

FYS2150

Lab Report: Elasticity

Nicholas Karlsen

April 12, 2018

Abstract

Determining the Young's modulus of a brass rod by deflection and listening for the root frequency.

1 Introduction

2 Theory

2.1 Euler-Bernoulli beam theory

From the Euler-Bernoulli beam theory, it follows that the deflection of a beam supported by two points of distance l and a load mg halfway between the two knives is given by

$$h(m) = \frac{mgl^3}{48EI} \quad (1)$$

Where E denotes the Young's modulus of the beam and I the second moment area given by

$$I = \frac{\pi d^4}{4 \cdot 2^4} \quad (2)$$

Where d denotes the diameter of the beam.

Further, Eqn. 1 can be rewritten in the following way

$$E = \frac{4l^3g}{3\pi|A|d^4} \quad (3)$$

Where $A \equiv h(m)/m$, which can be obtained as the gradient from a linear fit on the data gathered when varying the load subjected and on the beam and recording the resulting deflection. Which gives the following relationship

$$h(m) = Am + B \quad (4)$$

2.2 Errors

When performing arithmetic operations on recorded data, the uncertainty in the data must also carry over to the derived results. How these uncertainties carry over in different operations can be found in Practical Physics [1].

3 Experimental Procedure

3.1 Three-point flexural test

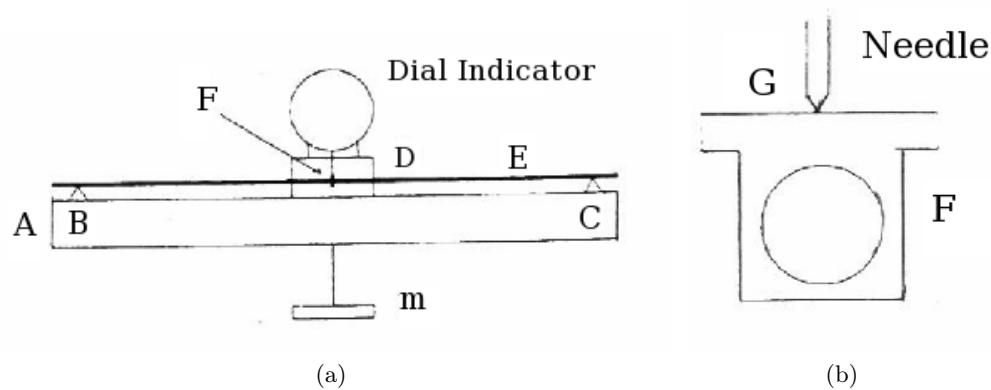


Figure 1: (a) shows the apparatus used for measuring the deflection of a rod and (b) a cross section of the apparatus at point F.

Using Fig. 1a as a reference; The brass rod, A, was laid on the "knives" B and C, in order to ensure that the dial gauge, manufactured by Baker¹, was positioned halfway between B and C, we measured the distance from the dial gauge to B and C using a measuring tape of type Lufkin pee wee 2m Y612CM with uncertainty $\pm 0.1\text{cm}$, adjusting the rod such that the difference in measurements would be sufficiently small. At the middle of the rod, there was a ring attached, as shown in Fig. 1b. The flat surface of the ring was in contact with the needle of the dial gauge at G. In order to ensure that the flat surface of the ring was at right angle with the needle, we turned the rod such that the reading of the dial gauge would be at a minimum, as the skewer the surface, the greater the reading. This process was repeated at the start of every attempt of the experiment.

After having prepared the apparatus, three masses of roughly 0.5g, 1kg and 2kg which we denoted m_a , m_b and m_c respectively. They were placed carefully in the tray denoted

¹I did not take note of the model number of the particular dial gauge that was used during the lab. While working on this report, i have become aware that each Baker dial gauge is individually calibrated. Therefore, i have no values for the instrumental error in the deflection measurements.

m in Fig. 1a, in different combinations so that we would get readings for the deflection of the rod at $\approx \{0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$ kg, recorded by reading the dial gauge.

Due to seemingly disturbing the system significantly when adding masses, we were worried that there might be a significant systematic error in the experiment. So we opted to repeat the readings in this experiment several times in order to investigate if the data in the later readings (when the system had been disturbed multiple times in succession) had an increase in its deviation.

