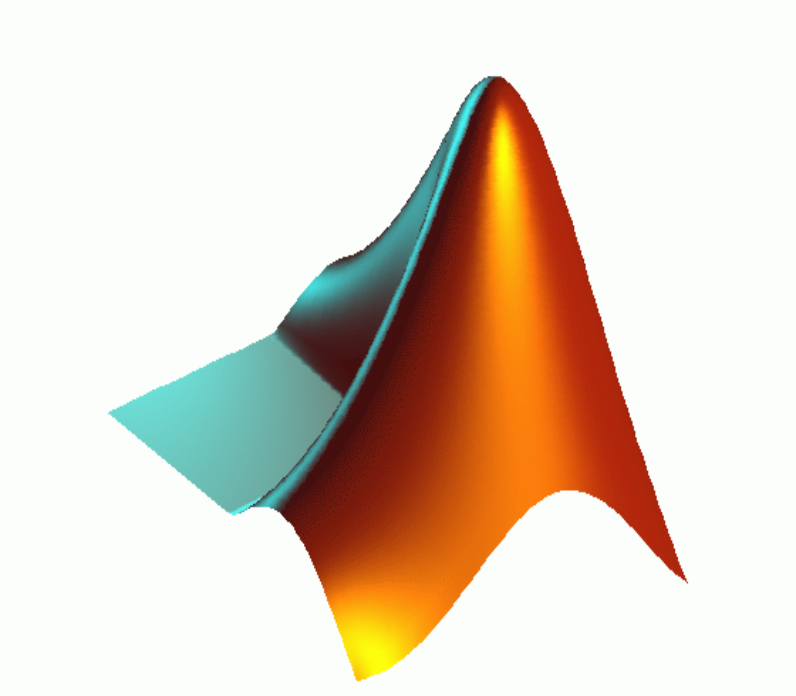
Introduction to Matlab



Marcus Kaiser

(based on lecture notes form Vince Adams and Syed Bilal Ul Haq)

MATLAB

- "MATrix LABoratory"
  (started as interactive interface to Fortran routines)
- Powerful, extensible, highly integrated computation, programming, visualization, and simulation package
- Widely used in engineering, mathematics, and science
- Why?

MATLAB's Appeal

- Interactive code development proceeds incrementally; excellent development and rapid prototyping environment
- Basic data element is the array
- This allows quick solutions to problems that can be formulated in vector or matrix form
- Large collection of *toolboxes*: collections of topic-related MATLAB functions that extend the core functionality significantly

MATLAB Toolboxes

Math and Analysis
- Optimization
- Statistics
- Symbolic/Extended Math
- Partial Differential Equations

