EGR 103L – Fall 2019

Laboratory 2 - Introduction to Python

Marcus Deans (md374)
Lab Section 2, Tuesdays 11:45-2:35
15 September 2019

Contents

List of Figures

1 Introduction

This program displays information gathered from physical experiments regarding the relationship between force and displacement in several beams. Moreover, the experience of creating this program is designed to enhance understanding of basic Python principles. Specifically relating to springs, a model was formed based on a first-order polynomial, intended to replicate the $F = -kx$ equation that represents the relationship between spring force and displacement. Lessons learned include:

- How to create, customize, and edit graphics in Python
- How to save graphics for usage elsewhere
- How to import Python resources into a LaTeX report

2 Data Obtained

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

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

Beam2.dat	
Mass	**Disp.**
(kg)	(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
1.2485	7.1716e-01
1.4982	1.2029e
1.7479	1.7899
1.9975	2.6526
2.2472	3.7465
2.4969	5.0663

Beam3.dat	
Mass	**Disp.**
(kg)	(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`	$5.1369e - 3$	$5.7333e - 3$
`Beam2.dat`	$4.8048e - 3$	$-2.1624e - 2$
`Beam3.dat`	$2.4163e - 3$	$5.8703e - 4$

Table 2: Table of Compliances and Initial Displacement Values

4 Conclusions

In conclusion, the previously measured values of displacement and the corresponding resultant force for three different beams was used to create graphical representations of the relationship for each beam. Calculations were performed in order to represent the relationship, with the graphical format providing a more accessible qualitative evaluation of the correlation between displacement and force for each beam.

Each beam appeared to, in effect, behave similarly to a spring, at least for small displacements as observed by the compliance with the graph. Beam 2 and Beam 3 were both found to have substantially higher deviation from the line of best fit compared to that of Beam 1. This may be attributable to the higher displacement that may have caused the beam to no longer act like a spring, thus no longer complying with the previously established line of best fit that held relatively true at lower displacements.

A Codes

A.1 run_beam1.m

```
"""
[Beam 1 File]
[Marcus Deans]
[3 September 2019]
Based on: RunCan.py
Written by: Michael R. Gustafson II

I understand and have adhered to all the tenets of the Duke Community Standard
in creating this code.
Signed: [md374]
"""
# %% Import modules
import numpy as np
import matplotlib.pyplot as plt

# %% Load and manipulate data
# Load data from Beam1.dat
beam_data = np.loadtxt('Beam1.dat')
# Copy data from each column into new variables
mass = beam_data[:,0].copy()
disp = beam_data[:,1].copy()
# Convert mass in kilograms to force in Newtons
force = mass * 9.81
# Convert displacement in inches to metres
disp = (disp * 2.54) / 100

# %% Perform calculations
# Use polyfit to get coefficients
p = np.polyfit(force, disp, 1)
print(p)

# %% Generate3 predictions
#Create 100 representational forces
force_model = np.linspace(min(force), max(force), 100)
# Calculate displacement predictions
disp_model = np.polyval(p, force_model)

# %% Generate and save
# Create a figure and axes
fig, ax = plt.subplots(num=1, clear=True)
ax.plot(force, disp, 'ko')
ax.plot(force_model, disp_model, 'k-')
ax.grid(True)

ax.set_xlabel('Force (Newtons)')
ax.set_ylabel('Displacement (metres)')
ax.set_title('Displacement vs. Force for Beam1.dat (md374)')

fig.tight_layout()

fig.savefig('Beam1Plot.eps')
fig.savefig('Beam1Plot.pdf')
```

