Image Processing and Data Visualization with MATLAB

MATLAB Graphics

(based on MATLAB Help)

Hansrudi Noser

June 28-29, 2010

UZH, Multimedia and Robotics
Summer School

Contents

Overview of Plotting

- Wide variety of techniques to display data graphically
- Graphs can be
  - Created
  - Annotated
  - Printed
  - Exported to standard graphics format

The Plotting Process

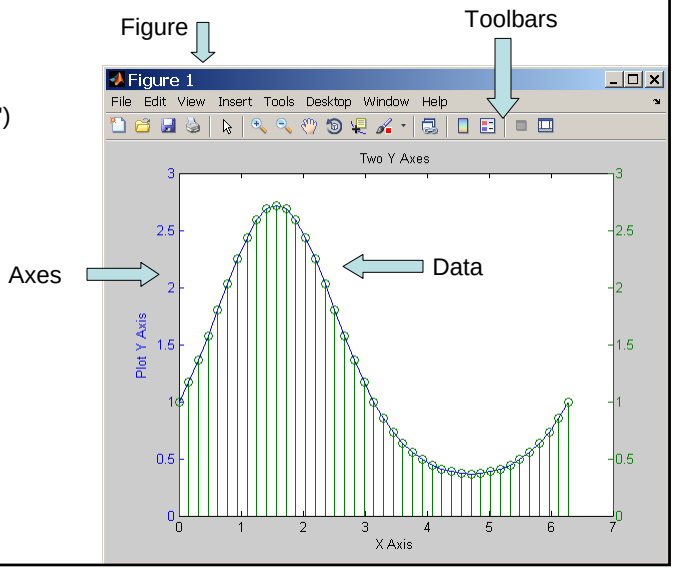
- Creating a graph
  - By interactive tools
  - By command interface
  - By plotting programs
- Exploring data
- Editing graph components
- Annotating graphs
- Printing and exporting graphs
- Adding and removing figure content
- Saving graphs for reuse

Graph Components

- MATLAB graphs are displayed in a special window, called a figure, containing menus and toolbars
- Within a figure you have axes, the coordinate system of the graph
- The data are visualized within the coordinate system, defined by the axes, with graphics objects like lines and surfaces
- The actual data is stored as properties of the graphics objects

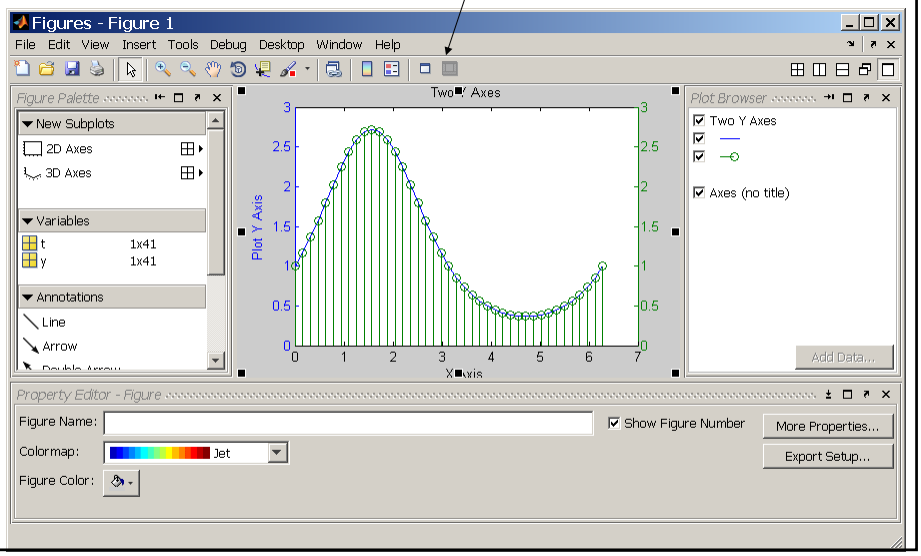
Example: Creating a graph with commands

```
>> t = 0:pi/20:2*pi;
y = exp(sin(t));
plotyy(t,y,t,y,'plot','stem')
xlabel('X Axis')
ylabel('Plot Y Axis')
title('Two Y Axes')
```



Plotting Tools

- You can enable the plotting tools for any graph, even one created using MATLAB commands
- See MATLAB help



Types of MATLAB Plots

- There exist many 2D and 3D types of plots supported by MATLAB
- Most 2D plots have 3D analogs
- In MATLAB, plot types beginning with ez are functions that plot functions passed as arguments (of ez…)

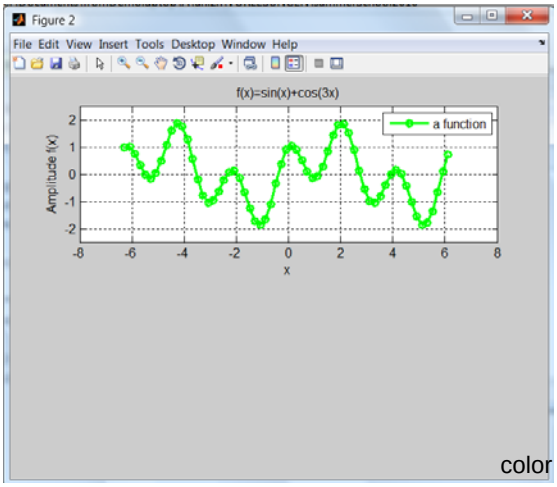
- 2D
  - Line graphs
  - Bar graphs
  - Area graphs
  - Direction graphs
  - Radial graphs
  - Scatter graphs
- 3D
  - Line graphs
  - Mesh and bar graphs
  - Area graphs and constructive objects
  - Surface graphs
  - Direction graphs
  - Volumetric graphs

