Scientific Computing Lecture Series
Introduction to MATLAB

Süleyman Yıldız*

*Scientific Computing, Institute of Applied Mathematics

Lecture III
Graphics, Visualizations and Symbolic Toolbox

Lecture III–Outline

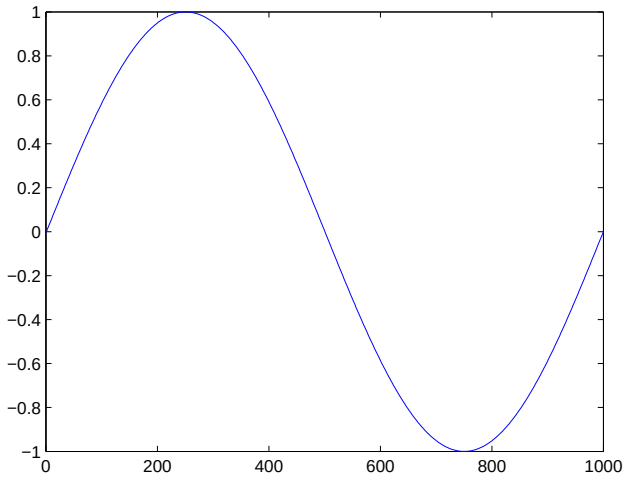
Basic Plotting

- plot() generates dots at each (x, y) pair and then connects the dots with a line

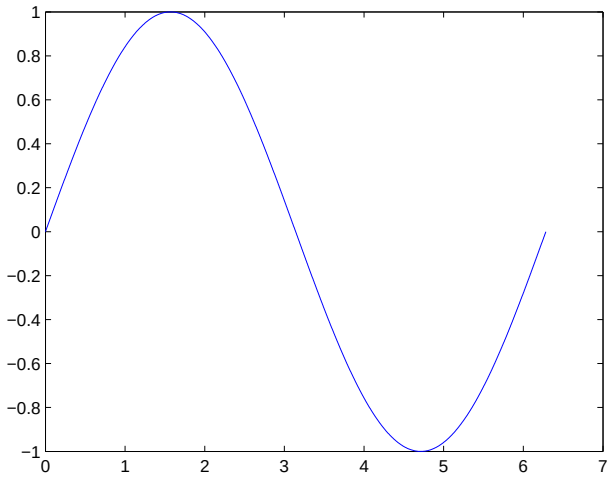
```
>> x = linspace(0,2*pi,1000);
>> y = sin(x);
>> plot(x,sin(x))
```

- Plot values against their index

```
>> plot(y)
```



(a) plot(y)



(b) plot(x,sin(x))

Basic Plotting

- figure To open a new Figure and avoid overwriting plots

```
>> x = [-pi:0.1:pi];
>> y = sin(x);
>> z = cos(x);
>> plot(x,y);   (automatically creates a new Figure!)
>> figure
>> plot(x,z);
```

- close Close figures

```
>> close 1
>> close all
```

- hold on/off Multiple plots in same graph

```
>> plot(x,y);   hold on
>> plot(x,z,'r');  hold off
```

Basic Plotting

- To make plot of a function look smoother, evaluate at more points
- x and y vectors must be same size or else you will get an error

```
>> plot([1,2],[1 2 3])
??? Error using ==> plot
Vectors must be the same lengths.
```

- To add a title

```
>> title('My first title')
```

- To add axis labels

```
>> xlabel('x-label')
>> ylabel('y-label')
```

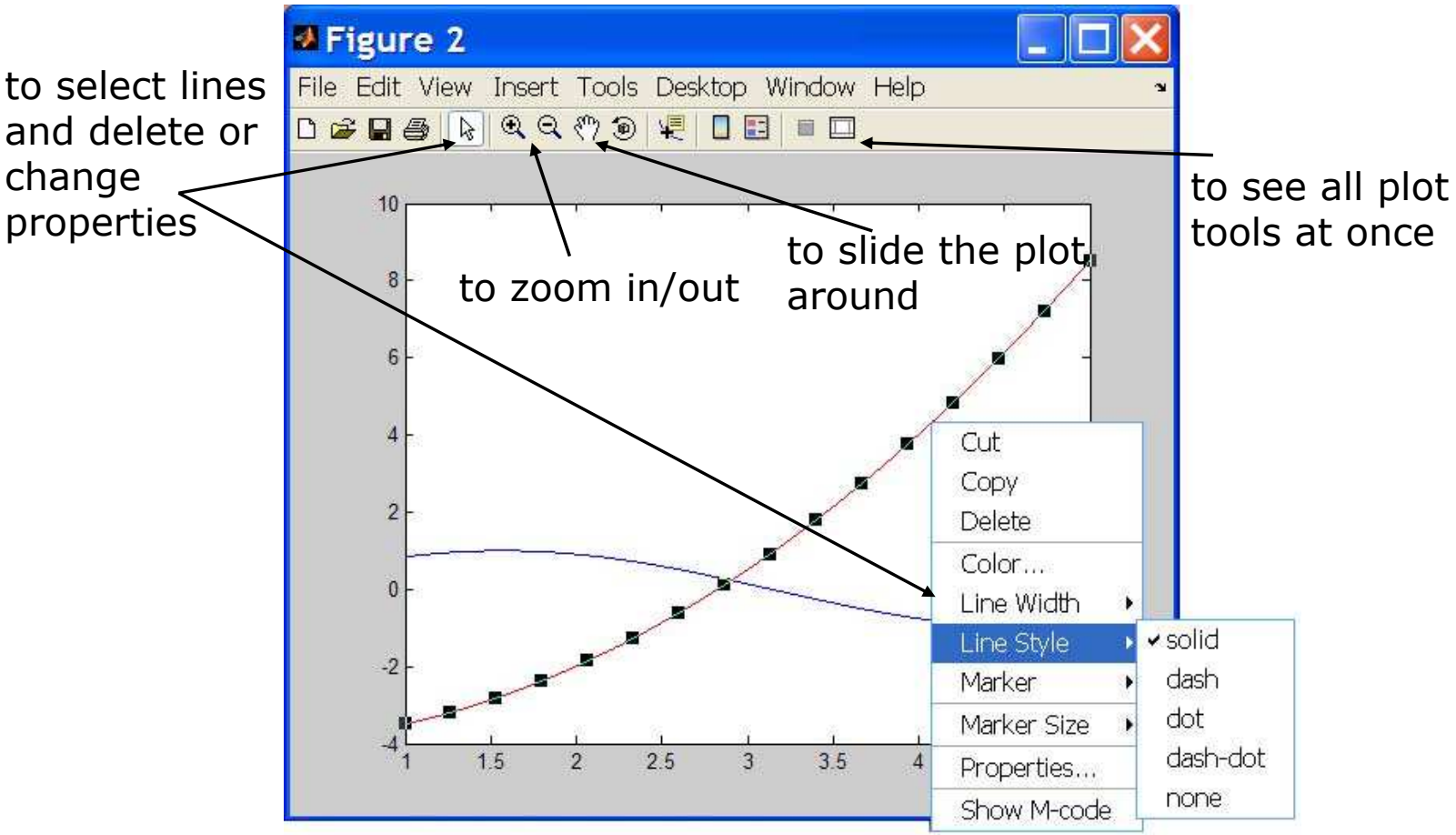
- Can change the line color, marker style, and line style by adding a string argument

```
>> plot(x,y,'k.-');
```