A.2 run_beam2.m

```
"""
[Beam 2 File]
[Marcus Deans]
[3 September 2019]
Based on: RunCan.py
Written by: Michael R. Gustafson II

I understand and have adhered to all the tenets of the Duke Community Standard
in creating this code.
Signed: [md374]
"""
# %% Import modules
import numpy as np
import matplotlib.pyplot as plt

# %% Load and manipulate data
# Load data from Beam1.dat
beam_data = np.loadtxt('Beam2.dat')
# Copy data from each column into new variables
mass = beam_data[:,0].copy()
disp = beam_data[:,1].copy()
# Convert mass in kilograms to force in Newtons
force = mass * 9.81
# Convert displacement in inches to metres
disp = (disp * 2.54) / 100

# %% Perform calculations
# Use polyfit to get coefficients
p = np.polyfit(force, disp, 1)
print(p)

# %% Generate3 predictions
#Create 100 representational forces
force_model = np.linspace(min(force), max(force), 100)
# Calculate displacement predictions
disp_model = np.polyval(p, force_model)

# %% Generate and save
# Create a figure and axes
fig, ax = plt.subplots(num=1, clear=True)
ax.plot(force, disp, 'ko')
ax.plot(force_model, disp_model, 'k-')
ax.grid(True)

ax.set_xlabel('Force (Newtons)')
ax.set_ylabel('Displacement (metres)')
ax.set_title('Displacement vs. Force for Beam2.dat (md374)')

fig.tight_layout()

fig.savefig('Beam2Plot.eps')
fig.savefig('Beam2Plot.pdf')
```

A.3 run_beam3.m

```
"""
[Beam 3 File]
[Marcus Deans]
[3 September 2019]
Based on: RunCan.py
Written by: Michael R. Gustafson II

I understand and have adhered to all the tenets of the Duke Community Standard
in creating this code.
Signed: [md374]
"""
# %% Import modules
import numpy as np
import matplotlib.pyplot as plt

# %% Load and manipulate data
# Load data from Beam1.dat
beam_data = np.loadtxt('Beam3.dat')
# Copy data from each column into new variables
mass = beam_data[:,0].copy()
disp = beam_data[:,1].copy()
# Convert mass in kilograms to force in Newtons
force = mass * 9.81
# Convert displacement in inches to metres
disp = (disp * 2.54) / 100

# %% Perform calculations
# Use polyfit to get coefficients
p = np.polyfit(force, disp, 1)
print(p)

# %% Generate3 predictions
#Create 100 representational forces
force_model = np.linspace(min(force), max(force), 100)
# Calculate displacement predictions
disp_model = np.polyval(p, force_model)

# %% Generate and save
# Create a figure and axes
fig, ax = plt.subplots(num=1, clear=True)
ax.plot(force, disp, 'ko')
ax.plot(force_model, disp_model, 'k-')
ax.grid(True)

ax.set_xlabel('Force (Newtons)')
ax.set_ylabel('Displacement (metres)')
ax.set_title('Displacement vs. Force for Beam3.dat (md374)')

fig.tight_layout()

fig.savefig('Beam3Plot.eps')
fig.savefig('Beam3Plot.pdf')
```