Programmatic Plotting

- Prepare data
- Select a window and position a plot region within the window
- Plot
- Set line and marker characteristics
- Set axis limits, tick marks, and grid lines
- Annotate the graph with axis labels, legend, and text
- Export graph

```
x=-2*pi:0.2:2*pi;
y = sin(x)+cos(3*x);

figure, subplot(2,1,1);

h=plot(x,y);

set(h,'LineWidth',2);
set(h,'Marker','o');
set(h,'Color','g');

axis([-8 8 -2.5 2.5])
grid on;

xlabel('x');
ylabel('Amplitude f(x)');
legend(h,'a function');
title('f(x)=sin(x)+cos(3x)');
```

Programmatic Plotting



print resolution of 200 dpi

tiff preview

File name

color eps format

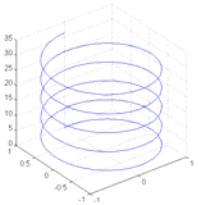
Example of export →

```
print -depsc -tiff -r200 myplot
```

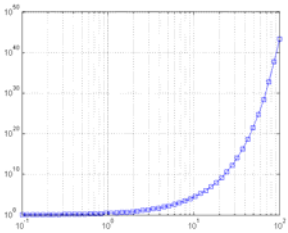
Contents

Line Plots

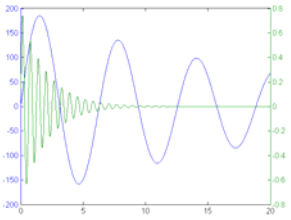
- plot
- plot3
- loglog
- semilogx
- semilogy
- plotyy



```
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
grid on
axis square
```



```
x = logspace(-1,2);
loglog(x,exp(x),'-s')
grid on
```



```
x = 0:0.01:20;
y1 = 200*exp(-0.05*x).*sin(x);
y2 = 0.8*exp(-0.5*x).*sin(10*x);
plotyy(x,y1,x,y2,'plot');
```

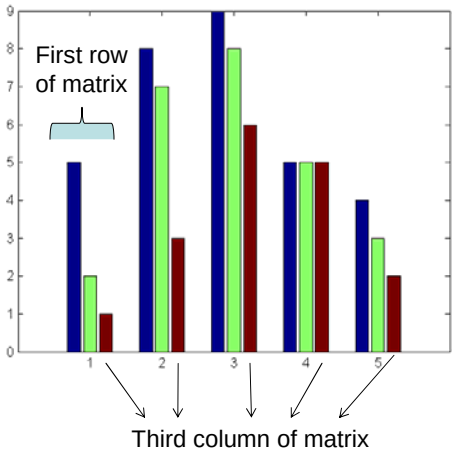
Contents

Bar Graphs

- Display vector or matrix data
- Useful for
  - Viewing results over a period of time
  - Comparing results from different data sets
  - Showing how individual elements contribute to an aggregate amount
  - Displaying discrete data

Grouped 2D Bar Graph

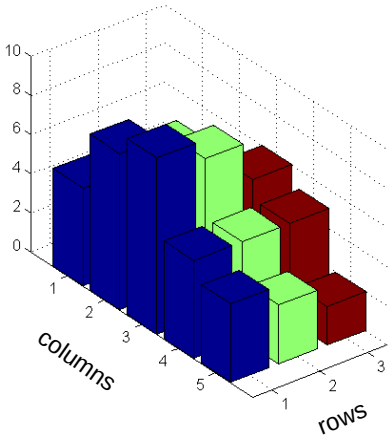
```
Y = [5 2 1
     8 7 3
     9 8 6
     5 5 5
     4 3 2];

bar(Y)
```



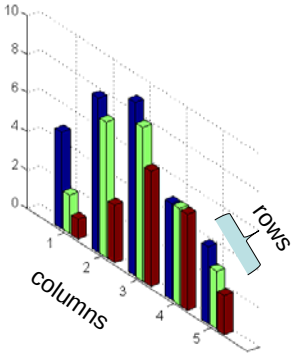
Each matrix element corresponds to a bar

Detached and Grouped 3D Bar Graphs

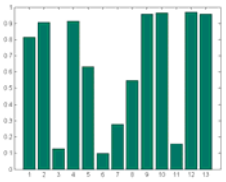
```
Y = [5 2 1
     8 7 3
     9 8 6
     5 5 5
     4 3 2];

bar3(Y)
```



