EGR 103L – Fall 2017

Laboratory 2 - Introduction to MATLAB Solutions

Ian Hanus (ih52)
Lab Section 1, Tuesday 8:30-11:30
September 17, 2017

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

The program created takes inputs of mass in kilograms and converts it into force in Newtons by multiplying by 9.81. It then takes inputs of displacement in inches and converts these to displacement in meters. The program then plots the points with the force as the independent variable and the corresponding displacements as the dependent variable. A best fit line for a first order polynomial is found and graphed along with the data points. Then 100 representational force values are created between the minimum and maximum forces and used to predict displacement. Next the slope and y-intercept of the best fit line are measured which, according to the equation Displacement = Compliance∗Force + Initial Displacement, means compliance is equal to the slope and initial displacement is equal to the y-intercept. Then, looking at the graphs for each data set, one can tell if the line fits the data well which determines if the beam is acting as a spring.

2 Data Obtained

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

Beam1.dat	
Mass (kg)	**Disp.** (in)
0	4.0908e-01
2.4018e-01	5.1145e-01
4.8037e-01	1.1085e+00
7.2055e-01	1.6810e+00
9.6073e-01	2.2016e+00
1.2009e+00	2.5752e+00
1.4411e+00	3.0958e+00
1.6813e+00	3.5045e+00
1.9215e+00	4.1159e+00
2.1616e+00	4.4975e+00

Beam2.dat	
Mass (kg)	**Disp.** (in)
0	1.4446e-01
2.4969e-01	4.5522e-02
4.9939e-01	1.1184e-01
7.4908e-01	2.0433e-01
9.9877e-01	4.3890e-01
1.2485e+00	7.1716e-01
1.4982e+00	1.2029e+00
1.7479e+00	1.7899e+00
1.9975e+00	2.6526e+00
2.2472e+00	3.7465e+00
2.4969e+00	5.0663e+00

Beam3.dat	
Mass (kg)	**Disp.** (in)
0	6.8650e-04
3.4637e-02	4.3857e-02
6.9273e-02	8.7320e-02
1.0391e-01	1.2922e-01
1.3855e-01	1.7391e-01
1.7318e-01	2.1621e-01
2.0782e-01	2.4016e-01
2.4246e-01	2.4016e-01
2.7709e-01	2.4016e-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`	5.1369e-03	5.7334e-03
`Beam2.dat`	4.8049e-03	−2.1624e-02
`Beam3.dat`	2.4163e-03	5.8703e-04

Table 2: Table of Compliances and Initial Displacement Values

4 Conclusions

In a spring, force is directly proportional to displacement. The results of the programs are the slope (compliance) and y-intercept (initial displacement) of the best fit line for the data given. As the best fit line is first order, if the line describes the data well then that data is also linear. Because the only data that is accurately represented by its best fit line is the data of Beam 1, the only beam that acts like a spring is Beam 1. The best fit lines of Beams 2 and 3 do not accurately represent the data from Beams 2 and 3, so Beams 2 and 3 do not act like springs.

A Codes

A.1 RunBeam1.m

```
 % RunBeam1.m
 % [Ian Hanus]
 % September 6, 2017
 % Based on: RunCam.m
 % Written by: Michael R. Gustafson II (mrg@duke.edu)

 % I have adhered to all the tenets of the
 % Duke Community Standard in creating this code.
 % Signed: [ih52]

%% 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;

%% 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')

%% 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);
grid('on')
%% Generate and save plots
```

```
%Turn hold on, plot the model values, and turn hold off
hold on
plot (ForceModel, DispModel, 'k-')
hold off

% Label and title the graph
xlabel('Force (Newtons)')
ylabel('Displacement (meters)')
title('Displacement vs. Force for Beam1.dat (ih52)')

% Save the graph to PostScript
print -deps Beam1Plot
```

A.2 RunBeam2.m

```
 % RunBeam1.m
 % [Ian Hanus]
 % September 6, 2017
 % Based on: RunCam.m
 % Written by: Michael R. Gustafson II (mrg@duke.edu)

 % I have adhered to all the tenets of the
 % Duke Community Standard in creating this code.
 % Signed: [ih52]

%% 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 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;

%% 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')

%% 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);
grid('on')
%% Generate and save plots
%Turn hold on, plot the model values, and turn hold off
hold on
```

```
plot (ForceModel, DispModel, 'k-')
hold off

% Label and title the graph
xlabel('Force (Newtons)')
ylabel('Displacement (meters)')
title('Displacement vs. Force for Beam2.dat (ih52)')

% Save the graph to PostScript
print -deps Beam2Plot
```

