Introduction to MATLAB

Presentation to EE1001 Class of the Department of Electrical Engineering at the University of Minnesota Duluth, November 1\textsuperscript{st}, 2016

Dr. Mohammed A. Hasan

The material of this presentation is taken from many sources including Textbooks and websites
Outline

Scientific Computing Software
What is MATLAB?
What are we interested in?
MATLAB History
Number Types
Build-in Functions
More Build-in functions
Where to Access MATLAB
Comparison with other Languages
Some other aspects of MATLAB
MATLAB as a Calculator
Numbers and Formats
Scientific Computing Software
MATLAB (OCTAVE, SCILAB, FREEMAT)
MATHEMATICA (WOLFAM ALPHA,...)
MAPLE

These offer symbolic and numeric computing environment

The following are some of the industries where matlab is used in research and real life applications:
• Computational science - biological data mining
• Embedded systems
• Aerospace industry
• Automobile industry
What is MATLAB?

• MATLAB is a Language for Technical Computing.
• MATLAB’s name is derived from MATrix LABoratory. It was originally designed for solving linear algebra type problems using matrices.
• It is essential in math and numeric computation, algorithm development, data acquisition, analysis, visualization, modeling, simulation, and prototyping
• Scientific and engineering graphics
• Application development, including graphical user interface
• Widely used in industry and academia
Open-Source Alternatives to MATLAB

Octave is one of the major free alternatives to MATLAB, others being FreeMat and Scilab. Scilab, however, puts less emphasis on syntactic compatibility with MATLAB than Octave does.

As in MATLAB, the syntax is matrix-based and provides various functions for matrix operations. It supports various data structures and allows object-oriented programming.

http://octave-online.net/
Using MATLAB?

- Matlab is not only a programming language, but a programming environment as well.

- You can perform operations from the command line, as a sophisticated calculator.

- Or you can create programs and functions that perform repetitive tasks, just as any other computer languages.

Try a simple operation now:
2 + 2 <enter>
To run a program, type its name:

demo <enter>

One of the most important features of the MATLAB interface is the help. It is very thorough and you can learn almost anything you need from it.

Let’s start doing something interesting with MATLAB (Help Manipulating Matrices)

\[
A = \begin{bmatrix}
16 & 3 & 2 & 13 \\
5 & 10 & 11 & 8 \\
9 & 6 & 7 & 12 \\
4 & 15 & 14 & 1
\end{bmatrix}
\]

This is what MATLAB displays after you hit <enter>
Let’s prove it is a magic square. Let’s get the sum of all columns by typing

\text{sum}(A)

The answer (ans) is:

\text{ans} =

\begin{array}{cccc}
34 & 34 & 34 & 34 \\
\end{array}
What is MATLAB? (continued)

• MATLAB is a high level language which has many specialized toolboxes for making things easier.
• How high?
What are we interested in?

- MATLAB is too broad for our purposes in this course.
- The features we are going to require is
Comparison with Other Languages

MATLAB is matrix-oriented, so what would take several statements in C or Fortran can usually be accomplished in just a few lines using MATLAB's built-in matrix and vector operations.

**FORTRAN:**

```fortran
real*8 A(10,10), B(10,10), C(10,10)
do i=1,10
  do j=1,10
    C(i,j) = A(i,j) + B(i,j)
  continue
continue
```

**MATLAB:**

```
C = A + B
```
MATLAB History

• Founded in 1984 by Jack Little and Cleve Moler
  – recognized the need among engineers and scientists for more powerful & productive computation environments beyond that provided by Fortran and C

• The Mathworks is Headquartered in Natick, Massachusetts

• Originally written to provide easy access to software developed by the LINPACK and EISPACK projects

• Flagship products are MATLAB and SIMULINK

• Just released ver. 13 (MATLAB 6.5, SIMULINK 5.0) with 10 new products and 33 updated products

• R13 compatible with UNIX, PC, and MAC OS X
Where to Access Matlab

MATLAB is accessible in most EE Labs and MWAH 102

Student version is affordable.
It costs $99.00

The Mathworks allows accessibility to all of their products with a fee of $50/year

MATLAB is available for MS Windows, Macintosh, Unix and other operating systems.
Where to Access MATLAB

In Windows systems MATLAB is started by double-clicking the mouse on the appropriate icon.
MATLAB Assignment & Operators

Assignment = a = b (assign b to a)
Addition + a + b
Subtraction - a - b
Multiplication * or .* a*b or a.*b
Division / or ./ a/b or a./b
Power ^ or .^ a^b or a.^b
MATLAB as a Calculator
(Examples)

