PRE-LAB #2: Plotting Data; Buttons and Switches

Notes:

In Experiment #1 we learned how to take measurements using the voltmeter at our lab bench. In Experiment #2, you will employ these skills in order to explore the behaviors of simple circuits. This pre-lab assignment will provide an overview of using the program MATLAB to plot the data points you collected in the previous lab, as well as introduce some background topics and terminology that we will need for Experiment #2. Finally, you will build a small circuit in preparation for experiment 2's Breakout Session.

Please use the **Notes** *margin on the right for both notes to yourself about the experiment as well as for feedback to your TA on the quality or clarity of the lab procedure. Thanks!*

Graphical Representation of Data

How can we depict our measurements in a manner that is easy to read, understand, and draw conclusions from? We can use **graphs**! But we must take care when creating a graph in order to avoid ambiguity. Well-measured data, when poorly plotted, can lead to erroneous conclusions and be very confusing to someone reading your report. Even your future self will likely have difficulty interpreting your own report.

Graphs (and charts) are very concise and useful methods of depicting a large amount of data. This portion of the lab outlines the necessary components for an informative graph. You will be required to draw a few graphs by had, but most will be produced using a powerful computing platform – MATLAB. So, in addition to an introduction to "good plotting habits", you will get a quick introduction to plotting graphs using MATLAB. **MATLAB** is *a high-level programming language and computing environment* that has become a very common tool among engineers. It is important that you get comfortable with it early in your academic career.

Plotting Graphs

Below are the details that are necessary when plotting a graph. Without these details, a person reading your lab report might not understand what your graph means and you will not receive full credit.

Notes:

Title/Caption

The title of your graph should give the reader an idea of *what* is plotted and *why* it matters. In a lab course like ECE110, it should be made clear which step (or question) in the procedure is being addressed by the graph.

Example caption:

Figure 6: The IV characteristic of a DC motor with a linear curve fit to the region after turn-on.

Axes labels and Units

The labels for your axes should tell the reader what physical quantity is being plotted. Calling your axes x and y is uninformative and is considered inadequate in a quantitative experimental setting. Common labels in ECE110 include *time (in seconds)* as the horizontal axis and *voltage (in Volts)* as the vertical axis or *voltage (V)* as the horizontal axis and *current (mA)* as the vertical axis. Always, where appropriate, include the units in the axis label.

Axes scales

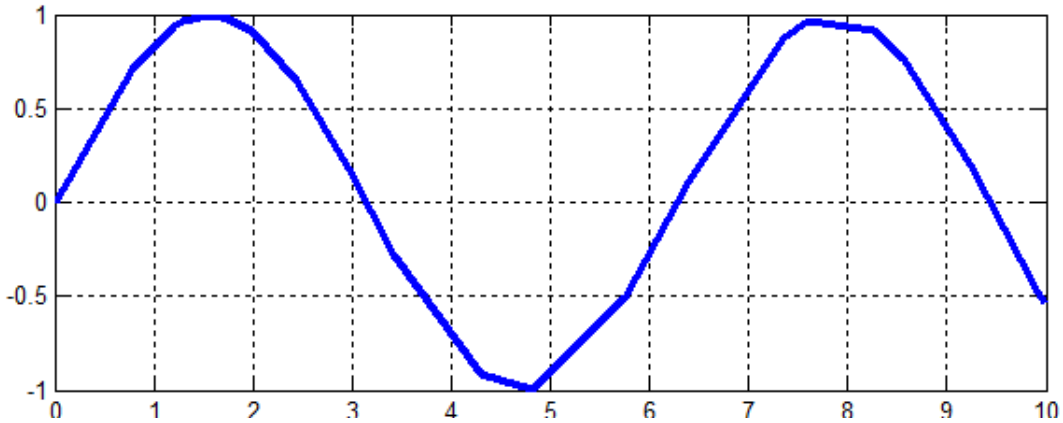
The scale of your axes is usually depicted by *labeling three or more divisions* with a numerical value. Sometimes your scale will be integer-valued and in other cases it might not be. Keep in mind that the scale of your graph should be chosen to show critical detail. If you choose a scale too large, the plot will be too small and the reader will have a hard time seeing important aspects of the curve.

Legend/plot labels

Legends are necessary when you have multiple curves on one graph. Each plot should be clearly labeled so that is clear what data are represented by each curve on your graph.

