MATLAB Plotting

MATH2070: Numerical Methods in Scientific Computing I

Location: http://people.sc.fsu.edu/~jburkardt/classes/math2070_2019/matlab_plotting/matlab_plotting.pdf

How can we illustrate our data and mathematical results?

MATLAB Plots

Plots represent our data, help us to see patterns and problems, and convince our viewers of our results.

Plotting allows us to visualize data. Before we do any analysis, this allows us to explore the data; after an analysis, we use various kinds of plots to analyze and publish our results. MATLAB has a rich set of plotting commands that we will explore, including

- **line plot**, a sequence of points (x_i, y_i) connected by lines;
- **line plots**, several graphs on a single plot;
- **scatter plot**, a set of points (x_i, y_i) indicated by markers;
- **bar plot**, a set of data y_i displayed as bar heights;
- **histogram plot**, data to be grouped into bins before bar plotting;
- **contour plot**, contour lines of a function $f(x, y)$;
- **color contour plot**, contour colors of a function $f(x, y)$;
- **vector plot**, a 2D display of vectors (u,v)(x,y), representing a gradient field, an ODE direction field, or a flow of some kind.
- **surface plot**, a 3D plot of the surface $z = f(x, y)$;
- **polygon_fill**, polygons filled with color.
- **tree plot**, representing a branching process;
- **graph plot**, a set of labeled points p_i, and a list of two-way connections $(p_i \leftrightarrow p_j)$.
- **digraph plot**, a set of labeled points p_i, and a list of one-way connections $(p_i \rightarrow p_j)$.

1 Exercise: Line plot of the BULGARIA data

The text data file *bulgaria_data.txt* records a sequence (y_i, p_i), of years and population. A line plot connects the data points, suggesting a smooth functional behavior.

- load the text data file

```
data = load ( 'bulgaria_data.txt' );
```

- plot year versus population (several tries):

```
plot ( data(:,1), data(:,2) );
plot ( data(:,1), data(:,2), 'r' );
plot ( data(:,1), data(:,2), 'b.', 'markersize', 20 );
plot ( data(:,1), data(:,2), 'go-' );
plot ( data(:,1), data(:,2), 'm--', 'linewidth', 3 );
```

- Force y axis to run from 0 to 10,000,000:

```
ylim ( [0.0, 10000000.0 ] )
```

- Customize the plot:

```
grid ( 'on' )
xlabel ( '<-- Year -->' )
ylabel ( '<-- Population -->' )
title ( 'Changes in Bulgarian Population' )
title ( 'Changes in Bulgarian Population', 'fontsize', 16 )
```

- save a PNG version:

```
print ( '-dpng', 'bulgaria_line_plot.png' );
```

2 Exercise: Line plots of the PRICE data

The file *price_data.txt* is a table of average monthly prices for 11 consumer products, between February 2008 and February 2018. There are 241 records, and each record contains 13 items: the month, the year, and the prices of 11 goods. Columns 3, 10 and 13 contain the prices of bananas, gas, and milk, respectively. We want a single plot that shows line plots for all three items together. There are two ways to do this.

- load the file, extract data, add month label:

```
  data = load ( 'price_data.txt' );
  bananas = data(:,3);
  gas = data(:,10);
  milk = data(:,13);
  month = 1:241;
```

- plot data all at once, or in three steps:

```
  plot ( month, bananas, month, gas, month, milk, 'linewidth', 3 );
  clf ( );
  hold ( 'on' );
  plot ( month, bananas, 'linewidth', 3 );
  plot ( month, gas, 'linewidth', 3 );
  plot ( month, milk, 'linewidth', 3 );
  hold ( 'off' )
```

- Finish the plot:

```
  legend ( 'Bananas', 'Gas', 'Milk' );
  grid ( 'on' )
  xlabel ( 'Month Index', 'fontsize', 16 );
  ylabel ( 'Price in Dollars', 'fontsize', 16 );
  title ( 'Prices, Feb 2008-Feb 2018', 'fontsize', 16 );
  print ( '-dpng', 'price_line_plots.png' );
```

3 Exercise: Scatter plot of the FAITHFUL data

The file *faithful_data.txt* contains 272 observations of the Old Faithful geyser. Each line records 2 items: the eruption length in minutes, and the wait in minutes until the next eruption. It might seem reasonable that the two variables are related. A long eruption might mean a long subsequent wait, for instance. Such patterns can be investigated using a scatter plot, in which we simply put a mark at each point (x, y) for which we have data.

