Lecture 6: Plotting in MATLAB

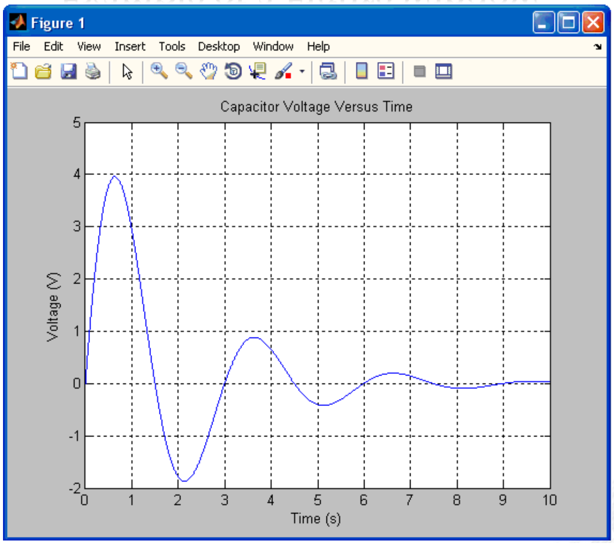
Dr. Mohammed Hawa
Electrical Engineering Department
University of Jordan

EE201: Computer Applications. See Textbook Chapter 5.

A picture is worth a thousand words

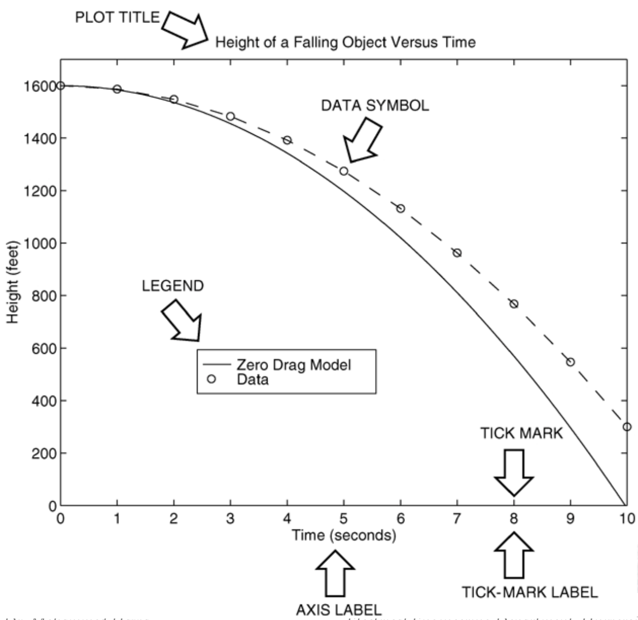
- MATLAB allows you to plot data sets for better visualization and interpretation.
- There are different types of plots available in MATLAB (*see next*) including 2D and 3D plots.
- You can control all aspects of the plot: lines, colors, grids, labels, etc.
- Plotting clear and easy-to-read figures is an important skill, which you gain from experience.
- For pointers, read in your textbook the *Requirements for a Correct Plot* (Table 5.1-1, page 221), and *Hints for Improving Plots* (Table 5.1-3, page 226).

Example of a Figure window



5-11

Nomenclature for a typical *xy* two-dimensional plot.

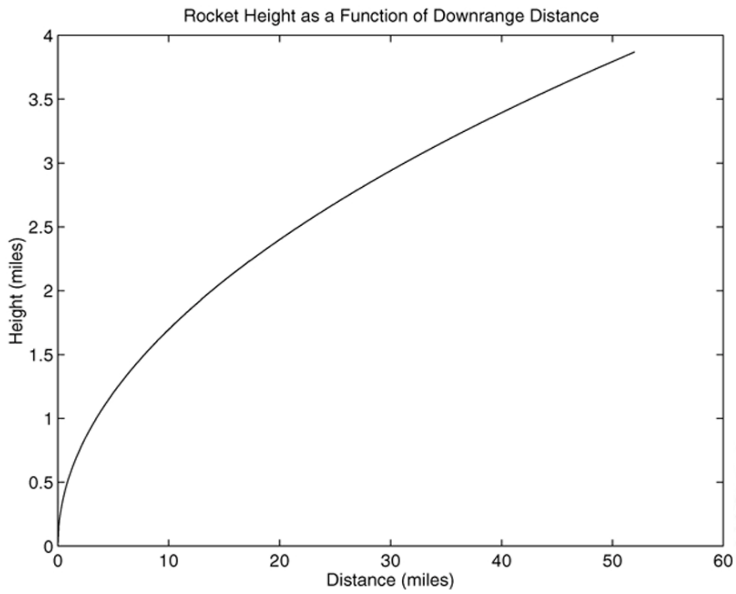


Example: Plot $y = 0.4 \times \sqrt{1.8x}$ for $0 \le x \le 52$, where y represents the height of a rocket after launch, in miles, and x is the horizontal (downrange) distance in miles.

```
>> x = 0:0.1:52;
>> y = 0.4*sqrt(1.8*x);
>> plot(x,y);
>> xlabel('Distance (miles)');
>> ylabel('Height (miles)');
>> title('Rocket Height vs. Distance');
```

Notice that for each x there is y; so MATLAB plots one array against another.
Also notice how we added the axes labels and plot title.
The resulting plot is shown on the next slide.

The autoscaling feature in MATLAB selects tick-mark spacing.



The plot will appear in the Figure window. You can use the plot in other applications in several ways:

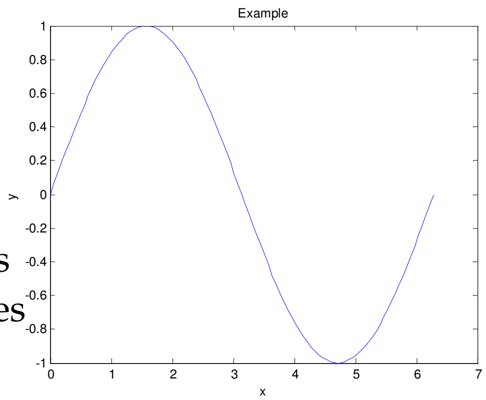
1. You can print a hard copy of the figure by selecting **File | Print** menu item in the Figure window.
2. You can save the plot to a file to be used later. You can save the plot by selecting **File | Save As** menu item. Possible file formats include: *.fig (MATLAB format), *.bmp, *.eps, *.jpg, *.png, *.tif, *.pdf, …. Another way to save is **File | Export Setup** that allows specifying options for the output file, then selecting **Export**.
3. You can copy a figure to the clipboard and then paste it into another application using the **Edit | Copy Figure** menu item. For options, use **Edit | Copying Options** menu item.

When you have finished with the plot, close the figure window by selecting **File | Close** menu item in the figure window.

If you do not close the window, it will not re-appear when a new `plot` command is executed. However, the figure will still be updated.

