`	Compute sum of elements of v
`prod(v)`	Compute product of elements of v
`min(v)`	Minimum element of v
`max(v)`	Maximum element of v
`size(v)`	Gives dimensions of v
`mean(v)`	Mean value of elements in v
`std(v)`	Standard deviation of elements

Standard Deviation, Median, Covariance

If rpm_raw is a 3*N matrix containing values for 3 variables, then:

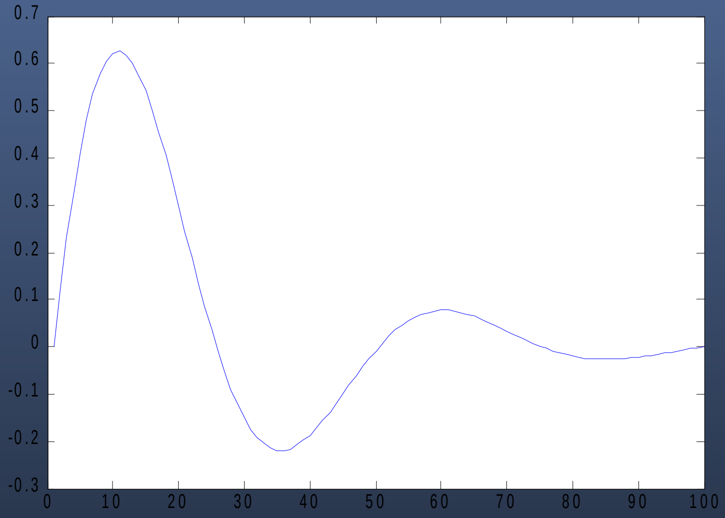
```
>> median(rpm_raw) % median along each column
ans =
        1080       1083.5          1004

>> var(rpm_raw) % variance (deviation from mean)
ans =
       306.4        244.9        356.25

>> std(rpm_raw) % standard deviation along each column (root of
   var)
ans =
      17.504       15.649        18.875

>> cov(rpm_raw) % covariance of the data
ans =
       306.4       -34.76        32.192
      -34.76        244.9       -165.21
      32.192      -165.21        356.25
```

Display Facilities

- plot(.)

Example:

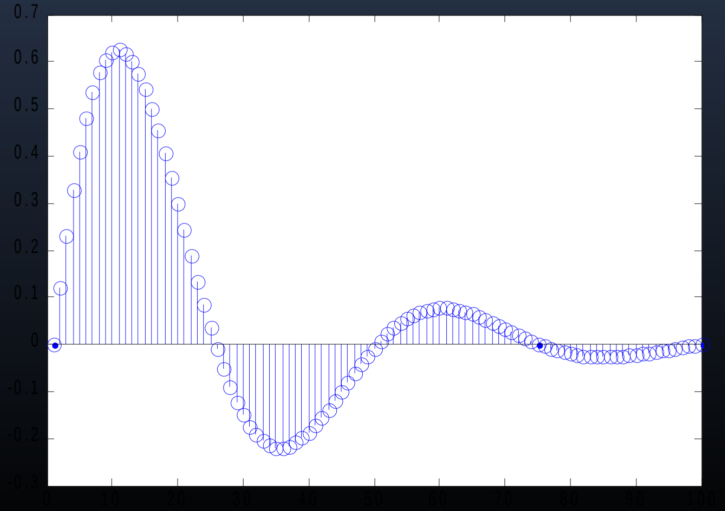
```
>>x=linspace(0,4*pi,100);
>>y=sin(x);
>>plot(y)
>>plot(x,y)
```



- stem(.)

Example:

```
>>stem(y)
>>stem(x,y)
```



Display Facilities

- title(.)

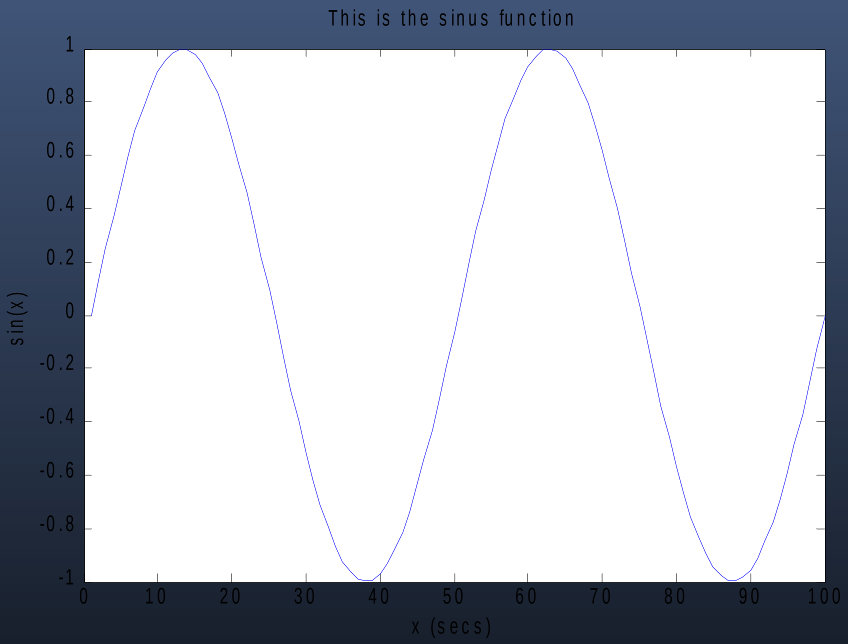
```
>>title('This is the sinus function')
```

