What is Matlab?

- A software environment for interactive numerical computations

- Examples:
  - Matrix computations and linear algebra
  - Solving nonlinear equations
  - Numerical solution of differential equations
  - Mathematical optimization
  - Statistics and data analysis
  - Signal processing
  - Modelling of dynamical systems
  - Solving partial differential equations
  - Simulation of engineering systems

Matlab Background

- **Matlab** = **Mat**rix **Lab**oratory
- Originally a user interface for numerical linear algebra routines (Lapak/Linpak)
- Commercialized 1984 by The Mathworks
- Since then heavily extended (defacto-standard)

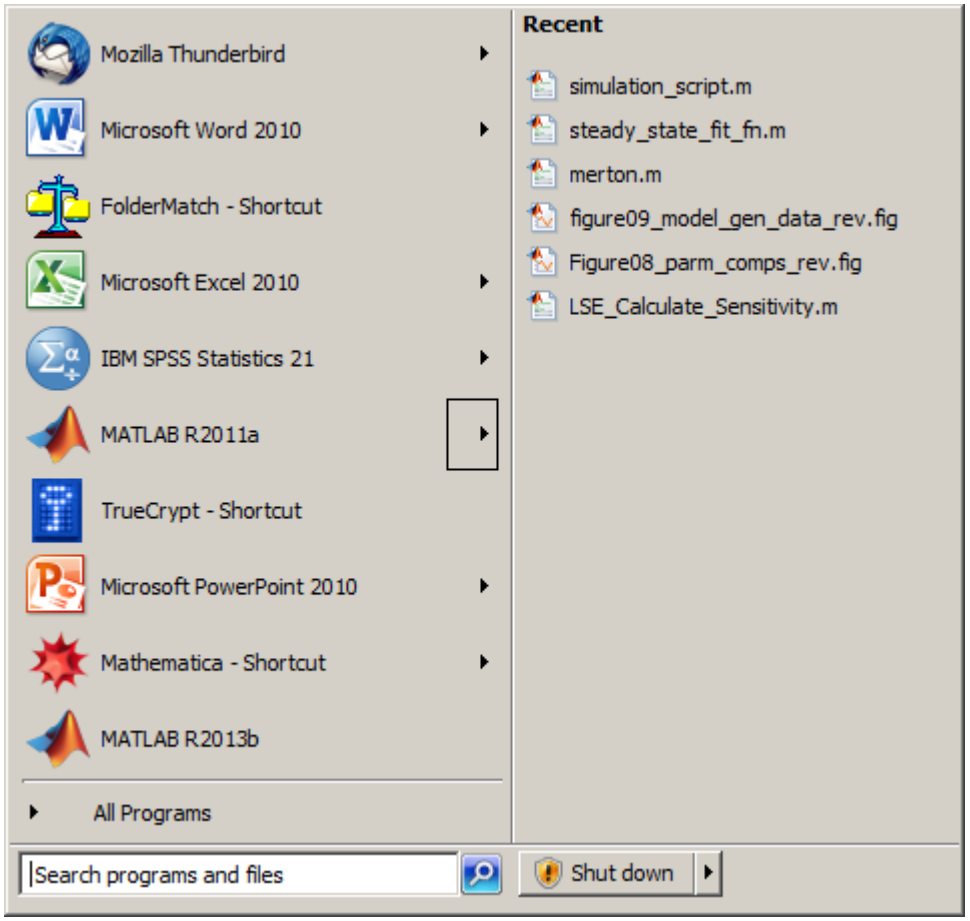
- Alternatives
  - Octave (free)
  - Julia (free)
  - R (sort of) and (free)