One Data Set: `plot`

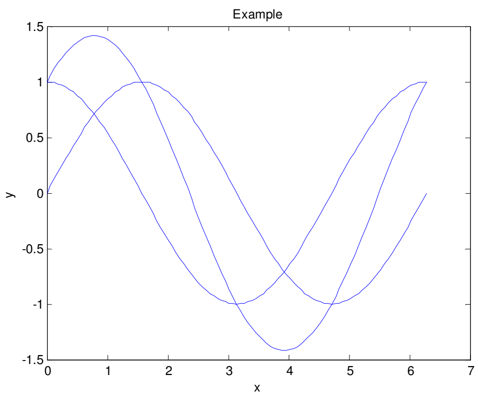
```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
plot(x,y1);
xlabel('x');
ylabel('y');
title('Example');
```



`plot(y1)`: Plots values of y1 versus their indices if y1 is a vector.

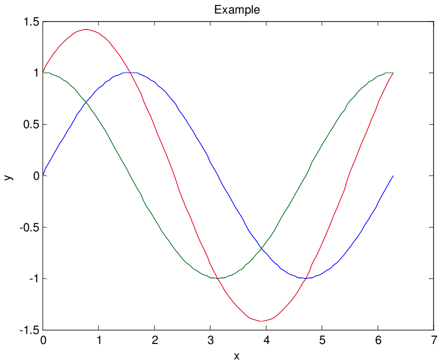
Multiple Data Sets: `plot`, `hold`

```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
y3 = sin(x)+cos(x);
plot(x,y1);
hold on;
plot(x,y2);
plot(x,y3);
xlabel('x');
ylabel('y');
title('Example');
hold off;
```



Or better use one `plot` command

```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
y3 = sin(x)+cos(x);
plot(x,y1,x,y2,x,y3);
xlabel('x');
ylabel('y');
title('Example');
% Notice the auto coloring
% by  MATLAB
```

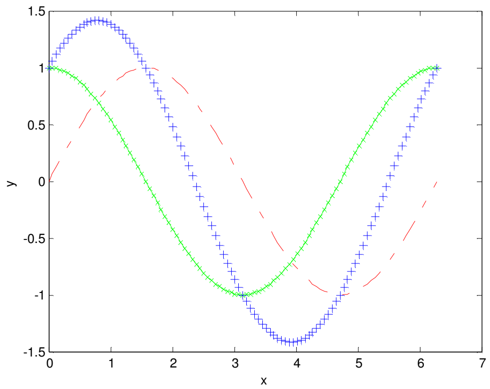


Colors, Data Markers & Line Types

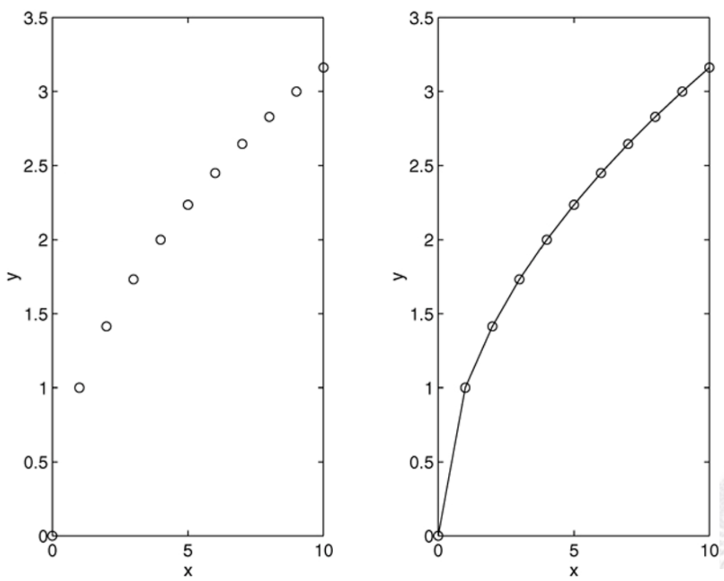
- You can also specify your own line styles in the plot command.
- For full details enter `help plot` in MATLAB.

```
b     blue          .     point              -       solid
g     green         o     circle             :       dotted
r     red           x     x-mark             -.      dashdot
c     cyan          +     plus               --      dashed
m     magenta       *     star             (none)    no line
y     yellow        s     square
k     black         d     diamond
w     white         v     triangle (down)
                    ^     triangle (up)
                    <     triangle (left)
                    >     triangle (right)
                    p     pentagram
                    h     hexagram
```

```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
y3 = sin(x)+cos(x);
plot(x,y1,'r-.',x,y2,'g-x',x,y3,'b+');
xlabel('x');
ylabel('y');
```



Exercise: How did we use different data markers below?

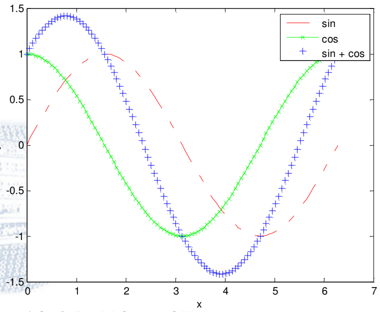


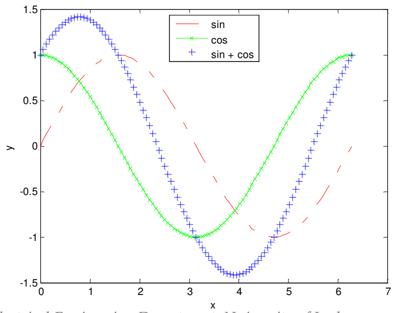
Legends

- With multiple lines on the same plot it is a good idea to add a legend.

```
legend('sin','cos','sin + cos');
legend('sin','cos','sin+cos','Location','North');
```

- You can also move the legend with the mouse.





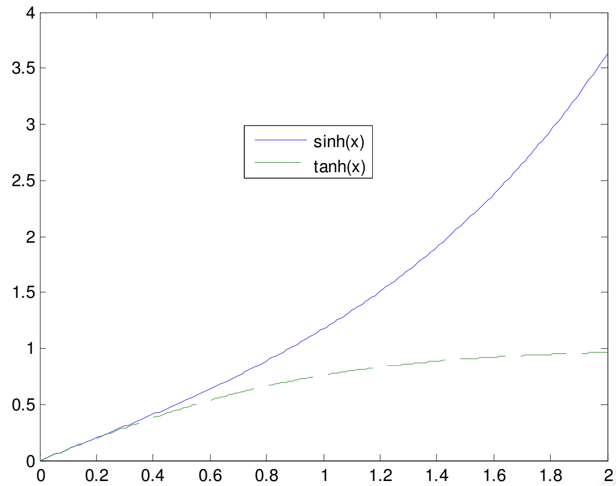
Labeling Curves and Data

The `legend` command automatically obtains from the plot the line type used for each data set and displays a sample of this line type in the legend box next to the string you selected. The following script file produced the plot in the next slide.

```
x = 0:0.01:2;
y = sinh(x);
z = tanh(x);
plot(x,y,x,z,'--');
legend('sinh(x)', 'tanh(x)');
```

`gtext('text')`: Places a string in the Figure window at a point specified by the mouse.
`text(x,y,'text')`: Places a string in the Figure window at a point specified by coordinates x, y.