- load the data:

```
data = load ( 'faithful_data.txt' );
```

- Use the plot() command or the scatter() command:

```
  plot ( data(:,1), data(:,1), 'ro', 'markersize', 15 );
  scatter ( data(:,1), data(:,2), 'filled' );
```

- Finish the plot:

```
grid ( 'on' )
title ( 'Old Faithful Eruption / Pause Durations' )
xlabel ( '<-- Eruption Duration (minutes)-->' )
ylabel ( '<-- Pause Duration (minutes) -->' )
print ( '-dpng', 'faithful_scatter_plot.png' );
```

What relationship between eruption time and wait time is suggested by the scatter plot of the data?

4 Exercise: Bar plot of the SCHOOLYEAR data

Both line and scatter plots assume each data item has the form (x, y). Sometimes, however, instead of x data, we have some kind of label. Thus, we might have a list of car models and their prices. A bar plot allows us to represent the data as bars, identified by their data labels, with heights proportional to the data values.

MATLAB has some trouble reading data files that contain a mixture of text and numeric data, so instead, we will get our data from a MATLAB function.

- Get country name and schoolyear length from function *schoolyear_data.m*:

```
  [ country, days ] = schoolyear_data ( );
```

- Make a horizontal bar plot of days, and use country names as labels:

```
  barh ( days );
  set ( gca, 'yticklabel', country );
```

- Finish the plot:

```
  grid ( 'on' );
  xlabel ( '<-- School Year in Days -->', 'FontSize', 16 );
  title ( 'International Variation in School Year', 'FontSize', 24 );
  print ( '-dpng', 'schoolyear_bar_plot.png' );
```

5 Exercise: Histogram of the SNOWFALL data

Suppose we want to display a large set of observations of some numerical quantity, $(x_1, x_2, ..., x_n)$. If the data has some slowly varying trend, a line plot might be suitable. If the data is not too numerous, a bar plot might work. But if the data is numerous and somewhat chaotic, we are better off with a *histogram*, which simplifies the data by grouping it into a small number of bins. A histogram then shows how many data items fall into each range.

- load the file *snowfall_data.txt* and extract column 10:

```
data = load ( 'snowfall_data.txt');
inches = data(:,10);
```

- Create a bar plot (bad idea), then switch to histogram, then specify number of bins:

```
bar ( inches );
histogram ( inches );
histogram ( inches, 25 );
```

- Finish the plot:

```
grid ( 'on' );
xlabel ( '<-- Total Yearly Snowfall in Inches -->' );
ylabel ( '<-- Frequency -->' );
title ( 'Yearly Snowfall in Michigan' );
print ( '-dpng', 'snowfall_histogram.png' );
```

What features of the data does the histogram allow you to see right away?

6 Exercise: Contour plot of the VOLCANO data

We may have a collection of measurements of temperature, or elevation, or pollution levels, sampled at regular points on a grid. One way to visualize such data is to use a contour plot. Contours can be indicated by a sequence of curves, each of which connects points with the same value of the measurement; a more vivid image can be done using colors.

- access an 82×61 array **z** from data file *volcano_data.txt*:

```
z = load ( 'volcano_data.txt');
```

- Make a line contour plot:

```
contour ( z, 'linewidth', 3 );
```

- Make a color contour plot with colorbar:

```
contourf ( z );
colorbar ( );
```

- Make a surface plot with colorbar:

```
  surf ( z, 'edgecolor', 'none' );
  colorbar ( );
  xlabel ( '<-- X -->' );
  ylabel ( '<-- Y -->' );
  zlabel ( '<-- Z -->' );
  title ( 'Surface plot of volcano height' );
  print ( '-dpng', 'volcano_surface.png' );
```

7 Exercise: Vector plot of the HEX2() function

Recall the *hex2()* function that we tried to minimize in a previous lab:

$$f(x,y) = 2x^2 - 1.05x^4 + x^6/6 + xy + y^2$$

It would be interesting to see a plot displaying both the contour lines of this function, and the field of negative gradient vectors that point towards local minimizers.