1. basic arithmetic operator = + - * / ^ ()
   e.g.
   >> 2+3/4*5
   >> 3^2*4
   >> 3-4/4-2
   >> (1+i)*( -1+3*i)
   >> (1+i)/( -1+3*i)

   \( i = \sqrt{-1} \)

2. extended arithmetic - accidental error
   inf: Infinity (dividing by 0)
   NaN: Not a number (0/0)
   1/0 = 
   -1/0 =
   0/0 =
Symbolic Calculation

• Example: Suppose we would like to compute $z^2$ and $zz^*$, where $z$ is a complex number $z = x + iy$

```matlab
>> syms x y real
>> z = x + i*y
z = x + i*y
>> square = expand(z^2)
square = x^2 + 2*i*x*y - y^2
>> zstar = expand(z*conj(z))
zstar = x^2 + y^2
```
Numbers and Formats

• Format short
• Format long

Examples:  \( \pi = 3.1416 \)

```
format long

\( \pi = 3.1415926535897 \ldots \)
```

MATLAB limit accuracy (enough for most cases): 64 bits, it store number as

• large as \( 2 \times 10^{308} \), and as small as \( 2 \times 10^{(-308)} \)

• Store any number 15 significant figures:
  - e.g. \( 1.23456789023456 \) (14 figures, can handle)

• notation for very large or small number, e.g. \( -1.34e+03 \)
  , and \( 1.34e-05 \)
Variables

- Variable names can contain up to 63 characters
- Variable names must start with a letter followed by letters, digits, and underscores.
- Variable names are case sensitive
- Combination of letter and number, case sensitive
  - a, x1, z2453, A, com_c,
- Not allowed: com-c, 2p, %x, @sign
- Avoid using special names: eps (= $2^{-54}$), pi, etc
- Complex numbers: i, j = sqrt(-1), unless you change them
- For suppressing output (i.e. don't want to show output) use semi-colon ;)
- Hidden: x = -13; (semi-colon).
Variables (continued)

- Variables have not to be previously declared
- Variable names are case sensitive

```
>> x=5;
>> x1=2
ans
  2
```
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ans</td>
<td>default variable name for results</td>
</tr>
<tr>
<td>Pi</td>
<td>Value of 3.1459…</td>
</tr>
<tr>
<td>eps</td>
<td>Smallest incremental number</td>
</tr>
<tr>
<td>inf</td>
<td>Infinity</td>
</tr>
<tr>
<td>NaN</td>
<td>Not a number e.g. 0/0</td>
</tr>
<tr>
<td>realmin</td>
<td>The smallest usable positive real number</td>
</tr>
<tr>
<td>realmax</td>
<td>The largest usable positive real number</td>
</tr>
</tbody>
</table>
Number Types

No need for types. i.e.,

```plaintext
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.
Build-in Functions

1. Trig functions: sin, cos, tan, sec = 1/sin, cosec = 1/cos, cotan=1/tan

>> x=pi/4; y=cos(x)
0.7071

>> x1=2;

2. inverse trig function
   e.g. asin, acos, atan--> answer returned in radians, so

>> asin(1)
ans
1.5729…
3. Exponential  \((y=e^x)\):
Matlab syntax: \(>>y=\exp(x)\)
logarithm (log): log to base e,
\(\log10\): log to base 10

4. square root: \(\sqrt{\text{.}}\).
\(\text{e.g.}\)
\(>>x = 9; \sqrt{x}, \exp(x), \log(\sqrt{x}), \log10(x^2+6)\)

5. \(a=[1 2 5]; u=\text{roots}(a); \ %\text{find the roots of a polynomial of coefficient 1, 2, 5.}\)
More Built-in functions

- \texttt{mean(A)}: mean value of a vector
- \texttt{max(A), min (A)}: maximum and minimum
- \texttt{sum(A)}: summation
- \texttt{sort(A)}: sorted vector
- \texttt{median(A)}: median value
- \texttt{std(A)}: standard deviation
- \texttt{det(A)}: determinant of a square matrix
- \texttt{dot(a,b)}: dot product of two vectors
- \texttt{Cross(a,b)}: cross product of two vectors
- \texttt{inv(A)}: Inverse of a matrix A
- \texttt{abs(z)}: magnitude of a number
Common Matlab Commands

1. find
2. size
3. length
4. rand, randn
5. randperm
6. detrend
7. repmat
8. reshape
9. fliplr, flipud
10. rot90
11. sort, sortrows
12. pinv
13. unwrap
14. conv
15. abs, angle
16. imag, real, conj
17. diff
18. cumsum
19. gradient
20. phantom
21. radon, iradon
22. montage
23. pixval
24. eval, feval
More Common Matlab Commands

- `cd`, `chdir`: change directory
- `pwd`: show present working directory
- `dir`, `ls`: list directory contents
- `delete`: delete a file
- `type`: show contents of a file
- `edit`: edit a file in the MATLAB editor
- `path`: get/set path
- `clear`: clear a variable from the workspace
- `clc`: clear the command window screen
- `who`, `whos`: list workspace variable information
- `which`: show location of a file in the path
- `what`: list MATLAB-specific files in a dir.
- `why`: succinct answers to any question
The Colon Operator

• The colon (:) is used in defining a vector range