Application of the `legend` command.
I moved the legend to an empty space using the mouse.



The `grid` and `axis` Commands

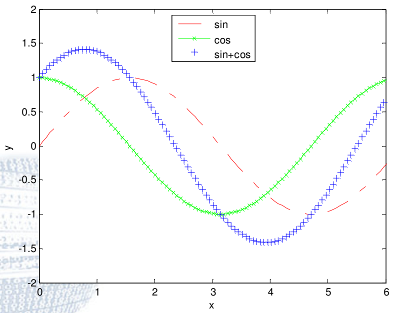
MATLAB will automatically determine the maximum and minimum values for the axes. You can use the `axis` command to override the MATLAB selections for the axis limits. The syntax is `axis([xmin xmax ymin ymax])`. This command sets the scaling for the *x*- and *y*-axes to the minimum and maximum values indicated.

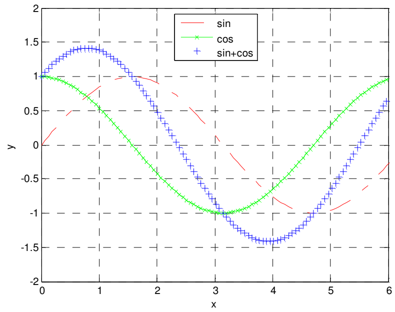
The `grid` command displays gridlines at the tick marks corresponding to the tick labels. Type `grid on` to add gridlines; type `grid off` to stop plotting gridlines. When used by itself, `grid` toggles this feature on or off, but you might want to use `grid on` and `grid off` to be sure.

`axis` and `grid` commands

```
axis([0 9 -2 2]);
axis([0 6 -2 2]);
```

```
grid on;
grid off;
```

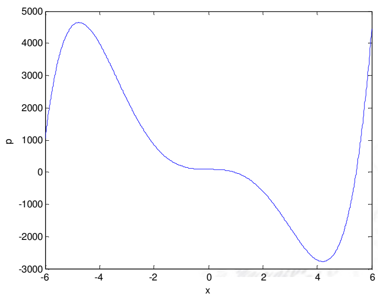




Homework #1
Plotting Polynomials with the `polyval` Function.

To plot the polynomial $3x^5 + 2x^4 - 100x^3 + 2x^2 - 7x + 90$ over the range $-6 \leq x \leq 6$ with a spacing of 0.01, you type

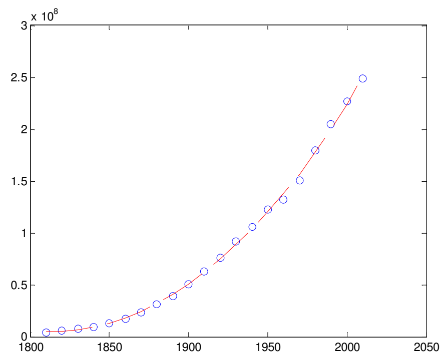
```
>> x = -6:0.01:6;
>> p = [3,2,-100,2,-7,90];
>> plot(x,polyval(p,x));
>> xlabel('x');
>> ylabel('p');
```



Homework #2

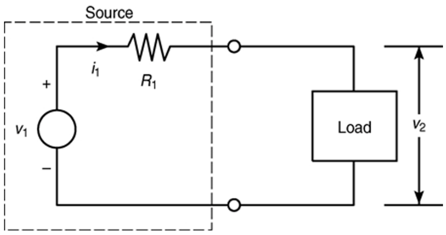
- The `polyfit` function is based on the least-squares method. It fits a polynomial of degree n to data described by the vectors x and y, where x is the independent variable.
- Syntax: `p = polyfit(x,y,n)`
- It returns a row vector p of length n+1 that contains the polynomial coefficients in order of descending powers.
- For the following census data, draw the actual points and the best 5th order polynomial fit for such data.


```
year = 1810:10:2010;
population = 1e6*[3.9 5.3 7.2 9.6 12.9 17.1
23.1 31.4 38.6 50.2 62.9 76. 92. 105.7 122.8
131.7 150.7 179. 205. 226.5 248.7];
coeff = polyfit(year, population, 5)
f = polyval(coeff, year);
plot(year, population, 'bo', year, f, 'r--');
```



Homework #3
Graphical solution of an Electrical System

- Load is governed by:
- $i1 = 0.16\,(e^{0.12v_2} - 1)$
- What is the equation for the practical source? Assume:
- $R1 = 30\Omega, v_1 = 15V$
- Find the correct value for $v2$ between 0 and 20V, and also i_1 value



Solution

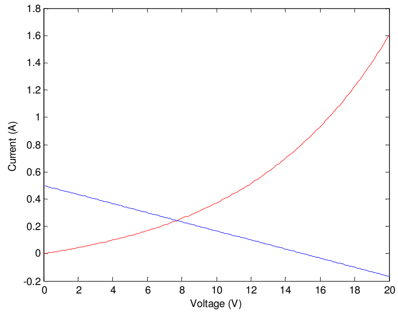
- The equation for the power supply is:

$$v_2 = v_1 - Ri_1$$

$$i_1 = \frac{15 - v_2}{30}$$

- If we draw both equations we can see the solution point (the one that satisfies both equations).

```
v2 = [0:0.1:20];
i_load = ...
0.16*(exp(0.12*v2) - 1);
i_source = (15-v2)/30;
plot(v2, i_load, 'r', ...
v2, i_source, 'b');
```



More Than One Figure Window

- What happens if you enter the following?

```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
plot(x,y1);
title('Plot #1');
plot(x,y2);
title('Plot #2');
```

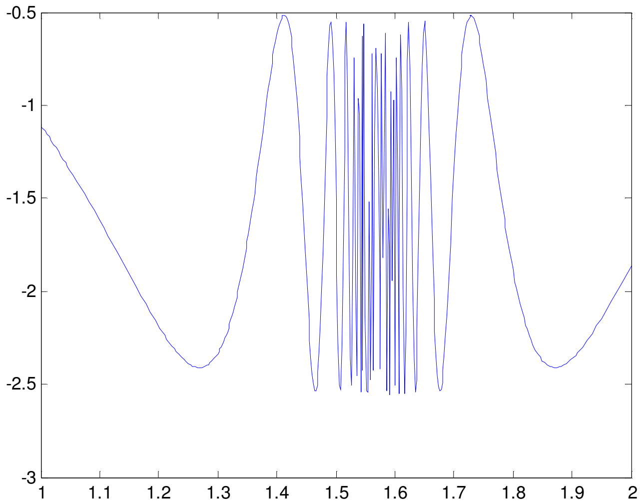

More Than One Figure Window

- … you end up with one figure window and it contains a plot of $y = \cos(x)$.
- To open a new figure window enter the command `figure` before making the second plot.

```
plot(x,y1);
title('Plot #1');
figure;
plot(x,y2);
title('Plot #2');
```