- xlabel(.)

```
>>xlabel('x (secs)')
```

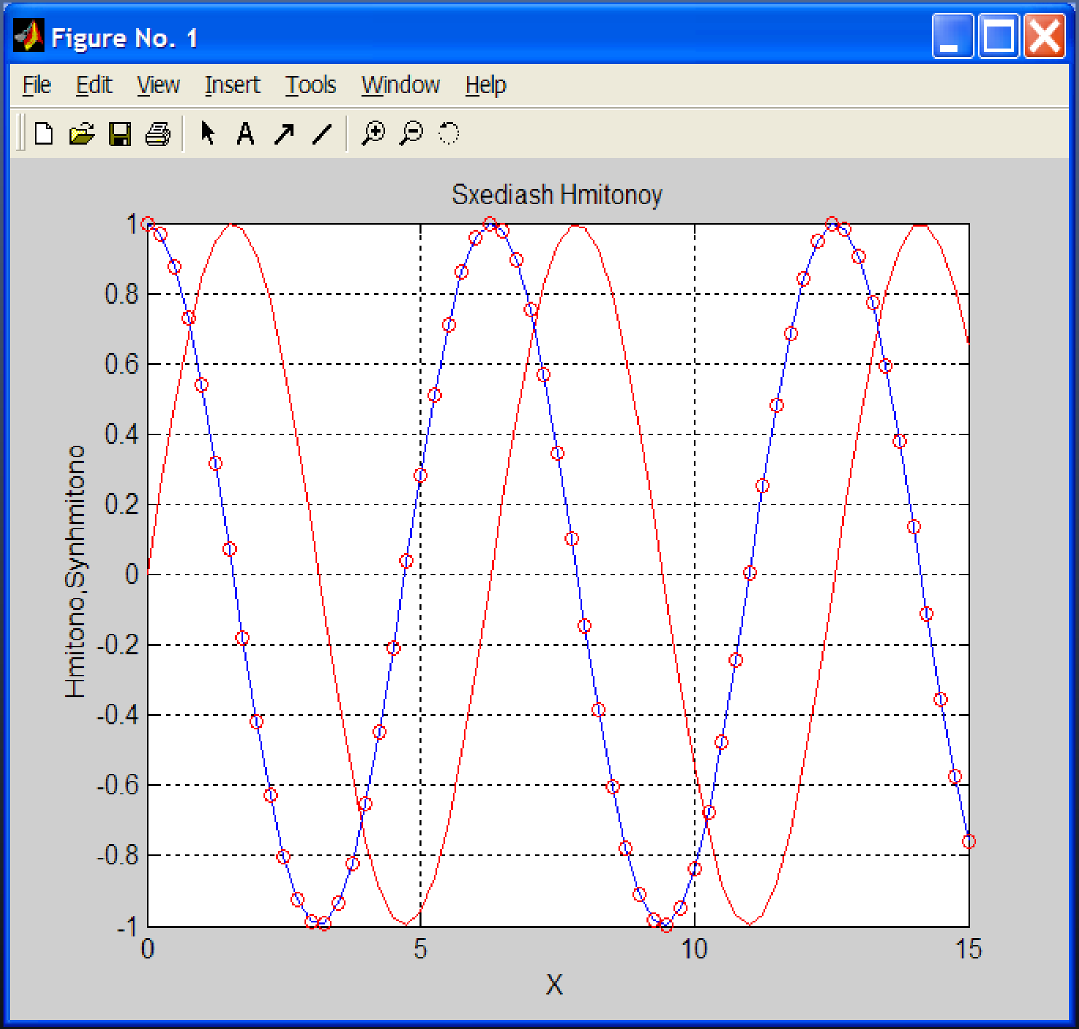
- ylabel(.)

```
>>ylabel('sin(x)')
```



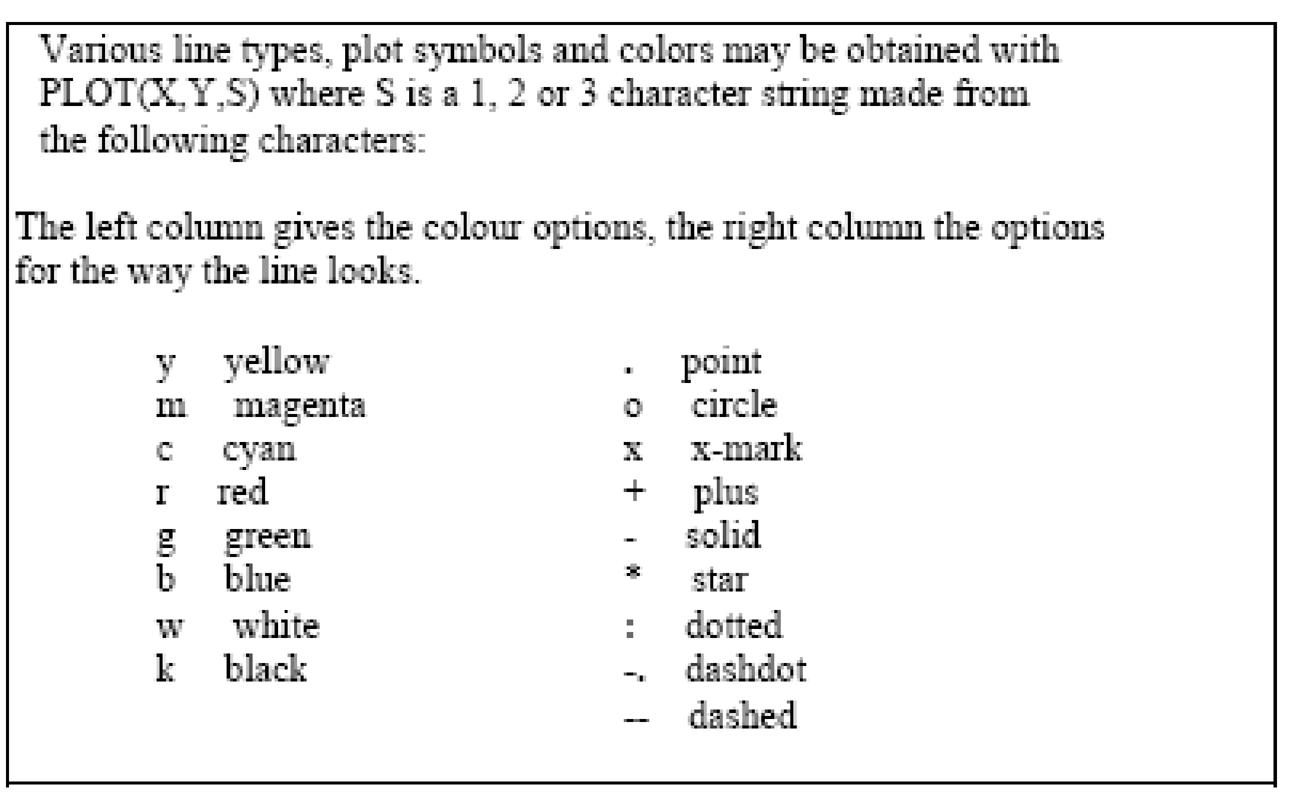
Multiple lines in one graph

```
>> figure(1)

>> clf

>> x = 0:0.25:15;

>> y = sin(x);

>> plot(x,y,’r’)

>> grid on

>> title('Sxediash Hmitonoy')

>> xlabel('X')

>> ylabel(’Hmitono,Synhmitono‘)

>> z = cos(x);

>> hold on

>> plot(x,z)

>> plot(x,z,’or’)
```



Alternatively: *plot(x,y,'r',x,z,'or')*

Line format options



Various line types, plot symbols and colors may be obtained with PLOT(X,Y,S) where S is a 1, 2 or 3 character string made from the following characters:

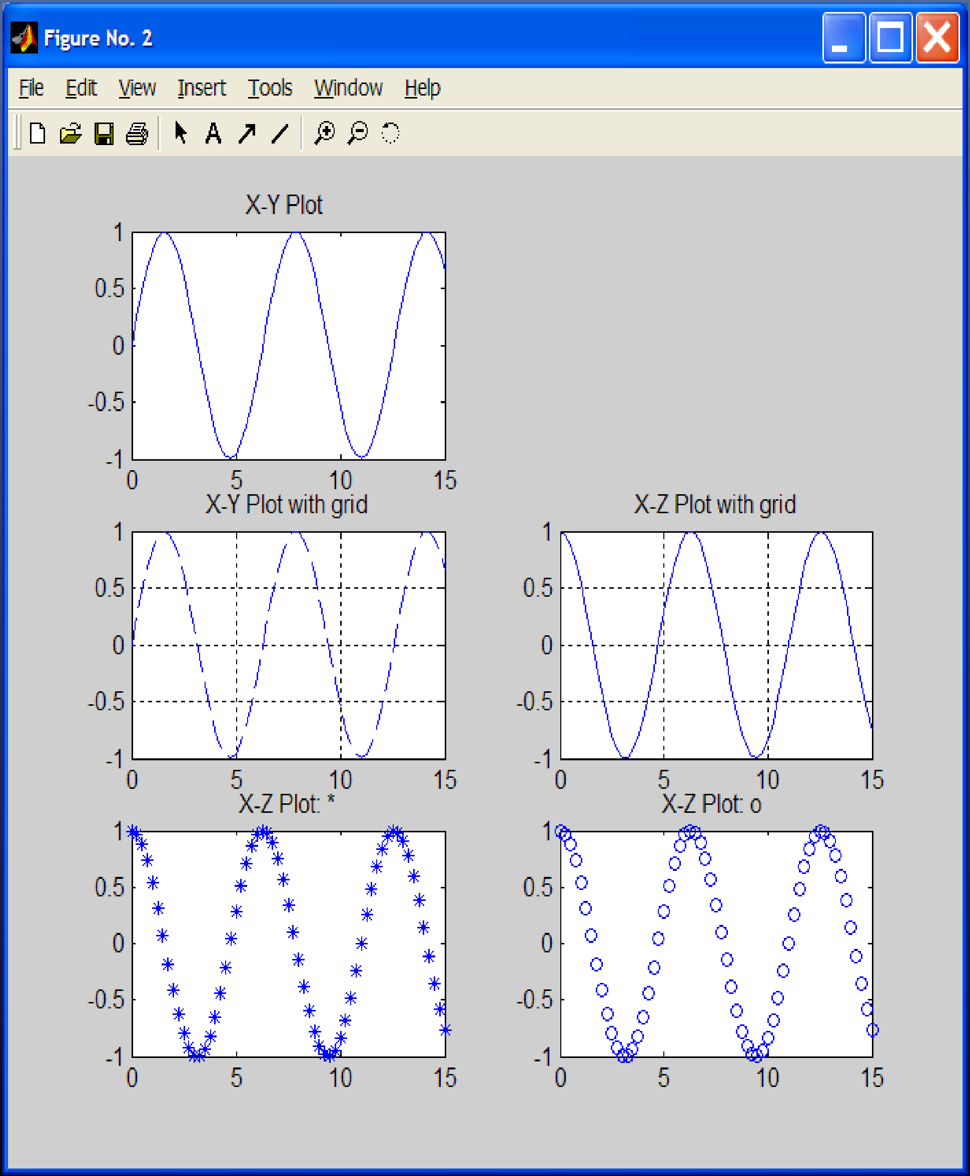
The left column gives the colour options, the right column the options for the way the line looks.

y	yellow	.	point
m	magenta	o	circle
c	cyan	x	x-mark
r	red	+	plus
g	green	-	solid
b	blue	*	star
w	white	:	dotted
k	black	-.	dashdot
		--	dashed

- >> plot(x, y, ’b’) plots the graph in blue as a solid line (default)
- >> plot(x, y, ’bo’) plots graph as blue circles
- >> plot(x, y, ’g:’) graph appears as a dotted green line.
- >> plot(x, y1,’k’, x, y2,’:b’) first curve is a black line and second is blue dots