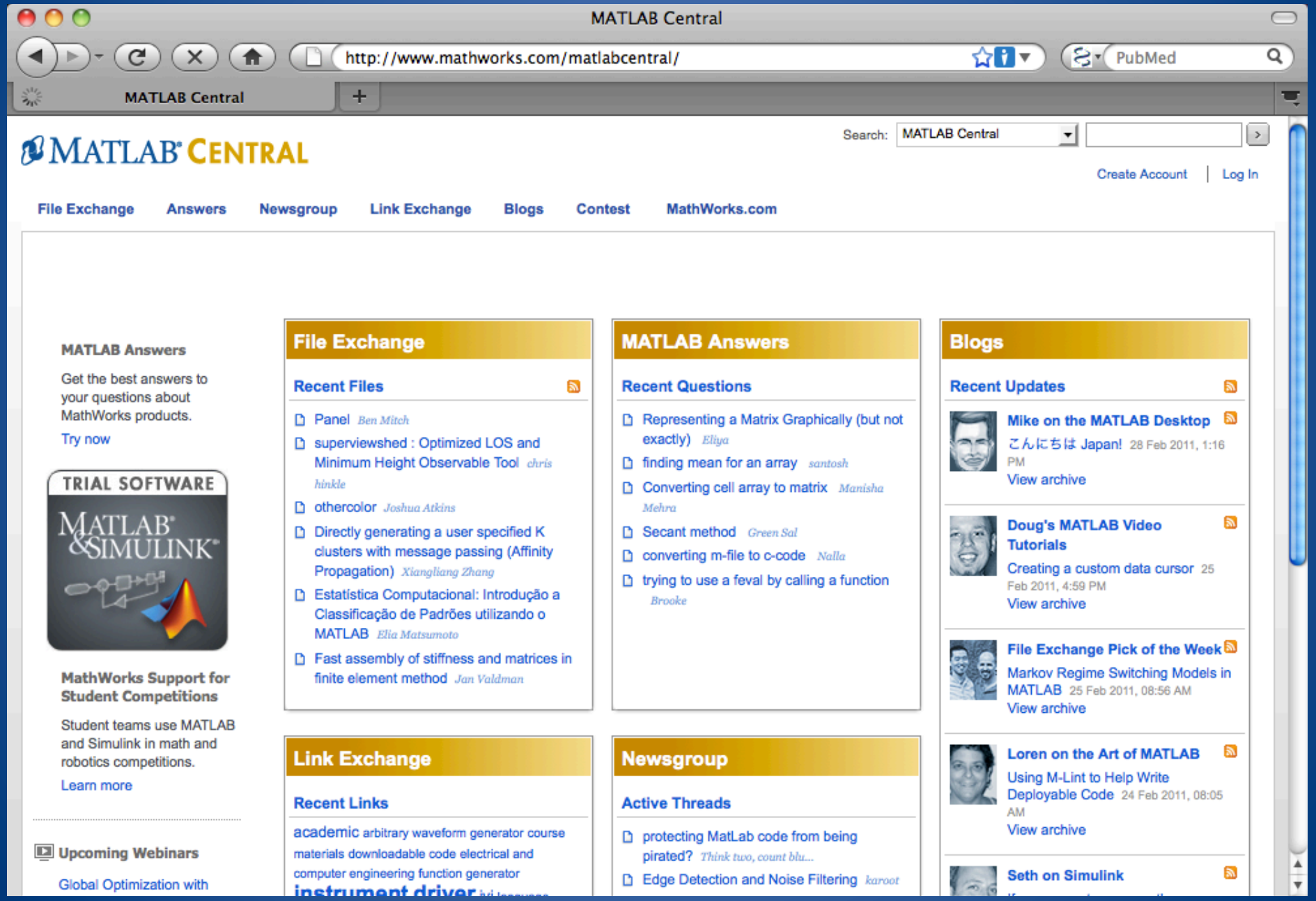
Data Acquisition and Import
- Data Acquisition
- Instrument Control

Signal & Image Processing
- Signal Processing
- Image Processing
  - Wavelet
- Filter Design