Notes:

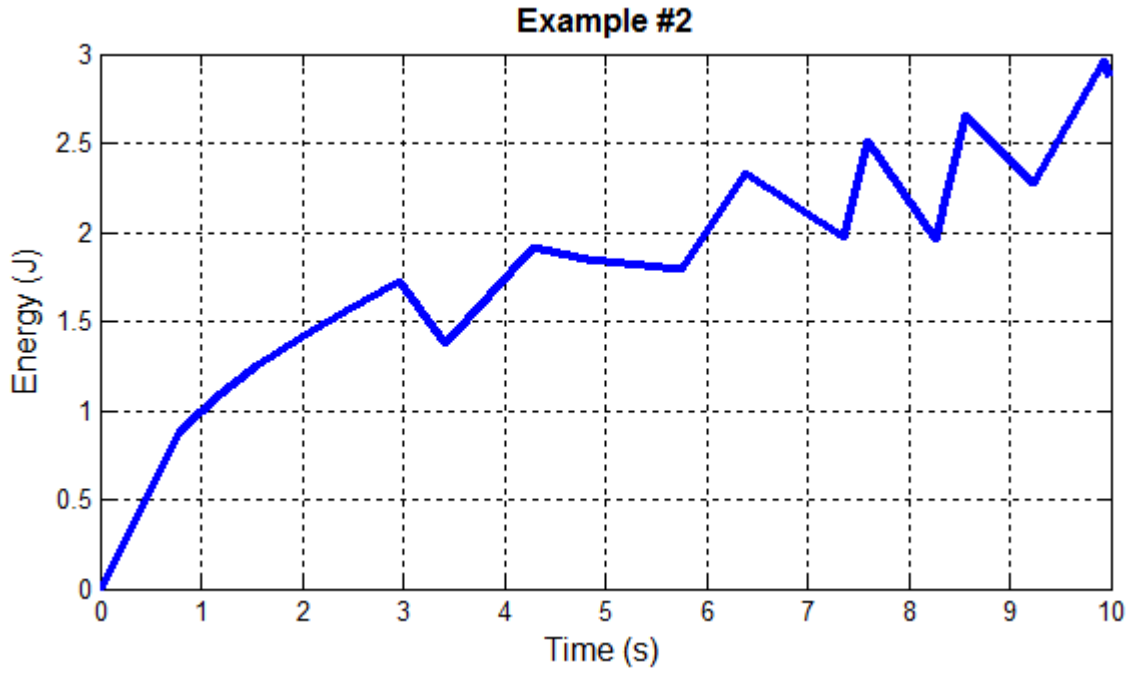
Below are some examples of graphs generated with various data sets. Identify whether each graph is acceptable or not. If you feel a graph is inadequate, clearly state why. This is a good time to discuss your thoughts with your classmates.



Question 1: What is wrong with the graph above? Consider the key features of a good graph described earlier.

No title or caption, no axes labels, no units

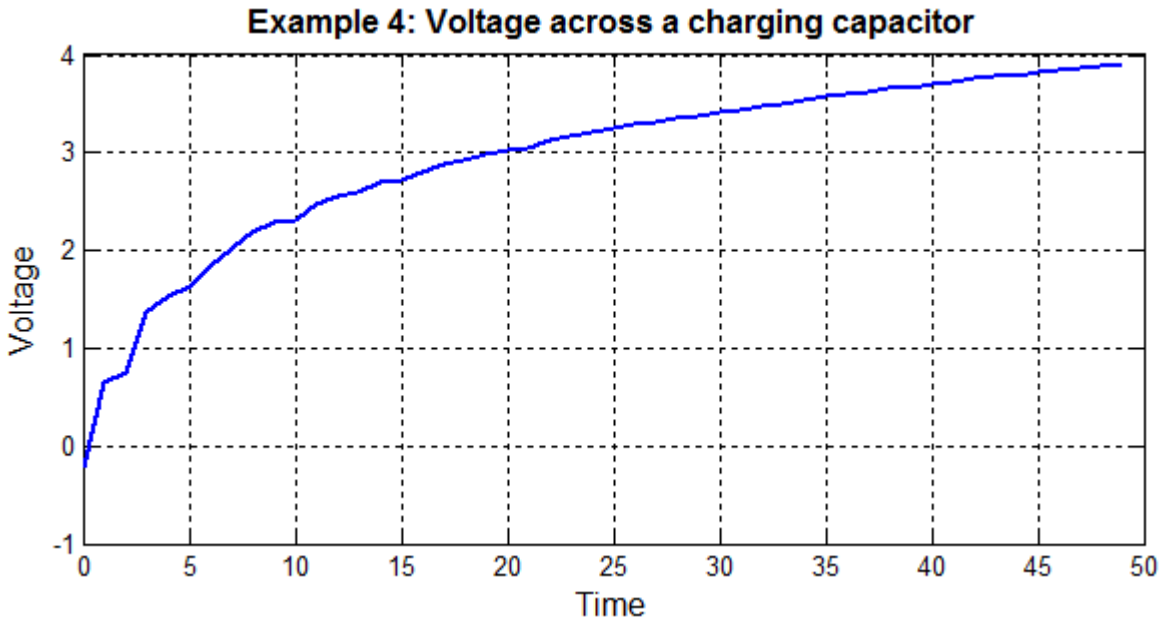
Notes:



Question 2: What is wrong with the graph above?

Title does not tell us anything about the data. A more-descriptive title and/or caption is needed.

Notes:



Question 3: What is wrong with the graph above?

No units

Notes:

Starting MATLAB

MATLAB is a very popular program used in a wide range of scientific computing and engineering applications. For our purposes we will use it to collect, manipulate, and plot the data we've measured in lab. You can think of it as the best graphing calculator you've ever owned! It is accessible from all the university Engineering Workstations (EWS) computers (like the ones at your lab benches). You can also access it on your personal computer via the Citrix service provided by EWS. Visit https://it.engineering.illinois.edu and search for "Citrix" for full instructions. You can save your MATLAB scripts, labeled as .m files, on your EWS drive, or on your local drive if you are using Citrix.

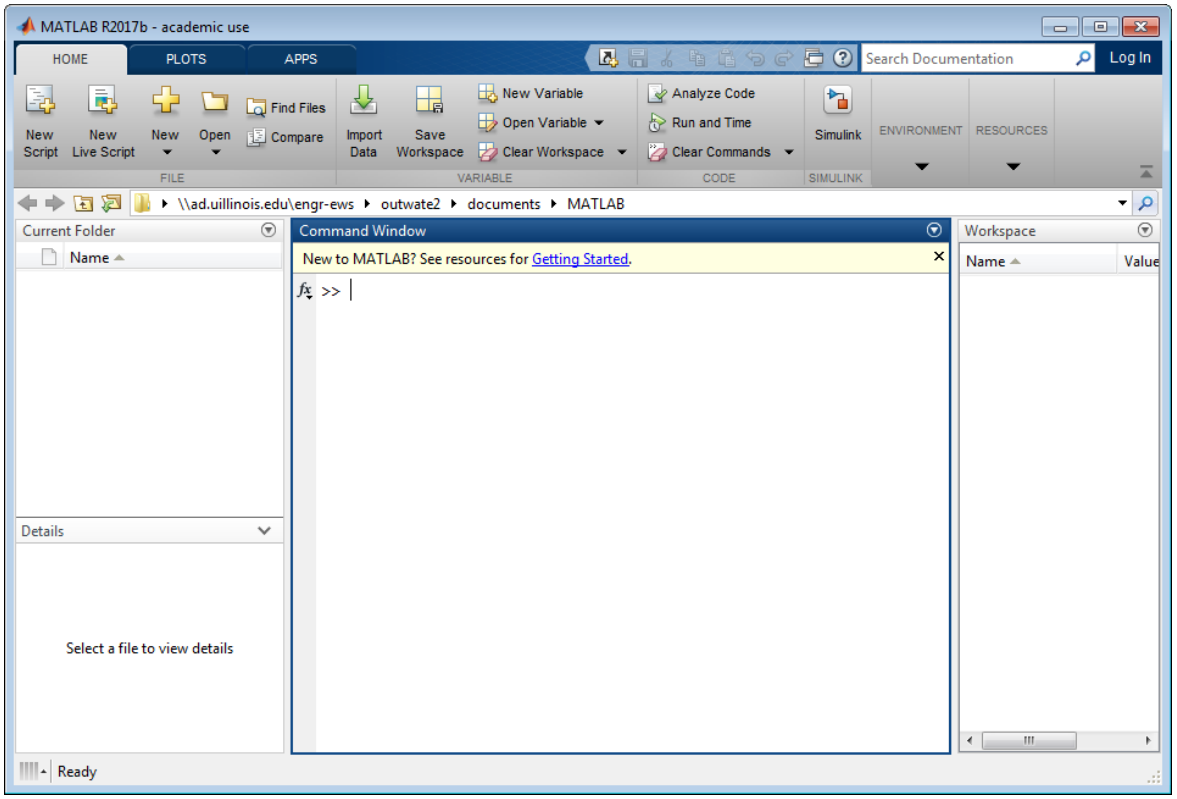


***Figure 1**: A new instance of MATLAB*

Notes:

Plotting graphs in MATLAB

Throughout the semester you will be asked to perform mathematical operations on your data and to generate your own graphs based on data collected during an experiment. In general, it will be expected that you create these graphs using MATLAB and attach them to the end of your lab report. Given that your graph may be separated from its corresponding portion of the experiment, it becomes *very important* that you give each graph a useful title and make a note in that section of the experiment so the reader knows where to find it.

Task 1: Below are the basic commands for generating a plot in MATLAB. Go ahead and enter these commands into the *Command Window* in MATLAB right now. At the command prompt '>' type each line and hit *Enter*. When the command prompt returns after each line is typed, enter the next line. Notice what changes when you hit *Enter* in the workspace and history windows. Note: the semicolon is optional – try entering the first line without it.

```
x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
y = [0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100];

figure(1);
plot(x, y);
title('Parabola');
xlabel('X Axis (x units)');
ylabel('Y Axis (y units)');
```

If you don't already have experience using MATLAB, these commands might be a little cryptic. To shed a little light on the issue, we've broken things down a little further. For further info about any command you can type "`help command`" or "`doc command`" at the command prompt, substituting the appropriate command label.

Further explanation:

- Creating variables

  ```
  x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
  ```

  This line of the code creates a variable named "x". This particular variable is an *array*, meaning that it is a collection of variables. Arrays can be multidimensional. The above array is a 1D array, containing 11 values.