Multiple plots in one figure

```
figure(2)

subplot(3,2,1)
plot(x,y,'-')
title('X-Y Plot ')

subplot(3,2,3)
plot(x,y,'--')
grid on
title('X-Y Plot with grid')

subplot(3,2,4)
plot(x,z)
grid on
title('X-Z Plot with grid')

subplot(3,2,6)
plot(x,z, 'o')
title('X-Z Plot: o ')

subplot(3,2,5)
plot(x,z, '*')
title('X-Z Plot: *')
```

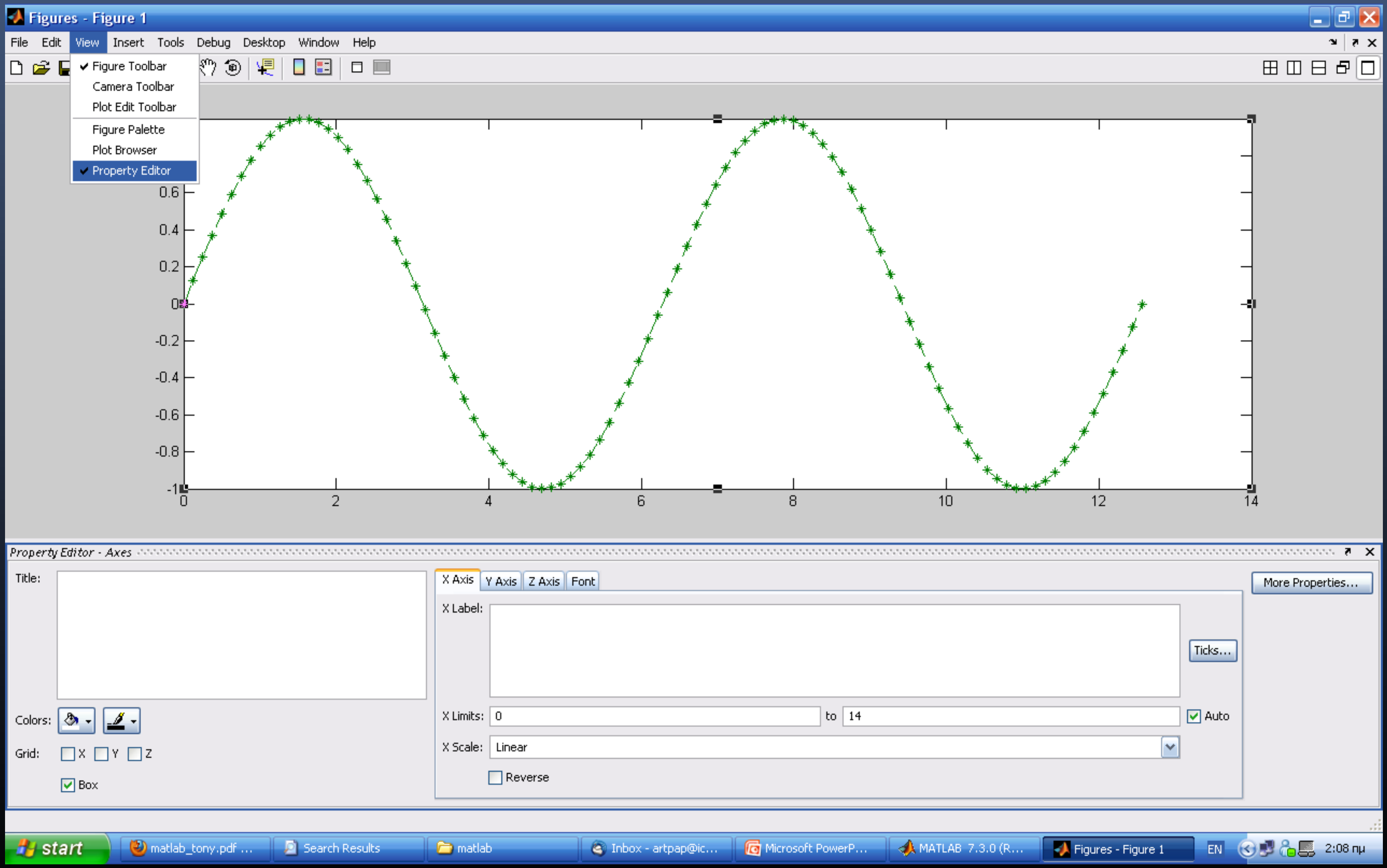


Controlling the Axes

- The axis command lets you to specify your own limits: axis([xmin xmax ymin ymax])
- You can use the axis command to make the axes visible or invisible: axis on / axis off
- The grid command toggles grid lines on and off: grid on / grid off

Using matlab gui

- Select view->property editor form the figure



Title:

Colors:

Grid: □ X □ Y □ Z

☑ Box

X Label:

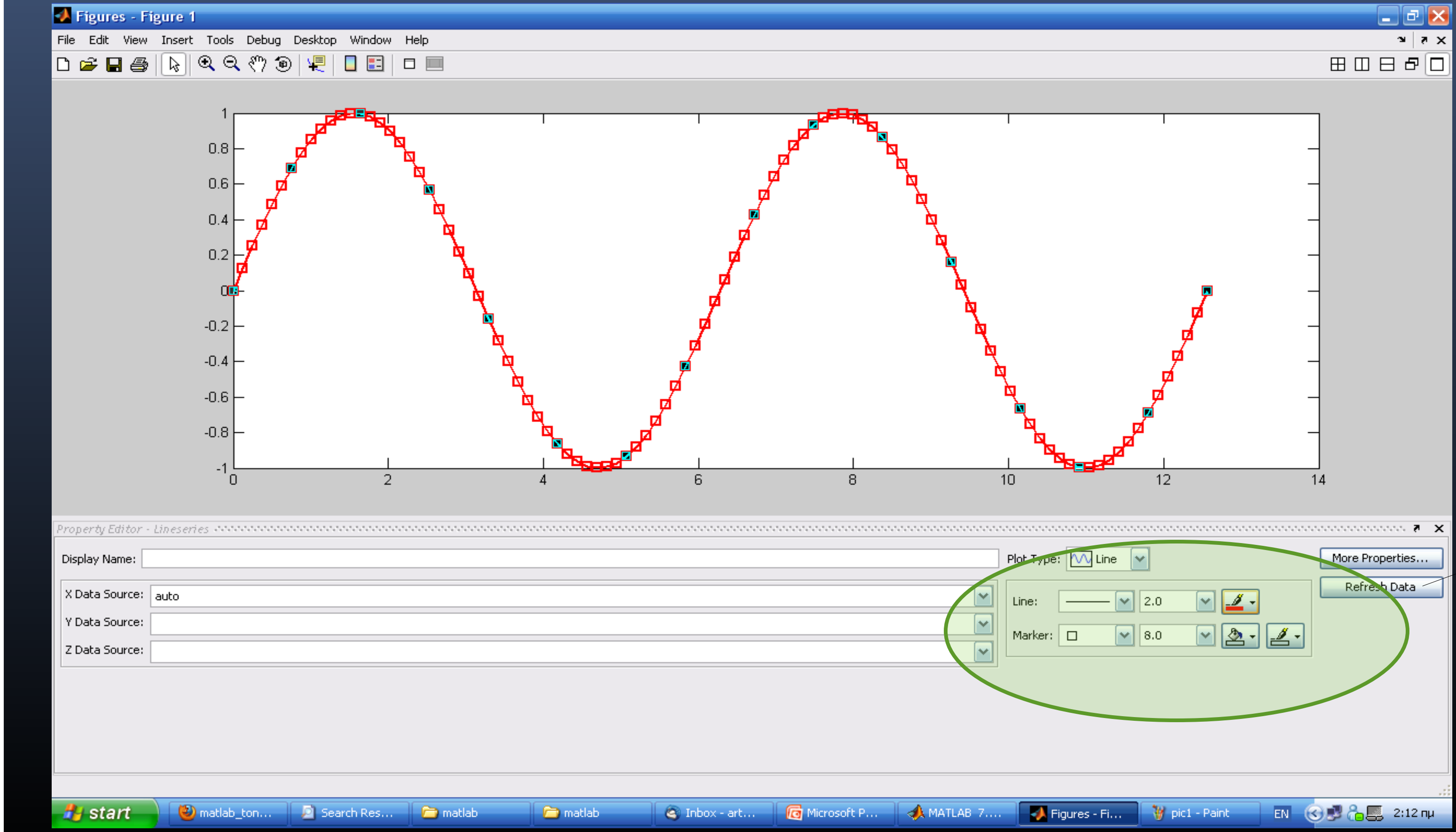
X Limits: 0 to 14 ☑ Auto

X Scale: Linear

□ Reverse

Using matlab gui (cont.)