- Basic legend syntax:

legend('First plotted','second plotted', 'Location','Northwest')

Playing with the Plot

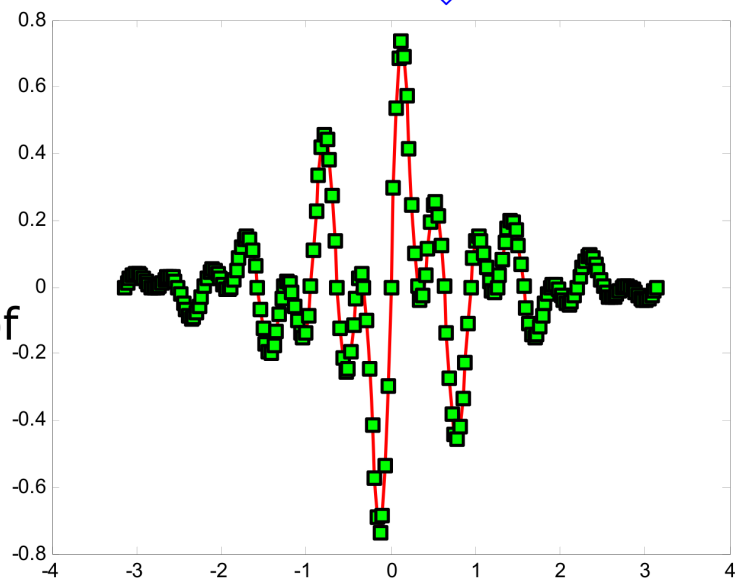


Line and Marker Options

- Everything on a line can be customized

```
» plot(x,y,'--s','LineWidth',2,...
          'Color', [1 0 0], ...
          'MarkerEdgeColor','k',...
          'MarkerFaceColor','g',...
          'MarkerSize',10)
```

You can set colors by using a vector of [R G B] values or a predefined color character like 'g', 'k', etc.



- See **doc line_props** for a full list of properties that can be specified

Basic Plotting

- semilogx logarithmic scales for x-axis
- semilogy logarithmic scales for y-axis
- loglog logarithmic scales for the x,y-axes
- plotyy 2D line plot: y–axes both sides

```
>> plotyy(X1,Y1,X2,Y2) (plot X1,Y1 using left axis and X2,Y2 using right axis)
```

- errorbar errors bar along 2D line plot

```
>> errorbar(X,Y,E)   (create 2D line plot from data X, Y
                       with symmetric error bars defined E)
>> errorbar(X,Y,L,U) (create 2D line plot from data X, Y  with upper error
                       bar defined by U and lower error bar defined by L)
```

Basic Plotting

- bar, barh Vertical, horizontal bar graph

```
>> x = 100*rand(1,20);
>> bar(x)
>> xlabel('x');
>> ylabel('values');
>> axis([0 21 0 120]);
>> title('First Bar');
```

- pie, pie3 2D, 3D pie chart

```
>> x = 100*rand(1,5);
>> pie(x)
>> title('my first pie');
>> legend('val1','val2','val3','val4','val5');
```

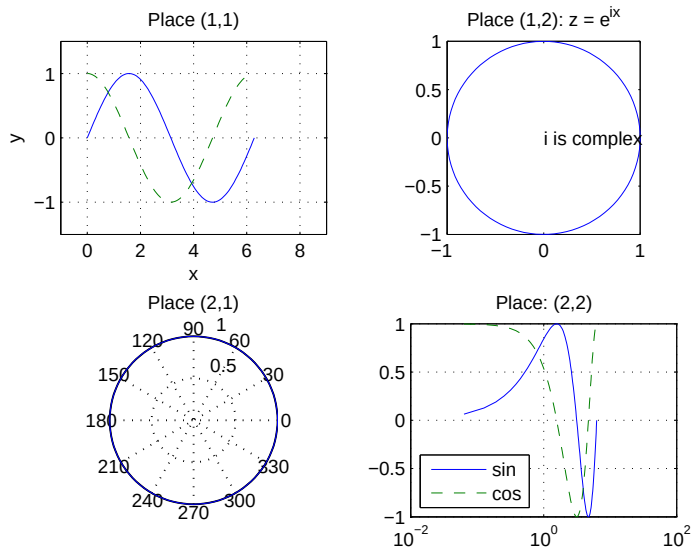
- area Filled are 2D plot

```
>> x = [-pi:0.01:pi]; y=sin(x);
>> plot(x,y); hold on;
>> area(x(200:300),y(200:300));
>> area(x(500:600),y(500:600)); hold off
```