A.3 RunBeam3.m

```
 % RunBeam1.m
 % [Ian Hanus]
 % September 6, 2017
 % Based on: RunCam.m
 % Written by: Michael R. Gustafson II (mrg@duke.edu)

 % I have adhered to all the tenets of the
 % Duke Community Standard in creating this code.
 % Signed: [ih52]

%% 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 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;

%% 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')

%% 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);
grid('on')
%% Generate and save plots
%Turn hold on, plot the model values, and turn hold off
hold on
```

```
plot (ForceModel, DispModel, 'k-')
hold off

% Label and title the graph
xlabel('Force (Newtons)')
ylabel('Displacement (meters)')
title('Displacement vs. Force for Beam3.dat (ih52)')

% Save the graph to PostScript
print -deps Beam3Plot
```

B Figures

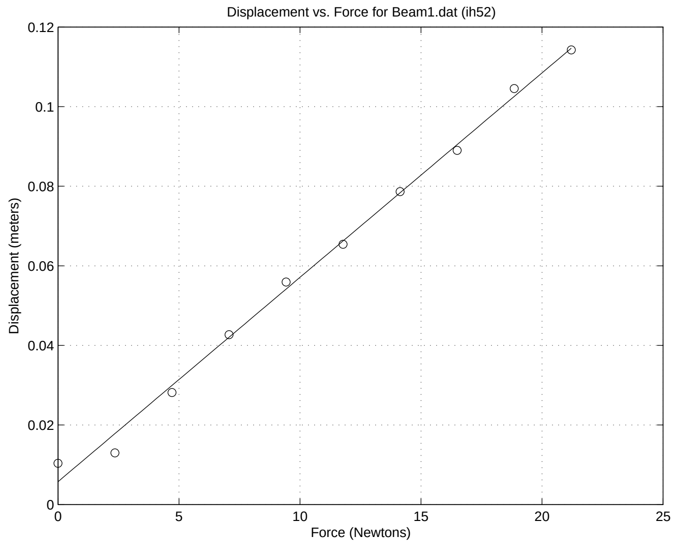

Figure 1: Displacement vs. Force for Beam 1

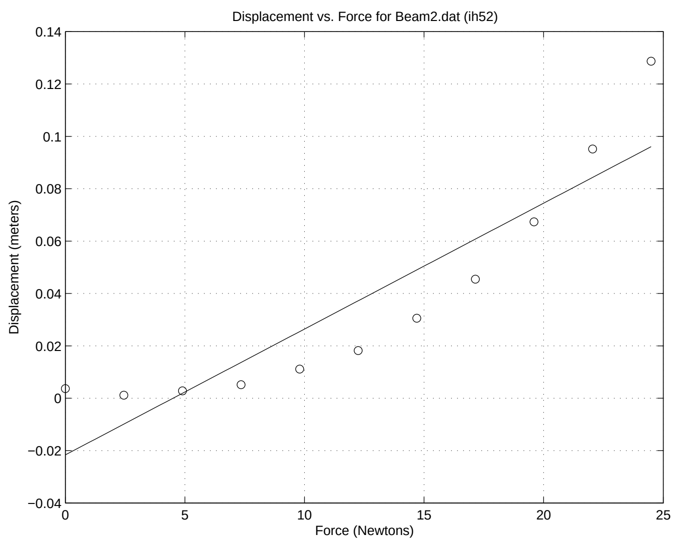

Figure 2: Displacement vs. Force for Beam 2

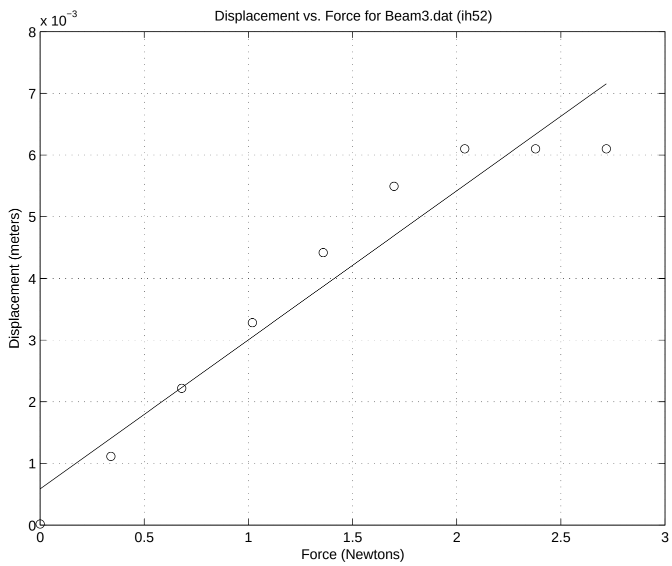

Figure 3: Displacement vs. Force for Beam 3