**SA=135cm^2**

**Week 7: MATLAB Practice**

**Objectives**

* Learn to use MATLAB to perform mathematical calculations, define variables and arrays, and create 2D plots
* Learn to create and use MATLAB scripts (m files)
* Use MATLAB to plot and curve fit data collected to be used as an empirical model
* Use a numerical model to predict system behavior & compare results to actual system performance

**Introduction**

Engineers need to be comfortable creating and using mathematical models to understand and predict system behavior. In this lab, you will create a predictive model (based on measured data), which will be used to predict the solar power available to your vehicle and the maximum torque of your motor.

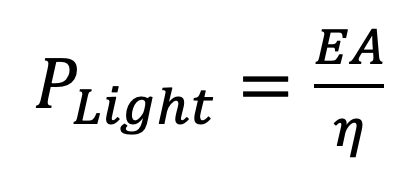
The purpose of this lab is to introduce you to Matlab, a computing software package that is very popular for technical computing. Matlab is commonly used within the engineering community because it is powerful, yet fairly easy to learn, even by those with little to no programming experience. In this lab exercise, you will learn and practice the basics of Matlab. It should be emphasized that what you will learn in this exercise will not make you an expert in Matlab, but will merely introduce you to the subject. It is highly recommended that you obtain a Matlab programming guide to further become acquainted with the software and its vast capabilities.

**Lab Procedure**

You should create a well-documented script (.m file) for this lab. All of the programming that you do for this lab should be done in this script. Be sure to avoid hard-coding, but instead, create variables for all values that can easily be changed.

**Part 1a: Creating a Predictive Model for the Solar Cells**

The goal of this part is to create a predictive model relating the input variable, Illuminance, that can be measured from the test setup and produce a value for the power available. To create a predictive model, you need to measure an input variable and an output parameter. Equation 1 is the equation for the power available in the light (in Watts):



(1)

where E is the illuminance (in lux), A is the area covered by the light (in m2), η is the luminous efficacy (in lumens/W). The area covered by the light will be determined by how big you think your solar cell will be (check Sparkyville Marketplace for the available solar cells).

Based on what you believe to be reasonable values for the area of your solar collector (array of solar cells - check Sparkyville Marketplace for the available solar cells), create variables to assign these values and use them in the equations.

Constant Variables:

* A: Surface area in m2
* η (luminous efficacy) of sunlight = 110 lumens/W

Independent Variables:

* E: Create a vector of uniformly spaced illuminance values between 100 lux and 100,000 lux with 100 values.

Using these values, create a plot of available power vs. input vector (Power vs. Illuminance). You should have a plot with labeled axes (and units), legend, and title. Illuminance will be on the x-axis, and Power on the y-axis.

Now take measurements of the actual illuminance in direct sun and in the shade. Record your measurements on the worksheet provided. Note the condition (ex. Cloudy day, sunny day, extra bright, shade from tree, shade from building, etc), the measured values, and the units.

**Part 1b: Creating an Interactive Program**

Now that you have your equation for solar power, let’s use Matlab to help solve the equation if we are to provide a single desired value of illuminance. For this, we are going to be using the “input” ability of Matlab to help write an interactive program. The goal of this program is to take an input provided by the user (through the keyboard) of the realistic illuminance value, and solve the power equation using that user-provided value. You will need to create **two** “input prompts” asking the user for the illuminance value in direct sun, and another asking the user for the illuminance value in the shade.

To do this, use the input command in your M-file to ask the user what is the value of illuminance in direct sun. Then again asking the user for the illuminance value in the shade. You will need to create a new variable to store the user input value, so that it can then be used in the predictive equation. Below is an example of how to use the “input” feature.

Prompt1 = ‘How far away do you have to travel? Answer in miles.’;

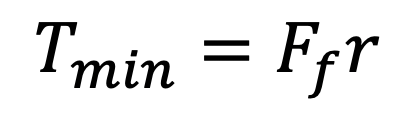
user\_distance = input(prompt1);

Assign new variable names for these two measured input values and solve the power equation to find two new Power data points. Plot each point on the original plot from above versus the expected output power. **Include your code and a screenshot of your Power vs. Illuminance plot with the predicted Power Available in Direct Sunlight and predicted Power Available in Shade data points on your Worksheet.** Please use the “snipping tool” or “screen capture” to grab a quick picture, not your cell phone cameras!