Subplot

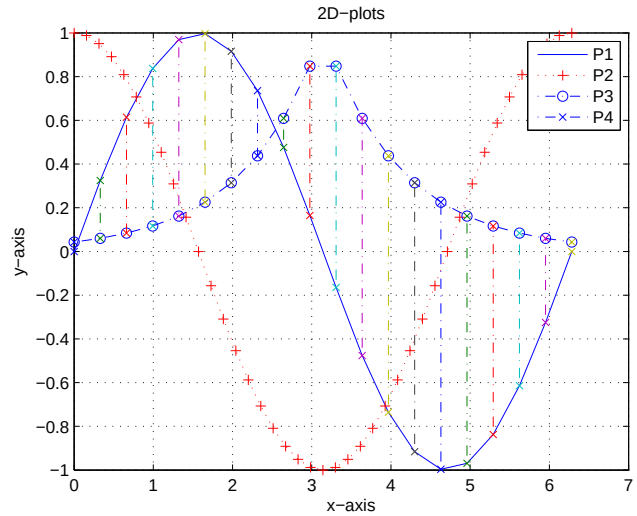
- subplot() Multiple plots in the same figure

```
>> x = linspace(0,2*pi);
>> subplot(2,2,1); plot(x, sin(x), x, cos(x), '--')
>> axis([-1 9 -1.5 1.5])
>> xlabel('x'), ylabel('y'), title('Place (1,1)'), grid on
>> subplot(2,2,2); plot(exp(i*x)), title('Place (1,2): z = e^{ix}')
>> axis square, text(0,0, 'i is complex')
>> subplot(2,2,3); polar(x, ones(size(x))), title('Place (2,1)')
>> subplot(2,2,4); semilogx(x,sin(x), x,cos(x), '--')
>> title('Place: (2,2)'), grid on
>> legend('sin', 'cos', 'Location', 'SouthWest')
```



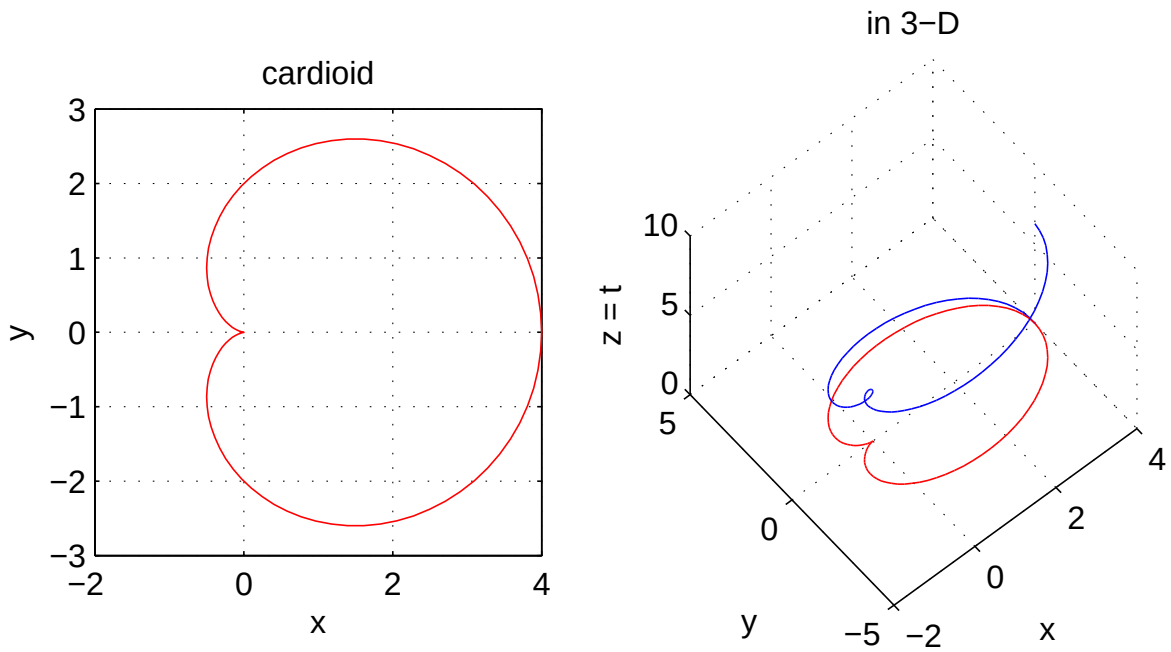
Continue...

```
>> x1 = linspace(0,2*pi,20); x2 = 0:pi/20:2*pi;
>> y1 = sin(x1); y2 = cos(x2); y3 = exp(-abs(x1-pi));
>> plot(x1, y1), hold on                          % "hold on" holds the current picture
>> plot(x2, y2, 'r+:'), plot(x1, y3, '-.o')
>> plot([x1; x1], [y1; y3], '-.x'), hold off
>> title('2D-plots')                              % title of the plot
>> xlabel('x-axis')                               % label x-axis
>> ylabel('y-axis')                               % label y-axis
>> grid                                           % a dotted grid is added
>> legend('P1', 'P2', 'P3', 'P4')                 % description of plots
>> print -deps fig1                               % save a copy of the image in a file
                                                  % called fig1.eps
```