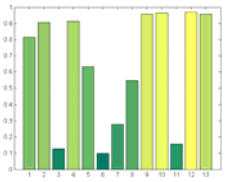

```
bar3(Y,'grouped')
```



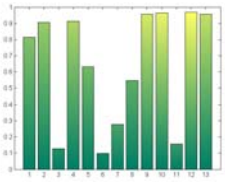
Coloring Bars According to Height



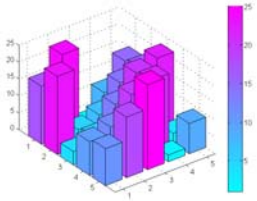
Same color

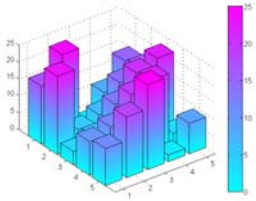


Color for each bar according to height



Interpolated shading according to height

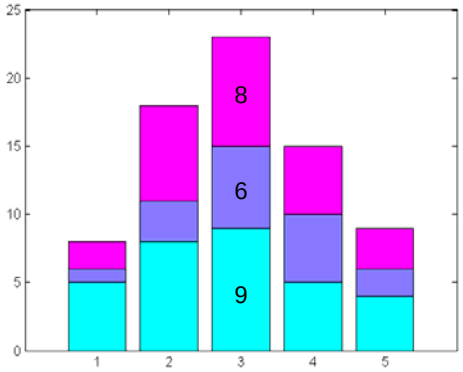




Stacked Bar Graphs

- Show contributing amounts

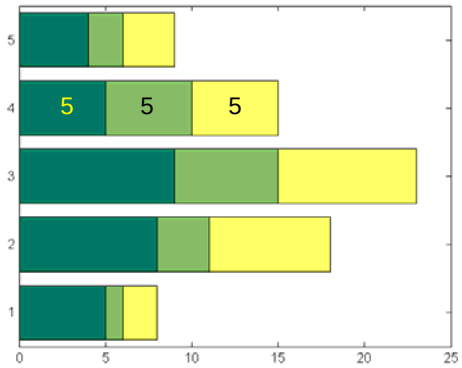
```
Y = [5 1 2
     8 3 7
     9 6 8
     5 5 5
     4 2 3];

bar(Y,'stack')
colormap cool
```



Rows contain contributing amounts of sum

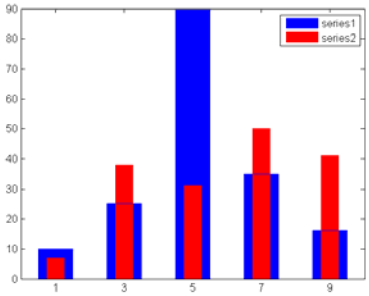
Horizontal Bar Graphs

```
Y = [5 1 2
     8 3 7
     9 6 8
     5 5 5
     4 2 3];

barh(Y,'stack')
colormap summer
```

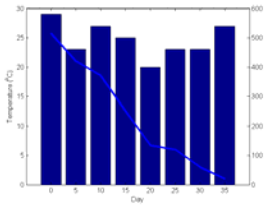


Overlaying Bar Graphs

```
x=[1 3 5 7 9];
y1=[10 25 90 35 16];
K=0.5;

 bar1=bar(x, y1, 'FaceColor', 'b', 'EdgeColor', 'b');
set(bar1,'BarWidth',K);
hold on;

y2=[7 38 31 50 41];
bar2=bar(x, y2, 'FaceColor', 'r', 'EdgeColor', 'r');
set(bar2,'BarWidth',K/2);
hold off;

legend('series1','series2')
```



Overlaying a line ⇨

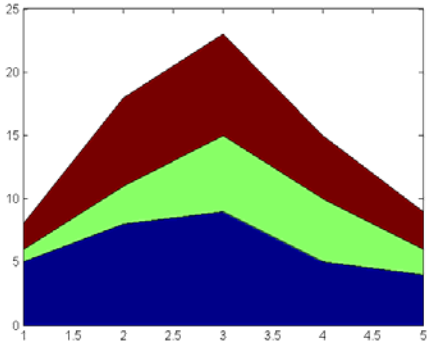


Area Graphs Showing Contributing Amounts

Area plots the values in each column of a matrix as a separate curve and fills the area between the curve and the *x*-axis

Area graphs are useful for showing how elements in a vector or matrix contribute to the sum of all elements at a particular *x* location

```
Y = [5 1 2
     8 3 7
     9 6 8
     5 5 5
     4 2 3];

area(Y);
```



Comparing Data Sets with Area Graphs

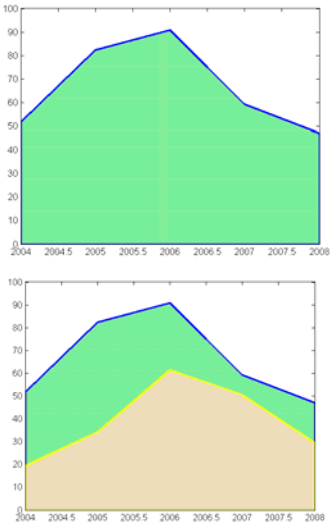
- Create a vector containing the income from sales
- Create a vector containing the years in which the sales took place
- Create a vector of profits for the same five-year period

```
sales = [51.6 82.4 90.8 59.1 47.0];

x = 2004:2008;

profits = [19.3 34.2 61.4 50.5 29.4];
```

