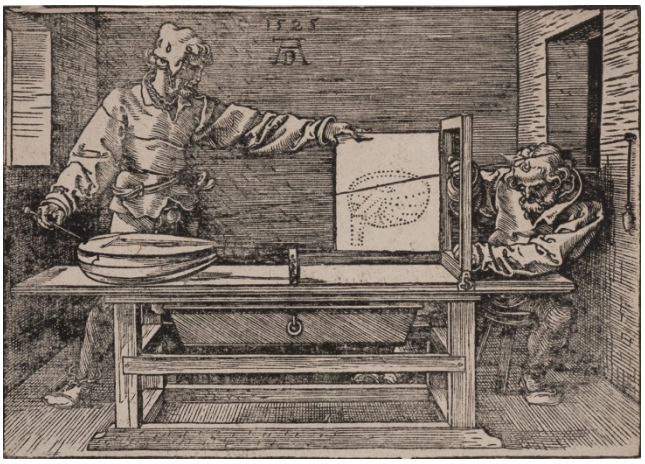
MATLAB Plots

MATH1070: Numerical Mathematical Analysis

Location: http://people.sc.fsu.edu/~jburkardt/classes/math1070_2019/matlab_plots/matlab_plots.pdf

---



How can we illustrate our data and mathematical results?

MATLAB Plots

How can we create plots that 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;
- **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 $f(x, y)$;
- **surface plot**, a 3D plot of the surface $z = f(x, y)$;
- **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 Line Plot: the BULGARIA Data

Load the text data file *bulgaria_data.txt*, which records a sequence (y_i, p_i), of years and population:
We think of the data as samples of a smoothly varying population function $p(y)$. A line plot will suggest this behavior by connecting the data points with line segments.

Exercise 1: Create the Bulgaria line plot:

1. load the text data file

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

2. plot column 1 (year) versus column 2 (population):

```
> plot ( data(:,1), data(:,2) );
```

3. Customize the plot:

```
> plot ( data(:,1), data(:,2), 'linewidth', 3 );
> grid ( 'on' )
> xlabel ( '<-- Year -->' )
> ylabel ( '<-- Population -->' )
> title ( 'Changes in Bulgarian Population' )
```

4. save a PNG version:

```
  > print ( '-dpng', 'bulgaria.png' );
```

2 Line Plots: 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.

Exercise 2:

1. load the text data file *price_data.txt*;
2. create a month counter:

```
> month = 1 : 241;
```

3. tell MATLAB to wait for multiple plots:

```
> hold on;
```

4. plot months versus banana prices (column 3):

```
> plot ( month, data(:,3) );
```

5. add similar plot commands for gas and milk;
6. tell MATLAB the multiple plots are done:

```
> hold off;
```

7. save a PNG copy as *price.png*;

3 Scatter Plots: the FAITHFUL Data

The file *faithful_data.txt* contains 272 observations of the Old Faithful geyser. Each line records 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.

Exercise 3:

1. load the data file *faithful_data.txt*:

```
> data = load ( 'faithful\_data.txt' );
```

2. plot column 1 versus column 2; the extra argument 'ro' marks data with red points, no connection:

```
> plot ( data(:,1), data(:,1), 'ro' );
```

3. save a PNG copy as *faithful.png*;

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

4 Bar Plot: 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.

Exercise 4:

1. access the MATLAB M file *schoolyear_data.m*:

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

2. Make a horizontal bar plot of the school days:

```
> barh ( days );
```

3. Label each bar with this horrible command:

```
> set ( gca, 'yticklabel', country );
```

4. save a PNG copy as *schoolyear.png*;

5 Histogram: 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.

Exercise 5:

1. access the data file *snowfall_data.txt*:

```
> data = load ( 'snowfall_data.txt');
```

2. Copy out column 10 as "inches":

```
> inches = data(:,10);
```

3. Create a bar plot of the inches, and convince yourself it is not a good presentation of the data:

```
> bar ( inches) ;
```

4. Instead, create a histogram, allowing MATLAB to choose the number of bins:

```
> histogram ( inches );
```

5. save a PNG copy as *snowfall.png*;

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

6 Contours: 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.

Exercise 6:

1. access the data file *volcano_data.txt*:

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

2. Make a line contour plot:

```
> contour ( z );
```

3. Make a color contour plot:

```
> contourf ( z );
```

4. save a PNG copy as *volcano.png*;

7 Surfaces: 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.

Exercise 7:

1. create X and Y grid arrays:

```
> [X,Y] = meshgrid ( -8 : 0.5 : 8 );
```

2. create the Z grid array:

```
> R = sqrt ( X.^2 + Y.^2 + eps );
> Z = sin ( R ) ./ R;
```

3. Start with a line contour plot:

```
> contour ( X, Y, Z );
```

4. Compare a color contour plot:

```
> contourf ( X, Y, Z );
```

5. Finish with a 3D surface plot:

```
> surf ( X, Y, Z );
```

6. save a PNG copy as *mexican_hat.png*;

8 Trees: 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.

Exercise 8:

1. create a tree description:

```
> nodes = [ 0, 1, 2, 2, 4, 4, 4, 1, 8, 8, 10, 10 ];
```

2. display the tree

```
> treeplot ( nodes );
```

3. Create labels for the nodes:

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

4. Find out where the nodes are:

```
> [ x, y ] = treelayout ( nodes );
```

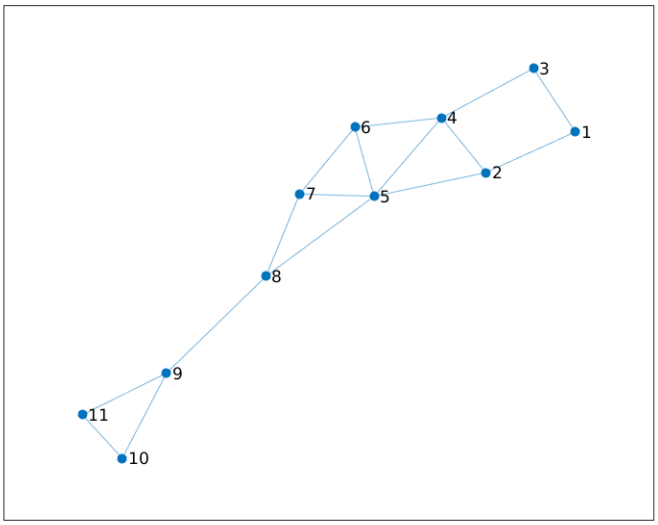
5. Attach labels to the nodes:

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

6. save a PNG copy as *genealogy.png*;

9 Graphs: 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:



Exercise 9:

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

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

2. create a graph called G:

```
> G = graph ( s, t );
```

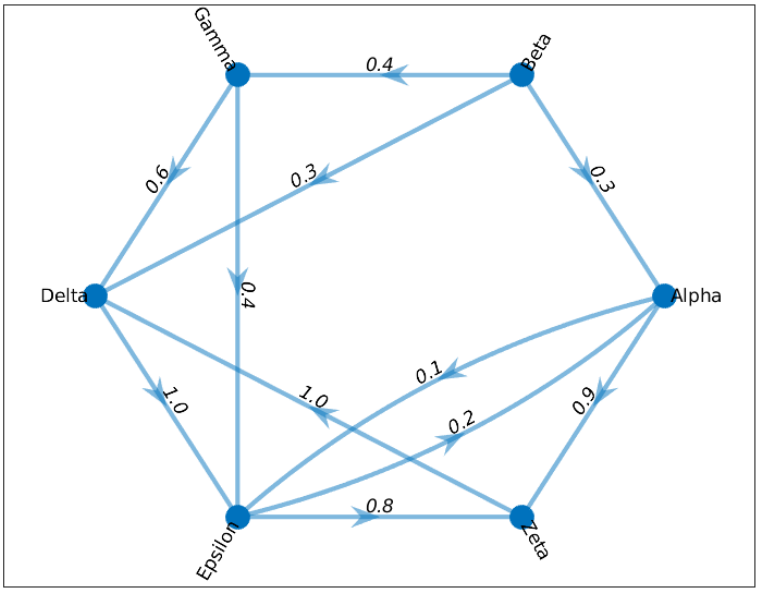
3. Plot the graph

```
> plot ( G );
```

4. save a PNG copy as *network.png*;

10 Graphs: 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.



Exercise 10:

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

```
>   G = digraph ( s, t, nodenames );
```

3. set a label for each node:

```
> node_labels = { 'Alpha', 'Beta', 'Gamma', 'Delta', 'Epsilon', 'Zeta' };
```

4. set a label for each edge:

```
> edge_labels = { '0.1', '0.9', '0.3', ...,'1.0' };
```

5. Plot the graph

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

6. save a PNG copy as *web.png*;