```matlab
>> x = 7:12
x =
    7     8     9    10    11    12

>> x = 7:2.5:12
x =
    7.0000    9.5000   12.0000

>> x = 7:-1:12
x =
    Empty matrix: 1-by-0

>> x = 12:-1:7
x =
    12    11    10     9     8     7
```
1. row vectors

\[ a = [1\ 2\ 3] \text{ or } a = [1, 2, 3] \]

\[ V = [1\ 3\ \sqrt{5}] \], what is length(V)

- space vitally important: e.g. \( v_2 = [3+4\ 5] \), \( v_3 = [3\ +4\ 5] \);

- add vector of the same length:

\( V + v_3 \), \( v_4 = 3*v_3 \), \( v_5 = 2*V - 3*v_4 \), \( v_6 = \)

\( V + v_2 \) wrong! since dimension must agree

build a row vector from existing ones: e.g. \( w = [1\ 2\ 3] \), \( z = [8, 9] \),

\( cd = [2*z - w] \), sort(cd) (ascending order)

look at value of particular entries: e.g. \( w(2) = ?...=2 \)

- set \( w(3) = 100 \), then \( w = ?? \)

\( w = [1\ 2\ 100] \)
2. column vector
   e.g. \( c = [1; 3; \sqrt{5}] \) or \( c2 = [3 \ return \ 4 \ return \ 5] \)
   \( c3 = 2*c - 5*c2 \)

3. column notation : a shortcut for producing row vectors
   e.g. 1:100
       3:7
       5:0.1:6
       1:-1 -- \rightarrow []
       0.32:0.1:0.6
       -0.4:-0.3:-2
Vectors and Matrices

A vector

\[ x_1 = [1 \ 2 \ 5 \ 1] \]

\[
\begin{bmatrix}
1 & 2 & 5 & 1
\end{bmatrix}
\]

A matrix

\[ x_2 = [1 \ 2 \ 3; 
5 \ 1 \ 4; 
3 \ 2 \ -1] \]

\[
\begin{bmatrix}
1 & 2 & 3 \\
5 & 1 & 4 \\
3 & 2 & -1
\end{bmatrix}
\]

transpose

\[ y = x_1' \]

\[
\begin{bmatrix}
1 \\
2 \\
5 \\
1
\end{bmatrix}
\]
Entering a Simple Matrix

```
>> a = [1 2; 3 4]
a =
    1     2
    3     4
>> b = [5:6; 7:1:8] % Using the colon operator
b =
    5     6
    7     8
>> a*b % Full matrix multiplication
ans =
    19    22
    43    50
>> a.*b % Element-by-element multiplication
ans =
    5    12
    21    32
>> x=1:3;y = x' % transpose
   y =
    1
    2
    3
```
Some Useful Matrices

- **zeros(M,N)  MxN**  
  matrix of zeros  
  
  \[
  x = \text{zeros}(1,3) \\
  x = \\
  \begin{bmatrix}
  0 & 0 & 0 \\
  \end{bmatrix}
  \]

- **ones(M,N)  MxN**  
  matrix of ones  
  
  \[
  x = \text{ones}(1,3) \\
  x = \\
  \begin{bmatrix}
  1 & 1 & 1 \\
  \end{bmatrix}
  \]

- **rand(M,N)  MxN**  
  matrix of uniformly distributed random numbers on (0,1)  
  
  \[
  x = \text{rand}(1,3) \\
  x = \\
  \begin{bmatrix}
  0.9501 & 0.2311 & 0.6068 \\
  \end{bmatrix}
  \]
Indices in MATLAB

- The matrix indices begin from 1 (not 0 (as in C))
- The matrix indices must be positive integers

A(-2), A(0)

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

A(4,2)
Error: ??? Index exceeds matrix dimensions.
Plotting in MATLAB

MATLAB supports many types of graph and surface plots:

- line plots (x vs. y),
- filled plots, bar charts, pie charts,
- parametric plots, polar plots, contour plots,
- density plots, log axis plots,
- surface plots, parametric plots in 3 dimensions and spherical plots.
Plotting Data (continued)

- The basic plotting function is `plot`.
- Axis labeling is available as well as a legend.
- Title and label strings can contain some LaTeX.
- Linestyles and colors are completely customizable.
Example 1 - Plot the function $\sin(x)$ between $0 \leq x \leq 4\pi$

Steps:

a- Create an $x$-array of 100 samples between 0 and $4\pi$.

$x=\text{linespace}(0,4*\pi,100);$%

b- Calculate $\sin(.)$ of the $x$-array

$y=\sin(x);$%

c- Plot the $y$-array

$\text{plot}(y)$
Example 2: Plotting Multiple Data Sets in One Graph

- Multiple x-y pair arguments create multiple graphs with a single call to plot.

