

ENGR101 Lab 5

[Part 1: Plotting in MATLAB](#)

[Part 2: Mapping in MATLAB](#)

[Part 3: Turning in Your Assignments](#)

[Exercise 1: Plotting in MATLAB](#)

[Exercise 2: Plotting A Histogram in MATLAB](#)

[Exercise 3: Mapping](#)

Submission Requirements:

All labs are due at 11:59pm the day after your lab section. See Part 3 of this lab on how to submit these files.

- Exercise 1: Save your work as lab5_1.m
- Exercise 1: Save your graph as lab5_1.png
- Exercise 1: Save your data as lab5_1.csv
- Exercise 2: Save your work as lab5_2.m
- Exercise 2: Save your graph as lab5_2.png
- Exercise 3: Save your work as lab5_3.m
- Exercise 3: Save your graph as lab5_3.png

Additional Note for Reading in Data

In Lab 4, we learned several functions to read data from a file and to write data to a file. Another useful type of file is the Comma Separated Values file (.csv). CSV files are commonly used with programs like Excel. The following CSV functions allow us to read and write CSV files:

`M = csvread('filename')` reads in a datum from a .csv file and stores it in the variable M.

`csvwrite('filename', M)` writes variable M to a .csv file where 'filename' is the name of the file.

These functions will be very helpful for Project 3...

Note

Make sure the filename includes the extension (i.e. csv) at the end of the filename.
Example: `csvread('LovePlotting.csv')`.

Part 1: Plotting in MATLAB

One important part of engineering is communicating technical results. Engineers use tools such as graphs and figures to visually communicate their findings. MATLAB has several functions and capabilities to perform graphing.

plot(X, Y) plots 2D data Y versus X in a new figure.

X and Y must be the same size. Multiple data can be plotted on a single graph by adding the next variables directly after the initial (i.e. **plot**(x1, y1, x2, y2)).

Note

X and Y must be the same length for plot to work.

plot(X, Y, 'LineSpec') plots 2D data Y versus X with additional parameters in LineSpec.

LineSpec includes options for line style, marker, and color, specified in this order. For example '-+y' is read as a yellow (y) solid line (-) made of plus sign points (+). See the [Appendix](#) at the end of this lab for tables detailing the markers and their descriptions for line style, marker, and color. You may also MATLAB's [help page](#) on the **plot**() function.

title('text') adds the 'text' title to the top center of the figure.

xlabel('text') adds the 'text' below the x axis of the graph in the figure.

ylabel('text') adds the 'text' below the y axis of the graph in the figure.

axis([xlb xup ylb yup]) sets the limits of the x and y axes.

xlb is the lower bound of the x axis and xup is the upper bound of the x axis.

ylb is the lower and yup is the upper bound of the y axis.

legend('text') adds the 'text' to the graph indicating what the various data lines represent in the graph of the figure.

For example, **legend**('f(x)', 'f(y)') would label the lines of f(x) and f(y). The default placement for legend is in the top right corner of the graph. If this is an unsatisfactory location, then relocate the legend using the command syntax **legend**('text', 'Location', 'southeast'). This code would place the legend at the bottom right corner of the graph following general cardinal directions. Any cardinal direction can be used such as 'north', 'east', 'northwest', etc. For all locations, refer to the Legend Property [MATLAB help page](#).

Note

Order matters for `legend()`. If you plot $f(x)$ first and then $f(y)$, you use `legend('f(x)', 'f(y)')`.

grid on places a grid on the background of the graph.

grid off removes a grid on the background of the graph (default).

hold on is one method to add another line to an already existing figure.

hold off will turn off this method and return to plotting each line on a new figure.

subplot(m, n, p) separates the figure space into different quadrants so multiple graphs can appear in the same figure.

The subplot command must come before the plot command to indicate where the graph should be plotted on the figure. (m, n, p) describe the location within the figure. For example, `subplot(2,2,1)` indicates that the graph should go in the top left corner of a 2 x 2 matrix within the figure; `subplot(2,2,4)` would place the graph in the bottom right corner of a 2 x 2 matrix within the figure.

histogram(x) creates a histogram graph of X.

You will learn more about this function in Exercise 2.