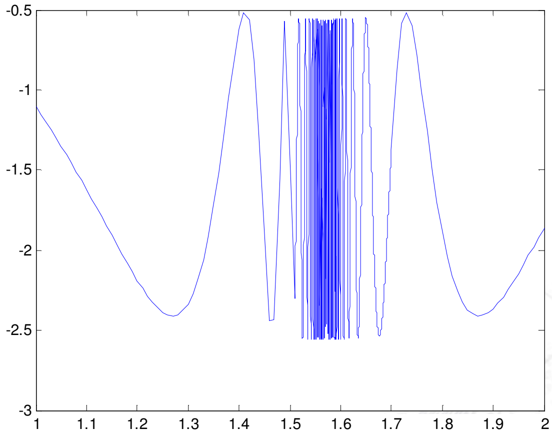

The `fplot` command is a "smart" plotting function. Example:

```
f = @(x) (cos(tan(x)) - tan(sin(x)));
fplot(f,[1 2]);
```



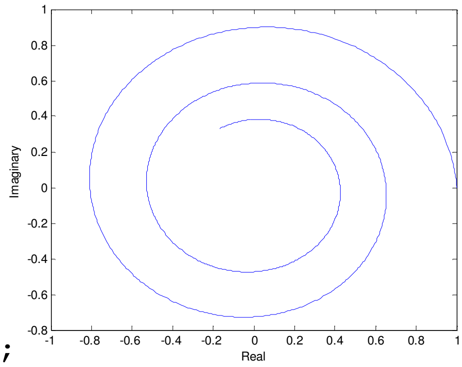
The `plot` command is more common than the `fplot` command because it gives more control. Also when you `type fplot` you see it actually uses `plot`.

```
f = @(x) (cos(tan(x)) - tan(sin(x)));
t=[1:0.01:1.5, 1.51:0.0001:1.7, 1.71:0.01:2];
plot(t, f(t));
```



Complex Plot: Real vs. Imaginary

```
n = [0:0.01:10];
y = (0.1+0.9j).^n;
plot(y);
xlabel('Real');
ylabel('Imaginary');
```



- Similar to:

```
plot(real(y),imag(y));
```

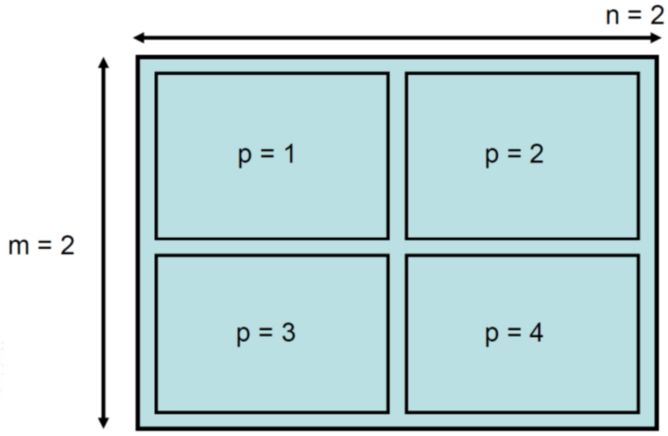

Subplots

You can use the `subplot` command to obtain several smaller "subplots" in the same figure. The syntax is `subplot(m,n,p)`. This command divides the Figure window into an array of rectangular panes with *m* rows and *n* columns. The variable `p` tells MATLAB to place the output of the `plot` command following the `subplot` command into the *p*th pane.

For example, `subplot(3,2,5)` creates an array of six panes, three panes deep and two panes across, and directs the next plot to appear in the fifth pane (in the bottom-left corner).

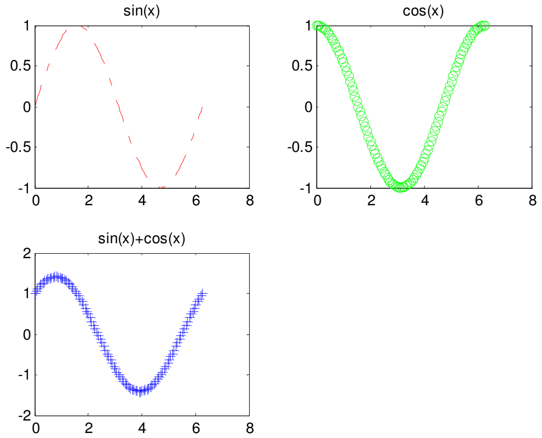
Subplots

- `subplot(m,n,p)`



Example

```
x = 0:2*pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
y3 = sin(x)+cos(x);
subplot(2,2,1);
plot(x,y1,'r-.');
title('sin(x)');
subplot(2,2,2);
plot(x,y2,'go');
title('cos(x)');
subplot(2,2,3);
plot(x,y3,'b+');
title('sin(x)+cos(x)');
```

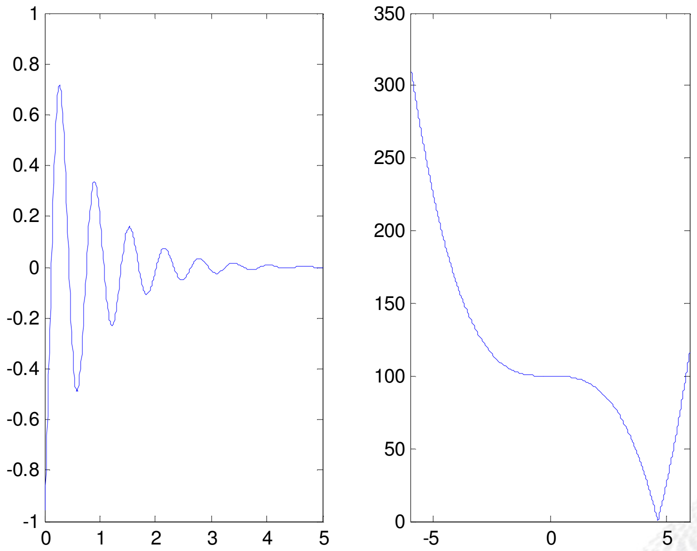


Homework:
The following script file shows two plots of the functions
$y = e^{-1.2x} \sin(10x + 5)$ for $0 \leq x \leq 5$
and $y = |x^3 - 100|$ for $-6 \leq x \leq 6$.

```
x = 0:0.01:5;
y = exp(-1.2*x).*sin(10*x+5);
subplot(1,2,1);
plot(x,y);
axis([0 5 -1 1]);
x = -6:0.01:6;
y = abs(x.^3-100);
subplot(1,2,2);
plot(x,y);
axis([-6 6 0 350])
```

The figure is shown on the next slide.

Application of the `subplot` command.