B Figures

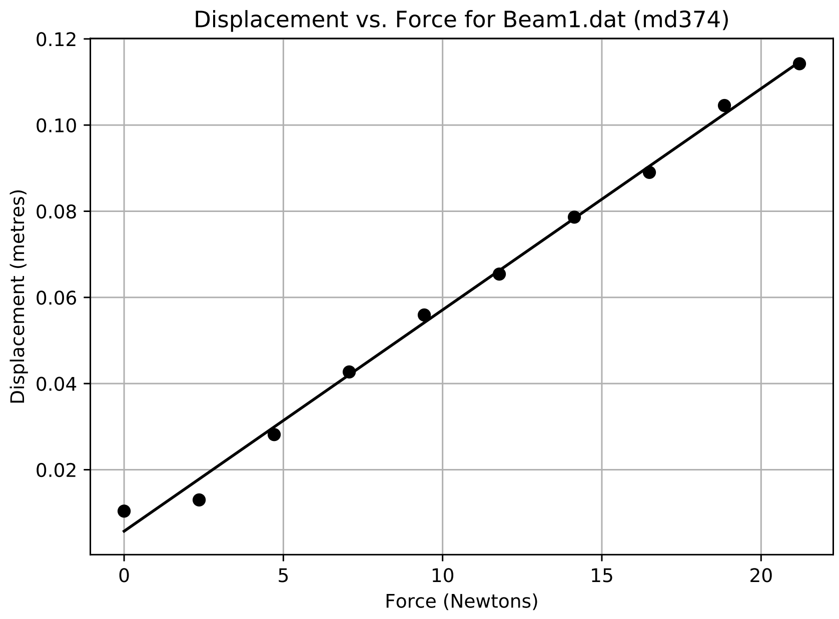

Figure 1: Displacement vs. Force for Beam 1

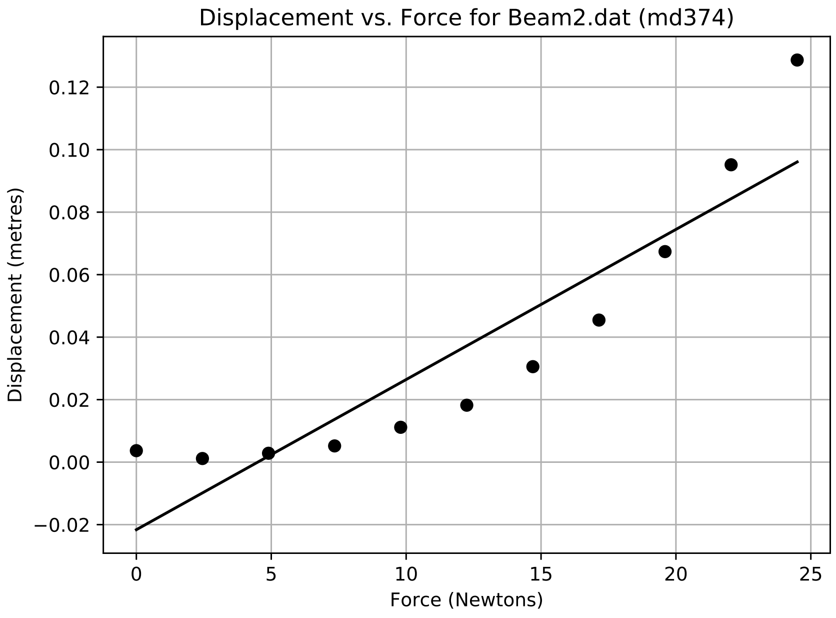

Figure 2: Displacement vs. Force for Beam 2

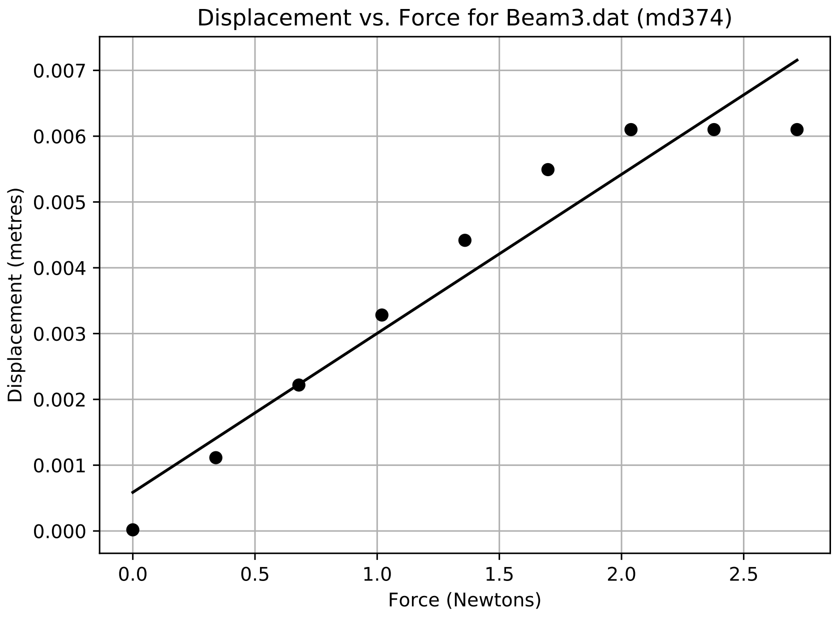

Figure 3: Displacement vs. Force for Beam 3