- Select the line to format it



Display Name:

X Data Source: auto

Y Data Source:

Z Data Source:

Plot Type: Line

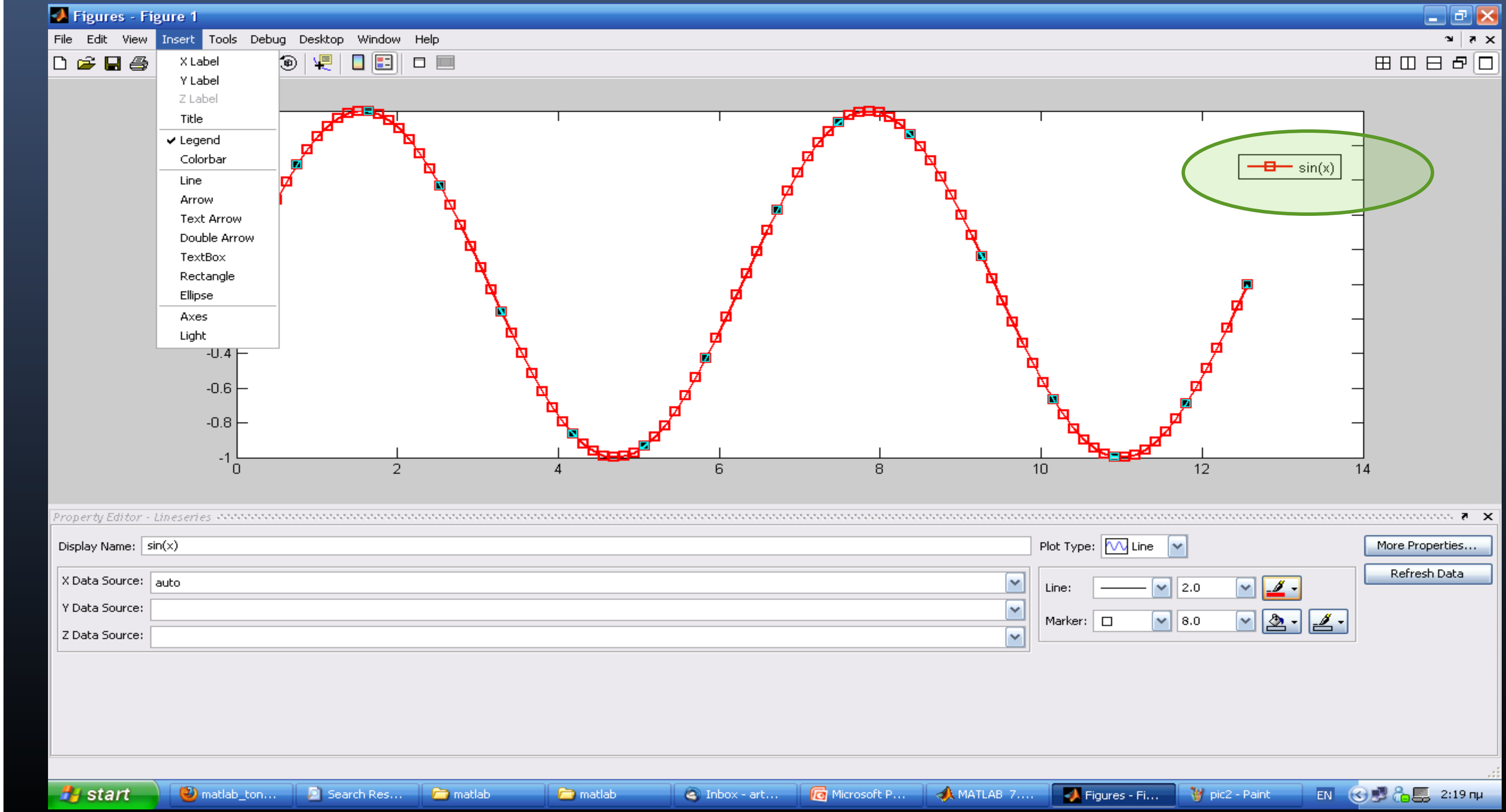
Line: 2.0

Marker: □ 8.0

Available options

Using matlab gui (cont.)

- Enter a display name and insert legend(or title/X label/Y label)



Display Name: sin(x)

X Data Source: auto

Y Data Source:

Z Data Source:

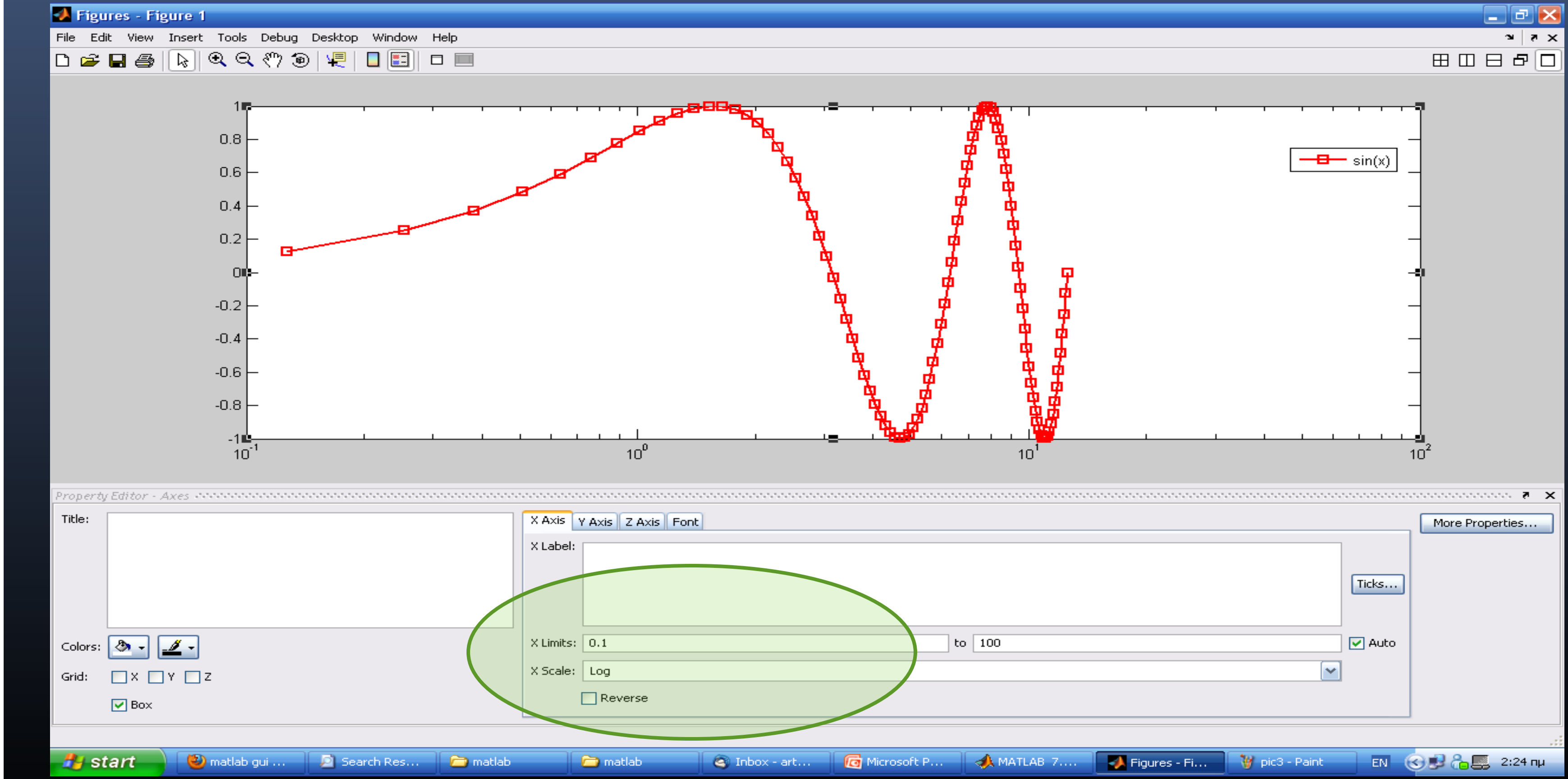
Plot Type: Line

Line: 2.0

Marker: □ 8.0

Using matlab gui (cont.)