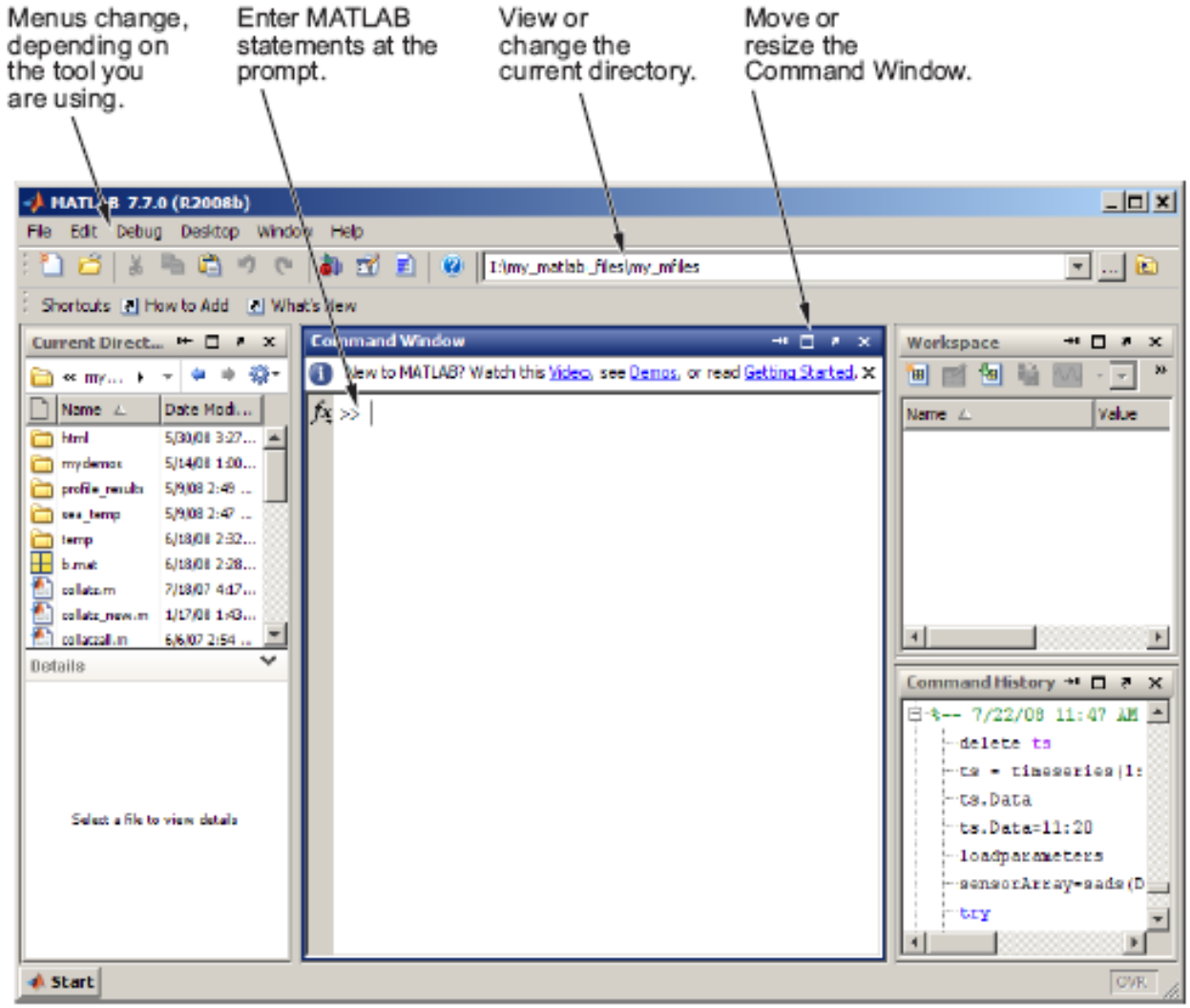
Computational Biology
- Bioinformatics
- SimBiology

Neuroscience
- Time Series Analysis (TISEAN)
- fMRI analysis (SPM)
- Data conversion (Biosignals)
- Electrophysiology (FIND toolbox)
- Brain Connectivity Toolbox
- EEG analysis (EEGLAB)
- Neural Network

Toolboxes, Software, & Links



Main Matlab Window



Working with Matrices and Arrays

Since Matlab makes extensive use of matrices,

the best way for you to get started with MATLAB

is to learn how to handle matrices.

What data can be represented as a matrix?

- Separate the elements of a row with blanks or commas.
- Use a semicolon ; to indicate the end of each row.
- Surround the entire list of elements with square brackets, [ ].

A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]

- MATLAB displays the matrix you just entered:

  A =

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

- Once you have entered the matrix, it is automatically remembered in the MATLAB workspace. You can simply refer to it as A.

- Keep in mind, variable names are case-sensitive

- When you do not specify an output variable, MATLAB uses the variable *ans*, short for *answer,* to store the results of a calculation.
- **Subscripts**

  The element in row i and column j of A

  is given by A(i,j).

  So to compute the sum of the elements in the fourth column of A, we have:

  A(1,4) + A(2,4) + A(3,4) + A(4,4)

  Which produces:

  ans = 34

- **The Colon Operator**
- For example: 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

- 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.

- **Arrays**

  Arithmetic operations on arrays are done element by element. This means that addition and subtraction are the **same** for arrays and matrices, but that *multiplicative* operations are **different**. MATLAB uses a dot, or decimal point, as part of the notation for multiplicative array operations.

