ALASTAIR MCLEAN

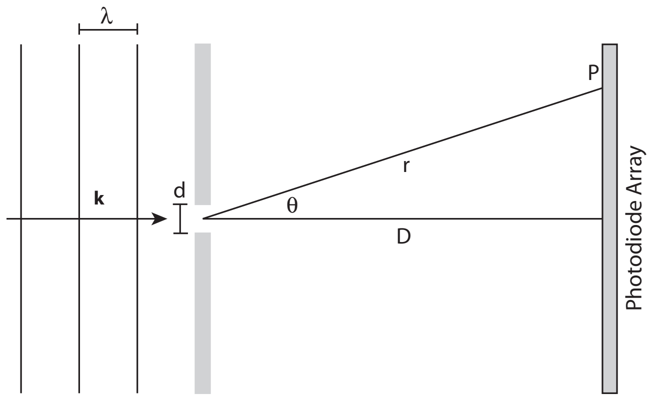
ASSIGNMENT 1

The Python Assignment. To complete this assignment you will have to read the Jupyter notebooks that have been provided onQ that provide an introduction of Python. In particular, you will need to be able to write a short function and plot a curve. Apart from the resources that have already been mentioned, help and instruction will be provided in class time.

For the last question, you will need to be able to load a Jupyter notebook, run it using the *Run All* command and then change the values of parameters, in this case the non-linear damping coefficient k_2, and interpret the results.

This assignment was first used in 2015 and we did the coding in MATLAB. The assignment was done in groups during a problem class.

Problem 1 *Single Slit Diffraction Pattern*



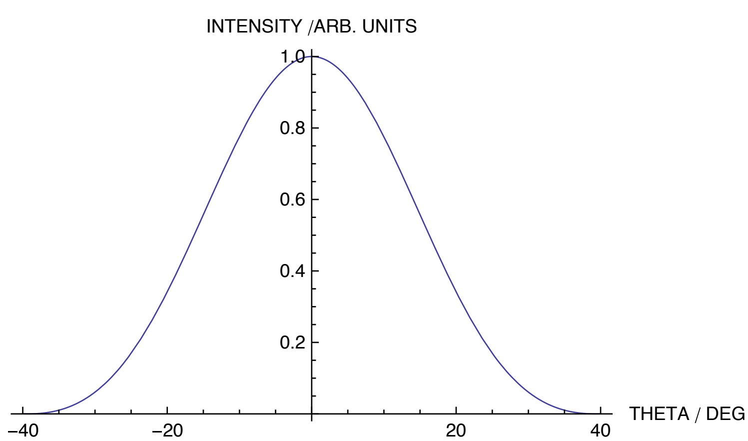
The diffraction of light from a single slit of width d as a function of β is described by

$$I = I_\circ \left(\frac{\sin\beta}{\beta} \right)^2 \tag{1}$$

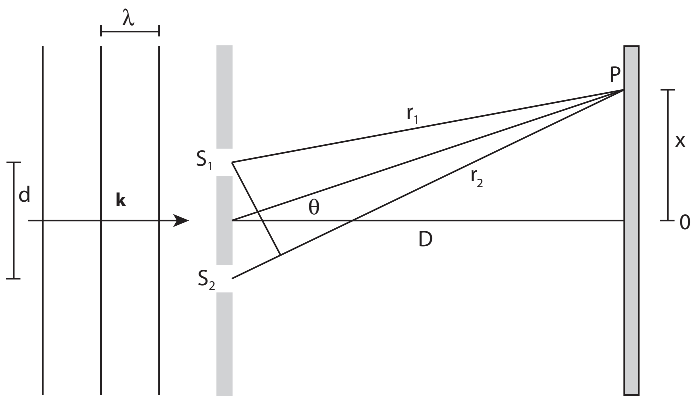
where β is:

$$\beta = \frac{\pi d}{\lambda} \sin\theta,$$

and θ is the angle corresponding to point P on the screen. I generated the plot of $I/I_\circ$ using Mathematica. Generate a similar plot with Python using the same parameters: λ = 632.8 nm (helium neon laser), d = 1 μm, and $-40^\circ < \theta < +40^\circ$. Remember to convert θ to rad. Your solution should contain the Python code you used to generate the plot and a copy of the plot.