- Select X (or Y) axis and change to log scale



Title:

Colors:

Grid: □ X □ Y □ Z

☑ Box

X Label:

X Limits: 0.1 to 100 ☑ Auto

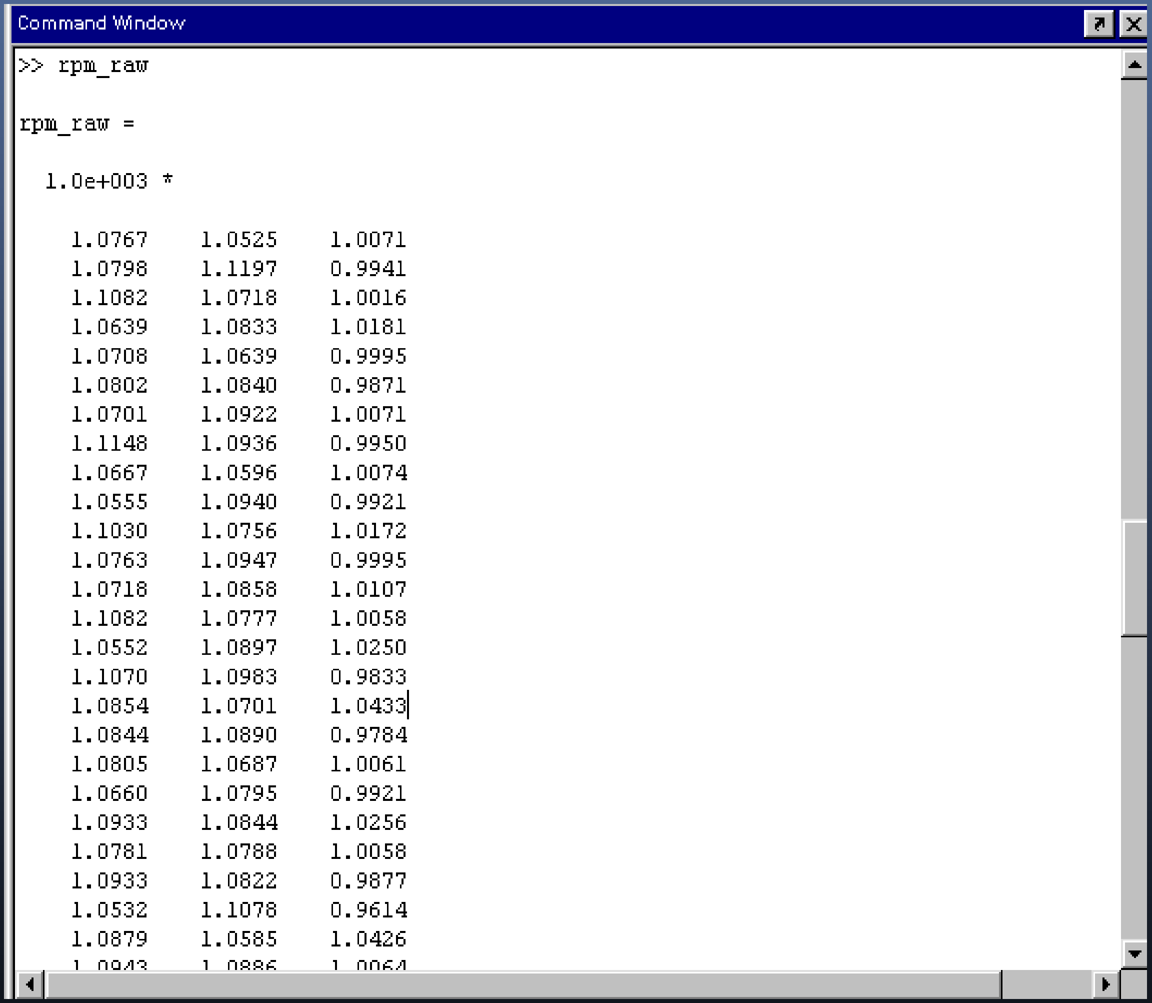
X Scale: Log

□ Reverse

Data Analysis: Histogram

- N=hist(Y) bins the elements of Y into 10 equally spaced containers and returns the number of elements in each container.
- N=hist(Y,M),where M is a scalar uses M bins
- N=hist(Y,X),where X is a vector, returns the distribution of Y among bins with centers specified by X.

Vibration Sensors Data

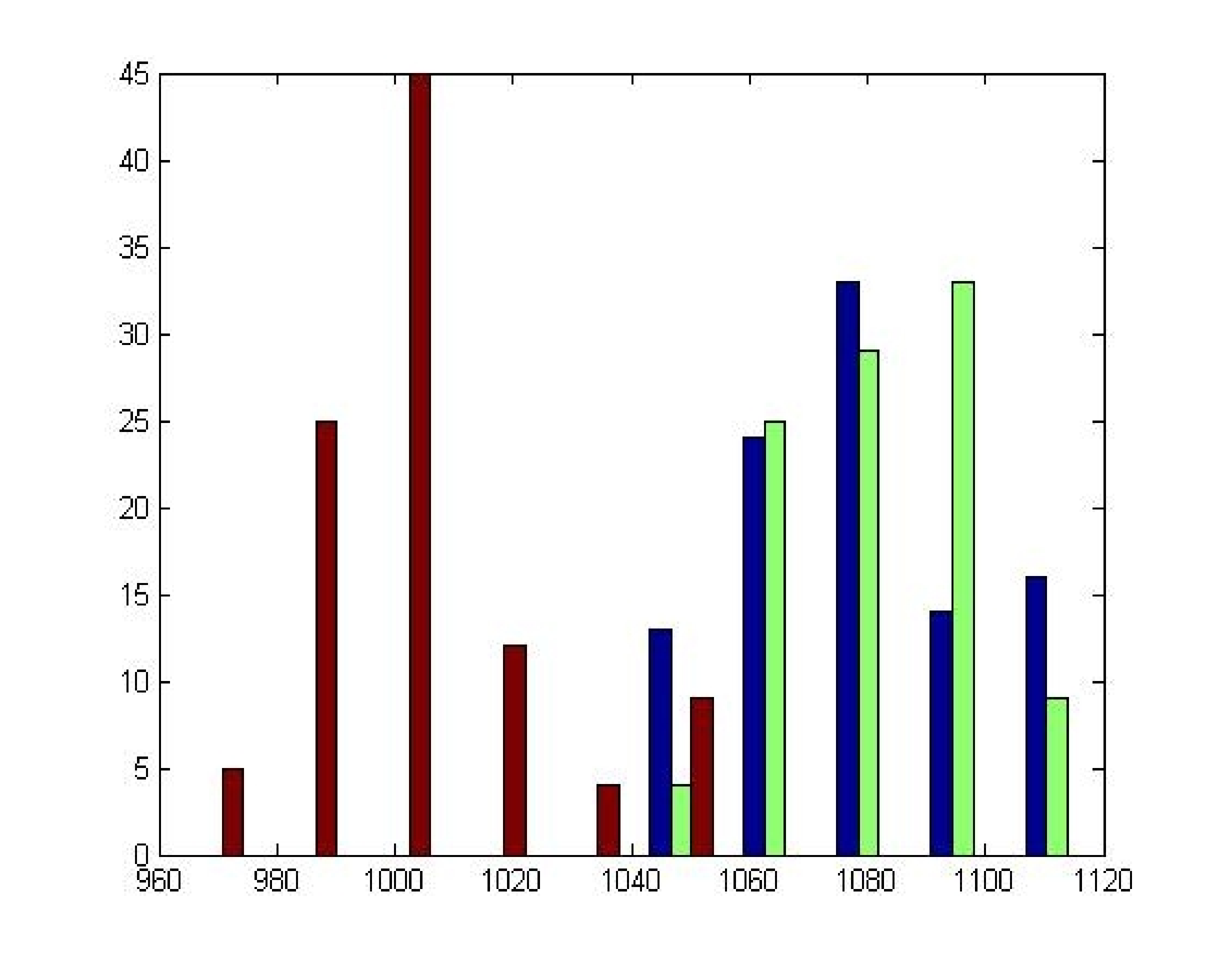


```
>> rpm_raw

rpm_raw =

  1.0e+003 *

    1.0767    1.0525    1.0071
    1.0798    1.1197    0.9941
    1.1082    1.0718    1.0016
    1.0639    1.0833    1.0181
    1.0708    1.0639    0.9995
    1.0802    1.0840    0.9871
    1.0701    1.0922    1.0071
    1.1148    1.0936    0.9950
    1.0667    1.0596    1.0074
    1.0555    1.0940    0.9921
    1.1030    1.0756    1.0172
    1.0763    1.0947    0.9995
    1.0718    1.0858    1.0107
    1.1082    1.0777    1.0058
    1.0552    1.0897    1.0250
    1.1070    1.0983    0.9833
    1.0854    1.0701    1.0433
    1.0844    1.0890    0.9784
    1.0805    1.0687    1.0061
    1.0660    1.0795    0.9921
    1.0933    1.0844    1.0256
    1.0781    1.0788    1.0058
    1.0933    1.0822    0.9877
    1.0532    1.1078    0.9614
    1.0879    1.0585    1.0426
    1.0943    1.0886    1.0064
```

Each column is the raw rpm sensor data from a different sensor used in an instrumented engine test. The rows represent the times readings were made.