Lastly, the distance between the knives, $l_{B,C}$, was measured using the measuring tape and a micrometer of type Moore & Wright 1965 MI with uncertainty ± 0.01 mm. The measuring tape was used to measure the distance between the outer edges of the "knives" at B and C, $l_{B\ outer}, l_{C\ outer}$. The micrometer was then used to measure the knives thickness l_{knife} , which was needed as the contact points between the rod and the knives are (assumed) at the middle of the identical knives. Since there are two contact points, $l_{B\ outer, C\ outer} - l_{knife} = l_{B,C}$

3.2 Measuring the speed of sound in the rod

The brass rod, with a ring attached to it (same as before), was laid to rest on the flat side of the ring on a solid surface such that the rod is held up by the ring, and nothing else. We also made sure that the rod was not to be disturbed in any way while it was vibrating. When hit with a hammer, it will emit a sound consisting of different frequencies. Following are the two different methods we used for determining the root frequency of the rod. During both experiments, we ensured there were no significant noise pollution during our recording (By which i mean people performing the same experiment as us).

3.2.1 By hearing for beats

A speaker was connected to a signal generator. We started the signal generator at 1200Hz and hit the brass rod with a plastic hammer on the the flat surface on one end of the rod. By ear, there was an audible beat due to the superposition of the two signals. We adjusted the signal generator such that the the frequency of the beat was minimized, and there was essentially no audible difference between the two signals. We did this by trying above and below where we thought the root frequency was, eventually zeroing in on a value.

3.2.2 By Fourier transform

A USB microphone was placed close to the rod, and faced towards it. The microphone was connected to a computer running matlab, with a script that collects audio data from it and Fourier transforms it using FFT. The recordings made were made with a sampling frequency of 8×1024 Hz and varying durations. As before, we hit the rod using a plastic hammer and recorded the data. A total of 7 recordings were made.

3.3 Other measurements

3.3.1 Mass

In order to accurately measure the mass of the rough loads and the rod, the balance scale (Ohaus triple beam balance) which we used had to be calibrated. We did this by weighing a set of three reference weight on the scale, and comparing their measured value to the measured value of the rough loads and the rod using a linear fit. When placing the masses on the scale, we made sure to position the masses in the center of the scale plate and not take a reading until the needle of the balance scale was not sufficiently stable.

3.3.2 Length and thickness of the rod

The length of the rod was measured using the measuring tape, and the thickness using the micrometer. In order to accurately determine the thickness, accounting for any irregularities in the rod due to deformation etc. The thickness was measured several times in different places on the rod, so that we could calculate the mean thickness.

4 Results

4.1 Length and mass measurements

Table 1: Mass of rough load and reference

Stated mass	Measured reference	Measured rough
500g	500.0g	500.1g
1000g	999.9g	1000.3g
2000g	2000.1g	2000.5g

4.2 Results from Three-point flexural test

Table 2: Deflection of rod

Attempt no.	h(0kg) [mm]	h(0.5kg) [mm]	h(1kg) [mm]	h(1.5kg) [mm]	h(2.0kg) [mm]	h(2.5kg) [mm]	h(3.0kg) [mm]	h(3.5kg) [mm]
1	9.44	8.72	8.00	7.28	6.58	5.84	5.15	4.43
2	9.42	8.70	7.98	7.26	6.53	5.80	5.09	4.39
3	9.42	8.71	7.98	7.26	6.53	5.80	5.09	4.37
4	9.41	8.69	7.97	7.25	6.52	5.79	5.08	4.36
5	9.42	8.70	7.98	7.26	6.70	5.87	5.19	4.51