3-D Plotting: plot3

```
>> t = linspace(0,2*pi); r = 2 * ( 1 + cos(t) );
>> x = r .* cos(t); y = r .* sin(t); z = t;
>> subplot(1,2,1), plot(x, y, 'r'), xlabel('x'), ylabel('y')
>> axis square, grid on, title('cardioid')
>> subplot(1,2,2), plot3(x, y, z), xlabel('x'), ylabel('y'), hold on
>> axis square, grid on, title('in 3-D'), zlabel('z = t')
>> plot3(x, y, zeros(size(x)), 'r'), view(-40, 60)
```



Specialized Plotting Functions

- polar — to make polor plots

```
>> polar(0:0.01:2*pi,cos((0:0.01:2*pi)*2))
```

- quiver — to add velocity vectors to a plot

```
>> [X,Y] = meshgrid(1:10,1:10);
>> quiver(X,Y,rand(10),rand(10));
```

- stairs — plot piecewise constant functions

```
>> stairs(1:10,rand(1,10));
```

- fill — draws and fills a polygon with specified vertices

```
>> fill([0 1 0.5],[0 0 1],'r');
```

- stem, stem3 — plot 2D, 3D discrete sequence data
- scatter, scatter3 — 2D, 3D scatter plot

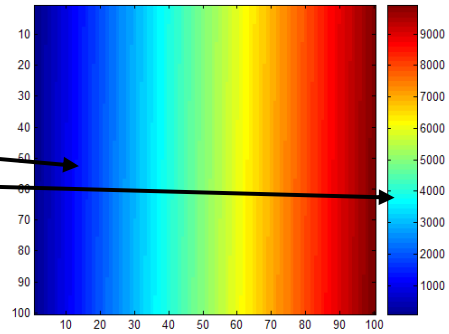
Other Useful Plotting Commands

- axis square- makes the current axis look like a box
- axis tight- fits axes to data
- axis equal- makes x and y scales the same
- axis xy- puts the origin in the bottom left corner (default for plots)
- axis ij- puts the origin in the top left corner (default for matrices/images)
- close([1 3])- closes figures 1 and 3
- close all- closes all figures (useful in scripts/functions)

Visualizing Matrices

- Any matrix can be visualized as an image

```
» mat=reshape(1:10000,100,100);
» imagesc(mat);
» colorbar
```



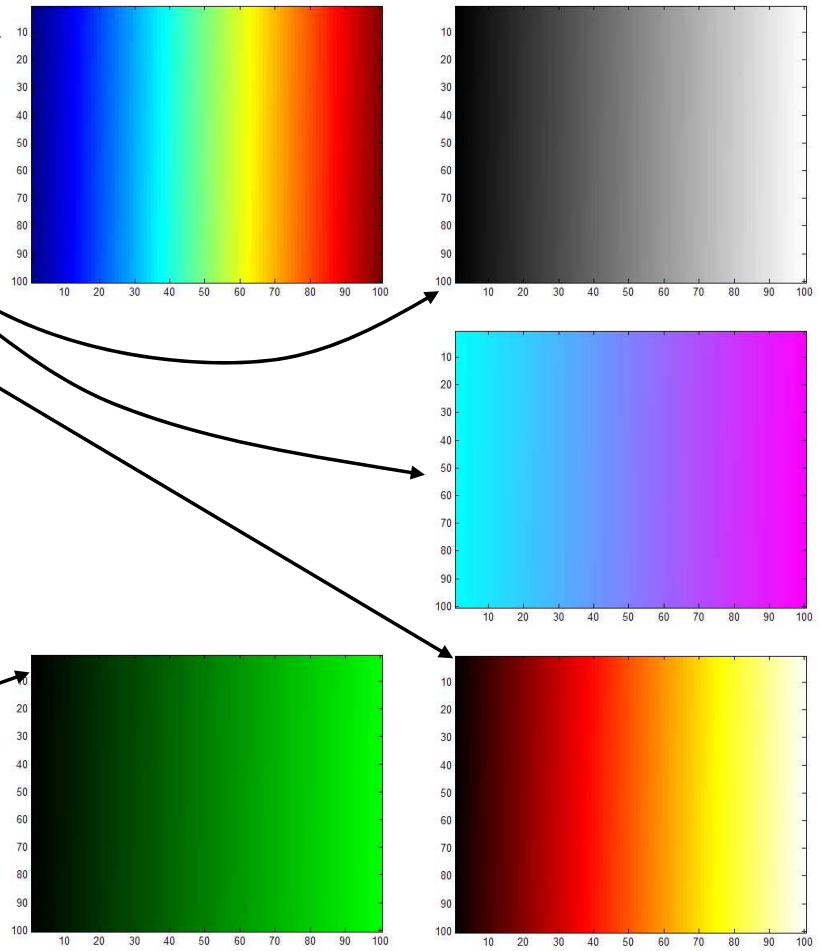
- **imagesc** automatically scales the values to span the entire colormap

- Can set limits for the color axis (analogous to **xlim**, **ylim**)

```
» caxis([3000 7000])
```

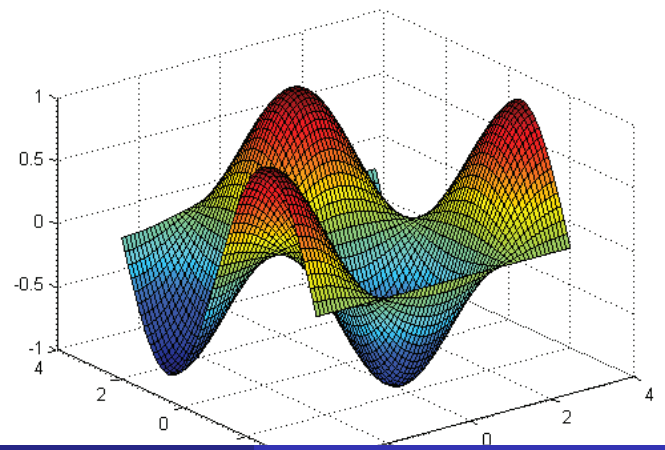
Colormaps

- You can change the colormap:
  - » `imagesc(mat)`
    default map is jet
  - » `colormap(gray)`
  - » `colormap(cool)`
  - » `colormap(hot(256))`
- See `help hot` for a list
- Can define custom colormap
  - » `map=zeros(256,3);`
  - » `map(:,2)=(0:255)/255;`
  - » `colormap(map);`