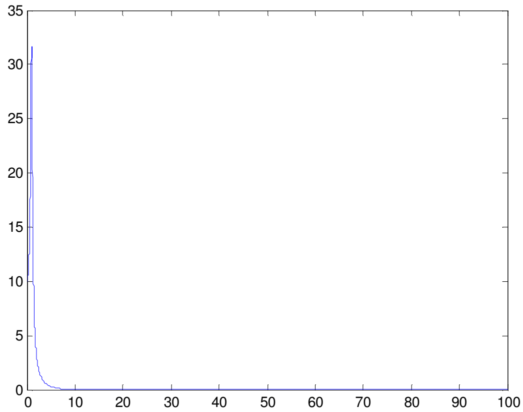


Log-scale Plots

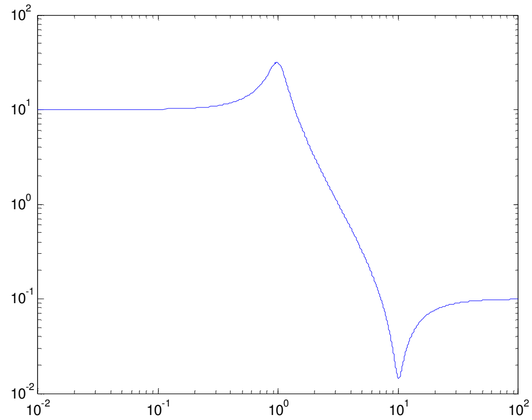
- Why use log scales? Linear scales cannot properly display wide variations in data values.
- MATLAB has three commands. The appropriate command depends on which axis you want to be a log scale.
- `loglog(x,y)`: both scales logarithmic.
- `semilogx(x,y)`: x-axis is logarithmic and y-axis is rectilinear.
- `semilogy(x,y)`: y-axis is logarithmic and x-axis is rectilinear.
- The syntax is similar to the `plot` command.

$$y = \sqrt{\frac{100(1 - 0.01x^2)^2 + 0.02x^2}{(1 - x^2)^2 + 0.1x^2}} \qquad 0.1 \le x \le 100$$


```
x = [0.1:0.01:100];
y = sqrt((100*(1-0.01*x.^2).^2 ...
+0.02*x.^2) ...
./ ((1-x.^2).^2+0.1*x.^2));
plot(x,y);
```




```
x = [0.1:0.01:100];
y = sqrt((100*(1-0.01*x.^2).^2 ...
+0.02*x.^2) ...
./ ((1-x.^2).^2+0.1*x.^2));
loglog(x,y);
```



Logarithmic Plots

It is important to remember the following points when using log scales:

1. You cannot plot negative numbers on a log scale, because the logarithm of a negative number is not defined as a real number.
2. You cannot plot the number 0 on a log scale, because $\log_{10} 0 = \ln 0 = -\infty$. You must choose an appropriately small number as the lower limit on the plot.

(continued…)

Logarithmic Plots (continued)

3. The tick-mark labels on a log scale are the actual values being plotted; they are not the logarithms of the numbers. For example, the range of *x* values in the plot in the above Figure is from $10^{-2} = 0.01$ to $10^{2} = 100$.

4. Gridlines and tick marks within a decade are unevenly spaced. If 8 gridlines or tick marks occur within the decade, they correspond to values equal to 2, 3, 4, . . . , 8, 9 times the value represented by the first gridline or tick mark of the decade.

(continued…)

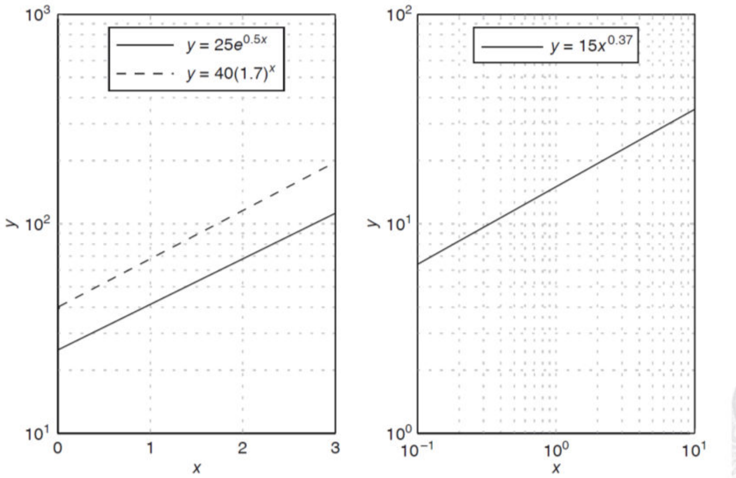
Logarithmic Plots (continued)

5. Equal distances on a log scale correspond to multiplication by the same constant (as opposed to addition of the same constant on a rectilinear scale).

For example, all numbers that differ by a factor of 10 are separated by the same distance on a log scale. That is, the distance between 0.3 and 3 is the same as the distance between 30 and 300. This separation is referred to as a *decade* or *cycle.*

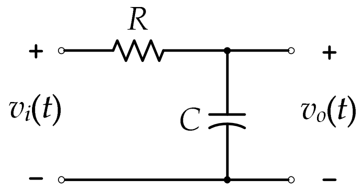
The plot shown in the above Figure covers four decades in *x* (from 0.01 to 100) and four decades in *y.*

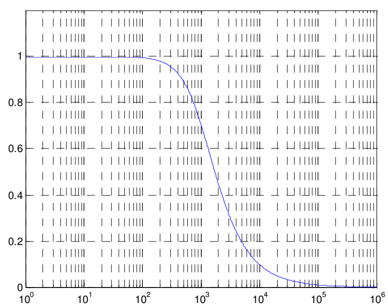
Homework: reproduce the following plots. What commands did you use?



Homework

- For the first-order RC circuit, which acts as a LPF, the output to input ratio is:
- $|H(\omega)| = \frac{|V_o(\omega)|}{|V_i(\omega)|} = \left|\frac{1}{1+j\omega RC}\right|$
- Sketch this frequency response function using `semilogx`. Assume: $R = 1k\Omega, C = 1\mu F$





Solution

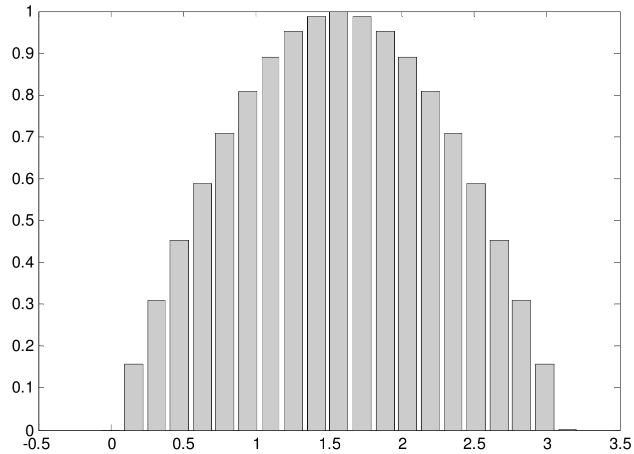
```
omega = 0:1:1e6;
h = abs(1./(1+i*omega*1e3*1e-6));
semilogx(omega, h);
axis([0 1e6 0 1.2]);
grid on;
```