Do the measured points sit on your plotted vector line?

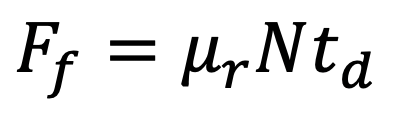
**Part 2a: Creating a Predictive Model for the Motor**

The goal of this part is to create a predictive model relating the input variable, mass of the vehicle, that can be measured from the test setup and produce a value for the torque required to get the vehicle to move. To create a predictive model, you need to measure an input variable and an output parameter. The equation below is the equation used to make this prediction and will be programmed in this code. Equation 2 is the equation for the minimum Torque (Tmin in Nm) required to move:



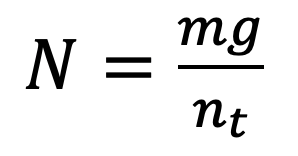
(2)

Where Ff is the Friction Force (in N), r is the radius of the tires (in m). The radius will be determined by how big you think your tires will be (check Sparkyville Marketplace for the available parts). Use equation 3 to find the Friction Force (Ff in Newtons):



(3)

Where td is the number of wheels connected to the drive axle (the number of wheels connected to the drive axle depends on your design, but today we will assume 2-wheel drive, so td =2) and use equation 4 to find the downward force (Normal Force) per wheel (N in Newtons):



(4)

Where m is the mass of the vehicle, passengers, and luggage (in kg) and nt is the number of wheels on the vehicle (the number of wheels on your vehicle depends on your design, but today we will assume 4 wheels, so nt=4).

Constant Variables:

µr (coefficient of friction) = 0.6

g (acceleration due to gravity) = 9.81 m/s2

td (wheels connected to drive axle) = 2

nt (total wheels on the vehicle) = 4

r (estimated radius of wheels in m) = Your Choice

Independent Variables:

* m: Create a vector of uniformly spaced mass values between 0 kg and 1 kg with 100 total values.

Using these values, solve for the torque at each mass data point. Create a plot of minimum required torque vs. mass (Torque vs. Mass). You should have a plot with labeled axes (and units), legend, and title.

Now take measurements of the actual mass of each passenger and piece of luggage. Record your measurements on the worksheet provided. Feel free to take the mass of any additional materials and add it to your notes and observations.

**Part 2b: Creating an Interactive Program**

Now that you have your equations, let’s use Matlab to help solve the equations if we are to provide a single desired value of weight. For this, we are going to be using the “input” ability of Matlab to help write an interactive program. The goal of this program is to take an input provided by the user (through the keyboard) of the realistic weight value, and solve the equations using that user-provided value.

To do this, use the input command in your M-file to ask the user what is the total expected mass of all passengers, luggage, and vehicle frame. You will need to create a new variable to store the user input value, so that it can then be used in the predictive equations. Below is an example of how to use the “input” feature.

Prompt1 = ‘How far away do you have to travel? Answer in miles.’;

user\_distance = input(prompt1);

Assign new variables for these measured values. Then solve for the minimum required torque due to your estimated mass, and plot the total mass point on the original plot from above versus the minimum required torque. **Include your code and a screenshot of your Torque vs. Mass plot and the predicted Minimum Torque Required to Move on your Worksheet. Include your predicted Torque in N-m.**

**When the entire Matlab script file is run, the output should be two plots:**

* **One for Power Available vs. Illuminance and the measured points marked on the plot**
* **One for Axle Torque vs Mass and the measured point marked on the plot**