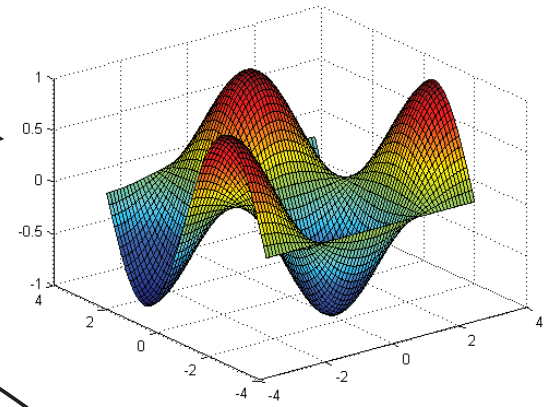
Surf

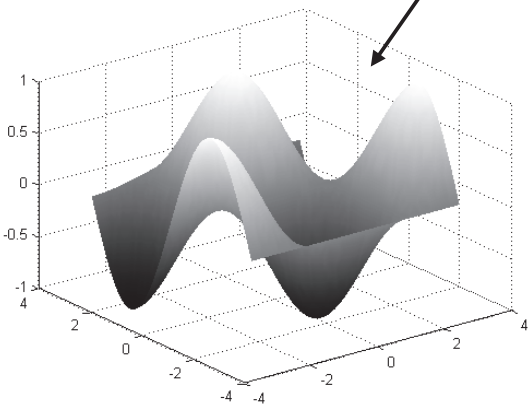
- meshgrid(x,y) produces grids containing all combinations of x and y elements, in order to create the domain for a 3D plot of a function $z = f(x, y)$
- surf puts vertices at specified points in space x, y, z, and connects all the vertices to make a surface
    - Make the x and y vectors
      » `x=-pi:0.1:pi;`
      » `y=-pi:0.1:pi;`
    - Use meshgrid to make matrices (this is the same as loop)
      » `[X,Y]=meshgrid(x,y);`
    - To get function values, evaluate the matrices
      » `Z =sin(X).*cos(Y);`
    - Plot the surface
      » `surf(X,Y,Z)`

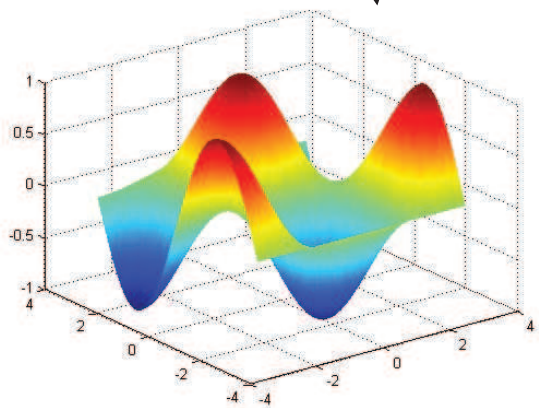


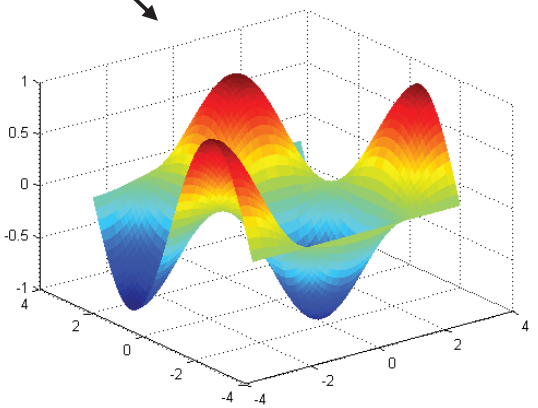
Surf Options

- See **help surf** for more options
- There are three types of surface shading
  - `» shading faceted`
  - `» shading flat`
  - `» shading interp`
- You can change colormaps
  - `» colormap(gray)`