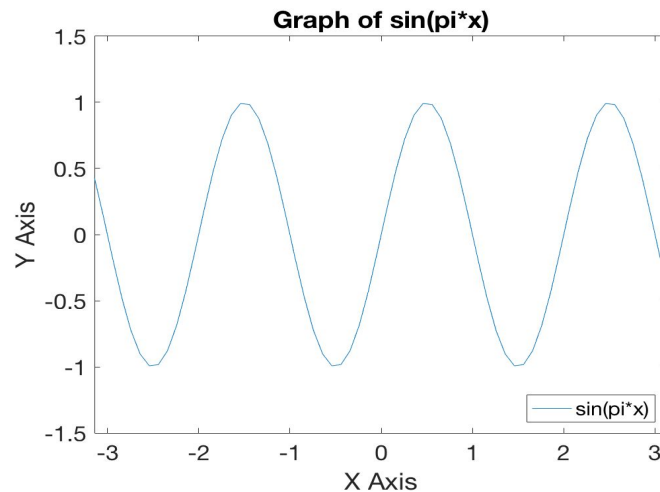
print('filename', formattype) saves the file in a specified format.

In this lab, we will use `print()` to save our images with the name 'filename' in format type '-dpng'. For more file types, see [MATLAB's help page](#) on `print()`.

Here are some examples of using the above functions to create different graphs. The first example creates a single graph. The second example creates two graphs next to each other.

Single Graph

The figure below displays a graph of $\sin(\pi \cdot x)$. Remember that ' π ' is a built in and defined variable in MATLAB. DO NOT CHANGE IT !!!



To make this image, the first step is to create vectors with all the elements for variable x and then all the elements for variable y . Please note that both variables have the same size.

```
x = [-pi:0.1:pi];  
y = sin(pi.*x);
```

To plot the output, we use the plot function with the x variable on our horizontal axis and the y variable on the vertical axis. MATLAB will plot each pair of elements in x and y as a point. Then it will draw a line through all the points.

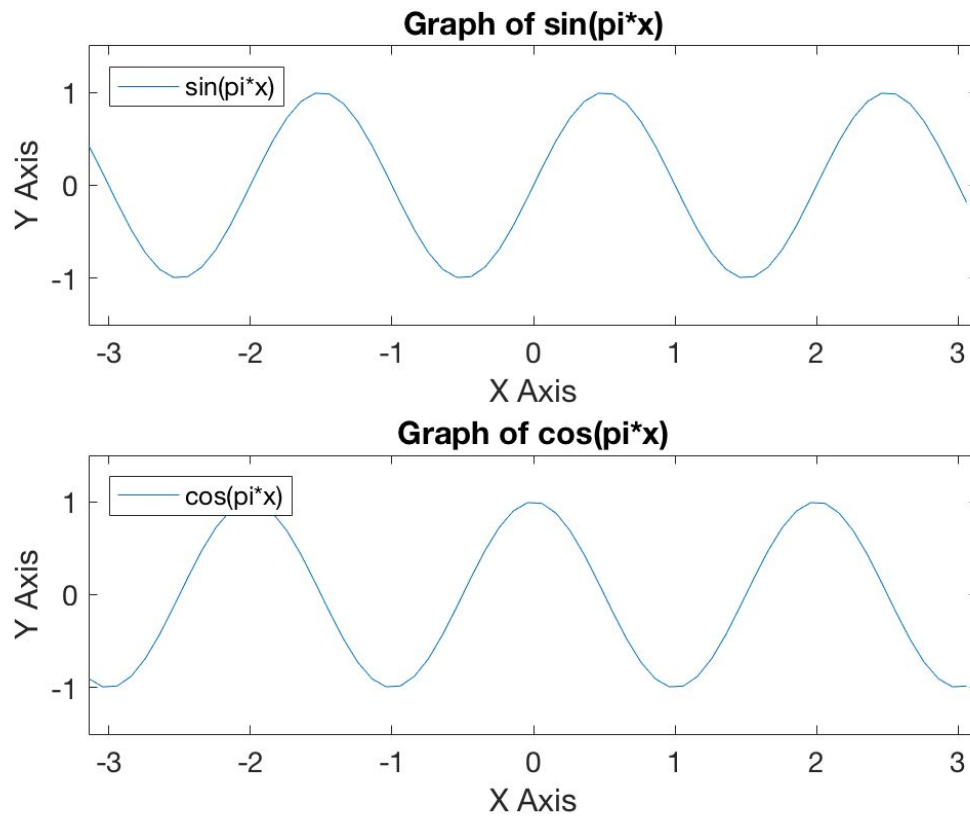
```
plot(x,y);
```

Now, we can label the axes, set the x-axis, add the title, and add a legend. The final line sets the font size of the title and axis of the graph.

```
xlabel('X Axis');  
ylabel('Y Axis');  
axis([-pi pi -1.5 1.5]);  
title('Graph of sin(pi*x)');  
legend('sin(pi*x)', 'Location', 'southeast');  
set(gca, 'fontsize', 18);
```

Multiple Graphs

To plot multiple graphs on the same figure, we will use **subplot()** with the above procedure for making a single graph.