Notes:

Alternatively, you can create a variable in the same way you would enter data into a spreadsheet. You can do this by using the menu under the *Workspace* (upper left corner of the MATLAB window) to create new variable. Once you have created and named a variable you can simply double click the variable name in your *Workspace* and a spreadsheet will open displaying its contents. Once you have this spreadsheet open, you can edit or add to the content associated with that variable name by selecting a cell and entering the desired numbers.

- Creating Plots

```
figure(1);
```

This command generates a new window in which your plot (figure 1) will appear. This command is not strictly required for creating plots in MATLAB but it is good practice. If a figure is not open before you call the plot function, MATLAB will automatically open one for you – making this command unnecessary. If a figure window is already open, MATLAB will plot into that window and *overwrite any graph that may have been there*. Use this function to avoid overwriting your plots by accident.

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

This command calls a function and gives it two parameters. The first parameter, x, is used as the horizontal axis and the second parameter, y, is the vertical axis. It is important to note that in this case, the variable names were chosen to illustrate how the plot function works. When you're creating a plot for an experiment, you should give your variables meaningful names (in case you forget what they were for). For example, if our data represented voltage and current measured in Lab 1, you can name your variables something like "voltage_lab1" and "current_lab1."

Task 2: So we just created a plot with some arbitrary values that we assigned to the arrays x and y. Now let's create two new arrays named voltage and current, using the values you collected in Lab 1 (back in Table 3 of Experiment #1). Once you've created your arrays, plot the variables. You can enter your code, line by line at the command prompt. Your code should look like the following, but with your measured values:

```
figure(2);
voltage = [THE VOLTAGES VALUES YOU MEASURED IN LAB 1, Table 3];
current = [THE CURRENT VALUES YOU MEASURED IN LAB 1, Table 3];
plot(voltage, current);
```

Notes:

At this point you should have a simple plot showing the IV relationship of the single motor from Lab 1. Now we need to label our axes and give the plot a title. Doing so is quick and easy from the command line. The commands are shown below. Substitute an appropriate title and appropriate axis labels.

Title and Labels

```
title('Plot title');
xlabel('X Axis (x units)');
ylabel('Y Axis (y units)');
```

These function calls should be pretty self-explanatory. Just remember that the text you want as a title or label must have single quotes around it.