**Week 7 Data Sheet: MATLAB**

| **Part 1: Solar Cells** |
| --- |
| **Testing Data**   | **Source** | **Condition** | **Measurement Conditions**  *(i.e. sunny day, windy, cloudy, hot, etc.)* | **Measurement Result(s)** | **Units** | | --- | --- | --- | --- | --- | | Solar | Direct Sun | sunny day, hot, decently clear skies | 5637 | lume  ns/w | | Solar | Shaded | sunny day, hot, decently clear skies, shade provided by a large tree and metal structure | 1028 | lume  ns/w |   **Notes and Observations:** *(be sure to include any information here that can help you better understand the conditions, set-up, and results later)* |
| **MATLAB Code:** *(insert code here)*  *%Define Constant Variables*  *A=1.35; %m^2*  *n=110; %lumens/W*  *%Define Independent Variables*  *E=linspace(100,100000,100); %E=illumance(inLUX)*  *P=(E\*A)/n;*  *%creating interactive programs*  *prompt1='What is the illuminance value in sunlight?';*  *sun\_value=input(prompt1);*  *prompt2='what is the illuminance value in shade?';*  *value\_shade=input(prompt2);*  *%Plotting Data Points*  *Psun=sun\_value\*A/n;*  *PShade=value\_shade\*A/n;*  *%create Graph*  *plot(E,P,'k',value\_shade,Pshade,'bx',sun\_value,Psun,'r\*')*  *%Graph Labeling*  *xlabel('power');*  *ylabel('illuminance');*  *title('Power VS Illuminance Graph');* |
| **Available Power vs. Illuminance** *(insert plot here)* |
| **Power Available in Direct Sun (Psun) = 69.184W**  **Power Available in Shade (Pshade) = 12.616W** |
| **Part 2: Motor Torque** |
| **Testing Data**   | **Item** | **Mass** | **Units** | | --- | --- | --- | | Driver | 101 | g | | Passengers | 228 | g | | Luggage | 89 | g | | Building Materials | 1014 | g | | Other: Golf Clubs | 27 | g | | Other: Wheel Chairs | 37 | g | |  |  |  | | **TOTAL MASS** | 1548 | g |   **Notes and Observations:** *(be sure to include any information here that can help you better understand the materials selected and the passenger/luggage configuration)* |
| **MATLAB code:** *(insert code here)*  *%define constant variables*  *Mr=.6; %coefficient of friction*  *g=9.81; %acceleration due to gravity*  *td=2; %wheels on drive axel*  *nt=4; %wheels on car*  *r=29.5;*  *m=linespace(0,2,100); %kg*  *%defining equations*  *N=M/nt;*  *Ff=Mr\*N\*td;*  *Tmin=Ff\*r;*  *%Interactive Program Creation*  *prompt3='What is the total mass of the vehicle?';*  *CarMass=input(prompt3);*  *CarN=CarMass/nt;*  *CarFf=Mr\*CarN\*td;*  *CarTmin=CarFf\*r;*  *%Making a Graph*  *plot(M,TMin,'k',CarMass,CarTmin,'M\*');*  *%Labeling The Graph*  *xlabel('Mass (kg)');*  *ylabel('Torque (NM)');*  *title('Mass Vs. Torque Graph for Solar Car');*  *legend('Torque and Mass Linear Expression','Torque of Total Car Mass')* |
| **Minimum Torque Required vs. Mass** *(insert plot here)* |
| **Torque Required to Move (Tmin) for your configuration = 13.6998 N-m** |

**Questions of Evaluation**

*(Don’t answer these questions, they are meant to help you if you’re stuck)*

**Matlab Debugging:**

* Are you working in your script file (the Editor) or the Command window?
* Did you remember to clear your workspace (clear all), command window (clc), and close all figure windows (close all) before your code begins running?
* If an error occurs, what line does it say (in Command Window) it had a problem on?
* Can you decipher the error message?
* Did you meet the MATLAB name requirements for the script file?
* Did you remember to precede comments with %?
* If Matlab does not recognize a variable name, did you define it (put a value into this variable) before that point in your code?
* Remember that Matlab variables are case sensitive
* For plotting functions, you need to create a vector of values for your independent variable
* If you are trying to multiply two vectors together (element-by-element) did you remember to use a “.” before the operator (.\*, .^, ./, etc.)?
* If the plot is being overwritten, did you include a figure command before the plot command to tell it to make a new figure window?
* Did you remember to put the plot style in single quotes (ex. ‘r-‘, b’--‘, etc.)
* Did you remember to put text in single quotes inside of the legend command?
* Did you remember to include a ; after the command if you do not want the output to appear in the Command Window (or not have it if you do want it to be displayed)?
* Did you convert to all the correct units required for each equation?

**Note:** if you forget how to use a command, you can go back to the slides provided on Canvas from the lecture or use the ‘doc’ and ‘help’ commands (ex. help plot or doc plot).