- create vectors x and y, then tables X and Y. Define $Z(X,Y)$ by calling hex2():

```
x = linspace ( -2, +2, 31 );
y = linspace ( -2, +2, 31 );
[X,Y] = meshgrid ( x, y );
Z = hex2 ( X, Y );     % I modified hex2() to take two separate inputs.
```

- Estimate the gradient using finite differences:

```
h = 0.01;
[ dZdX, dZdY ] = gradient ( Z, h );
```

- Make one plot with both contours and vectors:

```
hold ( 'on' );
contour ( X, Y, Z, 25 );
quiver ( X, Y, -dZdX, -dZdY, 2.0 );
hold ( 'off' )
```

We could also have written a function to evaluate the gradient exactly, instead of calling MATLAB's `gradient()` function.

8 Exercise: Surface plot of the MEXICAN_HAT function

A contour plot can also be useful to examine the behavior of a function $z = f(x,y)$; even better is a 3D surface plot. In this case, we have to generate the (x,y) grid values, and set up a way to evaluate the function.

- create vectors x and y, then tables X and Y. Define $Z(X,Y)$ using an auxilliary variable R:

```
x = linspace ( -8, 8, 33 );
y = linspace ( -8, 8, 33 );
[X,Y] = meshgrid ( x, y );
R = sqrt ( X.^2 + Y.^2 + eps );
Z = sin ( R ) ./ R;
```

- Compare contour, color contour, and surface plots:

```
contour ( X, Y, Z );
contourf ( X, Y, Z );
surf ( X, Y, Z );
```

9 Exercise: polygon fill plot of MARIO

The MATLAB `fill()` command can be used to create a plot made up of one or more polygons, each filled with a specific color.

The form of the command is `fill(X,Y,C)`, where `X` and `Y` are the coordinates of a polygon's vertices, and `C` is a 3-component RGB color specification, that is, real numbers between 0 and 1. To get an idea of how this works, type

```
hold ( 'on' )
fill ( [0,1,0], [0,0,1], [1,0,0] )
fill ( [0.5, 1.0, 1.0, 0.5 ], [0.5, 0.5, 1.0, 1.0 ], [0,0,1] )
axis ( 'square' )  % (otherwise, a square looks like a rectangle)
hold ( 'off' )
```

This command can be used to make beautiful contour color plots, diagrams of meshes, tilings, and other images. However, to make an interesting plot often requires entering a lot of data.

To see the kind of plots this command can create, download the file *mario.m* and run it!

10 Exercise: Tree plot of the GENEALOGY data

A tree plot is way to illustrate a process which involves branching or dependence. A tree plot can illustrate all the paths that start at the initial point and branch one way or another at decision points.
The locations on a tree are called *nodes*, and the node representing the initial state is called the *root node*. If we assign each node an identifying index, then the structure of the tree can be described if each node identifies its "parent node", which connects it back to the root node.

1. create a tree description and display the tree:

```
nodes = [ 0, 1, 2, 2, 4, 4, 4, 1, 8, 8, 10, 10 ];
treeplot ( nodes );
```