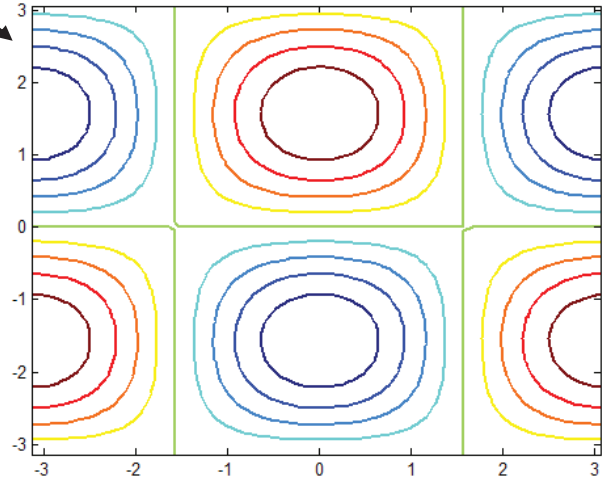


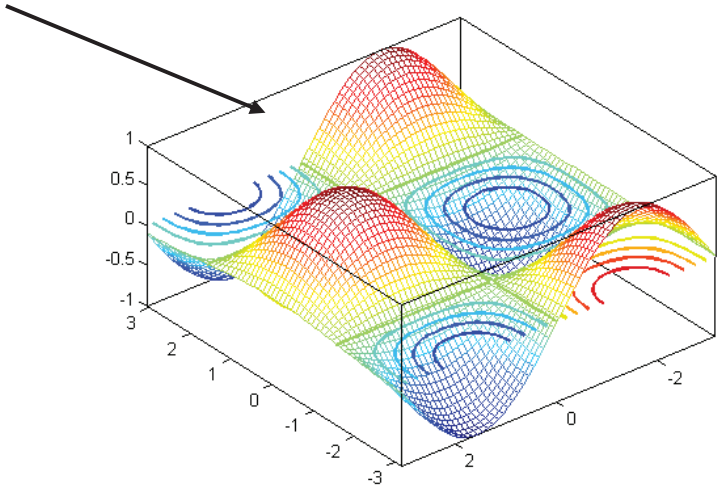




Contour

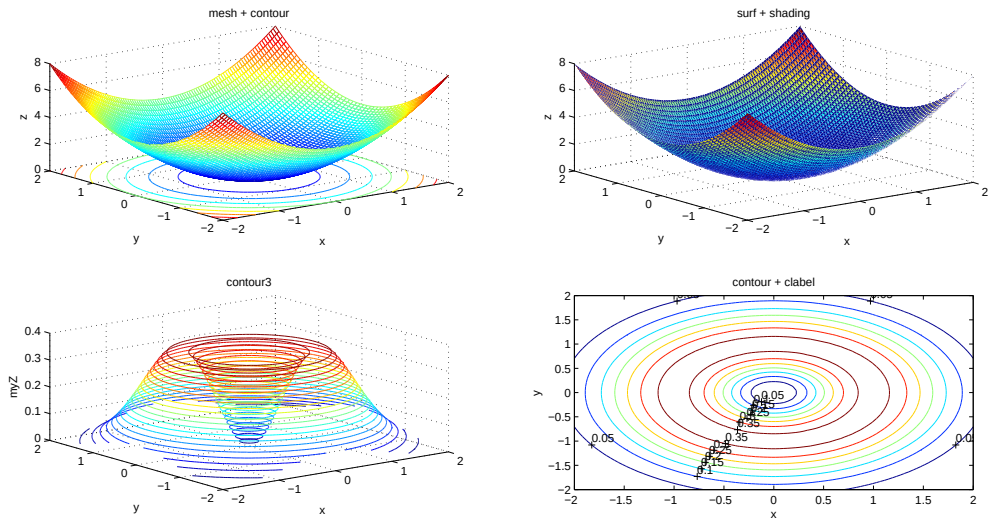
- You can make surfaces two-dimensional by using contour
  - » `contour(X,Y,Z,'LineWidth',2)`
    - ➢ takes same arguments as surf
    - ➢ color indicates height
    - ➢ can modify linestyle properties
    - ➢ can set colormap
  - » `hold on`
  - » `mesh(X,Y,Z)`





Continue...

```
>> x = linspace(-2,2); y = linspace(-2,2,50);
>> [X, Y] = meshgrid(x,y);
>> z = X.^2 + Y.^2;
>> subplot(2,2,1), mesh(x,y,z), xlabel('x'), ylabel('y')
>> zlabel('z'), hold on, contour(x,y,z), title('mesh + contour')
>> subplot(2,2,2), surf(x,y,z), xlabel('x'), ylabel('y')
>> zlabel('z'), shading interp, title('surf + shading')
>> myZ = z .* exp(-z);
>> subplot(2,2,3), contour3(x,y,myZ,20), xlabel('x'), ylabel('y')
>> zlabel('myZ'), title('contour3')
>> subplot(2,2,4), H = contour(x,y,myZ); xlabel('x'), ylabel('y')
>> title('contour + clabel'), clabel(H)
```



Vector Fields

- Visualize vector–valued function of two or three variables $\mathbf{F}(x, y) \in \mathbb{R}^2$ or $\mathbf{F}(x, y, x) \in \mathbb{R}^3$

Command	**Description**
feather	Plot velocity vectors along horizontal
quiver, quiver3	Plot 2D,3D velocity vectors from specified points
compass	Plot arrows emanating from origin
streamslice	Plot streamlines in slice planes
streamline	Plot streamlines of 2D, 3D vector data

```
[x,y] = meshgrid(0:0.1:1,0:0.1:1);
u = x;
v = -y;
figure
quiver(x,y,u,v)
startx = 0.1:0.1:1;
starty = ones(size(startx));
streamline(x,y,u,v,startx,starty)
```

Volume Visualization–Scalar Data

- Visualize scalar–valued function of two and three variables $f(x, y, z) \in \mathbb{R}$

Command	**Description**
contourslice	Draw contours in volume slice planes
flow	Simple function of three variables
isocaps	Compute isosurface end–cap geometry
isocolors	Compute isosurface and patch colors
isonormals	Compute normals of isosurface vertices
isosurface	Extract isosurface data from volume data
slice	Volumetric slice plot

Volume Visualization–Scalar Data

- contourslice

```
[X,Y,Z] = meshgrid(-2:.2:2);
V = X.*exp(-X.^2-Y.^2-Z.^2);
xslice = [-1.2,0.8,2];    yslice = [];    zslice = [];
contourslice(X,Y,Z,V,xslice,yslice,zslice)
view(3)
grid on
```

- slice

```
zslice = 0;
slice(X,Y,Z,V,xslice,yslice,zslice)
```

- isonormals

```
[x,y,z,v] = flow;
p = patch(isosurface(x,y,z,v,-3));
isonormals(x,y,z,v,p)
p.FaceColor = 'red';
p.EdgeColor = 'none';
daspect([1 1 1])
view(3);
axis tight
camlight
lighting gouraud
```

Volume Visualization–Vector Data

- Visualize vector–valued function of three variables $\mathbf{F}(x, y, z) \in \mathbb{R}^3$

Command	**Description**
interpstreamspeed	Interpolate stream–line vertices from flow speed
coneplot	Plot velocity vectors as cone
streamparticles	Plot stream particles
streamribbon	3D stream ribbon plot
streamslice	Plot streamlines in slice planes
streamtube	Create 3D stream tube plot

```
load wind
[sx,sy,sz] = meshgrid(80,20:10:50,0:5:15);
streamtube(x,y,z,u,v,w,sx,sy,sz);
view(3);
axis tight
shading interp;
camlight;
lighting gouraud
```

Saving and Loading Figures

- .fig format (MATLAB format for figures)
- openfig
  - this function is used to load previously saved MATLAB figures

```
>> openfig('figFileName')
```

- print
  - print -dformat filename (Ex: print -depsc 'figure.eps')

    eps is a cool format that stores your image in a vectorized way, which avoids quality loss after scaling. It is particularly useful when used within LaTeX.