Comparing Data Sets with Area Graphs

- Use area to display profits and sales as two separate area graphs within the same axes

```
area(x,sales,'FaceColor',[.5 .9 .6], …
'EdgeColor','b', 'LineWidth',2)


hold on
area(x,profits,'FaceColor',[.9 .85 .7], …
'EdgeColor','y', 'LineWidth',2)
```



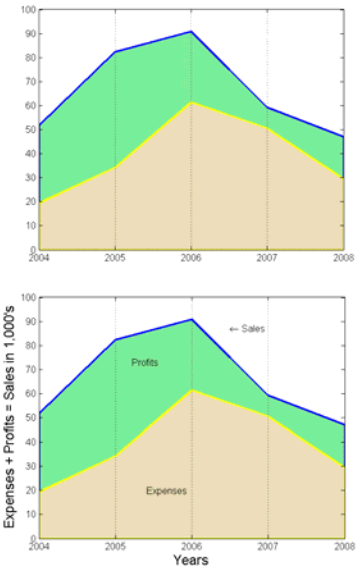
Comparing Data Sets with Area Graphs

- Improve the graph

```
set(gca,'XTick',x)
set(gca,'XGrid','on')
set(gca,'Layer','top')
```

- Annotate interactively

```
gtext('\leftarrow Sales')
gtext('Profits')
gtext('Expenses')

xlabel('Years','FontSize',14)
ylabel('Expenses + Profits = Sales in 1,000''s',…
'FontSize',14)
```



Contents

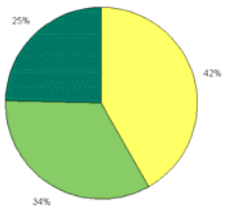
Pie Charts

- Pie charts are a useful way to communicate the percentage that each element in a vector or matrix contributes to the sum of all elements
- Example:
  - visualize the contribution that three products make to total sales
  - Given a matrix X where each column of X contains yearly sales figures for a specific product over a five-year period

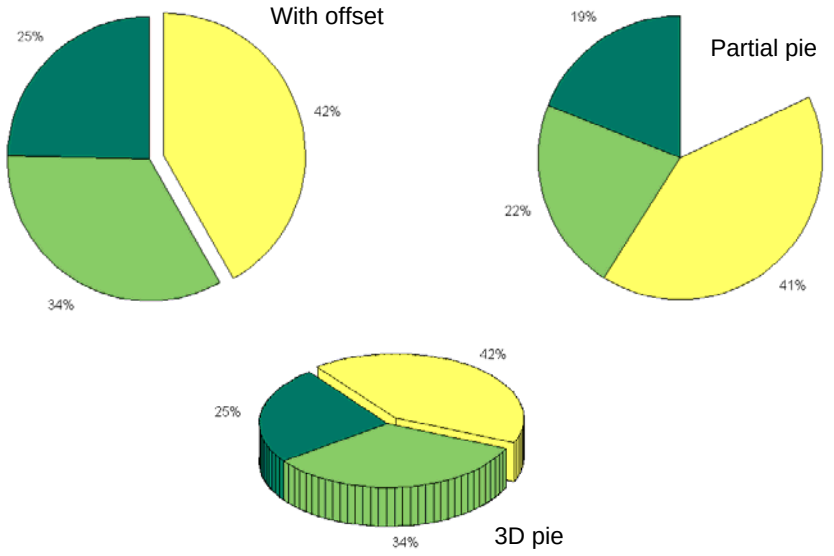
```
X = [19.3 22.1 51.6;
     34.2 70.3 82.4;
     61.4 82.9 90.8;
     50.5 54.9 59.1;
     29.4 36.3 47.0];

x = sum(X)
```

```
x =
  194.8000  266.5000  330.9000
```

```
pie(x)
colormap summer
```



Pie Chart Variants



Contents

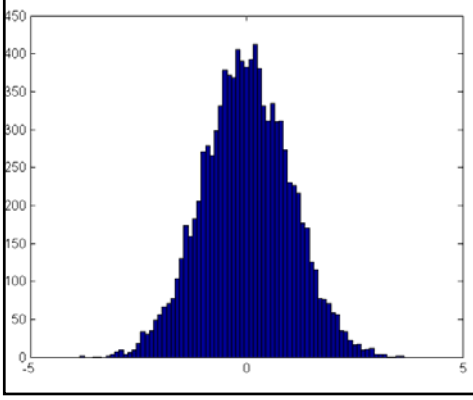
Histograms

- Show the distribution of data values across a data range.
  - The data range is divided into a certain number of intervals ("binning" the data)
  - the number of values that fall into each interval (or "bin") are tabulated
  - the values in the bins using bars or wedges of varying height are plotted
- Functions for creating histograms are
  - hist: Data in Cartesian coordinate system
  - rose: Data in polar coordinate