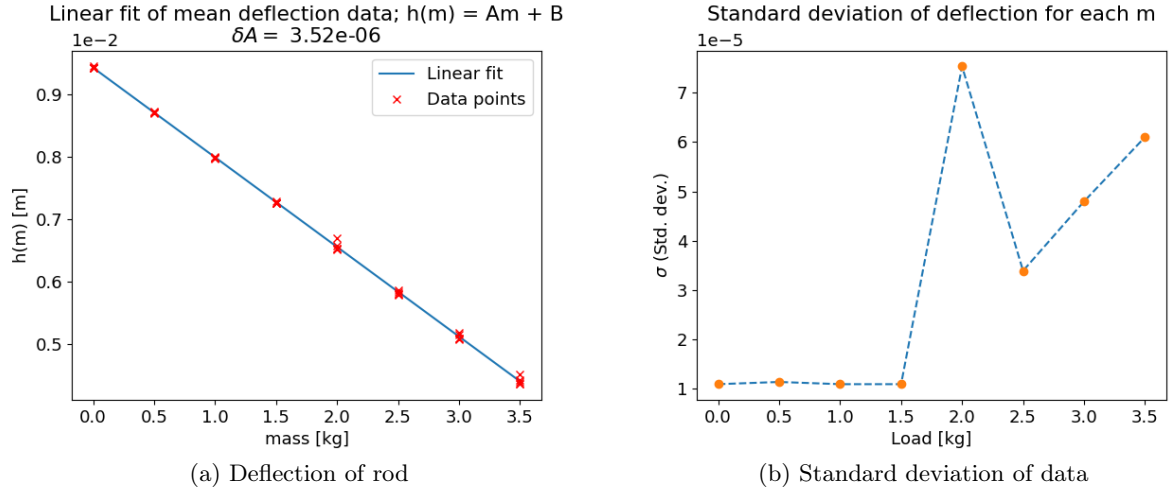


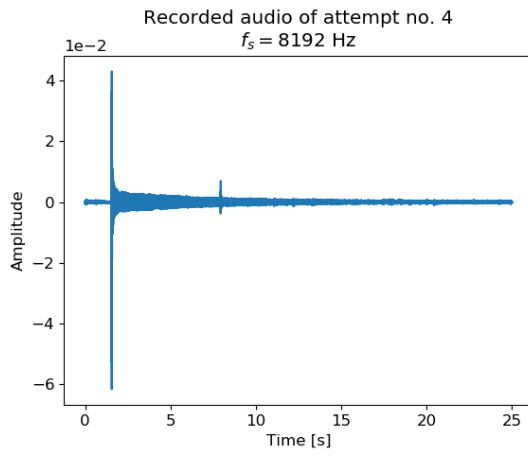
Figure 2: (a) Shows the deflection of the brass rod measured by the dial gauge. (b) Shows the standard deviation of the data points in (a) at their respective masses

Table 2 contains the deflection data recorded with the dial gauge where the loads listed are from the rough, uncalibrated masses. Their corrected value is listed in table 1.

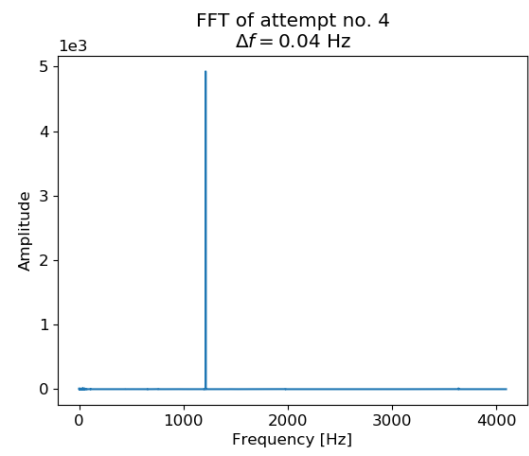
Fig. 2a contains all the recorded data, as well as a linear fit on the mean deflection for each load using corrected values for the mass, m . The error of the linear fit, $h(m) = Am + B$, $\delta A = 3.52e - 06$. Fig. 2b contains the standard deviation of the deflection values for each load.

4.3 Results from measuring the speed of sound in the rod

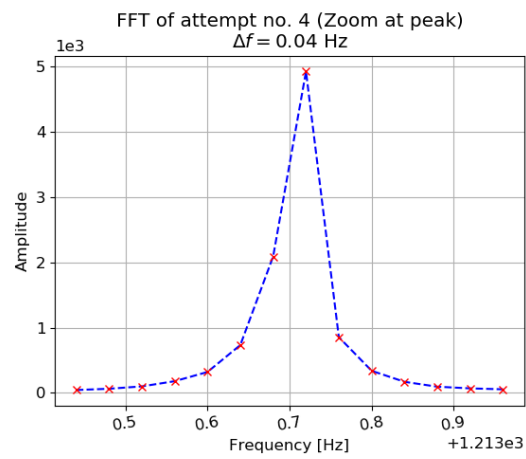
When hearing for beats, me and my lab-partner decided that the root frequency was ≈ 1240 Hz.