Complements

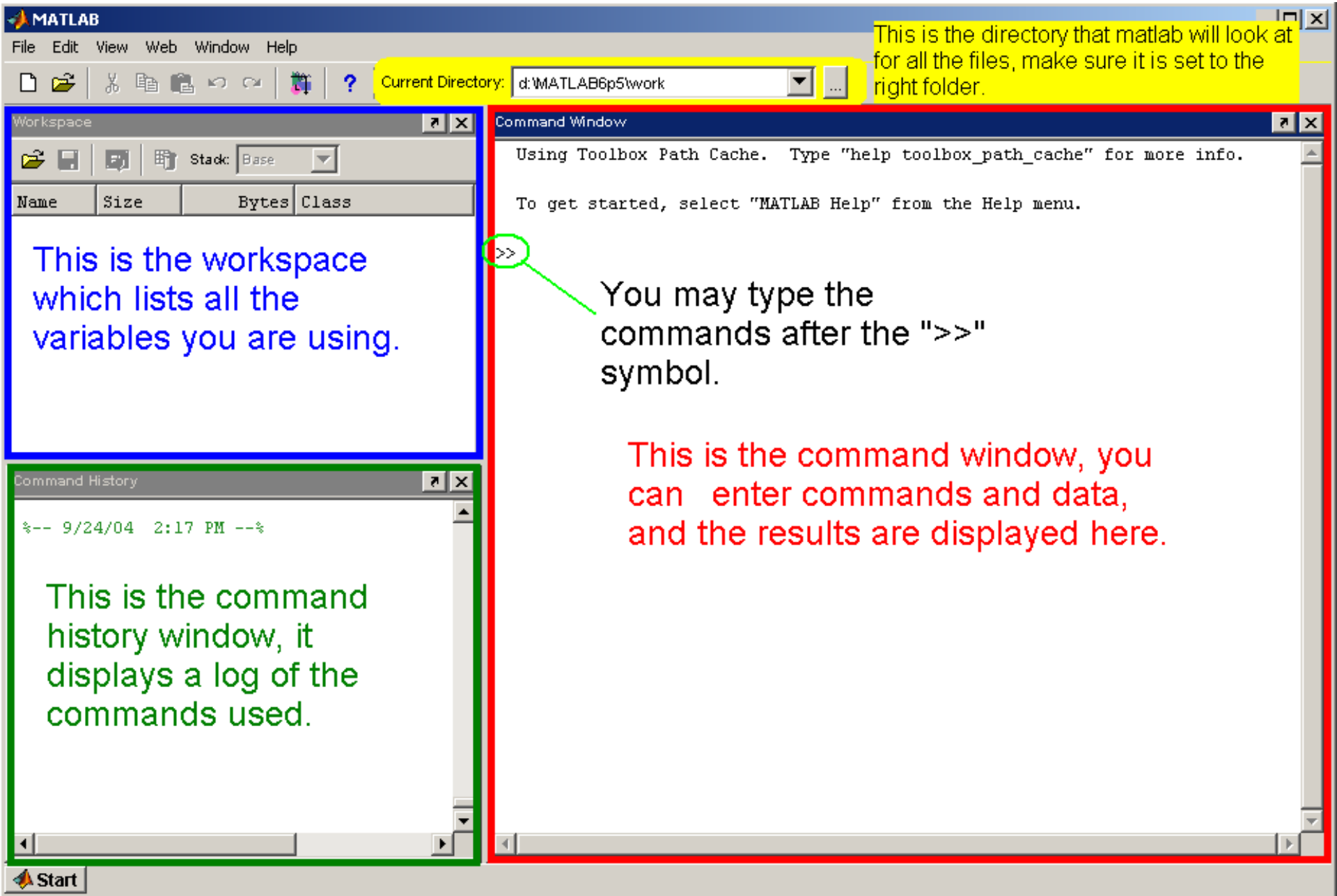
- Maple (symbolic)
- Mathematica (symbolic)

Start it up!

or



Display Windows



Help!

- help
- help command

Eg., help plot

- Help on toolbar
- demo
- Read a matlab primer:
  - **http://math.ucsd.edu/~driver/21d-s99/matlab-primer.html**
  - **https://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf**
- Try the tutorial in openwetware

Matlab Workspace

- Variables are stored here in the form of arrays (or matrices) which can be of different types: integers (`int`), real numbers (`double`), or alphabetical (`string`).
  - All numerical operations need to be performed on `double` precision, takes 8 bytes per value.
  - `Who, whos` – current variables in workspace
  - `Save, load` – save or load variables to *.mat file
  - `Clear all` – clear all variables

First steps

- Assignments (not equations!!!!)

```
>>x=4;
>>y=2;
>>z=2*x+3*y;
>>z
z =
    14
```

- You must make sure there's data in a variable before you use it in a computation!