```matlab
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-.25);
y3 = sin(x-.5);
plot(x,y,x,y2,x,y3)
```
Example 3: Displaying Multiple Plots in One Figure using

\texttt{subplot(m,n,p)}

This splits the figure window into an m-by-n matrix of small subplots and selects the pth subplot for the current plot.

Example:

\begin{verbatim}
t = 0:pi/10:2*pi;
[X,Y,Z] = cylinder(4*cos(t));
subplot(2,2,1); mesh(X);
subplot(2,2,2); mesh(Y);
subplot(2,2,3); mesh(Z);
subplot(2,2,4); mesh(X,Y,Z);
\end{verbatim}
Multiple Plots

```matlab
x = linspace(-5,5); % define x
y1 = sin(x); % define y1
figure % create new figure
subplot(2,2,1) % first subplot
plot(x,y1) title('First subplot')
y2 = sin(2*x); % define y2
subplot(2,2,2) % first subplot
plot(x,y2) title('Second subplot')
y3 = sin(4*x); % define y3
y4 = sin(6*x); % define y4
subplot(2,2,3) % third subplot
plot(x,y3) title('Third subplot')
subplot(2,2,4) % fourth subplot
plot(x,y4) title('Fourth subplot')
```
Various line types, plot symbols and colors may be obtained with plot(x,y,s) where s is a character string made from one element from any or all the following 3 columns:

<table>
<thead>
<tr>
<th>b</th>
<th>blue</th>
<th>.</th>
<th>point</th>
<th>-</th>
<th>solid</th>
</tr>
</thead>
<tbody>
<tr>
<td>g</td>
<td>green</td>
<td>o</td>
<td>circle</td>
<td>:</td>
<td>dotted</td>
</tr>
<tr>
<td>r</td>
<td>red</td>
<td>x</td>
<td>x-mark</td>
<td>-.</td>
<td>dashdot</td>
</tr>
<tr>
<td>c</td>
<td>cyan</td>
<td>+</td>
<td>plus</td>
<td>--</td>
<td>dashed</td>
</tr>
<tr>
<td>m</td>
<td>magenta</td>
<td>*</td>
<td>star</td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>yellow</td>
<td>s</td>
<td>square</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>black</td>
<td>d</td>
<td>diamond</td>
<td></td>
<td></td>
</tr>
<tr>
<td>v</td>
<td></td>
<td></td>
<td>triangle (down)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>^</td>
<td></td>
<td></td>
<td>triangle (up)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;</td>
<td></td>
<td></td>
<td>triangle (left)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&gt;</td>
<td></td>
<td></td>
<td>triangle (right)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p</td>
<td></td>
<td></td>
<td>pentagram</td>
<td></td>
<td></td>
</tr>
<tr>
<td>h</td>
<td></td>
<td></td>
<td>hexagram</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

plot(x,y,'rs')
Plotting Vector Field with quiver
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t);

3D Plot
Peaks is an example function, useful for demonstrating 3D data, contouring, etc. Figure above is its default output.

\[ P = \text{peaks} \] - return data matrix for replotting...
Important Plotting Functions

- xlabel, ylabel, zlabel
  Axis labeling
- title
  Add a title
- legend
  Add a legend
- axis
  Control axis range and shape
- grid
  Turn grid on/off
- hold
  Holds axes for additional plots
- Subplot
  Divide figure into subplots
- orient
  Figure printing orientation
- set, get
  Set/get plot object properties
More Plotting Functions

“help” or “doc” graph2d, graph3d, or specgraph

• loglog, semilogx, semilogy, polar, plotyy

• plot3, mesh, surf, fill3, bar, barh, comet, contour, feather, hist, pareto, pie, quiver, scatter, stem, stairs, errorbar, bar3, bar3h, meshc, meshz, pie3, slice, streamribbon, ribbon, trimesh, waterfall

• And many more!
http://www.engr.iupui.edu/~jschild/matlabtutorial/introductory_lessons/matlab_demo.htm
Scripts and Functions

- There are two kinds of M-files:

  - Scripts, which do not accept input arguments or return output arguments. They operate on data in the workspace. Any variables that they create remain in the workspace, to be used in subsequent computations.

  - Functions, which can accept input arguments and return output arguments. Internal variables are local to the function.

function y=my fun(x)

y=x.*sin(3*x.^2).*exp(-x.^2/4);

plot(x,y)
3D Plot Using `surf`

Example:
clear all
close all

\[
[x,y] = \text{meshgrid}([-2:.2:2]); \quad \% \text{set up 2-D plane}
\]

\[
Z = x.*\exp(-x.^2-y.^2); \quad \% \text{plot 3rd dimension on plane}
\]

figure

surf(x,y,Z,\text{gradient}(Z)) \quad \% \text{surface plot, with gradient(Z)}

\% determining color distribution

colorbar \quad \% \text{display color scale, can adjust}

\% location similarly to legend
Scripts and Functions

You can sum matrices of same size

\[
A = \begin{bmatrix} 1 & 2 \\ 4 & 5 \end{bmatrix}; \quad B = A'; \quad C = 0.5 \times (A + B)
\]

\[
C = \\
1 & 3 \\
3 & 5 
\]

or

\[
A = \begin{bmatrix} 1 & 2 \\ 4 & 5 \end{bmatrix}; \\
B = A'; \\
C = 0.5 \times (A + B)
\]

Note: The apostrophe (') indicates transpose operation, i.e. A' is the transpose of A.
Scripts and Functions

Functions are M-files that accept input arguments and return output arguments. They operate on variables within their own workspace. See an example below

