EGR 103L – Fall 2016

Laboratory 2 - Introduction to MATLAB

CEMAL YAGCIOGLU (cy111)
Lab Section 05, WEDNESDAY 11:45 AM - 2:35 PM
SEPTEMBER 18, 2016

I understand and have adhered to all the tenets of the Duke Community Standard in completing every part of this assignment. I understand that a violation of any part of the Standard on any part of this assignment can result in failure of this assignment, failure of this course, and/or suspension from Duke University.

Contents

List of Figures

1 Introduction

My Matlab program loads the data for Beam1, and copies each column into new variables. It converts the Mass(kg) data to Force(N) by multiplying it by 9.81(N/kg). It converts the Displacement data in inches to meters by multiplying it by (2.54/100). With polyfit command, it finds first order fit polynomials. Then it generates predictions by creating 100 representational Force values, and calculating Displacement predictions. It plots Displacement as a function of Force and the plot values on the same graph. Finally it labels graph with appropriate titles, and saves the graph. It goes through the same procedure to Beam2 and Beam3 data. We are expected to observe these three data to see which graph fits most to the Displacement as a function of Force graph to conclude which Beam acts like a spring(acts according to Hooke's Law).

2 Data Obtained

The three data sets from the experiments are presented in Table 1.

Beam1.dat	
Mass (kg)	**Disp.** (in)
0.0000000e+00	1.0506442e-01
2.4969314e-01	1.5393050e-01
4.9938629e-01	9.1829900e-02
7.4907943e-01	1.9121739e-01
9.9877258e-01	3.3852003e-01
1.2484657e+00	6.4140472e-01
1.4981589e+00	1.1599771e+00
1.7478520e+00	1.8051665e+00
1.9975452e+00	2.6829435e+00
2.2472383e+00	3.6913222e+00
2.4969314e+00	5.1911317e+00

Beam2.dat	
Mass (kg)	**Disp.** (in)
0.0000000e+00	2.8577720e-03
2.4018292e-01	7.7785719e-01
4.8036585e-01	1.3496902e+00
7.2054877e-01	1.6343995e+00
9.6073170e-01	2.2155730e+00
1.2009146e+00	2.6345051e+00
1.4410975e+00	3.1922596e+00
1.6812805e+00	3.5522297e+00
1.9214634e+00	3.8456644e+00
2.1616463e+00	4.6163926e+00

Beam3.dat	
Mass (kg)	**Disp.** (in)
0.0000000e+00	2.5207794e-03
3.4636619e-02	4.6555129e-02
6.9273237e-02	8.9037799e-02
1.0390986e-01	1.3197719e-01
1.3854647e-01	1.7271271e-01
1.7318309e-01	2.1775915e-01
2.0781971e-01	2.4015748e-01
2.4245633e-01	2.4015748e-01
2.7709295e-01	2.4015748e-01

Table 1: Data from Three Beam Experiments

3 Calculation Results

A first-order polynomial fitting algorithm determined that the coefficients given in Table 2 produce the best-fit of the data to a straight line.

Data File	Compliance (m/N)	Init. Disp. (m)
`Beam1.dat`	4.8456e-03	-2.2280e-02
`Beam2.dat`	5.1680e-03	5.7105e-03
`Beam3.dat`	2.3913e-03	6.4745e-04

Table 2: Table of Compliances and Initial Displacement Values

4 Conclusions

For all three data we obtained a graph, a compliance value and a initial displacement value. Compliance value and Initial Displacement value was used to graph of data according to Hooke's law. Data for Beam2 was the most fitting to its own Hooke's Law graph. Thus, Beam2 acted most like a usual spring. The other two did not match well to the equation that modeled the force displacement relationship, therefore Beam1 and Beam3 did not act like a spring.

A Codes

A.1 DoneBeam1.m

```
%% Initialize the workspace
% Clear all variables
clear
% Change display to short exponential format
format short e

%% Load and manipulate the data
% Load data from Beam1.dat
load Beam1.dat
% Copy data from each column into new variables
Mass = Beam1(:,1);
Displacement = Beam1(:,2);
% Convert Mass to a Force measurement
Force = Mass*9.81;
% Convert Displacement in inches to meters
Displacement = (Displacement*2.54)/100;

%% Perform calculations
% Use polyfit to find first-order fit polynomials
P = polyfit(Force, Displacement, 1)

%% Generate predictions
% Create 100 representational Force values
ForceModel = linspace(min(Force),max(Force),100);
% Calculate Displacement predictions
DispModel = polyval(P, ForceModel);

%% Generate and save plots
% Bring up a figure window
figure(1)
% Clear the figure window
clf
% Plot Displacement as a function of Force
plot(Force, Displacement, 'ko')
% Turn hold on, plot the model values, and turn hold off
hold on
plot (ForceModel, DispModel, 'k-')
hold off
% Turn the grid on
grid on
% Label and title the graph
xlabel('Force (Newtons)')
ylabel('Displacement (meters)')
title('Displacement vs. Force for Beam1.dat (cy111)')
% Save the graph to PostScript
print -deps RunCanPlot
```