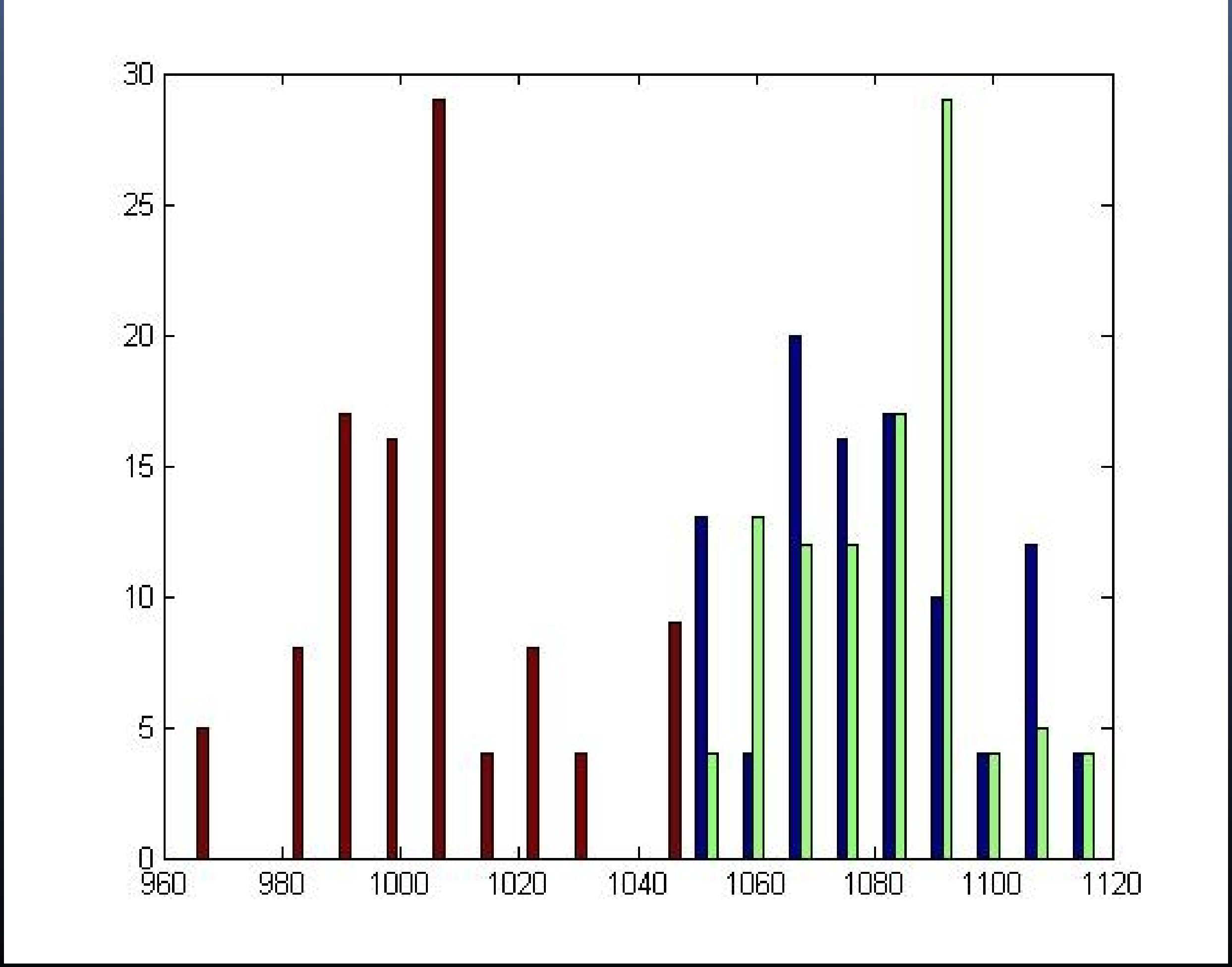
Histogram (cont.)

```
>> hist(rpm_raw) %histogram of the data
```



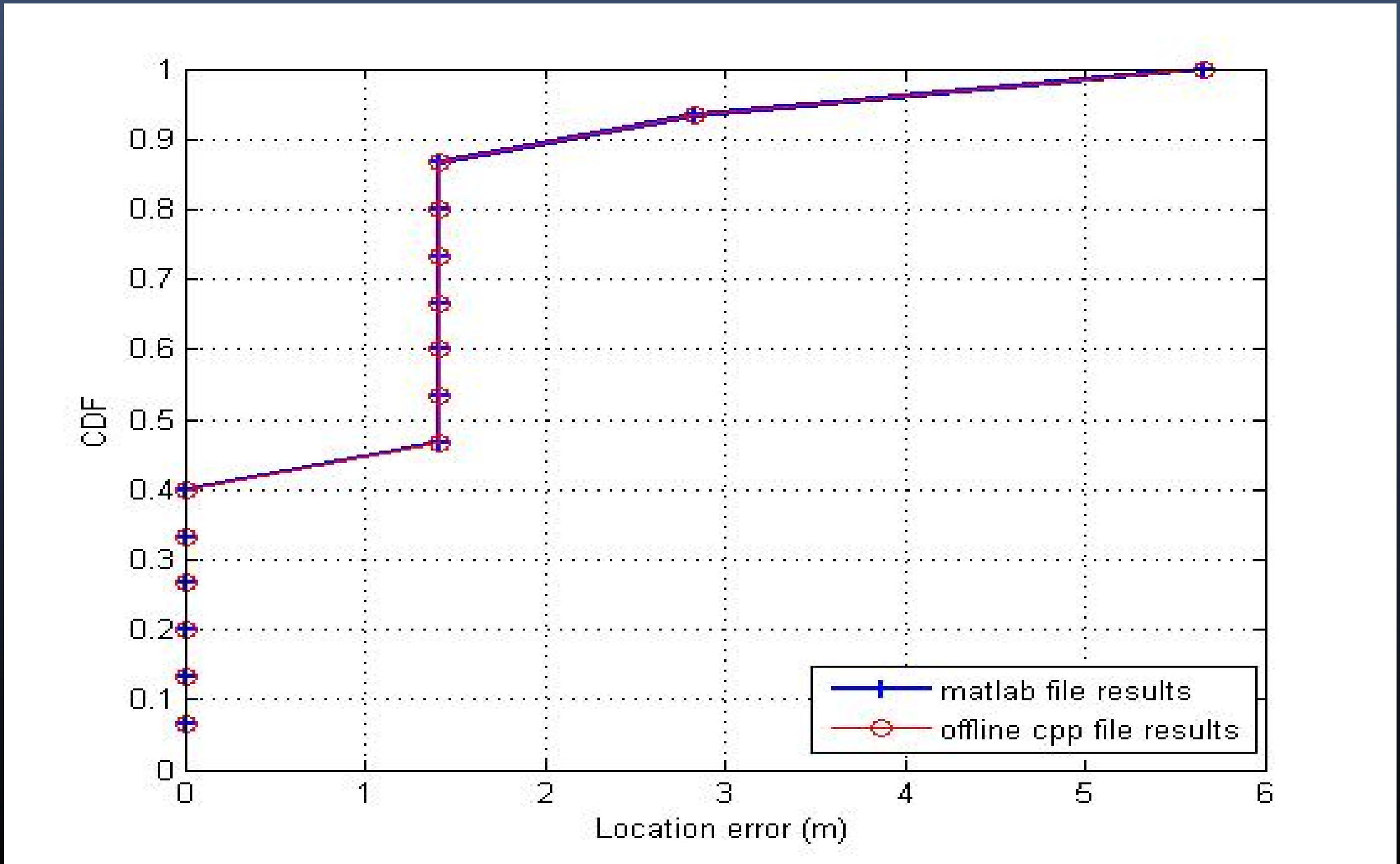
Histogram (cont.)

```
>> hist(rpm_raw, 20) %histogram of the data
```