```matlab
function y = average(x)
% AVERAGE Mean of vector elements.
% AVERAGE(X), where X is a vector, is the mean of vector elements.
[m,n] = size(x);
y = sum(x)/length(x);   % Actual computation
```

This file should be saved as avarage.m
Available Transforms

• $\texttt{fft,ifft,fft2,ifft2}$  Fast Fourier
• $\texttt{dct,idct,dct2,idct2}$  Discrete Cosine
• $\texttt{czt}$  Chirp-z
• $\texttt{radon, iradon}$  Radon
• $\texttt{hilbert}$  Hilbert
• $\texttt{dftmtx}$  Discrete Fourier matrix
• $\texttt{fftshift}$  Swap vector halves

• Signal Processing Toolbox (\texttt{help signal}) has many more functions for analysis and filtering
Reading and playing audio files

- MATLAB can read, play, write, and even record audio files
  - `wavread` or `auread` to load a file or create a signal yourself
  - `sound`, `soundsc`, or `wavplay` to play audio
  - `wavwrite` or `auwrite` to save the audio file
  - Use `wavrecord` to record sound using Windows audio input device

```
>> load handel;
>> % Handel's Hallelujah Chorus
>> sound(y,Fs);
>> load gong;
>> % a single bang on a gong
>> sound(y,Fs);
>> plot(y); axis tight;
>>
```
Programming in MATLAB

• Conditional Control
  - if, else, elseif
  - switch, case

• Loop Control
  - for, while, continue, break

• Error Control
  - try, catch

• Program Termination
  - return
Some Other Aspects of MATLAB

1. MATLAB is an interpreter -> not as fast as compiled code

2. Typically quite fast for an interpreted language

3. Often used early in development -> can then convert to C for speed

4. Can be linked to C/C++, JAVA, SQL, etc
Compiling m-files with mcc

- Using the MATLAB C/C++ Compiler, the MATLAB C/C++ Math Library, & the MATLAB C/C++ Graphics Library, MATLAB applications can be converted to standalone C or C++ apps.
- The MATLAB command `mcc -B sgl knot.m` converts the MATLAB M-file `knot.m` into C and compiles `knot.exe`.
- Running the knot executable in a DOS window reproduces the exact same colorfully lit surfaces produced by MATLAB.