2. Create labels for the nodes:

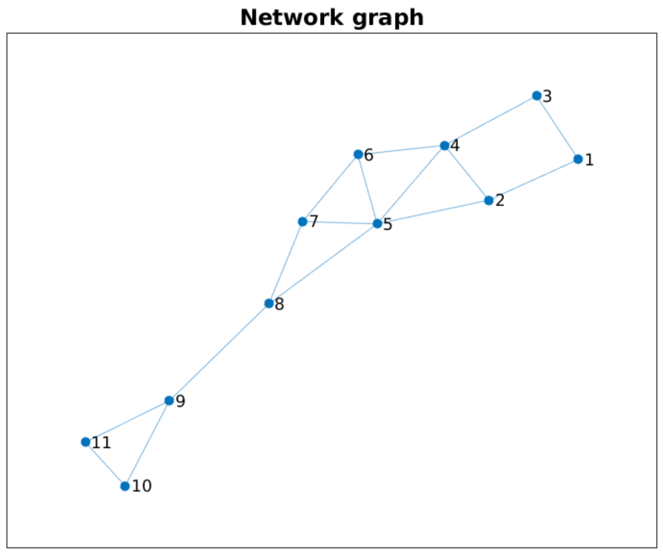
```
labels = { ...
  'Adam',
  'Bert',
  'Carl',
  'Dale',
  'Eddy',
  'Fred',
  'Gina',
  'Hank',
  'Inez',
  'Jane',
  'Kurt',
  'Leah' };
```

3. Find out where the nodes are, and attach the labels to the nodes:

```
[ x, y ] = treelayout ( nodes );
for i = 1 : length ( x )
  text ( x(i) + 0.1, y(i), labels{i} );
end
```

11 Exercise: Graph plot of the NETWORK data

Mathematically, a graph is a set of *nodes*, some of which are connected by *edges*. The edges usually represent two-way connections; one-way connections arise in "directed graphs".
The MATLAB function *graph()* can define a graph G using two lists of equal length, containing the starting node s and terminal node t of each edge.
For your exercise, consider the following network:



A plot of the network data.

1. use the picture above to fill in the missing data in these MATLAB commands that define the edges:

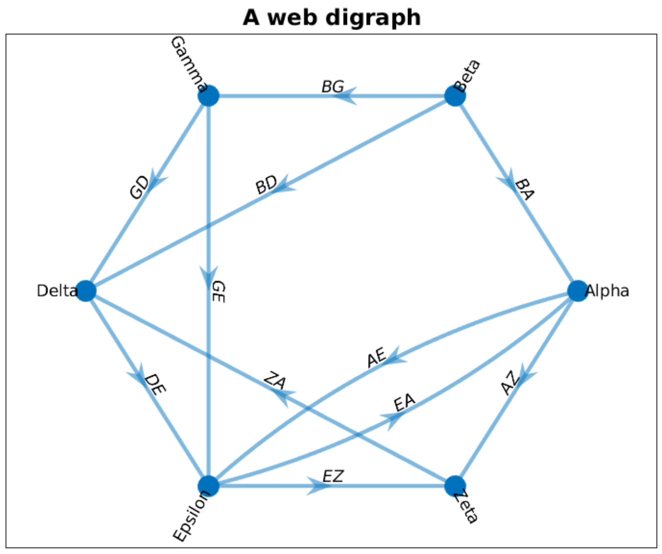
```
s = [ 1, 1, 2, ..., 10 ];
t = [ 2, 3, 4, ..., 11 ];
```

2. Use the edges to create a graph called G, and plot it:

```
G = graph ( s, t );
plot ( G );
title ( 'A Network graph' )
```

12 Exercise: Directed graph plot of the WEB data

A directed graph displays nodes connected by one-way edges. There may be a value associated with each edge, representing a weight, length or probability of access. A good display should include that information. For your exercise, consider the following network, and assume that nodes 1 through 6 are labeled Alpha, Beta, Gamma, Delta, Epsilon and Zeta, respectively.



A plot of the web data.

1. use the picture above to describe the graph of the network by filling in the s and t vectors:

```
s = [ 1, 1, 2, ..., 6 ];
t = [ 5, 6, 1, ..., 4 ];
```

2. Define the digraph and label each node and edge:

```
G = digraph ( s, t, nodenames );
node_labels = { 'Alpha', 'Beta', 'Gamma', 'Delta', 'Epsilon', 'Zeta' };
edge_labels = { 'AE', 'AZ', 'BA', 'BG', 'BD', 'GD', 'GE', 'DE', 'EA', 'EZ', 'ZA' };
```

3. Plot the graph

```
> plot ( G, ...
  'Layout', 'circle', ...
  'EdgeLabel', edge_labels, ...
  'NodeLabel', node_labels, ...
  'ArrowSize', 15 );
```

13 No Computing Assignment for this Lab!