Cartesian Histograms

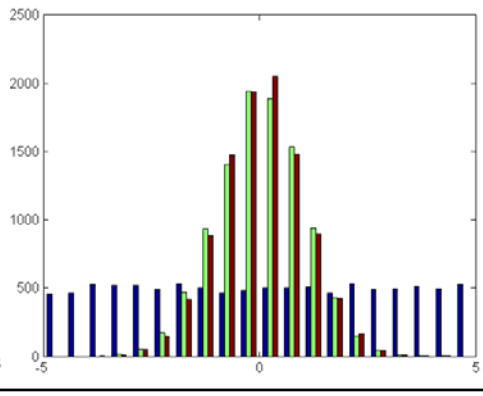
```
x = -4:0.1:4;
y = randn(10000,1);
hist(y,x)
```

Number and centers of bins
Specified by x




```
Y=randn(10000,3);
YY = rand(10000,1)*10-5;
Y(:,1) = YY;
hist(Y,20)
```

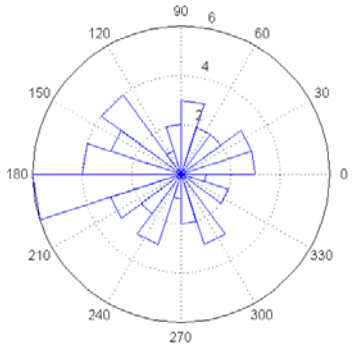
20 bins



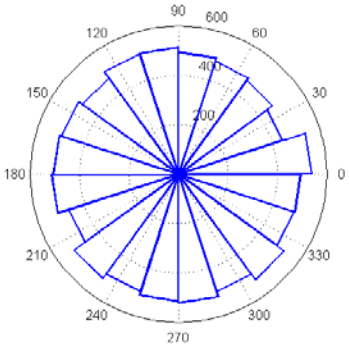
Polar Histograms

Data values given in radians

```
theta = 2*pi*rand(1,50);
rose(theta)
```




```
theta2 = 2*pi*rand(1, 10000);
figure, rose(theta2)
hline = findobj(gca,'Type','line');
set(hline,'LineWidth',1.5)
```



Contents

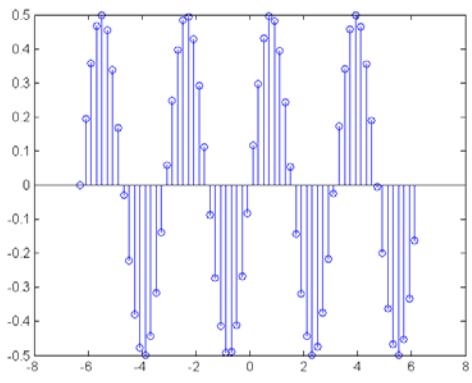
- Overview
- Line Plots
- Bar Graphs and Area Graphs
- Pie Charts
- Histograms
- Discrete Data Graphs
- Direction and Velocity Vector Graphs
- Contour Plots

Discrete Data Graphs

- Used for displaying discrete data such as
  - Number of accidents per month
  - Digital sampled values
  - …
- Stem and stair graphs:
  - stem: discrete sequence of *y*-data as stems from *x*-axis
  - stem3: discrete sequence of *z*-data as stems from *xy*-plane
  - stairs: discrete sequence of *y*-data as steps from *x*-axis

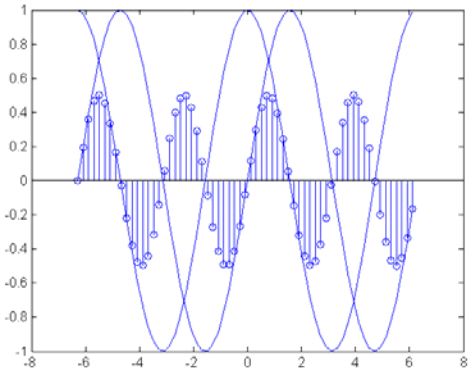
Stem plot

```
t = -2*pi: 0.2: 2*pi;
y = sin(t) .* cos(t);
stem(t,y)
```



Combined with line plot

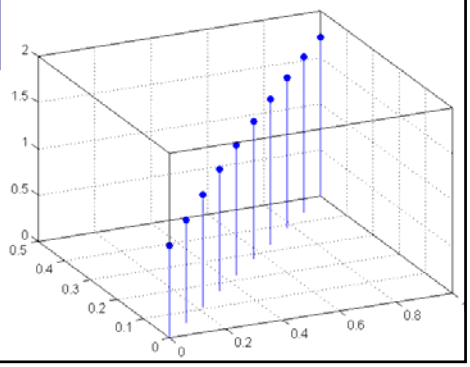
```
hold on
plot(t,sin(t))
plot(t,cos(t))
```



3D Stem Plot

Visualize discrete values of a function of 2 variables

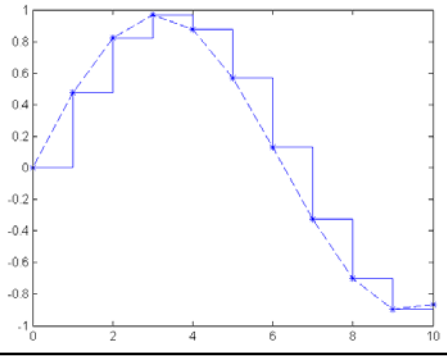
```
X = linspace(0,1,10);
% 10 equidistant values between 0 and 1
Y = X./2;
Z = sin(X) + cos(Y);

stem3(X,Y,Z,'fill')
view(-25,30)
% specify azimuth and elevation of view
```