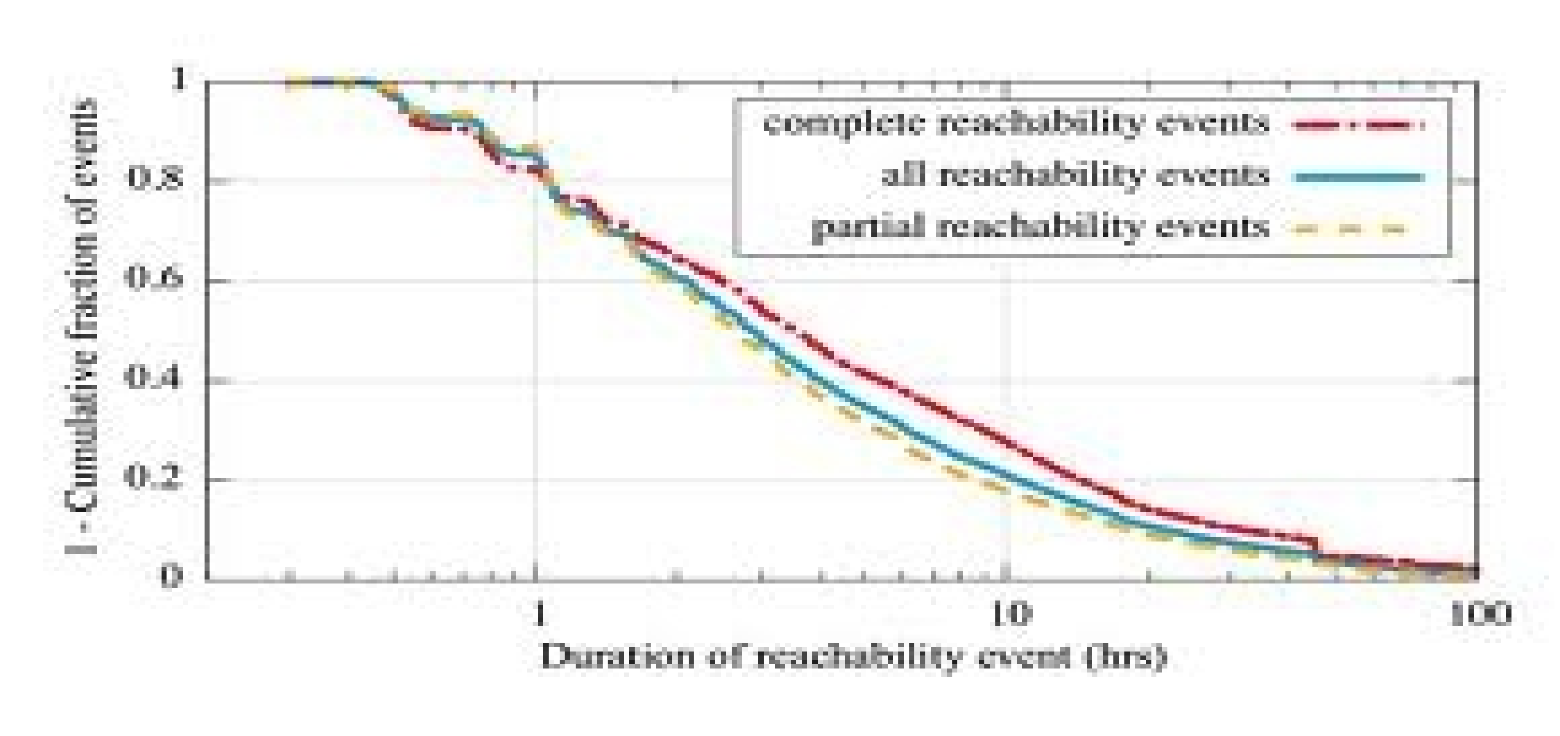
CDF plot

- **Cumulative Distribution Function**: describes the probability that a random variable *X* with a given probability distribution will be found at a value less than or equal to *x*
- >> cdfplot(X)



CCDF plot

- **Complementary Cumulative Distribution Function**: describes how often the random variable is *above* a particular level.



Correlation Coefficient

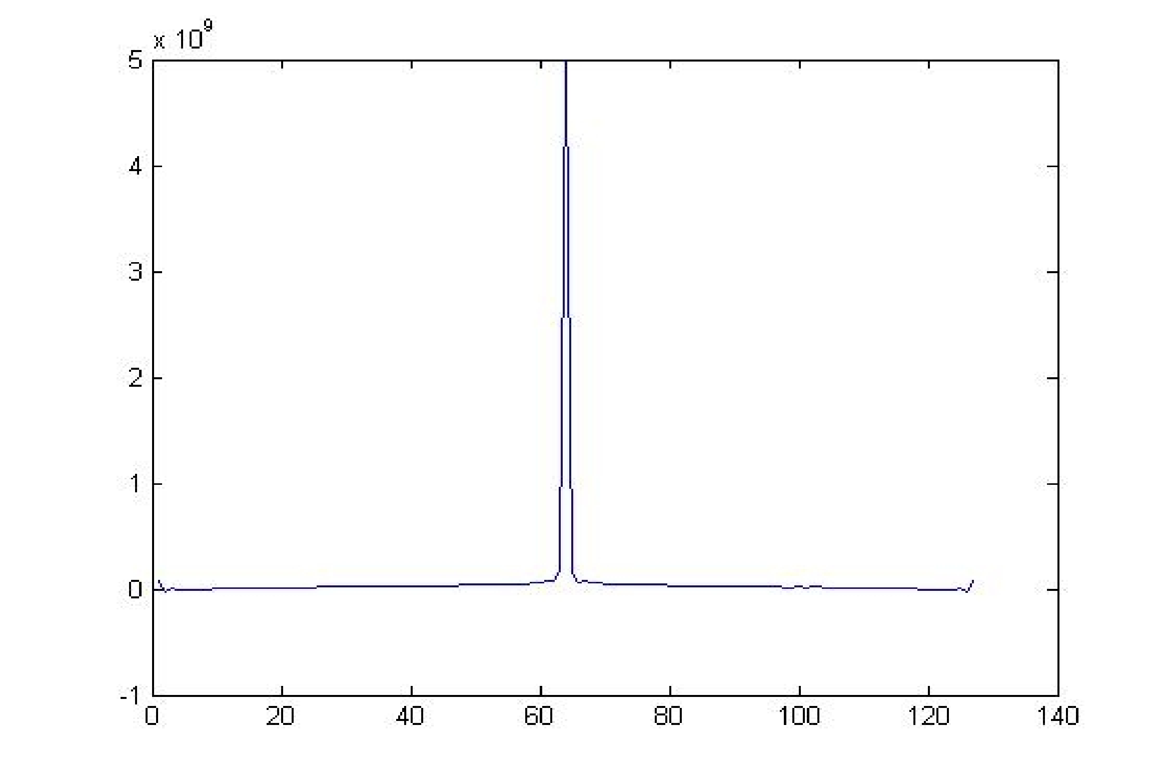
- Indicates relationship between random variables
- Values from -1 to +1
  - +1 indicates perfect positive (increasing) linear relationship
  - −1 indicates perfect decreasing (negative) linear relationship
  - 0 indicates no relationship/correlation

- corrcoef(x,y);

```
 >> ans
ans =
1.0000   -0.3835
-0.3835   1.0000
```

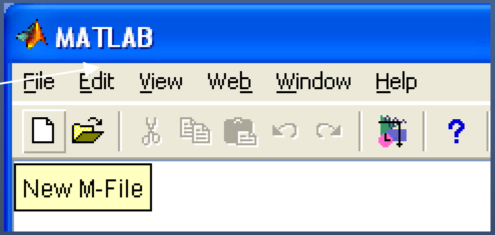
Correlation Coefficient(cont.)