Saving Figures

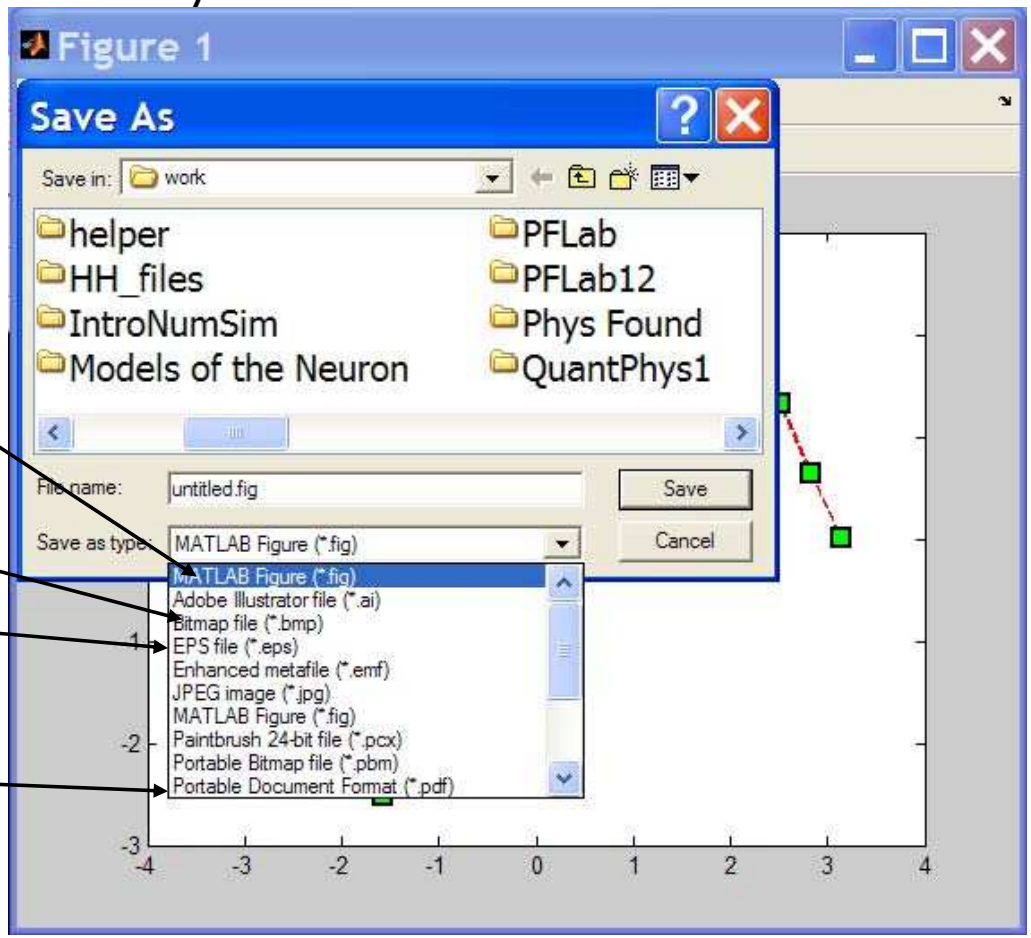
- Figures can be saved in many formats. The common ones are:

.fig preserves all information

.bmp uncompressed image

.eps high-quality scaleable format

.pdf compressed image



Symbolic Math Toolbox

- Numeric approach:
  - Always get a solution
  - Can make solutions accurate
  - Easy to code
  - Hard to extract deeper understanding
  - Numerical methods sometimes fail
  - Can take a while to compute
- Symbolic approach:
  - Analytical Solutions
  - Lets you intuit things about solution form
  - Sometimes can not be solved
  - Can be overly complicated

Symbolic Variables

- Symbolic variables are a type, like double or char
- To make symbolic variables, use sym

```
>> a=sym('1/3');
>> mat=sym([ 1 2;3 4]);
```

- Or use syms

```
>> syms a b c d
>> A = [a^2, b, c ; d*b, c-a, sqrt(b)]
A = [ a^2,     b,       c]
[ b*d, c - a, b^(1/2)]
>> b = [a;b;c];
>> A*b
ans =    a^3 + b^2 + c^2
b^(1/2)*c - b*(a - c) + a*b*d
```

Arithmetic, Relational, and Logical Operators

- Arithmetic Operations
  - ceil, floor, fix, cumprod, cumsum, real, imag, minus, mod, plus, quorem , round
- Relational Operations
  - eq, ge, gt, le, lt, ne, isequaln
- Logical Operations
  - and, not, or, xor, all, any, isequaln, isfinite, isinf, isnan, logical

See http://www.mathworks.com/help/symbolic/operators.html for more details

Symbolic Expressions

- expand multiplies out
- factor factors the expression
- inv computes inverse
- det computes determinant

```
>> syms a b
>> expand((a-b)^2)
ans = a^2 - 2*a*b + b^2
>> factor(ans)
ans = (a - b)^2
>> d=[a, b; 0.5*b a];
>> inv(d)
ans =
[ (2*a)/(2*a^2 - b^2), -(2*b)/(2*a^2 - b^2)]
[    -b/(2*a^2 - b^2),  (2*a)/(2*a^2 - b^2)]
>> det(d)
ans = a^2 - b^2/2
```