The figure was generated using the following code where **subplot(2,1,1)** designates the location of the sine graph and **subplot(2,1,2)** designates the location of the cosine graph. The first two arguments denote a 2x1 grid. The third argument specifies where to put the plot.

```

% Create a 2x1 grid of plots. Put the following plot in position 1
subplot(2,1,1);

% Plot sin(pi.*x)
x = [-pi:0.1:pi];
y = sin(pi.*x);
plot(x,y);

% Format the plot with the following settings
xlabel('X Axis');
ylabel('Y Axis');
axis([-pi pi -1.5 1.5]);
title('Graph of sin(pi*x)');
legend('sin(pi*x)', 'Location', 'northwest');
set(gca, 'fontsize', 14);

% In the existing 2x1 grid of plots, put the following plot in position 2
subplot(2,1,2);

% Plot cos(pi*x)
x = [-pi:0.1:pi];
y = cos(pi*x);
plot(x,y);

% Format with the following settings
xlabel('X Axis');
ylabel('Y Axis');
axis([-pi pi -1.5 1.5]);
title('Graph of cos(pi*x)');
legend('cos(pi*x)', 'Location', 'northwest');
set(gca, 'fontsize', 14);

```

Note

You may also open additional figure windows with the command **figure(n)**, where n is the figure number. See Lecture 8 Notes.

Part 2: Mapping in MATLAB

Another application of MATLAB plots is to map contours (e.g. peaks) of functions.

contour(M) displays a line-contour plot of a matrix M, similar to a topographic map, where the input matrix M is interpreted as peak heights.

contourf(M) also makes a counter plot of the input matrix M, but fills the map with colors that represent the peak heights of M. Each color can represent a range of values in M.

colormap 'name' sets the color theme to the color map specified by 'name'.

See the [Appendix](#) of this lab for a list of color map names.

colorbar displays a color bar on the figure.

[X, Y] = meshgrid(x,y) is a function that returns two matrices: X and Y. All rows in X are copies of vector x. Similarly, all columns in Y are copies of vector y. Both matrix X and matrix Y have dimensions [length(y), length(x)].

This command is useful in plotting surfaces such as for **contourf**(). To learn more about how this function works you can input this example into MATLAB and study the variable output (X and Y) as shown below.

```
>> x = 1:3;
>> y = 1:5;
>> [X Y] = meshgrid(x,y);
X =
     1     2     3
     1     2     3
     1     2     3
     1     2     3
     1     2     3
Y =
     1     1     1
     2     2     2
     3     3     3
     4     4     4
     5     5     5
```

Reference: Matlab help page on `meshgrid`().

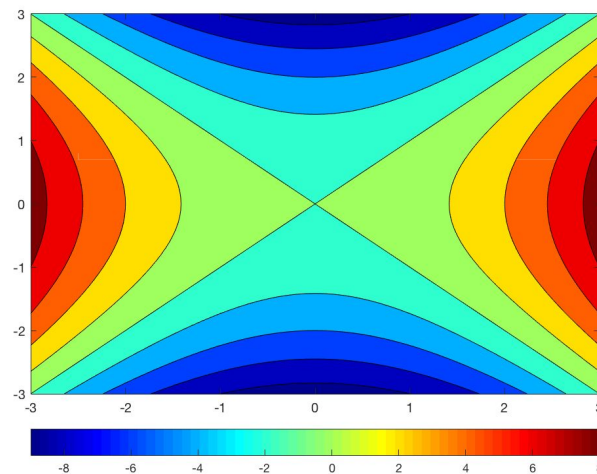
linspace(x1, x2, n) is a quick way to generate a row vector with n evenly spaced points in a dedicated range from x1 to x2. Note that the number of *points* is one greater than the number of

steps, which must be considered to obtain a clean step size. For example, the code below makes a vector that steps from 5 to 6 with a step size of 0.1, meaning it has 11 points in total.

```
>> x = linspace(5,6,11)
x =
    5.0    5.1    5.2    5.3    5.4    5.5    5.6    5.7    5.8    5.9    6.0
```

This function can be useful when generating a vector to be used in equations and various plots such as **contourf()**.