(a) Time domain



(b) Frequency domain



(c) Zoomed frequency domain

Figure 3: All of the plots generated for attempt no. 4

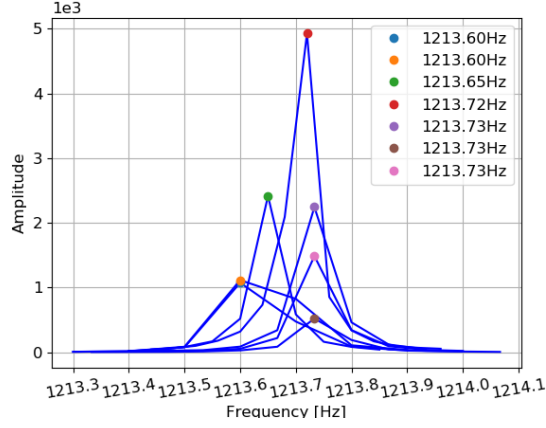


Figure 4: Zoomed frequency plot for all 7 attempts.

Fig. 3 contains the data and derived results from our fourth attempt of the experiment. We performed a total of 7 attempts, all of which yielded in similar results to attempt no. 4. The data yielded from all of the attempts is summarized in Fig. 4 which shows the peaks in the frequency domain in one plot. Table 3 contains all of the relevant numbers related to each attempt, where f denotes the root frequency, Δf the resolution of the frequency domain, t the time of the recording and f_s the sampling frequency.

Table 3: FFT data

Attempt no.	f [Hz]	Δf [Hz]	t [s]	f_s [Hz]
1	1213.60	0.10	10	8192
2	1213.60	0.10	10	8192
3	1213.65	0.05	20	8192
4	1213.72	0.04	25	8192
5	1213.72	0.04	25	8192
6	1213.72	0.07	15	8192
7	1213.73	0.07	15	8192

5 Discussion

Note STD.DEV of deflection increases with m (system is disturbed). Assume the disturbance is normally distributed, therefore error \rightarrow given by STD. DEV.

6 Conclusion

References

[1] G. L. Squires. *Practical Physics 4th Edition*. Cambridge University Press, 2001.

*

A Code

All of the code used to produce this report. Anything noteworthy should already be mentioned in the main body of the report. Note that when this code was written, readability was not a huge concern, so some of it may not be very easy to interpret.

scripts/FFTlyd.py

```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 """
4 Generates the same figures as FFTlyd.m
5 author: Nicholas Karlsen
6 """
7 import scipy.io as sio
8 import matplotlib.pyplot as plt
9 import numpy as np
10
11
12 # Sets font size of matplot
13 plt.rcParams.update({'font.size': 12})
14
15
16 def import_matlab(filename):
17     # Opens .mat file
18     mfile = sio.loadmat(filename)
19     # Fetches data
20     data = mfile.get("data")
21     energi = mfile.get("energi")
22     fut = mfile.get("fut")
23     L = mfile.get("L")
24     t = mfile.get("t")
25
26     return data, energi, fut, L, t
27
28
29 rel_path = "data/"
```



```

30 n = 1
31 mat_file = "forsok%i.mat" % n
32
33
34 def raw_fig(filename):
35     data, energi, fut, L, t = import_matlab(filename)
36     plt.plot(t, data)
37     plt.xlabel("Time [s]")
38     plt.ylabel("Amplitude")
39     plt.ticklabel_format(style='sci', axis='y', scilimits=(0,0))
40
41
42 raw_fig(rel_path + "forsok1.mat")
43 plt.title("Recorded audio of attempt no. 1\n$f_s = 8192$ Hz")
44 plt.savefig("raw_exp2_1.png")
45 plt.close()
46
47 raw_fig(rel_path + "forsok4.mat")
48 plt.title("Recorded audio of attempt no. 4\n$f_s = 8192$ Hz")
49 plt.savefig("raw_exp2_4.png")
50 plt.close()
51
52
53 def figure1(filename):
54     data, energi, fut, L, t = import_matlab(filename)
55     fut = np.transpose(fut)
56     fh = int(len(energi) / 2.0) # half lenght of data
57     # Only plot first half of data, as FF mirrors in half-way point.
58     plt.plot(fut[:fh], energi[:fh])
59     plt.xlabel("Frequency [Hz]")
60     plt.ylabel("Amplitude")
61     plt.ticklabel_format(style='sci', axis='y', scilimits=(0,0))
62
63
64 figure1(rel_path + "forsok1.mat")
65 plt.title("FFT of attempt no. 1\n$\Delta f=0.10$ Hz")
66 plt.savefig("energy_exp2_1.png")
67 plt.close()
68
69 figure1(rel_path + "forsok4.mat")
70 plt.title("FFT of attempt no. 4\n$\Delta f=0.04$ Hz")
71 plt.savefig("energy_exp2_4.png")
72 plt.close()
73
74 eigenfreqs = []
75
76
77 def figure2(filename, style="-", cross=0):
78     data, energi, fut, L, t = import_matlab(filename)
79     fut = np.transpose(fut)
80
81     fh = int(len(energi) / 2.0) # half lenght of data
82     ipeak = np.argmax(energi[:fh])