Great! At this point you should have a nicely formatted IV plot of your motor from Lab 1. But what if we want to add some additional features or modifications to out plots? Often times we may change the color and thickness of the data traces, or change the line types to dashes and dots (a good way to make lines distinct when printing out plots in black and white). Any change you want to make can be done from the command prompt, and often that is the preferred practice. But for now let's explore some ways to further modify our graph using the MATLAB plot tools.

Making Your Graph Pretty

MATLAB provides a very robust graphical interface for changing virtually any parameters of your plots. You can access the *plot tools* by selecting the icon on the figure window that is shown in the figure below.

***Figure 2**: Finding Plot tools on the MATLAB taskbar.*

Once you have the opened the plot tools, you can change the properties of most aspects of your graph by clicking on the thing you want to change and adjusting the properties displayed in the lower portion of the window. Alternatively, it is possible to adjust any of the plot properties from the command line but it is left to you to use the extensive documentation to find this information if needed.

Notes:

Task 3: Using the *plot tools* interface, adjust the settings on your IV plot so that it looks like Figure 3 (except with different data). To do this, you'll need to • add grid lines, • change the thickness of the curve, • change the data trace from a solid line to a dashed line, • adjust the font sizes for the title and labels. Don't worry about the exact font sizes and line properties. Also, be sure to ***enter your netID into the title***.

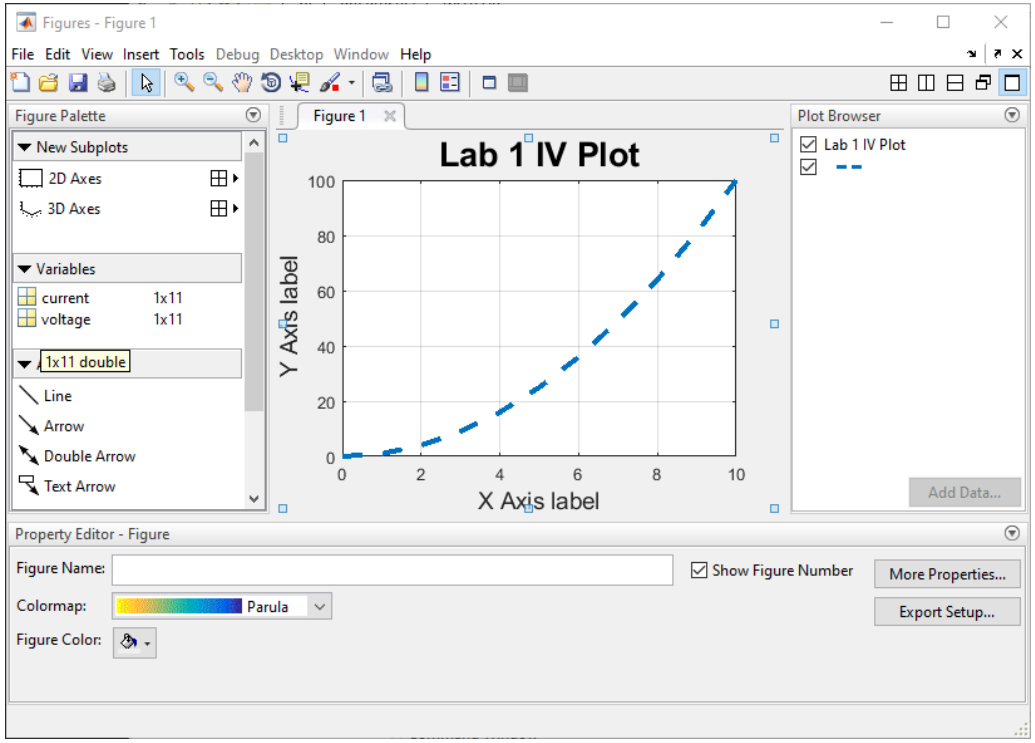


***Figure 3**: Example plot with Plot Tools windows*

Now let's try something a bit different.

Notes:

Multiple Plots in a Single Graph

Often times, we would like to compare two sets of data by plotting them on the same axes. You can do this using the "hold" command as shown below. To do this, we're going to create another variable, and plot it on the same graph as our previous plot. While we still have our previous plot open, return to the command prompt.