Contourf with Meshgrid



First, let's create a meshgrid with x ranging from -3 to 3 with an increment of 0.1 and y with the same increment.

```
[x y] = meshgrid(-3:0.1:3, -3:0.1:3);
```

We define the hyperbola function as the following.

```
z = x.^2 - y.^2;
```

Plot the function as a contour map. The default number of lines printed is 10. You can add a greater amount of lines by adding a number input after the z input.

```
contourf(x,y,z);
```

We change the color map to the 'jet' colormap:

```
colormap('jet');
```


We place a color bar on the bottom of the figure outside of the graph.

```
colorbar('southoutside');
```

Part 3: Turning in Your Assignments

Labs must be turned in through the relevant assignment, Lab 5, in Canvas. Be sure to submit all files listed under [Submission Requirements](#).

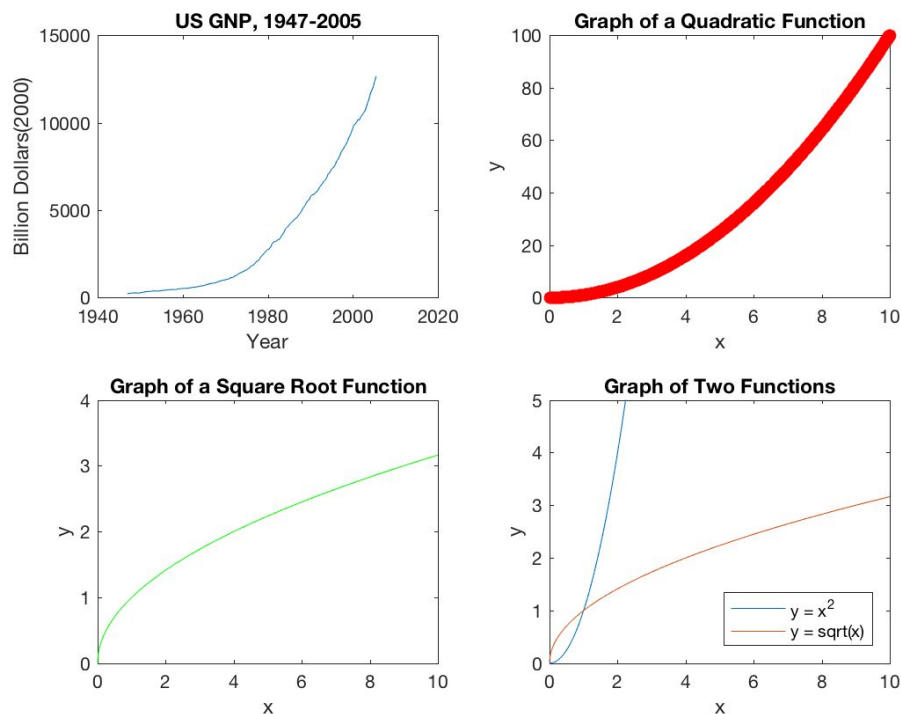
The name of your files must be exactly as specified in order to receive credit.

Exercise 1: Plotting in MATLAB

Download *lab5_1.m* and *lab5_1_GNP.csv*.*

The purpose of this exercise is to learn how to graph data from a .csv file and plot multiple graphs on the same figure. You will be using plotting functions to plot four line plots and save part of your data to a .csv file.

Your final output should be a matlab figure with four graphs as shown below. **Note: you may need to enlarge the figure window to duplicate the following display.** Please submit *lab5_1.m*, *lab5_1.csv*, and *lab5_1.png*.