```

```

83
84     eigenfreqs.append(fut[ipeak])
85
86     i = ipeak
87     while energi[i] > np.amax(energi[:fh]) * 0.01:
88         i -= 1
89
90     j = ipeak
91     while energi[j] > np.amax(energi[:fh]) * 0.01:
92         j += 1
93
94     plt.plot(fut[i:j], energi[i:j], color="blue", linestyle=style)
95     if cross == 1:
96         plt.plot(fut[i:j], energi[i:j], "rx")
97     else:
98         plt.plot(fut[ipeak], energi[ipeak], "o", label="%.2fHz" % fut[ipeak])
99
100     plt.grid("on")
101
102 figure2(rel_path + "forsok1.mat", style="—", cross=1)
103 plt.xlabel("Frequency [Hz]")
104 plt.ylabel("Amplitude")
105 plt.ticklabel_format(style='sci', axis='y', scilimits=(0,0))
106 plt.xticks(rotation=10)
107 plt.title("FFT of attempt no. 1 (Zoom at peak)\n$\Delta f=0.10$ Hz")
108 plt.savefig("freq_exp2_1.png")
109 plt.close()
110
111
112 figure2(rel_path + "forsok4.mat", style="—", cross=1)
113 plt.xlabel("Frequency [Hz]")
114 plt.ylabel("Amplitude")
115 plt.ticklabel_format(style='sci', axis='y', scilimits=(0,0))
116 plt.xticks(rotation=10)
117 plt.title("FFT of attempt no. 4 (Zoom at peak)\n$\Delta f=0.04$ Hz")
118 plt.savefig("freq_exp2_4.png")
119 plt.close()
120
121
122 for i in range(1, 8):
123     figure2(rel_path + "forsok%i.mat" % i)
124
125 plt.xlabel("Frequency [Hz]")
126 plt.ylabel("Amplitude")
127 plt.legend()
128 plt.ticklabel_format(style='sci', axis='y', scilimits=(0,0))
129 plt.xticks(rotation=10)
130 plt.savefig("freq_exp2_all.png")
131 plt.close()

```

```

1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 """
4 A collection of commonly used functions in FYS2150.
5 author: Nicholas Karlsen
6 """
7 import numpy as np
8
9
10 def stddev(x):
11     """
12     Finds the standard deviation, and standard deviation of
13     a 1D array of data x.
14     See. Eqn D. Page 24 squires
15     """
16     n = len(x)
17     sigma = np.sqrt((np.sum(x**2) - 1.0 / n * np.sum(x)**2) / (n - 1))
18     sigma_m = np.sqrt((np.sum(x**2) - 1.0 / n * np.sum(x)**2) / (n * (n -
19     1)))
20
21     return sigma, sigma_m
22
23 def linfit(x, y):
24     """
25     Finds the line of best-fit in the form y=mx+c given two
26     1D arrays x and y.
27     """
28     n = np.size(y)
29     D = np.sum(x**2) - (1.0 / n) * np.sum(x)**2
30     E = np.sum(x * y) - (1.0 / n) * np.sum(x) * np.sum(y)
31     F = np.sum(y**2) - (1.0 / n) * np.sum(y)**2
32
33     dm = np.sqrt(1.0 / (n - 2) * (D * F - E**2) / D**2)
34     dc = np.sqrt(1.0 / (n - 2) * (float(D) / n + np.mean(x)) *
35     ((D * F - E**2) / (D**2)))
36     m = float(E) / D
37     c = np.mean(y) - m * np.mean(x)
38
39     return m, c, dm, dc