  Example: A.*A

  the result is an array containing the squares of the integers

  ans =

256	**9**	**4**	**169**
25	**100**	**121**	**64**
81	**36**	**49**	**144**
16	**225**	**196**	**1**

- **Multivariate Data**

  MATLAB uses column-oriented analysis for multivariate statistical data. Each column in a data set represents a variable and each row an observation. The (i,j)th element is the ith observation of the jth variable.

  As an example, consider a data set with three variables:

  - **Heart rate**
  - **Weight**
  - **Hours exercise per week**

  For five observations, the resulting array might look like

- D = [72 134 3.2
  81 201 3.5
  69 156 7.1
  82 148 2.4
  75 170 1.2]

- Now you can apply MATLAB analysis functions to this data set. For example, to obtain the mean and standard deviation of each *column*, use

```
              mu = mean(D),      sigma = std(D)
mu =       75.8     161.8     3.48
sigma =    5.6303     25.499     2.2107
```

- **Entering Long Statements**

  If a statement does not fit on one line, use an ellipsis (three periods), ... , followed by **Return or Enter** to indicate that the statement continues on the next line. For example,

```
s = 1 -1/2 + 1/3 -1/4 + 1/5 - 1/6 + 1/7 ...
    - 1/8 + 1/9 - 1/10 + 1/11 - 1/12;
```

