MatLab Graphics

2 February 2004

Jacob D. Furst

2D Graphing

- Collect your data
- Prepare a place to put your graph
- Plot your graph
- If desired
  - Modify lines
  - Modify axes
  - Annotate
  - Print

Collect Your Data

- MatLab can plot many different forms of data – easiest to plot two arrays, one with x and one with f(x)

```
x = -pi:pi/100:pi;
fx = sin(x);
```

- Will also plot a single array as f(x) with the indices of the array as x values

Preparing a Place

- Create a figure

```
figure(1);
```

- Create a place within the figure for your graph – only do this if you plan to have multiple graphs

```
subplot (2, 2, 1);
```

- The first two arguments indicate the number of rows and columns for subplots, the third the position of the graph

Plot Your Graph

- Basic 2D plots

```
plot (x, fx);
```

- Plot multiple lines

```
cosx = cos(x);
plot (x, fx, x, cosx);
```

- MatLab will plot a matrix as column arrays
- X axis will be labeled 1 to number of columns

Plotting Variants

- loglog plots with both x and y axes logarithmic
- semilogx and semilogy make the x and y axis (respectively) log and the other linear
- plotyy allows you to have two y axes – one on the left and one on the right

What Else?

- MatLab allows you considerable control over the appearance of the graph
- Much of it can be specified interactively, by clicking in the figure window
- Much of it can also be controlled functionally, with MatLab commands

Line Style

Specifier	**Line Style**
-	solid line (default)
--	dashed line
:	dotted line
-.	dash-dot line

Example

```
x = -pi:pi/10:pi;
sine = sin(x);
cosine = cos(x);
plot (x, sine, ':', x, cosine,
  '--');
```

Marker Types

Specifier	Marker Type
+	plus sign
o	circle
*	asterisk
.	point
x	cross
s	square
d	diamond
^	upward pointing triangle
v	downward pointing triangle
>	right pointing triangle
<	left pointing triangle
p	five-pointed star (pentagram)
h	six-pointed star (hexagram)

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

Example

```
x = -pi:pi/10:pi;
sine = sin(x);
cosine = cos(x);
plot (x, sine, ':sr', x, cosine,
  '--ok');
```

Printing/Saving

- Use the `print` function to send your graph to the printer or to save as a file
- You can save to a file by specifying the file type and name

`print -d`*`fileformat filename`*

- Some common file format include jpeg, tiff, bitmap and eps

3D Graphing

- Occasionally, you want to plot a curve in 3D
- Uses the `plot3` function and three arrays (x, y, and z coordinates) to generate a curve in space
- The standard helix:

```
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t,'--or');
```

Plotting Surfaces

- Often, we'll want to plot surfaces, rather than curves, in 3D
- MatLab offers both the `surf` and `mesh` functions
- Use `mesh` to create a wireframe surface
- Use `surf` to create a shaded 3D surface

Example

- Create the X and Y coordinate matrices

```
[X, Y] = meshgrid(-pi:pi/10:pi);
```

- Calculate the function values

```
Z = sin (X) .* sin (Y);
```

- Plot the graph

```
mesh (Z);
surf (Z);
```

Caveat

- If your function is fairly smooth, plot fewer points and let MatLab interpolate the rest
- For example, change our last plot only a little, and look at the results:

```
[X, Y] =meshgrid(-pi:pi/100:pi);
Z = sin (X) .* sin (Y);
mesh (Z);
surf (Z);
```

Parametric Surfaces

- X, Y and Z are all matrices, in which the points of the surface are located at X(i,j), Y(i,j) and Z(i,j)

```
theta = -pi:pi/10:pi;
phi = (-pi/2:pi/20:pi/2)';
X = cos(phi)*cos(theta);
Y = cos(phi)*sin(theta);
Z = sin(phi)*ones(size(theta));
surf (X, Y, Z);
```