Interactive Calculations

- Matlab is interactive, no need to declare variables
- `>> 2+3*4/2`
- `>> a=5e-3; b=1; a+b`
- Most elementary functions and constants are already defined
- `>> cos(pi)`
- `>> abs(1+i)`
- `>> sin(pi)`
- `>> exp(1)`
- `>> log(exp(1))`
- There are some ”built in” numbers like: pi
- `sin(pi)` gives answer `1.2246e-016` `!?`
- `Try this (speakers on, not max volume tho)`
- `>> load handel`
- `>> sound(y,Fs)`

Vectors and Matrices

- Vectors (arrays) are defined as
- `>> v = [1, 2, 4, 5]`
- `>> w = [1; 2; 4; 5]`

$$v = \begin{bmatrix} 1 & 2 & 4 & 5 \end{bmatrix}$$

$$w = \begin{bmatrix} 1 \\ 2 \\ 4 \\ 5 \end{bmatrix}$$

- Matrices (2D arrays) defined similarly
- `>> A = [1,2,3;4,-5,6;5,-6,7]`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & -5 & 6 \\ 5 & -6 & 7 \end{bmatrix}$$

Matrix Operators

- All common operators are overloaded
- `>> v + 2`

- Common operators are available
- `>> B = A'`
- `>> A*B`
- `>> A+B`

- Note:
- Matlab is case-sensitive

  `A` and `a` are two different variables

Indexing Matrices

- Indexing using parentheses
  - `>> A(2,3)`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & -5 & 6 \\ 5 & 6 & 7 \end{bmatrix}$$

- Index submatrices using vectors of row and column indices
  - `>> A([2 3],[1 2])`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & -5 & 6 \\ 5 & 6 & 7 \end{bmatrix}$$

- Ordering of indices is important!
  - `>> B=A([3 2],[2 1])`
  - `>> B=[A(3,2),A(3,1);A(2,2);A(2,1)]`

$$B = \begin{bmatrix} 6 & 5 \\ -5 & 4 \end{bmatrix}$$

Indexing Matrices

- Index complete row or column using the colon operator
  - `>> A(1,:)`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & -5 & 6 \\ 5 & 6 & 7 \end{bmatrix}$$

- Can also add limit index range
  - `>> A(1:2,:)`
  - `>> A([1 2],:)`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & -5 & 6 \\ 5 & 6 & 7 \end{bmatrix}$$

- General notation for colon operator
  - `>> v=1:5`
  - `>> w=1:2:5`

$$v = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

$$w = \begin{bmatrix} 1 & 3 & 5 \end{bmatrix}$$

Matrix Functions

- Many elementary matrices predefined
- `>> help elmat;`
- `>> I=eye(3)`

- Elementary functions are often overloaded
- `>> help elmat`
- `>> sin(A)`

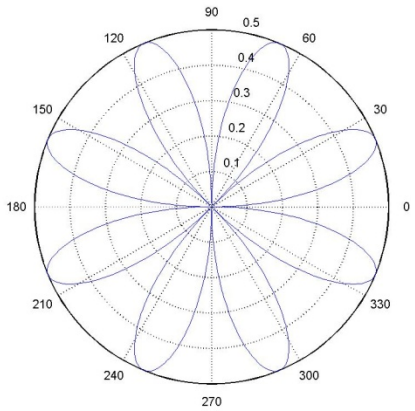
- Specialized matrix functions and operators
- `>> As=sqrtm(A)`
- `>> As^2`
- `>> A.*A`

- Note: in general, ".<operator>" is elementwise operation

Graphics

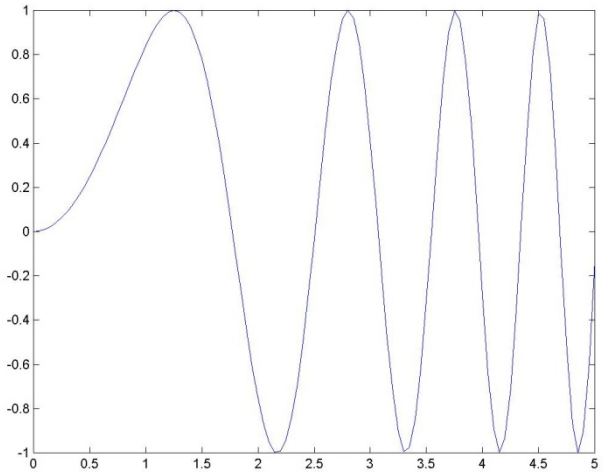
- Visualization of vector data is available
- `>> x=-pi:0.1:pi; y=sin(x);`
- `>> plot(x,y)`
- `>> plot(x,y,'s-')`
- `>> xlabel('x'); ylabel('y=sin(x)');`
- `>> plot(x,y'LineWidth', 4);`

What kind of graphics is possible in **Matlab**?