A.2 DoneBeam2.m

```
%% Initialize the workspace
% Clear all variables
clear
% Change display to short exponential format
format short e

%% Load and manipulate the data
% Load data from Beam2.dat
load Beam2.dat
% Copy data from each column into new variables
Mass = Beam2(:,1);
Displacement = Beam2(:,2);
% Convert Mass to a Force measurement
Force = Mass*9.81;
% Convert Displacement in inches to meters
Displacement = (Displacement*2.54)/100;

%% Perform calculations
% Use polyfit to find first-order fit polynomials
P = polyfit(Force, Displacement, 1)

%% Generate predictions
% Create 100 representational Force values
ForceModel = linspace(min(Force),max(Force),100);
% Calculate Displacement predictions
DispModel = polyval(P, ForceModel);

%% Generate and save plots
% Bring up a figure window
figure(1)
% Clear the figure window
clf
% Plot Displacement as a function of Force
plot(Force, Displacement, ’ko’)
% Turn hold on, plot the model values, and turn hold off
hold on
plot (ForceModel, DispModel, ’k-’)
hold off
% Turn the grid on
grid on
% Label and title the graph
xlabel(’Force (Newtons)’)
ylabel(’Displacement (meters)’)
title(’Displacement vs. Force for Beam2.dat (cy111)’)
% Save the graph to PostScript
print -deps RunCanPlot
```

A.3 DoneBeam3.m

```
%% Initialize the workspace
% Clear all variables
clear
% Change display to short exponential format
format short e

%% Load and manipulate the data
% Load data from Beam3.dat
load Beam3.dat
% Copy data from each column into new variables
Mass = Beam3(:,1);
Displacement = Beam3(:,2);
% Convert Mass to a Force measurement
Force = Mass*9.81;
% Convert Displacement in inches to meters
Displacement = (Displacement*2.54)/100;

%% Perform calculations
% Use polyfit to find first-order fit polynomials
P = polyfit(Force, Displacement, 1)

%% Generate predictions
% Create 100 representational Force values
ForceModel = linspace(min(Force),max(Force),100);
% Calculate Displacement predictions
DispModel = polyval(P, ForceModel);

%% Generate and save plots
% Bring up a figure window
figure(1)
% Clear the figure window
clf
% Plot Displacement as a function of Force
plot(Force, Displacement, 'ko')
% Turn hold on, plot the model values, and turn hold off
hold on
plot (ForceModel, DispModel, 'k-')
hold off
% Turn the grid on
grid on
% Label and title the graph
xlabel('Force (Newtons)')
ylabel('Displacement (meters)')
title('Displacement vs. Force for Beam3.dat (cy111)')
% Save the graph to PostScript
print -deps RunCanPlot
```

B Figures

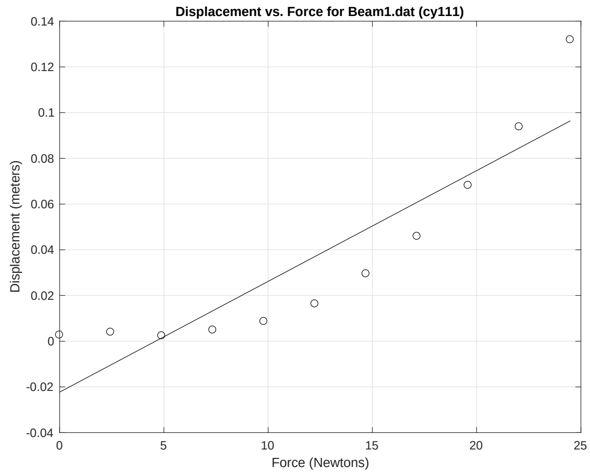

Figure 1: Displacement vs. Force for Beam 1

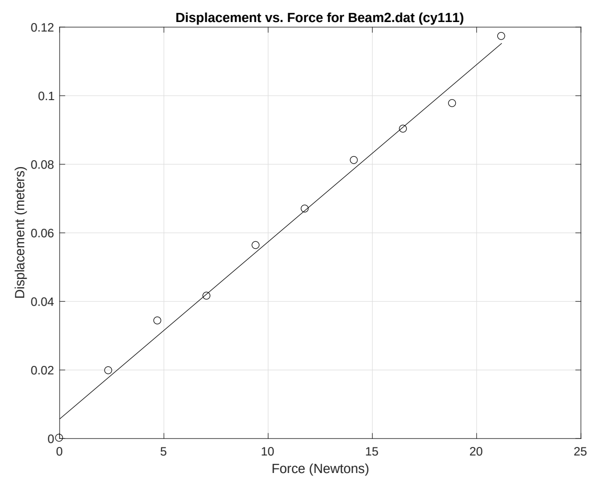

Figure 2: Displacement vs. Force for Beam 2

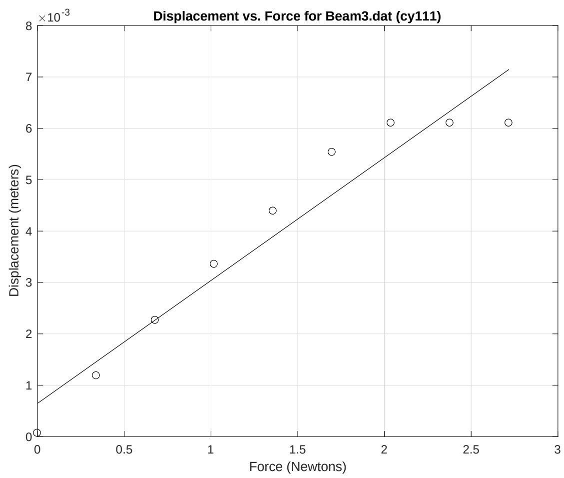

Figure 3: Displacement vs. Force for Beam 3