Q. What is the bandwidth of this LPF?

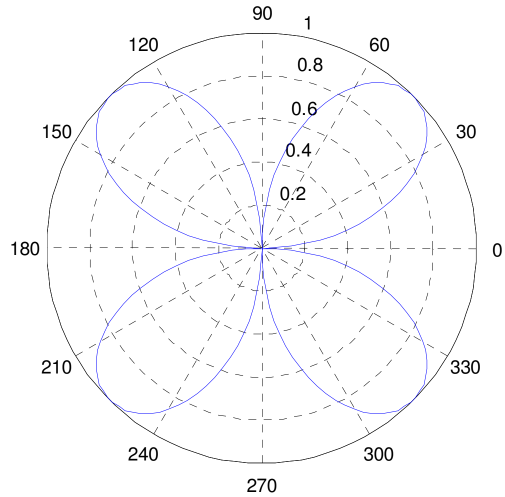
Specialized plot commands.

Command	Description
`bar(x,y)`	Creates a bar chart of `y` versus `x`.
`plotyy(x1,y1,x2,y2)`	Produces a plot with two *y*-axes, `y1` on the left and `y2` on the right.
`polar(theta,r,'type')`	Produces a polar plot from the polar coordinates `theta` and `r`, using the line type, data marker, and colors specified in the string `type`.
`stairs(x,y)`	Produces a stairs plot of `y` versus `x`.
`stem(x,y)`	Produces a stem plot of `y` versus `x`.

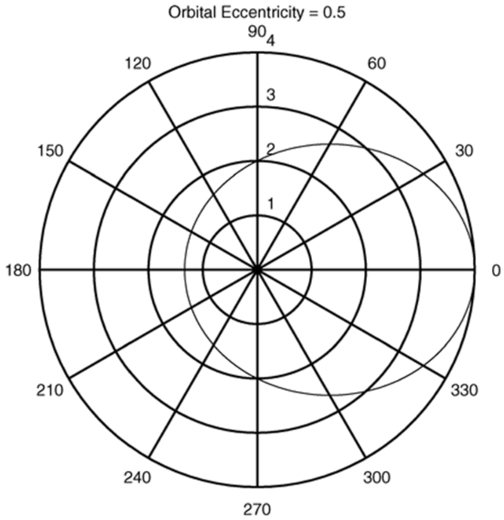

```
x = [0:pi/20:pi];
bar(x,sin(x));
```




```
theta = [0:pi/90:2*pi];
polar(theta , sin(2*theta));
grid;
```

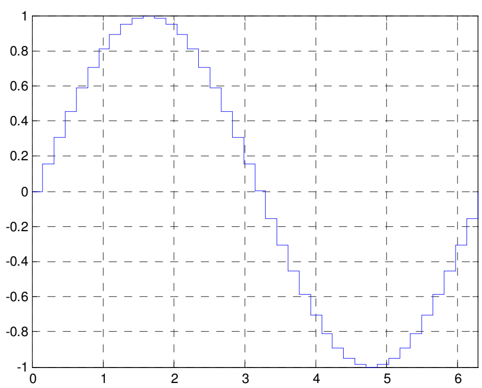


Homework: Reproduce the following plot for an orbit with an eccentricity of 0.5.

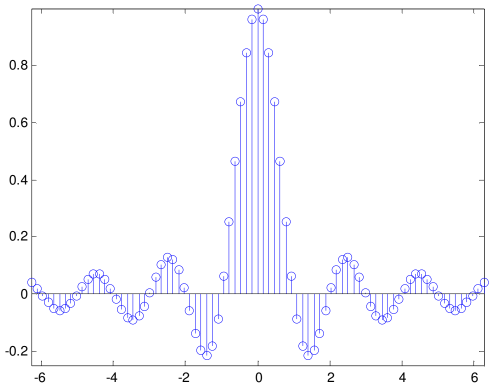


$$r = \frac{2}{1 - 0.5\cos(\theta)}$$

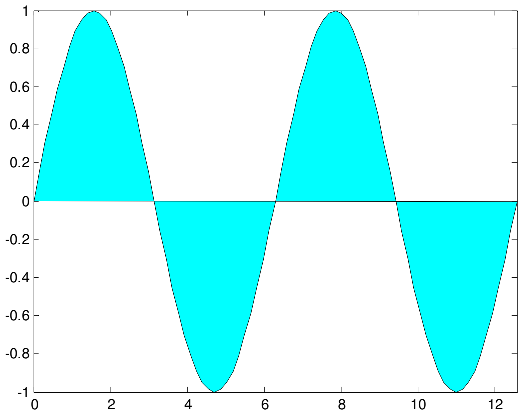

```
x = [0:pi/20:2*pi];
stairs(x,sin(x));
grid;
axis([0 2*pi -1 1]);
```



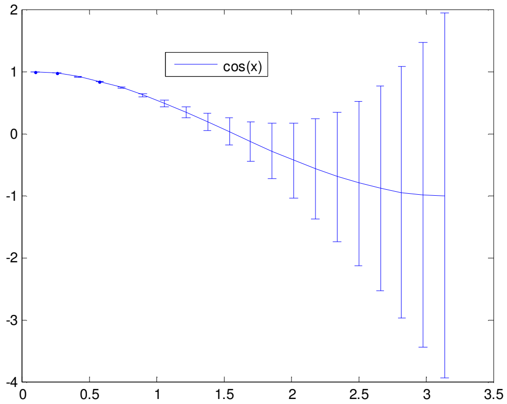
```
x = [-2*pi:pi/20:2*pi];
x = x + (~x)*eps;
y = sin(pi*x)./(pi*x);
stem(x,y);
axis([-2*pi 2*pi -.25 1]);
```




```
x = [-2*pi:pi/20:4*pi];
fill(x,sin(x),'c');
axis([0 4*pi -1 1]);
```



```
x = linspace(0.1, pi, 20);
approx = 1 - x.^2/2;
error = approx - cos(x);
errorbar(x, cos(x), error);
legend('cos(x)');
```