```matlab
>> mcc -B sgl knot.m
>> !knot.exe
```
Input/Output Files

To load data from a file:

```matlab
x = load('myfile.dat'); % load content of myfile.dat in an array x
```

To save data in a file:

```matlab
save myfile var1 var2 ... -ASCII
```
or

```matlab
save('myfile', 'var1', 'var2',...,'-ASCII')
```

```matlab
type mytextfile.txt; % display the content of the file
```
Printing to Files

- A wide variety of file formats are supported for printing; the general form is:
  ```
  >> print -driver -options filename
  ```
  e.g.
  ```
  >> print -dps filename
  print postscript file
  ```
  ```
  >> print -dpsc filename
  print colour postscript file
  ```
  ```
  >> print -depsc filename
  print colour encapsulated postscript file
  ```
Practice Problems

• Self-test Exercise
Plot the three functions \( \sin(x), x, \) and \( \frac{2x}{\pi} \) over the interval \([0, \pi/2]\), including the title "Bounds on \( \sin(x) \)" , a grid, a legend and making sure the x-axis corresponds to the plot interval.

• Plot the real part and imaginary part of the following signal

\[
x(t) = e^{t+2} \left(2t + 1\right) \quad 0 < t < 10
\]
The big problems for most MATLAB users are:

1. Finding the function that does what you want to do

2. Remembering the syntax (inputs, outputs) for that function

• `help function_topic_name`
• `doc function_topic_name`
• `helpwin`
• `helpdesk` (with internet connection)
• `lookfor` keyword
Mathworks On-line

• The MathWorks Inc. web site
http://www.mathworks.com/

• MATLAB Central
http://www.mathworks.com/matlabcentral/

• File Exchange
http://www.mathworks.com/matlabcentral/fileexchange/

• Newsgroup
http://newsreader.mathworks.com
>> help median

MEDIAN Median value.

For vectors, MEDIAN(X) is the median value of the elements in X. For matrices, MEDIAN(X) is a row vector containing the median value of each column. For N-D arrays, MEDIAN(X) is the median value of the elements along the first non-singleton dimension of X.

MEDIAN(X,DIM) takes the median along the dimension DIM of X.

Example: If X = [0 1 2
3 4 5]
then median(X,1) is [1.5 2.5 3.5] and median(X,2) is [1 4]

See also MEAN, STD, MIN, MAX, COV.
Useful Commands

>>help functionname

>>lookfor keyword
MathWorks Training Courses

http://www.mathworks.com/support/training
Some MATLAB Toolboxes

- Aerospace Blockset
  - Aerospace Toolbox
  - Bioinformatics Toolbox
  - Communications System Toolbox
  - Computer Vision System Toolbox
  - Control System Toolbox
  - Curve Fitting Toolbox
  - Data Acquisition Toolbox
  - Database Toolbox
  - Datafeed Toolbox
  - DO Qualification Kit (for DO-178)
  - DSP System Toolbox
  - Econometrics Toolbox
  - Embedded Coder
  - Filter Design HDL Coder
  - Financial Instruments Toolbox
  - Financial Toolbox
  - Fixed-Point Designer
  - Fuzzy Logic Toolbox
  - Gauges Blockset
  - Global Optimization Toolbox
  - HDL Coder
  - HDL Verifier

- Image Acquisition Toolbox
- Image Processing Toolbox
- Instrument Control Toolbox
- LTE System Toolbox
- Mapping Toolbox
- MATLAB Builder EX
- MATLAB Builder JA
- MATLAB Builder NE
- MATLAB Coder
- MATLAB Compiler
- MATLAB Distributed Computing Server
- MATLAB Production Server
- MATLAB Report Generator
- Model Predictive Control Toolbox
- Model-Based Calibration Toolbox
- Neural Network Toolbox

- IEC Certification Kit (for ISO 26262 and IEC 61508)
- Optimization Toolbox
- Parallel Computing Toolbox
- Partial Differential Equation Toolbox
- Phased Array System Toolbox
- Polyspace Bug Finder
- Polyspace Code Prover
- Polyspace Products for Ada
- Real-Time Windows Target
- RF Toolbox
- Robust Control Toolbox
- Signal Processing Toolbox
- SimBiology
- SimDriveline
- SimElectronics
- SimEvents
- SimHydraulics
- SimMechanics
- SimMechanics Link
- SimPowerSystems
- SimRF
Summary

Summary: MATLAB is very powerful software and easy to learn.

Give it a try

Thank you
Questions