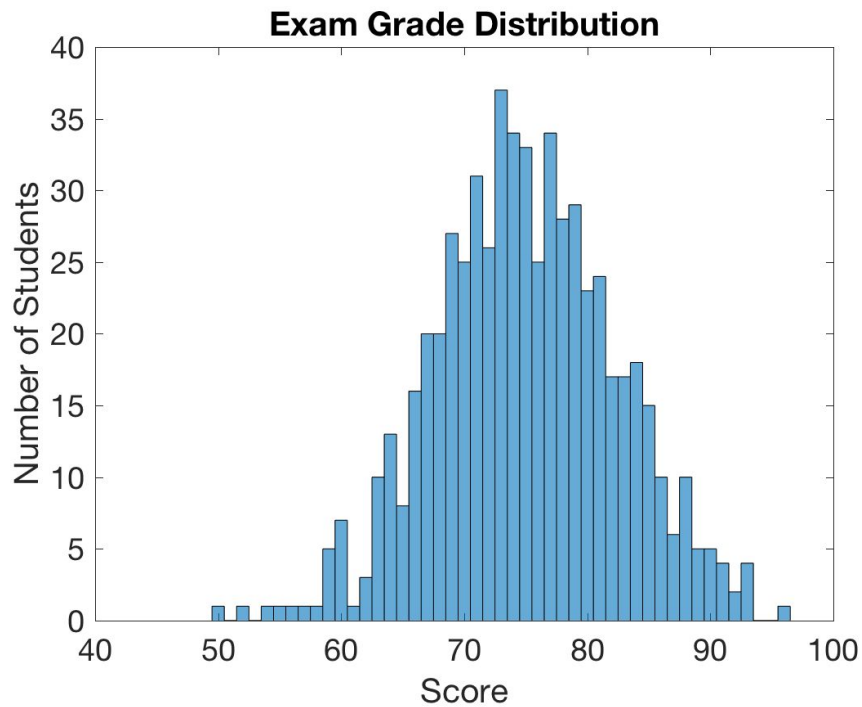


*GNP data taken from MATLAB data examples.

Exercise 2: Plotting a Histogram in Matlab

Download *lab5_2.m*.

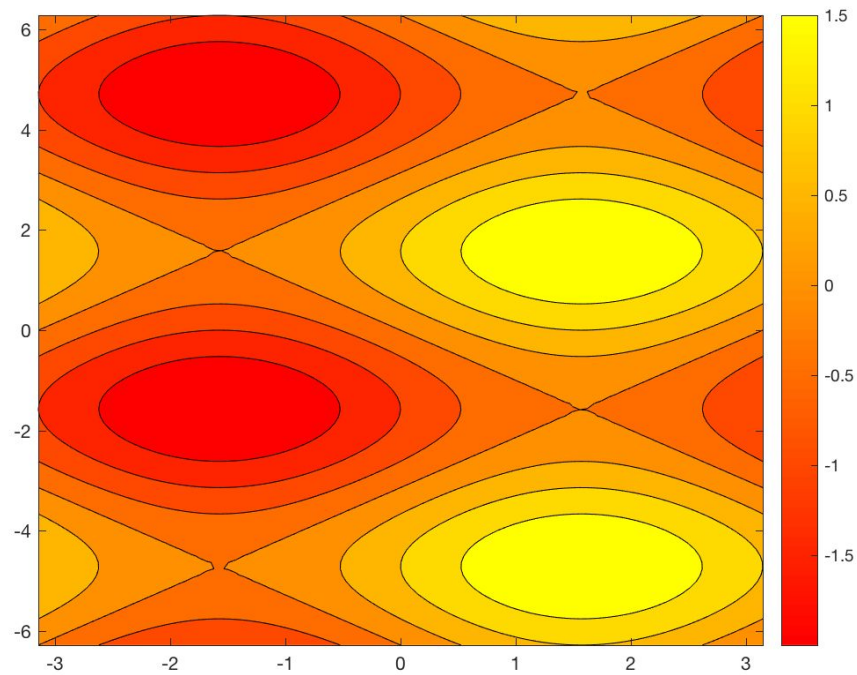
The purpose of this exercise is to learn how to plot a histogram. You will be using plotting functions to plot a histogram of a grade distribution. Your final figure should look like the figure below. Please submit both *lab5_2.m* and *lab5_2.png*.



Exercise 3: Mapping

Download *lab5_3.m*.

The purpose of this exercise is to learn how to plot the contours of a function. You will be using various functions to create vectors and create a contour map. Your final figure should look like the figure below. Please submit both *lab5_3.m* and *lab5_3.png*.












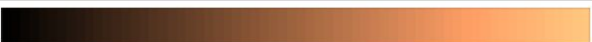







Appendix

Line Style	Description
-	Solid line (default)
--	Dashed line
:	Dotted line
-.	Dash-dot line

Marker	Description
o	circle
+	Plus sign
*	asterisk
.	point
x	cross
s	square
d	diamond
^	Upward-pointing triangle
v	Downward-pointing triangle
>	Right-pointing triangle
<	Left-pointing triangle
p	pentagram
h	hexagram

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

Colormap Name	Color Scale
parula	
jet	
hsv	
hot	
cool	
spring	
summer	
autumn	
winter	
gray	
bone	
copper	
pink	
lines	
colorcube	
prism	
flag	
white	