Stair Step Plot

- Plot holds the data at a constant *y* value for all values between *x(i)* and *x(i+1)*, where *i* is the index into the *x* data
- Plot is useful for drawing time-history plots of digitally sampled data systems

```
>> alpha = 0.01;
beta = 0.5;
t = 0:10;
f = exp(-alpha*t).*sin(beta*t);

stairs(t,f)
hold on
plot(t,f,'--*')
hold off
```



Contents

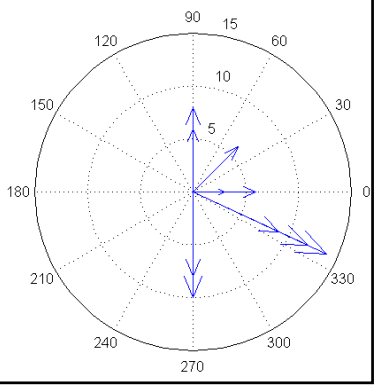
Direction and Velocity Vector Graphs

- Functions for displaying vectors
  - compass: vectors emanating from the origin of a polar plot
  - feather: vectors extending from equally spaced points along a horizontal line
  - quiver: 2-D vectors specified by (*u*,*v*) components
  - quiver3: 3-D vectors specified by (*u*,*v*,*w*) components

Compass Plots

```
wdir = [45 90 90 45 360 335 360 270 335 270 335 335];
knots = [6 6 8 6 3 9 6 8 9 10 14 12];

rdir = wdir * pi/180;
[x,y] = pol2cart(rdir,knots);
compass(x,y)
```

- Shows vectors emanating from the origin of a graph.
- The function takes Cartesian coordinates and plots them on a circular grid
- Example: Wind directions and strength

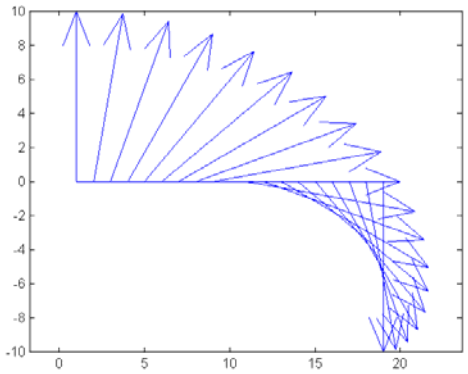


Feather Plots

- Displays vectors emanating from equally spaced points along a horizontal axis

Example:
Display vectors of length 10 and of angles from 90 to -90 degrees

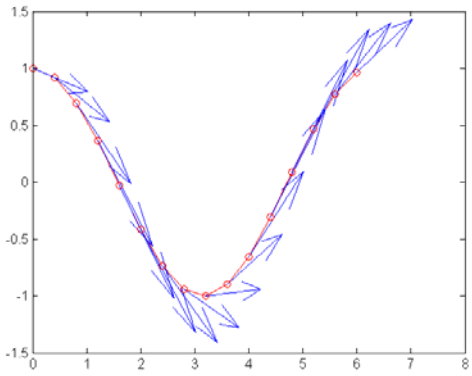
```
theta = 90:-10:-90;
r = ones(size(theta));
[u,v] = pol2cart(theta*pi/180,r*10);
feather(u,v)
axis equal
```



Quiver Plots

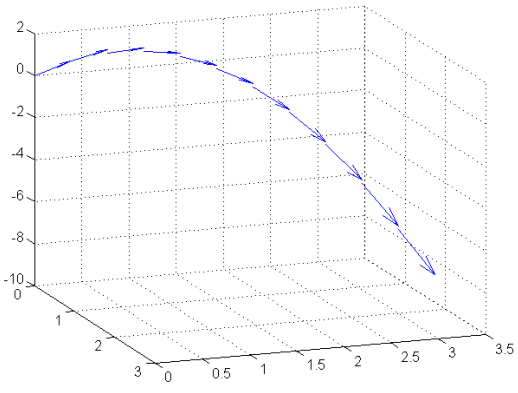
- A quiver plot displays velocity vectors as arrows with components (u,v) at the points (x,y)

```
x=0:0.4:2*pi;
y=cos(x);

u=gradient(x);
v=gradient(y);

quiver(x,y,u,v);

hold on;
plot(x,y,'or')
```