```

scripts/lab_data.py

```

1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 """
4 Contains all of the data collected in the
5 Elasticity lab, module 2 of FYS2150
6 author: Nicholas Karlsen
7 """
8
9 from pylab import *
10 import scipy.constants as const

```

```

11 import FYS2150lib as fys
12
13
14 rcParams.update({'font.size': 13}) # Sets font size of plots
15
16 def weight_data(set=1):
17     "set decides which data set the function returns."
18     set = set.lower() # Forces lowercase
19     sets = ["masses", "rod"]
20     # Mass of weights measured with balance
21     m_a_balance = 500.1e-3
22     m_b_balance = 1000.3e-3
23     m_c_balance = 2000.5e-3
24
25     # Mass of reference weights
26     m_reference = array([0.5, 1.0, 2.0])
27     m_reference_balance = array([500.0e-3, 999.9e-3, 2000.1e-3]) # Weighed
28
29     # Using linear fit to correct for error in balance
30     a, b, da, db = fys.linfit(m_reference, m_reference_balance)
31     # Corrected masses
32     m_a = (m_a_balance - b) / a # approx 500g
33     m_b = (m_b_balance - b) / a # approx 1000g
34     m_c = (m_c_balance - b) / a # approx 2000g
35
36     if set == sets[0]: # Return corrected masses
37         return m_a, m_b, m_c
38
39     m_rod_ring = np.array([2482.7, 2482.5, 2482.1]) * 1e-3
40     m_ring = 34.4 * 1e-3 #kg
41     m_rod = (mean(m_rod_ring) - m_ring - b) / a #kg
42
43
44     if set == sets[1]:
45         return m_rod
46
47     if set not in sets:
48         print "Invalid set"
49         print "List of valid sets:", sets
50         print "exiting..."
51         exit()
52
53
54 def E_sound(f, L, d, M):
55     '''
56     Returns youngs modulus given
57     f = root frequency
58     L = lenght between knives
59     d = diameter of rod
60     M = mass of rod
61     '''
62     return (16.0 * M * L * f**2) / (np.pi * d**2)
63

```

```

64
65 def E_sound_error(E, sd, sf, sL, sM, d, f, L, M):
66     return E * np.sqrt((2 * sd / d)**2 + (2 * sf / f)**2 +
67                        (2 * sL / L)**2 + (2 * sM / M)**2)
68
69
70 d = np.array([15.98, 15.99, 15.99, 16.00,
71              15.99, 15.99, 15.98, 15.99,
72              15.99, 15.99]) * 1e-3
73 d_mean = np.mean(d)
74 d_err = np.sqrt(fys.stddev(d)[1]**2 + (0.01e-3)**2) # Std dev of mean +
              instrumentation error
75
76 f_root = 1213.72
77 f_err = 0.04 # resolution of FFT
78 L_err = 1e-3
79 M_err = 1e-3
80
81 l_BC_outer = 133.9 * 1e-2
82 l_knife_diameter = 4.09 * 1e-3
83 l_BC = l_BC_outer - l_knife_diameter
84
85 E_sound = E_sound(f=f_root,
86                  L=l_BC,
87                  d=d_mean,
88                  M=weight_data("rod"))
89
90 print "E from root f = %e" % E_sound
91
92 print "E_err root = %e" % E_sound_error(E=E_sound,
93                                         sd=d_err,
94                                         sf=f_err,
95                                         sL=L_err,
96                                         sM=M_err,
97                                         d=d_mean,
98                                         f=f_root,
99                                         L=l_BC,
100                                         M=weight_data("rod"))
101
102
103 def experiment1_data():
104     m_a, m_b, m_c = weight_data("masses")
105     mass_dat = array(
106         [0, m_a, m_b, m_a + m_b, m_c, m_a + m_c,
107          m_b + m_c, m_a + m_b + m_c]) # [Kg]
108
109     # Round 1: (in order)
110     h_1 = array([9.44, 8.72, 8.00, 7.28, 6.58, 5.84, 5.15, 4.43]) * 1e-3 #
111         [m]
112     # Round 2: (in order)
113     h_2 = array([9.42, 8.70, 7.98, 7.26, 6.53, 5.80, 5.09, 4.39]) * 1e-3 #
114         [m]
115     # Round 3: (in order)