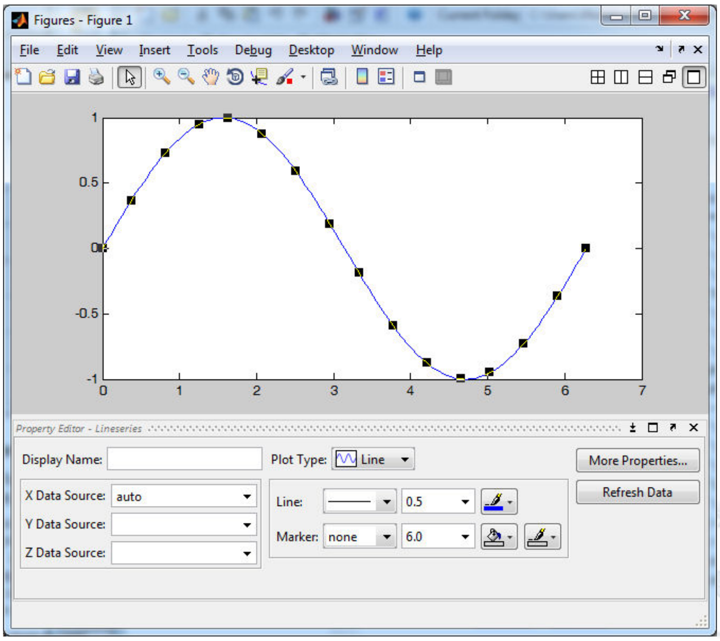


Interactive Editing of Plots in MATLAB

This interface can be advantageous in situations where:

- You want to add annotations such as lines, arrows, text, rectangles, and ellipses.
- You want to change plot characteristics such as tick spacing, fonts, bolding, colors, line weight, etc.

Select the Arrow (or **Tools| Edit Plot** from the menu) then double click on the portion you want to edit.

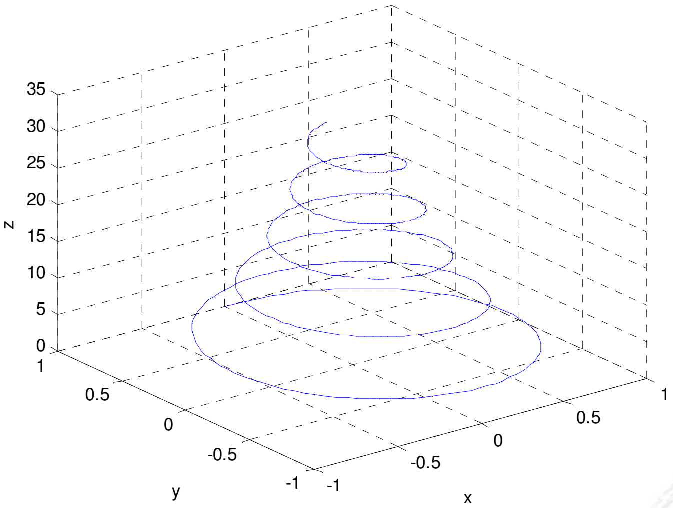


Three-Dimensional Line Plots

The following program uses the `plot3` function to generate the spiral curve shown in the next slide.

```
t = 0:pi/50:10*pi;
x = exp(-0.05*t).*sin(t);
y = exp(-0.05*t).*cos(t);
z = t;
plot3(x, y, z);
xlabel('x'),ylabel('y'),zlabel('z'),grid;
```

The curve $x = e^{-0.05t} \sin t$, $y = e^{-0.05t} \cos t$, $z = t$ plotted with the `plot3` function.



 Electrical Engineering Department, University of Jordan 55

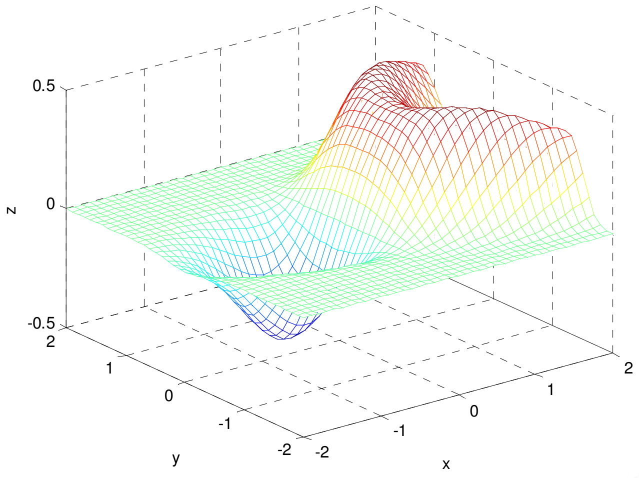
Surface Plots: `mesh` and `surf`

The following session shows how to generate the surface plot of the function $z = xe^{-[(x-y^2)^2+y^2]}$, for $-2 \le x \le 2$ and $-2 \le y \le 2$, with a spacing of 0.1. This plot appears in the next slide.

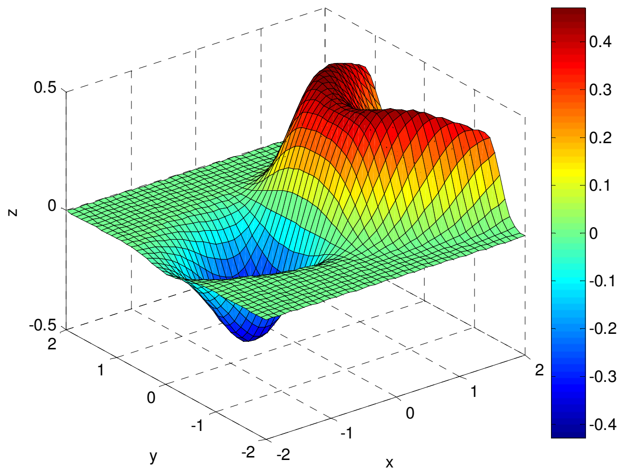
```
[X,Y] = meshgrid(-2:0.1:2);
Z = X.*exp(-((X-Y.^2).^2+Y.^2));
mesh(X,Y,Z);
xlabel('x'),ylabel('y'),zlabel('z');

[X,Y] = meshgrid(-2:0.1:2);
Z = X.*exp(-((X-Y.^2).^2+Y.^2));
surf(X,Y,Z);
xlabel('x'),ylabel('y'),zlabel('z'),colorbar
```

 Electrical Engineering Department, University of Jordan 56

A plot of the surface $z = xe^{-[(x-y^2)^2+y^2]}$ created with the `mesh` function.



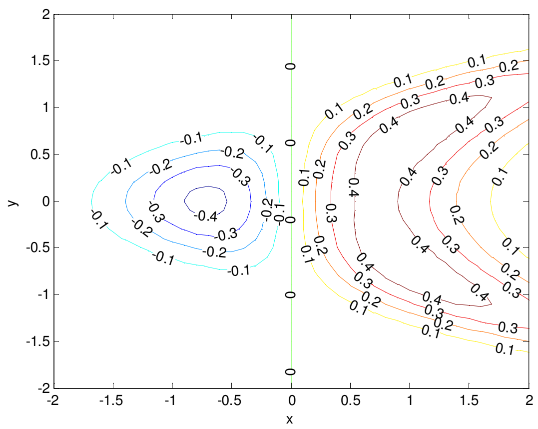
A plot of the surface $z = xe^{-[(x-y^2)^2+y^2]}$ created with the `surf` function.