3D Quiver Plot

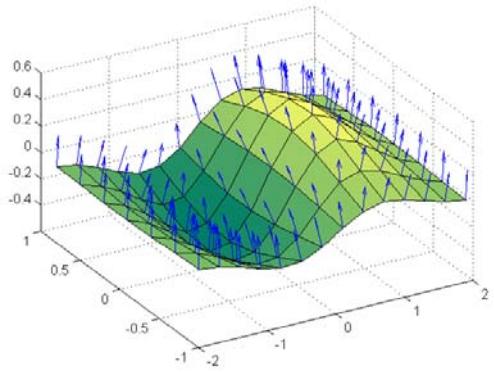
- Display path and velocity of a projectile




```
% initial velocity of projectile
vx = 2;     % velocity in x
vy = 3;     % velocity in y
vz = 10;    % velocity in z
a = -32;    % gravity acceleration

% time
t = 0:0.1:1 % time

% position of projectile
x = vx * t;
y = vy * t;
z = vz * t + 0.5*a*t.^2;

% velocity of projectile
u=gradient(x);
v=gradient(y);
w=gradient(z);

quiver3(x,y,z,u,v,w,0)
view([70 18])
```

Display of Surface Normals

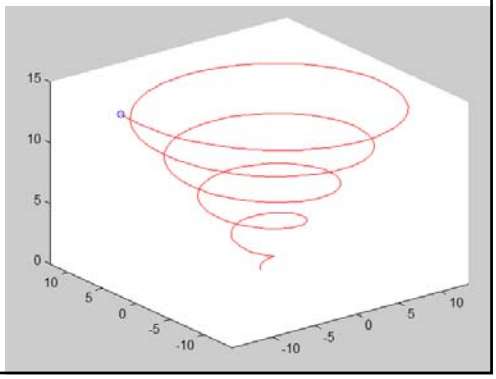
```
% 3D surface function
[X,Y] = meshgrid(-2:0.3:2,-1:0.3:1);
Z = X.* exp(-X.^2 - Y.^2);

% computation of normals
[U,V,W] = surfnorm(X,Y,Z);

% display of normals
quiver3(X,Y,Z,U,V,W,0.6);
hold on
surf(X,Y,Z);
colormap summer
view(-30,40)
axis ([-2 2 -1 1 -.6 .6])
hold off
```



Comet Plots

- A comet plot is an animated graph (2D or 3D) in which a circle (the comet *head*) traces the data points on the screen
- The comet *body* is a trailing segment that follows the head. The *tail* is a solid line that traces the entire function

```
% 3D parametric curve
t = 0:0.1:30;
x= t.*sin(t)/2;
y= t.*cos(t)/2;
z= t/2;

% comet plot
comet3(x,y,z);
```



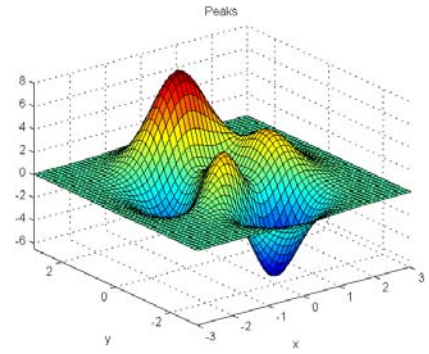
Contents

Contour Plots

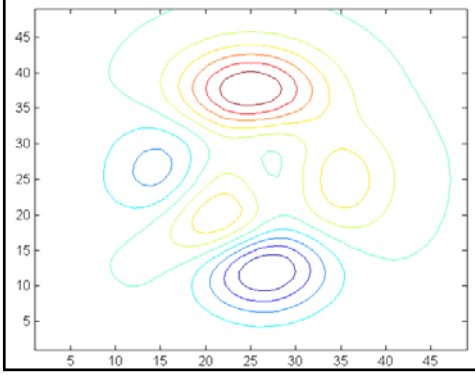
- Contour plots compute, plot, and label isolines (contour lines) for one or more matrices
  - contour: 2-D isolines generated from values given by a matrix Z
  - contour3: 3-D isolines generated from values given by a matrix Z.
  - Contourf:2-D contour plot and fills the area between the isolines with a solid color.
  - clabel: labels the isolines

Example: Test function peaks

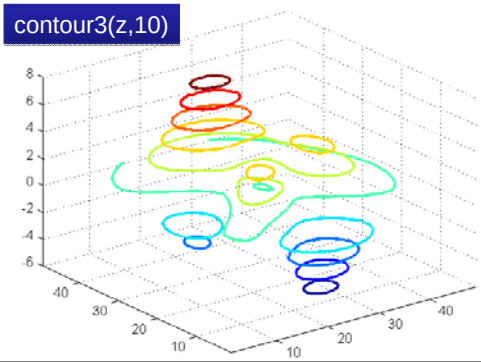
peaks
z = 3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) ...
- 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ...
- 1/3*exp(-(x+1).^2 - y.^2)



z=peaks(49);
contour(z,10); % 10 contour levels

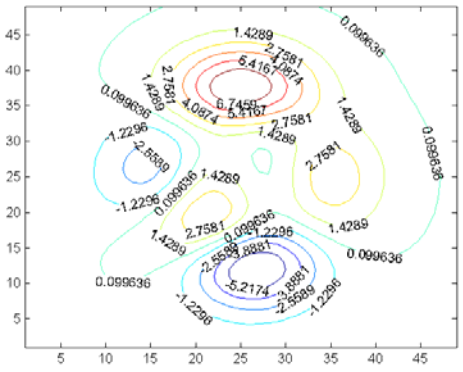


contour3(z,10)