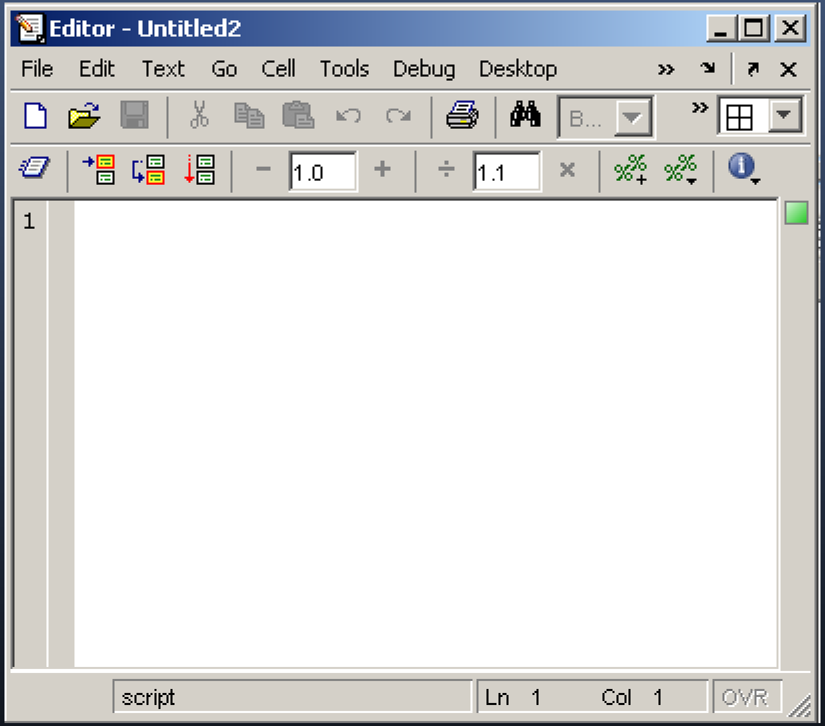
- >> c=xcorr(x,y);
- >> plot(c);



Use of M-File

Click to create a new M-File

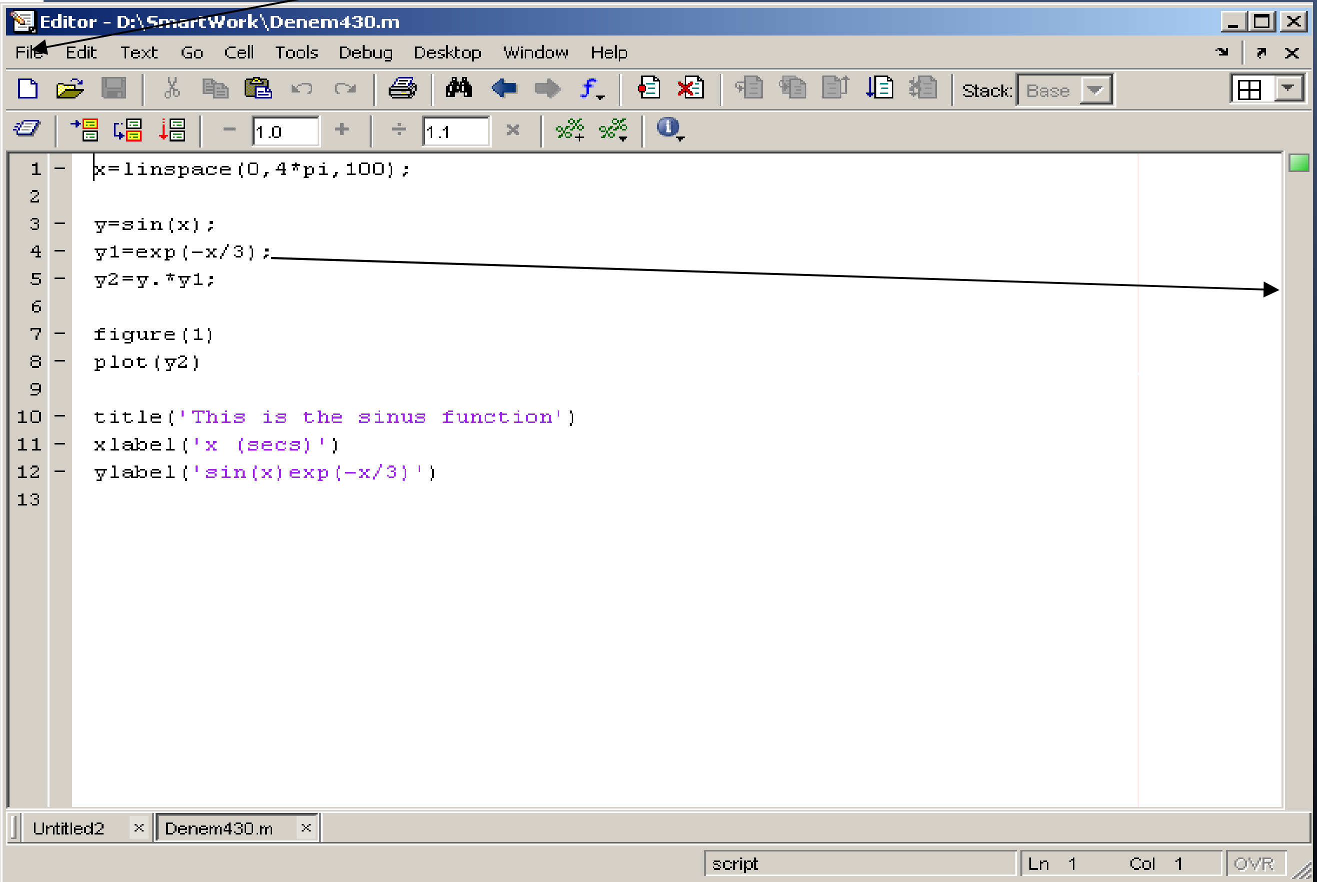




- Extension ".m"
- A text file containing script or function or program to run

Use of M-File

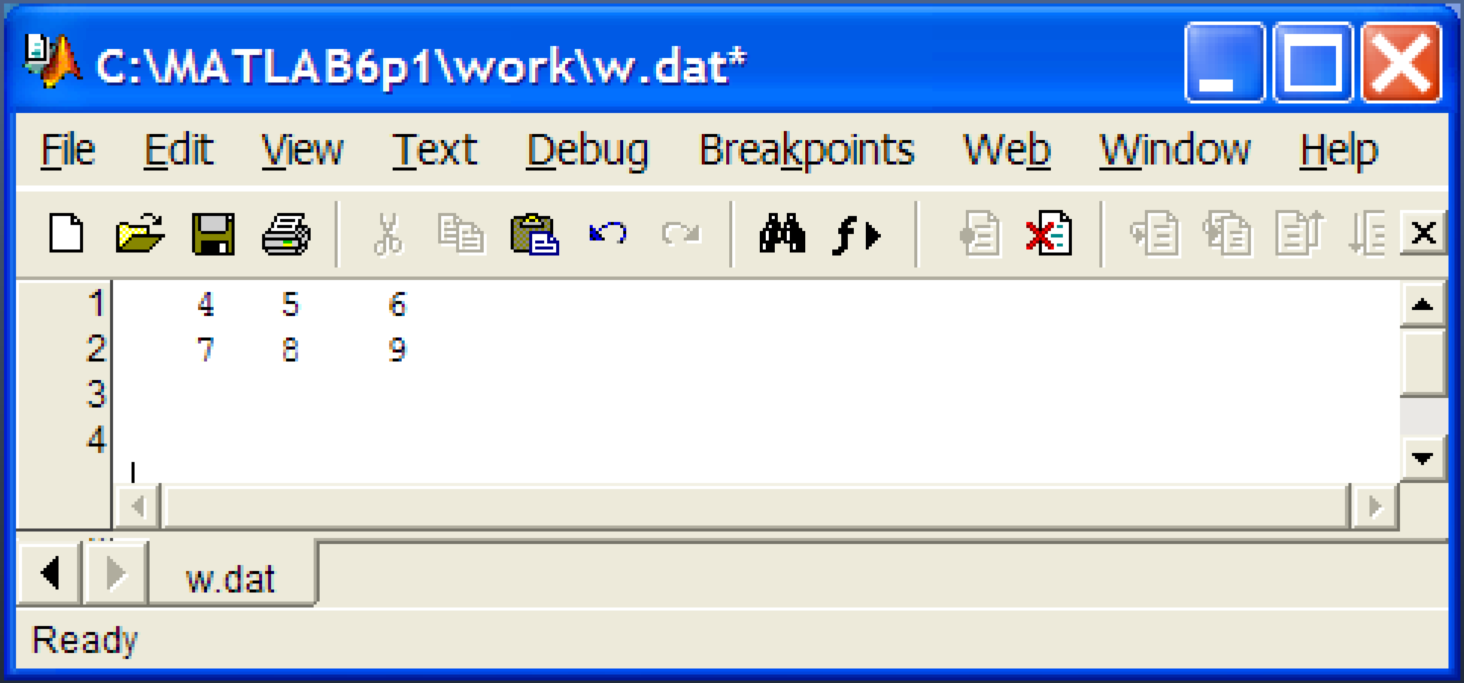
Save file as *Denem430*.m



```
x=linspace(0,4*pi,100);

y=sin(x);
y1=exp(-x/3);
y2=y.*y1;

figure(1)
plot(y2)

title('This is the sinus function')
xlabel('x (secs)')
ylabel('sin(x)exp(-x/3)')

```

If you include ";" at the end of each statement, result will not be shown immediately

Matlab Data files



```
1      4     5      6
2      7     8      9
3
4
```

```
>> load  w.dat              4 5 6
                            7 8 9
>> w

>> w(2,1)                   7
```

Useful Commands

- The two commands used most by

```
>>help functionname
```

```
>>lookfor keyword
```