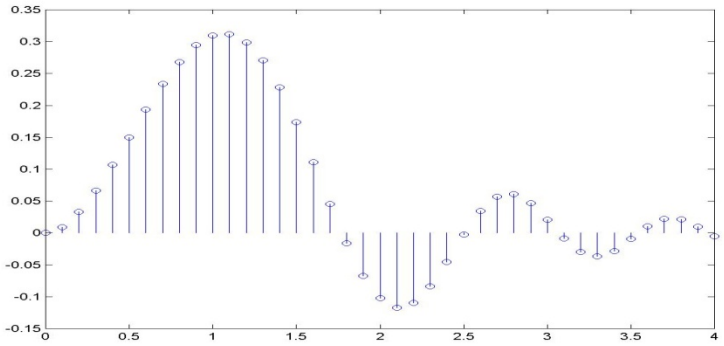
Polar plot:

```
t=0:.01:2*pi;
polar(t,abs(sin(2*t).*cos(2*t)));
```



Line plot:

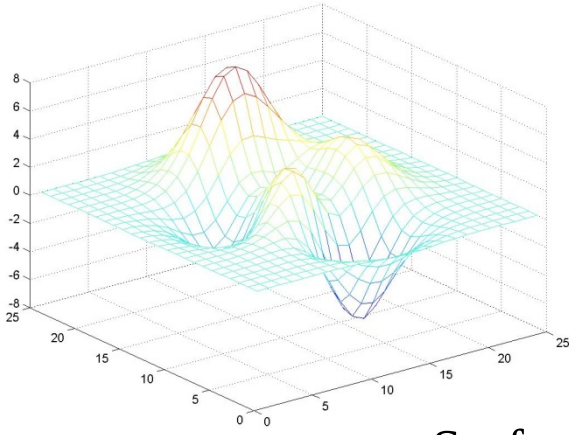
```
x=0:0.05:5;
y=sin(x.^2);
plot(x,y);
```



Stem plot:

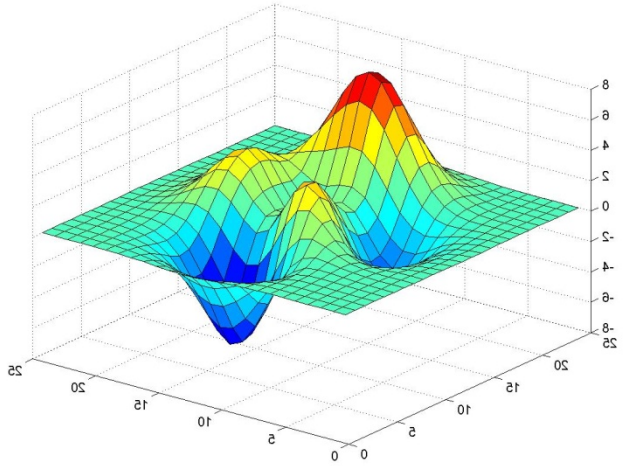
```
 x = 0:0.1:4;
 y = sin(x.^2).*exp(-x);
stem(x,y)
```

What kind of graphics is possible in **Matlab**?



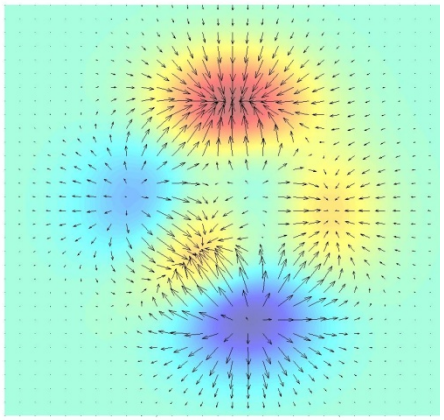
Mesh plot:

```
z=peaks(25);
mesh(z);
```