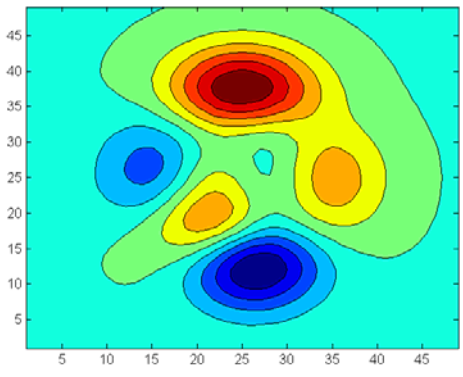


Labeled and Filled Contours

Z = peaks;
[C,h] = contour(Z,10);
clabel(C,h)



contourf(Z,10);



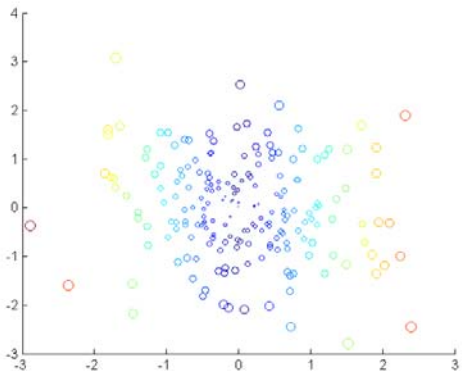
Contents

Scatter Plots

- scatter(X,Y,S,C)
  - Displays colored (C) markers with area S at (X, Y)
- scatter(X,Y,S) draws the markers at the specified sizes (S) with a single color.
  - This type of graph is known as a bubble plot
- scatter3(X,Y,Z,S,C)
  - Displays colored (C) markers with area S at (X, Y,Z)
- plotmatrix(X,Y) scatter plots the columns of X against the columns of Y
- plotmatrix(X) is the same as plotmatrix(X,X), except that the diagonal is replaced by hist(X(:,i))

Scatter / Bubble Plot

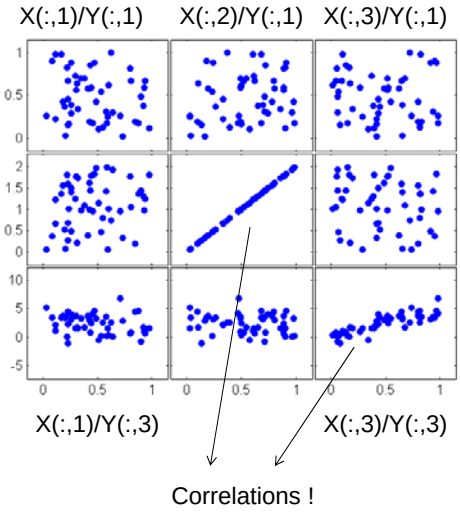
```
% normally distributed values X and Y
X=randn(200,1);
Y=randn(200,1);

% marker size depends on distance
% from (0,0)
S=20*sqrt(X.^2 +Y.^2);

% color depends on X value
C=abs(X*100);

% scatter plot (or bubble plot)
scatter(X,Y,S,C);
colormap jet;
```



Plotmatrix

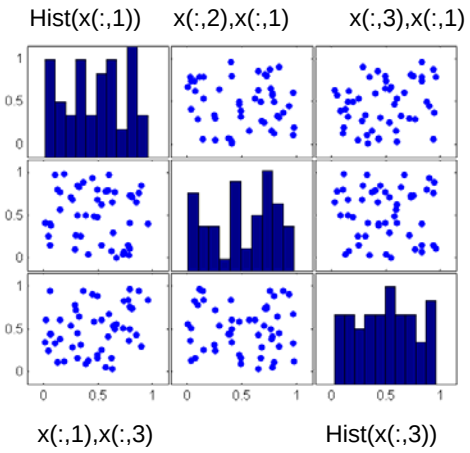
Random data in X
but with some correlations in Y

```
X=rand(50,3);
Y(:,1) = rand(50,1);
Y(:,2) =2*X(:,2);
Y(:,3) = 5*X(:,3)+randn(50,1);
plotmatrix(X,Y);
```



Plotmatrix

```
% random uniform data matrix
 x=rand(50,3);

% scatter plots with histograms
plotmatrix(x)
```

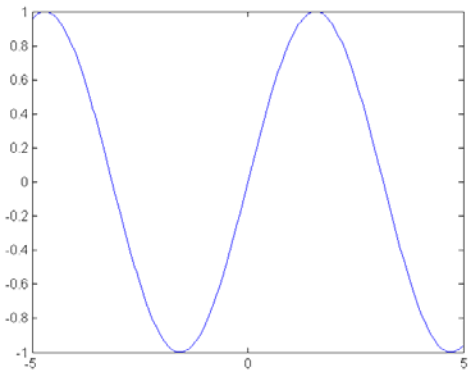


Contents

- …
- Pie Charts
- Histograms
- Discrete Data Graphs
- Direction and Velocity Vector Graphs
- Contour Plots
- Scatter Plots
- Function Plots

Function Plots (ez…)

- Plot functions with functions as arguments
  - fplot
  - ezcontour
  - ezmesh
  - …




```
fplot(@sin,[-5 5]);
```