Task 4: Enter the following commands, creating a plot with two different data traces.

```
current2 = 2*current;
hold on;
plot(voltage, current2);
hold off;
```

Note: After entering "hold on", MATLAB will put every curve you plot on the same axes until you enter "hold off".

Task 5: Now we should have another data trace plotted next to our blue dashed line. Now let's give each trace a name, and put a legend on our plot. In the Plot Tools window, click each trace shown in the Plot Browser window on the right. Once the trace is selected, you can enter a name in the Display Name property, shown below the plot. With each plot named, now add a legend by pressing the "Insert Legend" button on the top bar.

Figure 4: Insert Legend button highlighted

With these changes made, your plot should look something like the example shown in Figure 5.

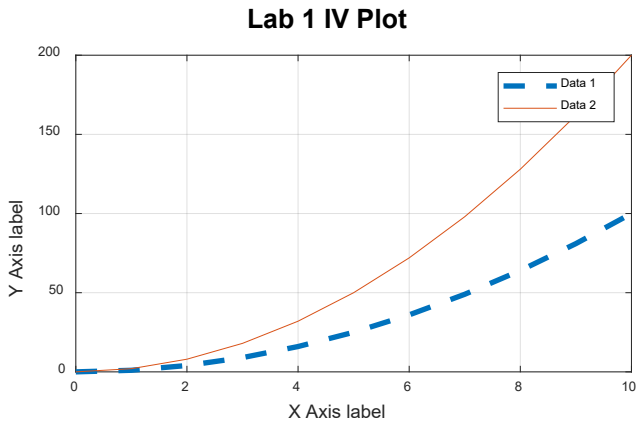


Figure 5: An example plot with two traces

Complete Summary Questions 1 through 3 (on the last page of this document).

Switching a Circuit

In the next lab, we would like to be able to control our car chassis by "switching" the motors on and off. Since our car chassis and "snap-action" switches are stored safely in your locker in the lab, we will practice first by building a circuit that uses two buttons to turn on and off two light-emitting diodes (LEDS). Locate two push buttons from your kit, as well as two red LEDs and two $330\ \Omega$ resistors. These resistors will have the color bands "orange orange brown" plus a third band (likely gold) that indicates tolerance. You may use either 4 AA batteries or a 9-V battery for power. Be very careful not to short your battery when building the circuit. Be ESPECIALLY careful not to accidentally short your battery when storing or transporting your circuit.

Using these parts, we will construct the circuit illustrated in the circuit schematic of Figure 6 and clearly explained in the physical diagram of Figure 7. The proper insertion of the button into the breadboard is explained in Figure 8.

Notes:

How to read the resistor color code: http://en.wikipedia.org/wiki/Electronic_color_code

You will want to learn a good mnemonic like the one here: http://www.orcadxcc.org/resistor_color_codes.html

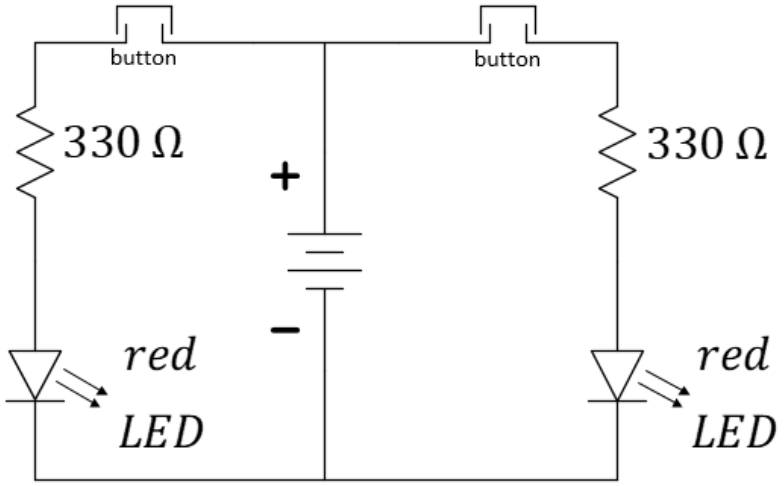
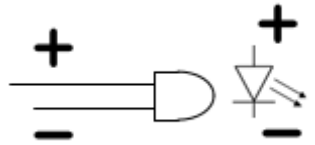


Figure 6: *Circuit schematic for switching LEDs.*

Notes:



All diodes have an anode (top) and a cathode (bottom). If the LED is inserted in reverse, it will not illuminate as the voltage is increased.