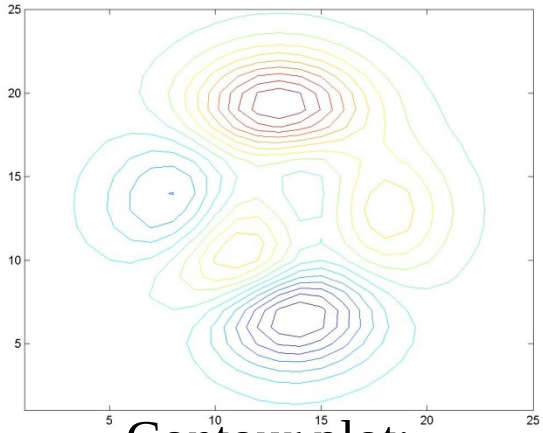


Surface plot:

```
z=peaks(25);surf(z);colormap(jet);
```



Quiver plot?



Contour plot:

```
z=peaks(25);contour(z,16);
```

Plot Continue…

- Adding plots to an existing graph:

  ```
  hold on
  ```

- Turn the above off

  ```
  hold off
  ```

- Multiple plots in one figure:

  ```
  subplot
  ```

  Create new plot windows

  ```
  figure
  ```

More Plotting

- Old plot got stomped
  - To open a new graph, type ‘figure’
- Multiple data sets:
  - Type ‘hold on’ to add new plot to current graph
  - Type ‘hold off’ to resume stomping
- Make your graph beautiful:
  - title(‘apples over oranges’)
  - xtitle(‘apples’)
  - ytitle(‘oranges’)

Plotting (con't…)

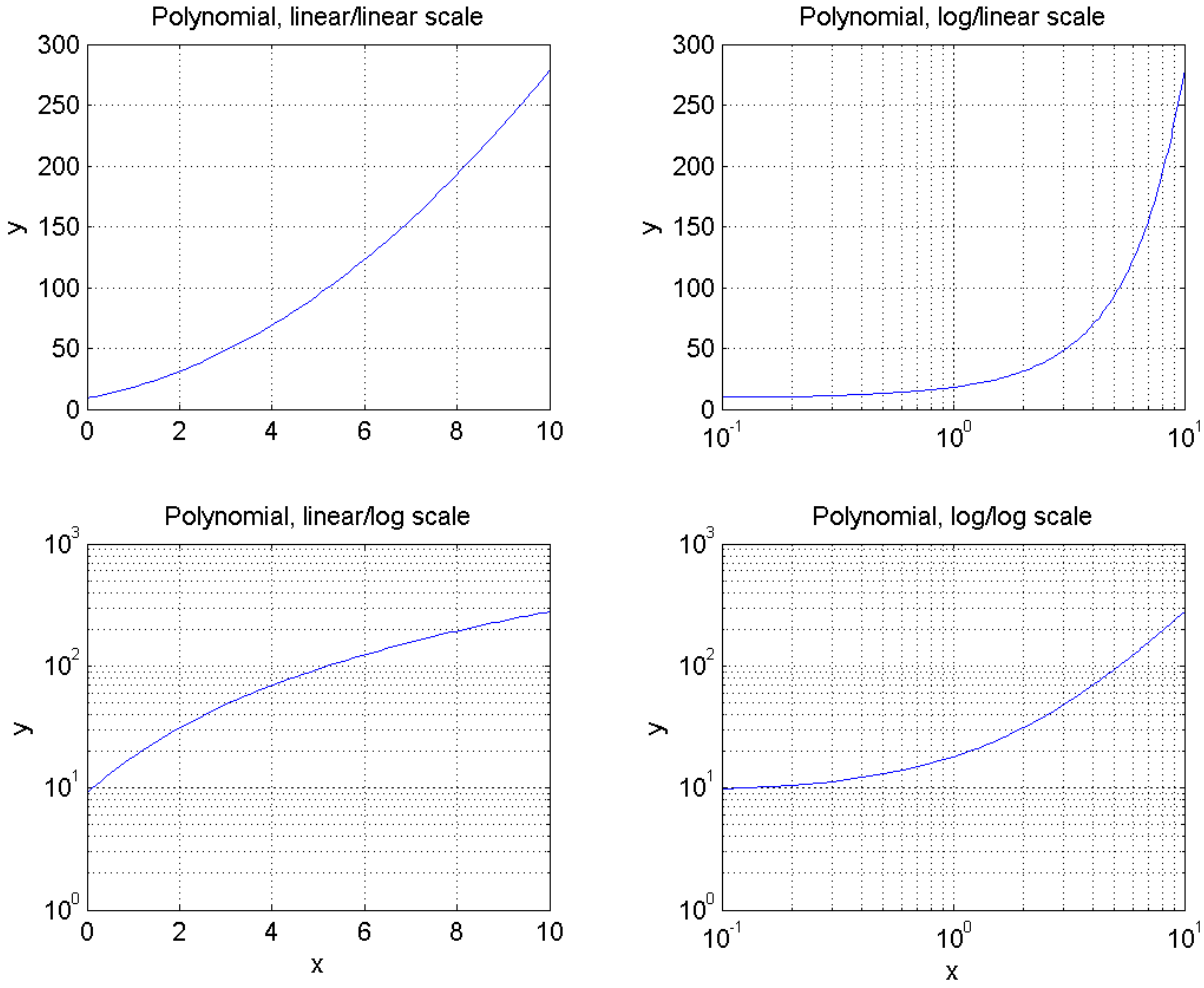
- **Example: (polynomial function)**
  plot the polynomial using linear/linear scale, log/linear scale, linear/log scale, & log/log scale:

$$y = 2x^2 + 7x + 9$$

```
% Generate the polynomial:
x = 0:10:100;
y = 2*x.^2 + 7*x + 9;

% plotting the polynomial:
figure (1);
subplot (2,2,1), plot (x,y);
title ('Polynomial, linear/linear scale');
ylabel ('y'), grid;
subplot (2,2,2), semilogx (x,y);
title ('Polynomial, log/linear scale');
ylabel ('y'), grid;
subplot (2,2,3), semilogy (x,y);
title ('Polynomial, linear/log scale');
xlabel('x'), ylabel ('y'), grid;
subplot (2,2,4), loglog (x,y);
title ('Polynomial, log/log scale');
xlabel('x'), ylabel ('y'), grid;
```

Plotting (con't…)



Plotting (con't…)

- Adding new curves to the existing graph:
- Use the **hold** command to add lines/points to an existing plot.
  - hold on – retain existing axes, add new curves to current axes. Axes are rescaled when necessary.
  - hold off – release the current figure window for new plots
- Grids and Labels:

Command	**Description**
grid on	Adds dashed grids lines at the tick marks
grid off	removes grid lines (default)
grid	toggles grid status (off to on, or on to off)
title ('text')	labels top of plot with text in quotes
xlabel ('text')	labels horizontal (x) axis with text is quotes
ylabel ('text')	labels vertical (y) axis with text is quotes
text (x,y,'text')	Adds text in quotes to location (x,y) on the current axes, where (x,y) is in units from the current plot.

Additional commands for plotting

color of the point or curve

Symbol	Color
y	yellow
m	magenta
c	cyan
r	red
g	green
b	blue
w	white
k	black

Marker of the data points

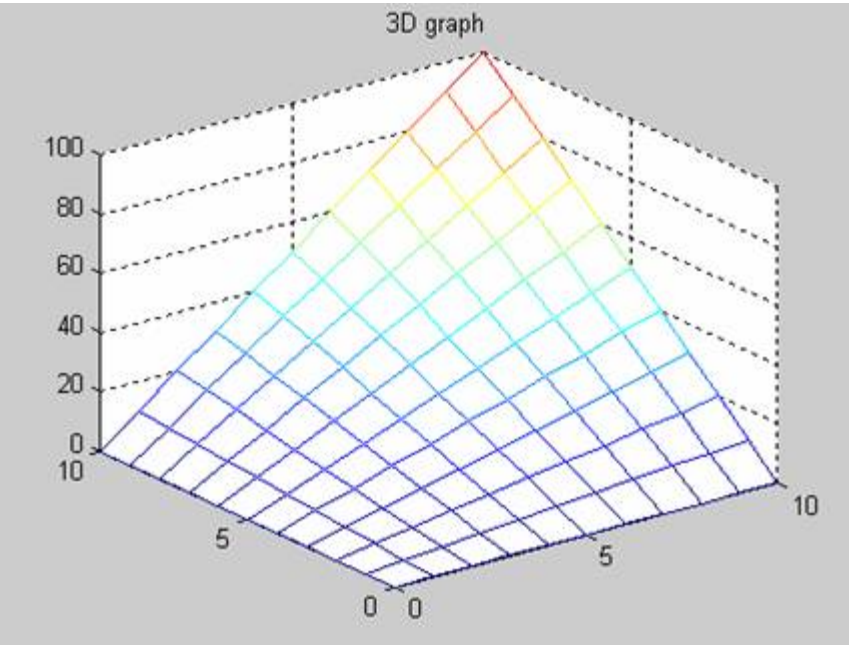
Symbol	Marker
.	•
o	◦
x	×
+	+
*	*
s	□
d	◊
v	∇
^	Δ
h	hexagram