```

```

114 h_3 = array([9.42, 8.71, 7.98, 7.26, 6.53, 5.80, 5.09, 4.37]) * 1e-3 #
    [m]
115 # Round 4: (in order)
116 h_4 = array([9.41, 8.69, 7.97, 7.25, 6.52, 5.79, 5.08, 4.36]) * 1e-3 #
    [m]
117 # Round 5: (in order)
118 h_5 = array([9.42, 8.70, 7.98, 7.26, 6.70, 5.87, 5.19, 4.51]) * 1e-3 #
    [m]
119
120 h_mean = (h_1 + h_2 + h_3 + h_4 + h_5) / 5.0
121
122 m, c, dm, dc = fys.linfit(mass_dat, h_mean)
123
124 mass = linspace(0, 3.5, 8)
125 h_mass = m * mass + c # h(m)
126
127
128 def plotdata():
129     h_sets = [h_1, h_2, h_3, h_4, h_5]
130     plot(mass, h_mass, label="Linear fit")
131     # errorbar(mass, m * mass + c, yerr=dm, color='blue', fmt='o',
label='Error Range')
132
133     for dat in h_sets:
134         plot(mass_dat, dat, "x", color="r")
135     plot(NaN, NaN, "xr", label="Data points")
136     xlabel("mass [kg]")
137     ylabel("h(m) [m]")
138     ticklabel_format(style='sci', axis='y', scilimits=(0,0))
139     legend()
140     title("Linear fit of mean deflection data;  $h(m) = Am + B\Delta A$ 
    = $ %.2e" % dm)
141     savefig("figs/h_m_fig.png")
142     close()
143 plotdata()
144
145 def plot_stddev():
146     """Plots the standard deviation of h(m)
147     as m is increased"""
148     deviation = np.zeros(len(h_1))
149     for i in xrange(len(h_1)):
150         deviation[i] = fys.stddev(array([h_1[i],
151                                         h_2[i],
152                                         h_3[i],
153                                         h_4[i],
154                                         h_5[i]])) [0])
155     plot(mass_dat, deviation, linestyle="--")
156     plot(mass_dat, deviation, "o")
157     ticklabel_format(style='sci', axis='y', scilimits=(0,0))
158     title("Standard deviation of deflection for each m\n")
159     xlabel("Load [kg]")
160     ylabel("$\sigma$ (Std. dev.)")
161     savefig("figs/h_m_deviation.png")

```

```

162         close()
163     plot_stddev()
164
165     # lengde mellom yttersidene til festepunktene til knivene
166     # PEE WEE 2m Y612CM LUFKIN +- 0.01cm
167     l_AB = 133.9 * 1e-2 # [m]
168     # diameter til festepunkter
169     # Moore & Wright 1965 MI +- 0.01mm
170     l_AB_diameter = 4.09 * 1e-3 # [mm]
171     # anta festepunktet er paa midtten saa trekk fra diameter totalt sett
172     l = l_AB - l_AB_diameter
173
174     #Maalinger av stangens diameter d paa forskjellige punkter
175     # Moore & Wright 1965 MI +- 0.01mm
176     d = array([15.98, 15.99, 15.99, 16.00, 15.99, 15.99, 15.98, 15.99,
177               15.99, 15.99]) * 1e-3 # [m]
178     d_m = mean(d); #m
179
180     A = abs((h_mass - c) / mass)
181
182     E = mean(4.0 * l**3 * const.g / (3 * pi * A * d_m**4)[1:-1])
183     print "E from deflection = %e"%E
184
185 def experiment_2():
186     ''' Data pertaining to the audio exp. '''
187
188     experiment1_data()

```