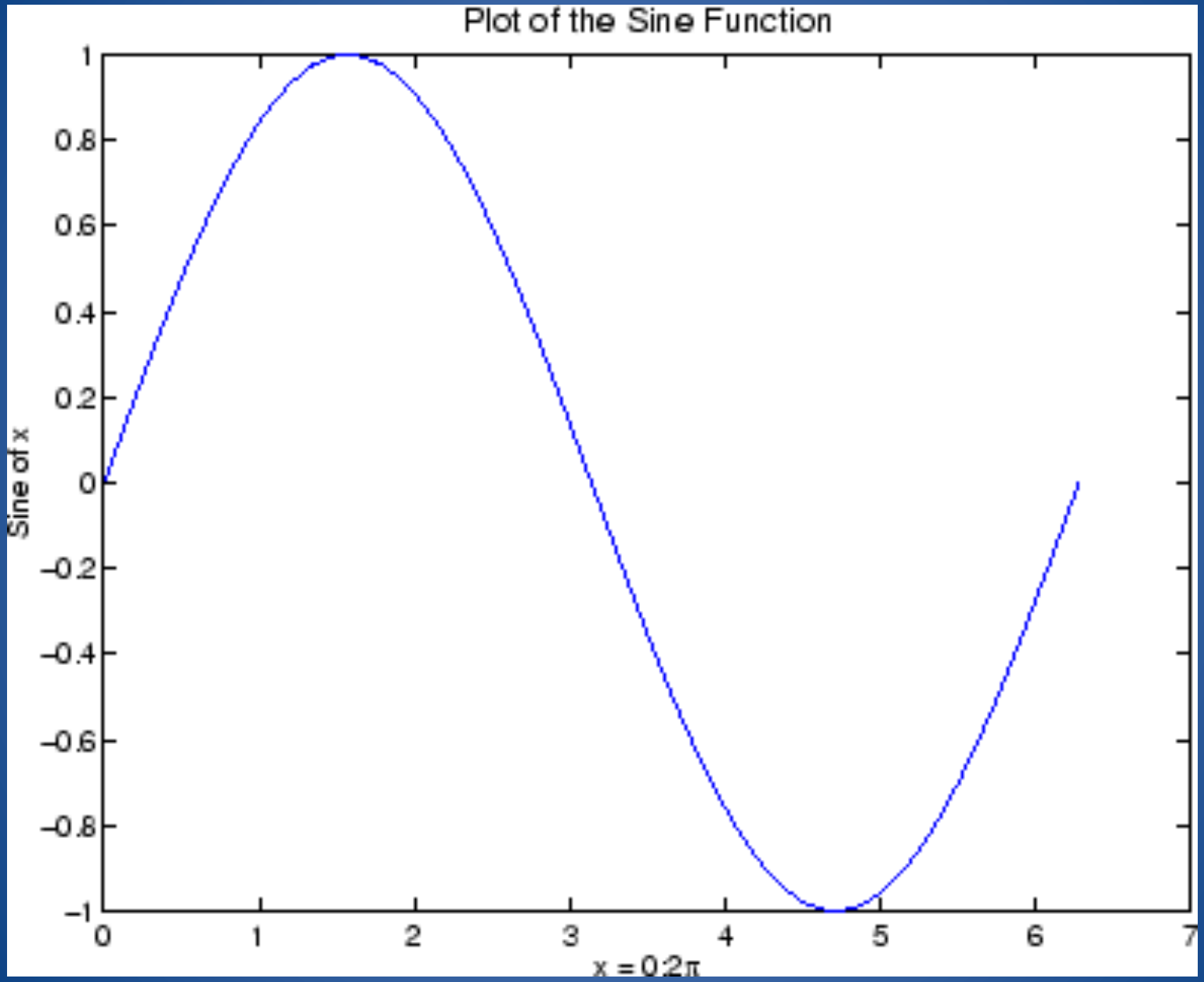
Graphics

- MATLAB provides a variety of techniques to display data graphically.
- Interactive tools enable you to manipulate graphs to achieve results that reveal the most information about your data.
- You can also edit and print graphs for presentations, or export graphs to standard graphics formats for presentation in Web browsers or other media.

Basic Plotting Functions

- The plot function has different forms, depending on the input arguments.
- If y is a vector, plot(y) produces a piecewise graph of the elements of (y) versus the index of the elements of (y).
- If you specify two vectors as arguments, plot(x,y) produces a graph of y versus x.
- You can also label the axes and add a title, using the ‘xlabel’, ‘ylabel’, and ‘title’ functions.

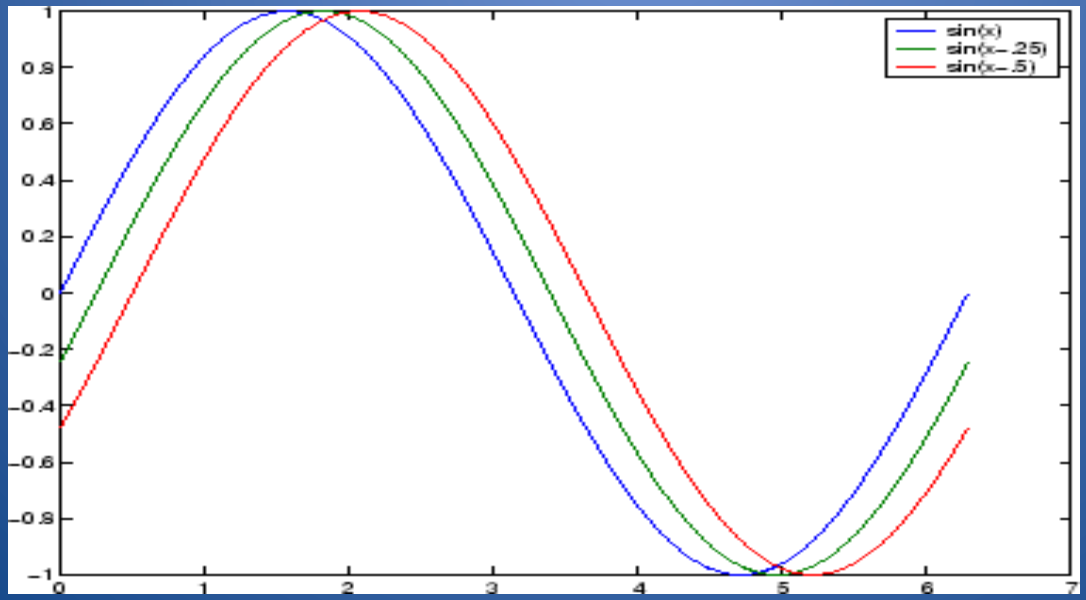
  Example:
```
xlabel('x = 0:2\pi')
ylabel('Sine of x')
title('Plot of the Sine Function','FontSize',12)
```


Plot of the Sine Function
Sine of x
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
1
2
3
4
5
6
7
x = 0:2π

- **Plotting Multiple Data Sets in One Graph**
  - Multiple x-y pair arguments create multiple graphs with a single call to plot.