Plot line styles

Symbol	Line Style
–	solid line
:	dotted line
–.	dash-dot line
– –	dashed line

3D - Plotting example

- x=[0:10]; y=[0:10]; z=x'*y;
- mesh(x,y,z); title('3-D Graph');



Graphics

- Bitmap images can also be visualized
- `>> load mandrill`
- `>> image(X); colormap(map)`
- `>> axis image off`

Programming with MATLAB:

- Files that contain code in the MATLAB language are called M-files. You create M-files using a text editor, then use them as you would any other MATLAB functions or command. There are two types of M-files: Scripts and Functions.

Programming in Matlab

- Script Files
  - List of commands to be executed sequentially. Useful when same sequence is executed many times. They should be saved with extension `script.m` and should be placed on work path to be called.
    - `>>script` runs the sequence
- Functions
  - Similar to script, but takes arguments
  - A `funcname.m` file starts with:

    ```
    function [output]=funcname(input)
    ```
  - Any function can be listed with: `>>type funcname`
- See Matlab editor (File>New>m-file)

Scripts

- Scripts: a bunch of code grouped together; doesn't accept argument or return output.
- Example: create a file called magicrank.m that calculates the rank of magic squares:

```
r = zeros(1,20);
for n = 1:20
    r(n) = rank(magic(n));
end
bar(r)
```

- Add the file into search path and type the statement: magicrank

M-Files

So far, we have executed the commands in the command window. But a more practical way is to create a M-file.

- The M-file is a text file that consists a group of MATLAB commands.
- MATLAB can open and execute the commands exactly as if they were entered at the MATLAB command window.
- To run the M-files, just type the file name in the command window. (make sure the current working directory is set correctly)

All MATLAB commands are M-files.

M-files → Functions

- M-files are macros of MATLAB commands that are stored as ordinary text files with the extension "m", that is *filename*.m
- example of an M-file that defines a function, create a file in your working directory named yplusx.m that contains the following commands:

Write this file:

```
function z = yplusx(y,x)
z = y + x;
```

-save the above file(2 lines) as yplusx.m

```
x = 2; y = 3;
z = yplusx(y,x)
```

Functions:

- Functions are M-files that can accept input arguments and return output arguments. The name of the M-file and of the function should be the same.
- For example, the M-file “ode45.m” is available in the directory: ~toolbox/matlab/funfun, you can see the file with

```
type ode45
```

Flow Control:

MATLAB has following flow controls:

- If statement
- Switch statement
- For loops
- While loops
- Continue statement
- Break statement

if … elseif … else … end

```
If A > B
   ‘greater’
elseif A < B
   ‘less’
elseif A = = B
   ‘equal’
else error(‘Unexpected
situation’)
end
```

for … end

```
for i = 1:m
   for j = 1:n
       H(i,j) = 1/(i+j)
   end
end
```

Suppressing Output:

If you simply type a statement and press **Enter,** MATLAB automatically displays the results on screen. If you end the line with a semicolon ‘;’, MATLAB performs the computation but does not display any result.

- Example: C = randn(5,1) v.s.

C = randn(5,1);

Useful Commands

- Single quote is transpose
- % same as // comment in C, Java
  No /* block comments */ (annoying)
- ; suppresses printing
- More:

max(x)	min(x)
mean(x)	median(x)
abs(x)	dot(x,y)
cross(x,y)	flops (flops in this session)

Useful Constants

- Inf infinity
- NaN Not a number (div by zero)
- eps machine epsilon
- pi 3.14159….
- And a variable
  ans most recent unassigned answer

DE lesson

- Get the files from today on openwetware
- Run the script files
- Change the parameter values and rerun