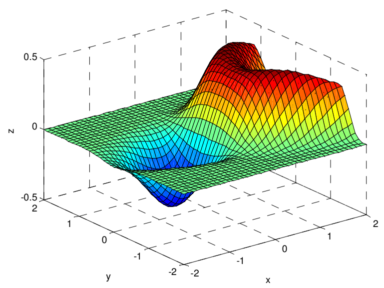


The following session generates the contour plot of the function whose surface plot is shown above; namely, $z = xe^{-[(x-y^2)^2+y^2]}$, for $-2 \le x \le 2$ and $-2 \le y \le 2$, with a spacing of 0.1. This plot appears in the next slide.

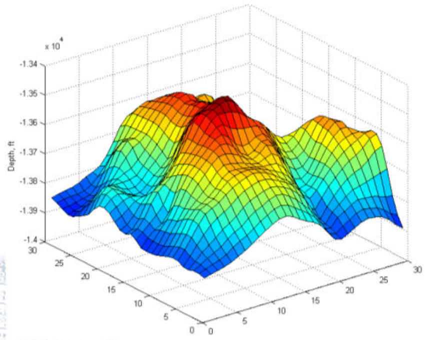
```
[X,Y] = meshgrid(-2:0.1:2);
Z = X.*exp(-((X-Y.^2).^2+Y.^2));
[cs, h] = contour(X,Y,Z);
xlabel('x'),ylabel('y'),zlabel('z');
clabel(cs, h, 'labelspacing', 72);
```

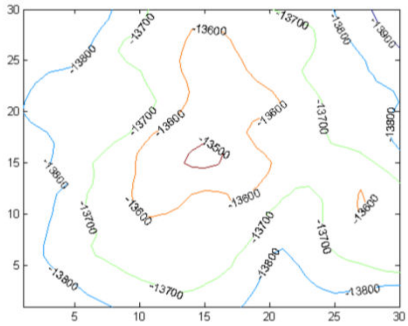

A contour plot of the surface $z = xe^{-[(x-y^2)^2+y^2]}$ created with the `contour` function.





Contours are useful for Terrain





Vector fields: `quiver`

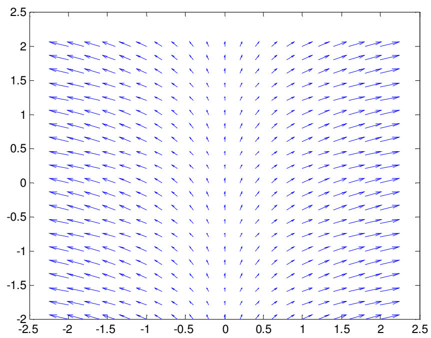
- `quiver` draws little arrows to indicate a gradient or other vector field.
- Although it produces a 2-D plot, it is often used in conjunction with contour. As an example, consider the scalar function of two variables: $V = x^2 + y$.
- The gradient of V is defined as the vector field: $\nabla V = \left(\frac{\partial V}{\partial x}, \frac{\partial V}{\partial y}\right) = (2x, 1)$

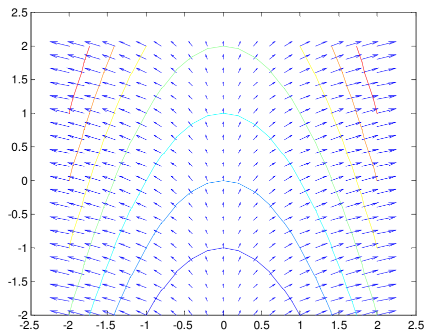
quiver

- The following statements draw arrows indicating the direction of the vector ∇V at points in the x-y plane (see next slide).

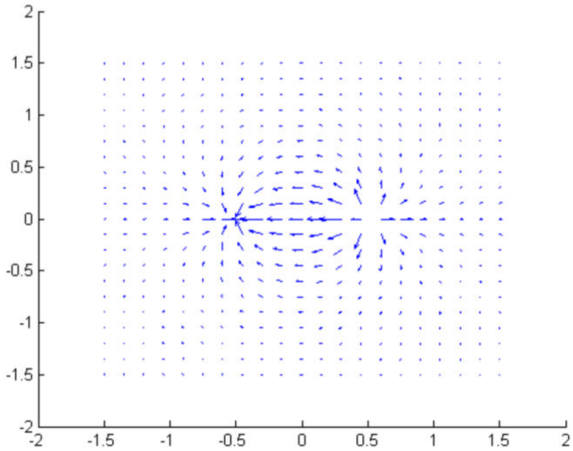
```
[x y] = meshgrid(-2:0.2:2, -2:0.2:2);
V = x.^2 + y;
dx = 2*x;
dy = ones(size(dx)); % dy same size as dx
quiver(x, y, dx, dy);
hold on;
contour(x, y, V);
hold off;
```


quiver alone; and with contour



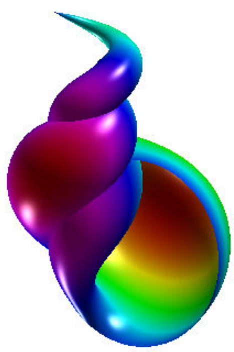


Useful for Electromagnetic Fields



Homework

```
% What is the output of this MATLAB code? Use help if you need.
figure;
t = linspace(0, 2*pi, 512);
[u,v] = meshgrid(t) ;
a = -0.2 ; b = .5 ; c = .1 ;
x = (a*(1-v/(2*pi)) .* (1+cos(u)) + c) .* cos(2*v);
y = (a*(1-v/(2*pi)) .* (1+cos(u)) + c) .* sin(2*v);
z = b*v/(2*pi) + a*(1-v/(2*pi)) .* sin(u);
surf(x,y,z,y);
shading interp;
axis off;
axis equal;
colormap(hsv(1024));
material shiny;
lighting gouraud;
lightangle(80, -40);
lightangle(-90, 60);
view([-150 10]);
```



Animation and Movies!

- A movies is just successive plots seen in quick succession.
- We can plot data repeatedly on a single figure.
- For example the function $y = \sin(x + t)$

```
x = 0:2*pi/100:2*pi;
for t = 0:0.05:5 % 5 seconds
  y = sin(x+t);
  plot(x,y,'k')
  pause(0.2); % 200 ms between frames
end
```


Homework: Creating Movies

- To create a movie a sequence of frames are "grabbed" from the figure, stored in an array and written out as .avi file.

```
nFrame = 1; % frame counter
x = 0:2*pi/100:2*pi;
for t=0:0.05:5
   y=sin(x+t);
   plot(x,y);
   pause(0.2);
   movie(nFrame) = getframe; % grab frame & store it
   nFrame = nFrame + 1;
end
movie2avi(movie,'animation.avi'); % save movie
```

Homework

- Solve as many problems from Chapter 5 as you can
- Suggested problems:
- Solve: 5.3, 5.5, 5.9, 5.11, 5.15, 5.20, 5.27, 5.29, 5.35, 5.36, 5.39.