For example:

```
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-.25);
y3 = sin(x-.5);
plot(x,y,x,y2,x,y3)
```



- **Specifying Line Styles and Colors**

  It is possible to specify color, line styles, and markers (such as plus signs or circles) when you plot your data using the plot command:

  plot(x,y,'*color_style_marker')*

  For example: plot(x,y,'r:+')

  plots a red-dotted line and places plus sign markers at each data point.

Adding Plots to an Existing Graph

When you type: hold on

MATLAB does not replace the existing graph when you issue another plotting command; it adds the new data to the current graph, rescaling the axes if necessary.

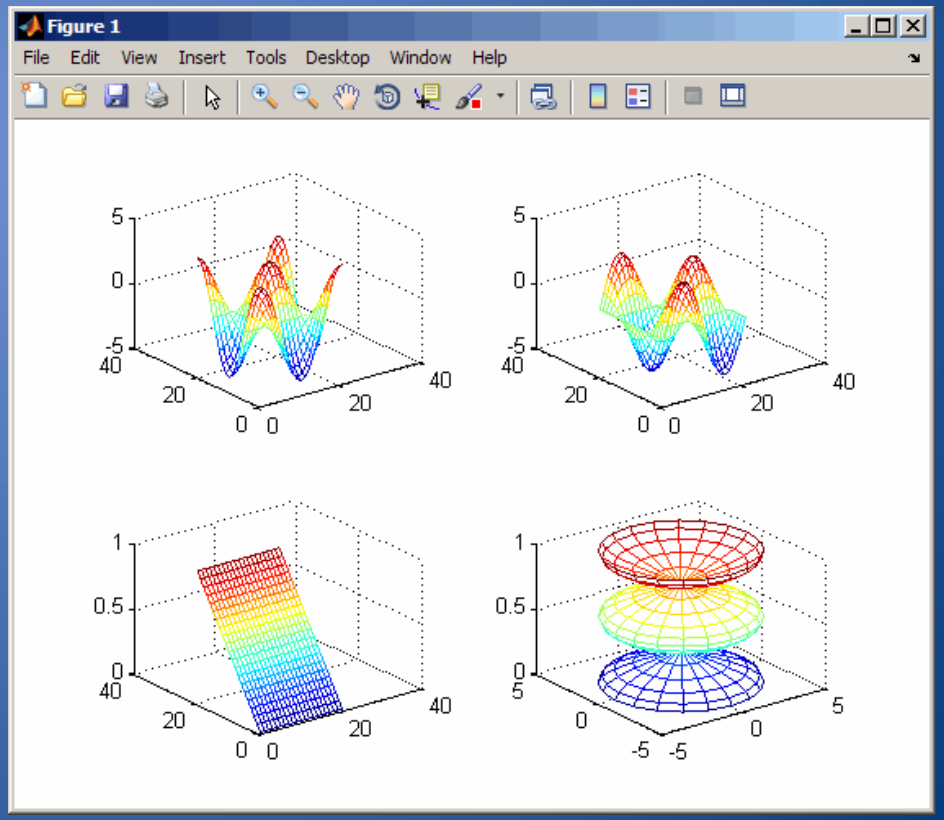
- **Displaying Multiple Plots in One Figure**

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:

```
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)
```



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.

MATLAB Resources

For this module:

Matlab Primer (chapter 2 and part of chapter 3):

http://www.mathworks.co.uk/help/pdf_doc/matlab/getstart.pdf

Other resources:

Getting Started with MATLAB 7: A Quick Introduction for Scientists and Engineers, R. Pratap, Oxford University Press, 2005.

Newsgroup: comp.soft-sys.matlab