- pretty makes it look nicer
- collect collect terms
- simplify simplifies expressions
- subs replaces variables with number or expressions
- solve replaces variables with number or expressions

```
>> g = 3*a +4*b-1/3*a^2-a+3/2*b;
>> collect(g)
ans =
(11*b)/2 + 2*a - a^2/3
>> subs(g,[a,b],[0,1])
ans = 5.5000
```

Symbolic Integration/Derivation

- Differentiation: diff(function,variable,degree)
- Integration: int(function,variable,degree,option)

```
>> syms x y t
>> f=exp(t)*(x^2-x*y +y^3);
>> fx=diff(f,x)
fx = exp(t)*(2*x - y)
>> fy=diff(f,y,2)
fy = 6*y*exp(t)
>> int(f,y)
ans = (y*exp(t)*(4*x^2 - 2*x*y + y^3))/4
>> int(f,y,0,1)
ans = (exp(t)*(4*x^2 - 2*x + 1))/4
```

Symbolic Summations/Limits

- Summation: symsum
- Limit: limit
- Taylor seris: taylor

```
>> syms x k
>> s1 = symsum(1/k^2,1,inf)
s1 = pi^2/6
>> s2 = symsum(x^k,k,0,inf)
s2 = piecewise([1 <= x, Inf], [abs(x) < 1, -1/(x - 1)])
>> limit(x / x^2, inf)
ans = 0
>> limit(sin(x) / x)
ans = 1
>> f = taylor(log(1+x))
f = x^5/5 - x^4/4 + x^3/3 - x^2/2 + x
```

Calculus Commands

Command	**Description**
diff	Differentiate symbolic
int	Definite and indefinite integrals
rsums	Riemann sums
curl	Curl of vector field
divergence	Divergence of vector field
gradient	Gradient vector of scalar function
hessian	Hessian matrix of scalar function
jacobian	Jacobian matrix
laplacian	Laplacian of scalar function
potential	Potential of vector field
taylor	Taylor series expansion
limit	Compute limit of symbolic expression
fourier	Fourier transform
ifourier	Inverse Fourier transform
ilaplace	Inverse Laplace transform

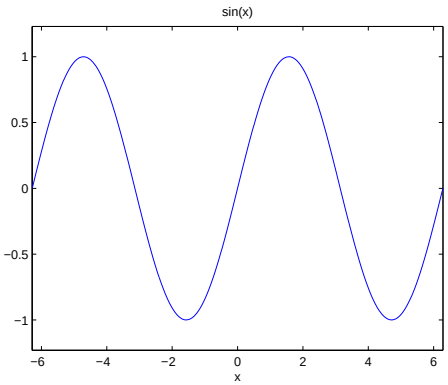
Linear Algebra Commands

Command	**Description**
adjoint	Adjoint of symbolic square matrix
expm	Matrix exponential
sqrtm	Matrix square root
cond	Condition number of symbolic matrix
det	Compute determinant of symbolic matrix
norm	Norm of matrix or vector
colspace	Column space of matrix
null	Form basis for null space of matrix
rank	Compute rank of symbolic matrix
rref	Compute reduced row echelon form
eig	Symbolic eigenvalue decomposition
jordon	Jordan form of symbolic matrix
lu	Symbolic LU decomposition
qr	Symbolic QR decomposition
svd	Symbolic singular value decomposition

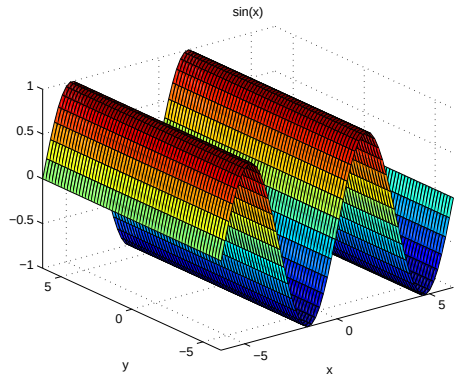
Symbolic Plotting

- Plot a symbolic function over one variable by using the figures/ezplot function

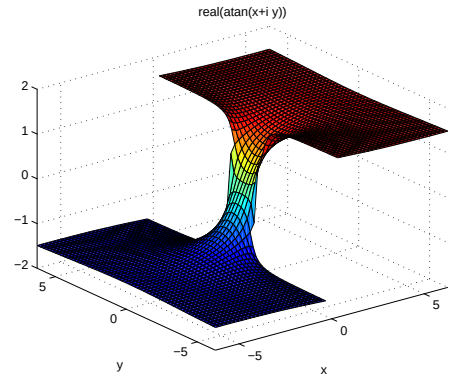
```
>> syms x
>> y = sin(x);
>> ezplot(y);
>> f = sin(x);
>> ezsurf(f);
>> ezsurf( 'real(atan(x+i*y))' );
```



(c) ezplot(y)



(d) ezsurf(f)



(e) ezsurf('real(atan(x+i*y))')

End of Lecture

1. 2D–3D Plotting
2. Visualizing Matrices
3. Vector Fields, Vector Visualization
4. Save, Load, Print
5. Symbolic Toolbox

For More Information

- http://iam.metu.edu.tr/scientific-computing
- https://iam.metu.edu.tr/scientific-computing-lecture-series
- https://www.facebook.com/SCiamMETU/

For More Information

- http://iam.metu.edu.tr/scientific-computing
- https://iam.metu.edu.tr/scientific-computing-lecture-series
- https://www.facebook.com/SCiamMETU/

...thank you for your attention !