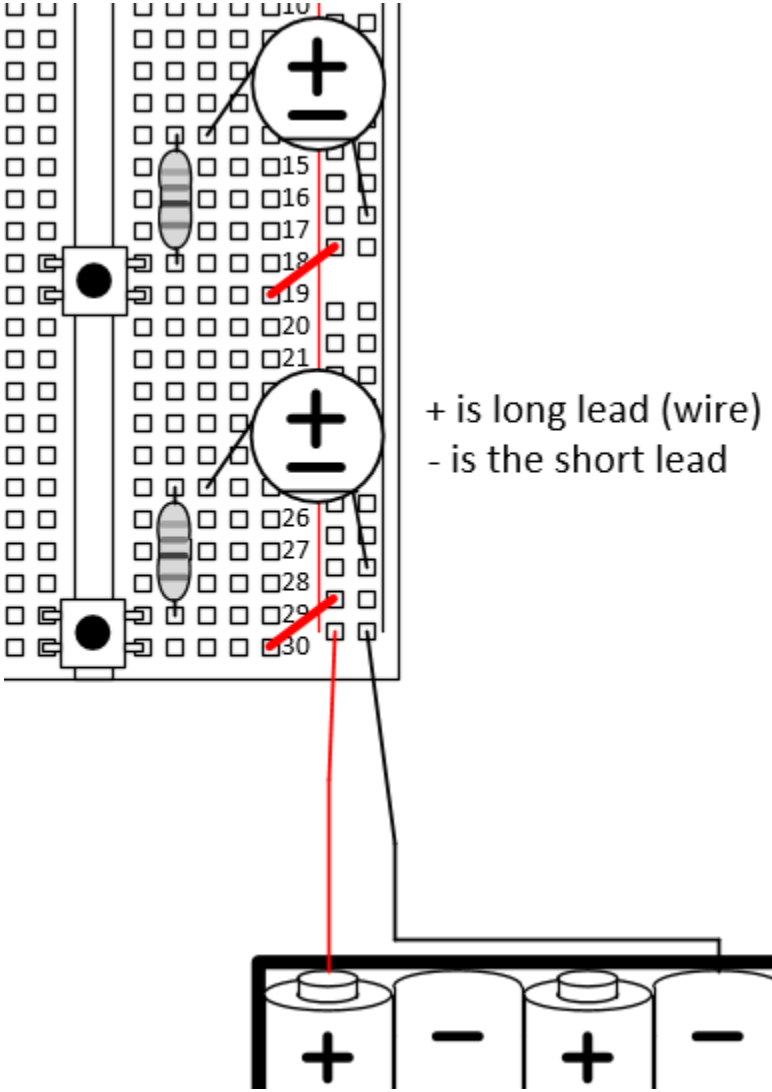


Figure 7: *Physical diagram for button-controlled motors. The barrel-to-wire adaptor will be needed.* ***Errata****: Our buttons are a little larger than the ones in the physical diagram above. Please correct the wiring accordingly. Also, the 4xAA battery pack may be replaced by your 9-V battery.*

Notes:

The LEDs of your kit have nice color!

Barrel-to-wire adaptor

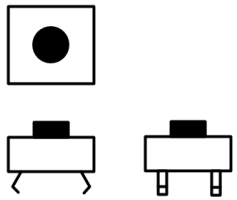


Figure 8: *Multiview projection of the button. Two flat wires will span the gap in the middle of your breadboard. These two wires are connected together when the button is pressed.*

Take the Summary page of this document as well as your MATLAB plot and scan them to PDF. This will be submitted to GradeScope.

When you arrive in lab, place the following items in a stack on the center table in 1005 ECEB and return to 1001 ECEB *before* the TAs begin laboratory Exercise #2:

1. The Summary page of this document and the MATLAB plot, stapled.
2. Your prelab hardware circuit.

Notes:

Name:_____ UIN: | | | | | | | | | | Section AB/BB: |

Notes:

PreLab 2 Summary

Question 1: Print your plot and make certain that your NetID is in the plot title. Scan it with this Summary when you submit to GradeScope.

Question 2: What does the following line of MATLAB code do?

```
time = (0:5)*1e-3;
```

Question 3: What happens if you try to plot two variables that do not have the same number of elements in them? (Try it for yourself and see what MATLAB does.)