Problem 2 *Double Slits of Finite Width*



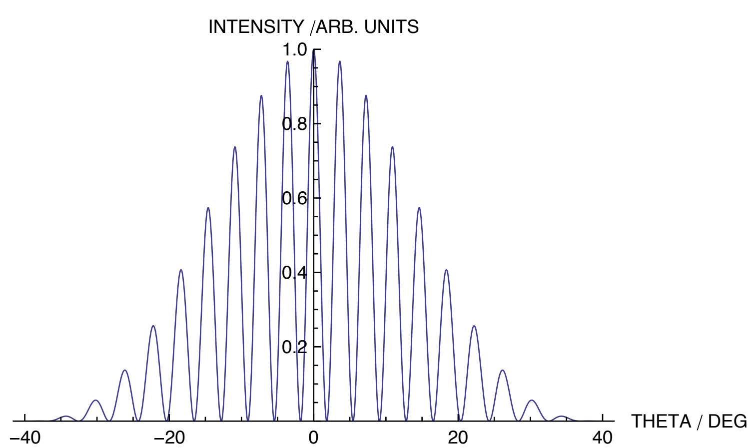
Now we have light interfering from two parallel slits. There is a plane wave incident from the left with wavelength λ. There are two parallel slits in a thin opaque screen and a glass slide on the right. The diffraction pattern produced by two parallel slits of width d and spacing a is:

$$I(\theta) = I_0 \frac{\sin^2 \beta(d)}{\beta(d)^2} \cos^2 \beta(a)$$

where

$$\beta(x) = \frac{\pi x}{\lambda} \sin\theta,$$

d is the width of each slit, and a is the spacing between each slit. Using Mathematica, I plotted the diffraction pattern ($I(\theta)/I_\circ$) that would be generated by slits of the following size and spacing with a helium neon laser over the same angular range as the last problem: λ = 632.8 nm (helium neon laser), d = 1 μm, and a = 10 μm. Using Python, produce a similar plot.



Problem 3 *Mean and Standard Deviation*

The most frequently used statistical measures are the *mean* $\bar{x}$ and the *standard deviation* s. If N measurements of a variable $x_1,\ x_2, \ldots x_i, \ldots x_N$ are obtained, the mean is

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,$$

and the standard deviation s, a measure of the width or spread of the distribution, and is given by

$$s = \sqrt{\frac{1}{(N-1)}\sum_{i=1}^{N}(x_i - \bar{x})^2}.$$

Python has built in function, `mean` and `std` to calculate the mean and standard deviation of an array. Write your own functions, `mymean` and `mystd`, to calculate the mean and standard deviation. Check that they give the same result to the Python NumPy functions (`mean` and `std`) for the following array:

$$x = (5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14)$$

Python

```
In[1]:  x=np.arange(5,15)
In[1]:  np.mean(x)
Out[8]:  9.5
```

Problem 4 *The Standard Error*

The quantity $s_\mu = s/\sqrt{N}$ is the **standard error** of the mean; i.e. the probable error of the mean. Our best estimate of the mean μ and its uncertainty are

$$\mu = \bar{x} \pm s_\mu.$$

It can be shown that the probabilities that $\bar{x}$ falls in the ranges $\mu \pm s_\mu$, $\mu \pm 2s_\mu$ and $\mu \pm 3s_\mu$ are 0.68, 0.95 and 0.997. These are the 68%, 95% and 99.7% confidence limits. Ten readings of a time interval produce the following values:

20.28, 21.26, 20.96, 20.70, 20.31, 21.16, 20.60, 20.36, 20.55, 19.95 s

Write a Python function called `mystderr` to calculate the standard error and using it ,and `mymean` and `mystd`, show that:

- $\bar{x} = 20.61 \qquad s = 0.42 \qquad s_\mu = 0.13$

so that $\mu = 20.61 \pm 0.13$ s

Cut and paste this if it is easier.[1]

[1] x=[20.28 21.26 20.96 20.70 20.31 21.16 20.60 20.36 20.55 19.95]

Problem 5 *Linear Regression*

The straight line

$$y = a_1 + a_2 x$$

is to be fitted to a data set containing N pairs (x_i, y_i) where the error is assumed to be only in y_i, i.e. x_i is arbitrarily accurate. The best estimates of a_1 and a_2 are found to be

$$a_1 = \frac{\sum x_i^2 \sum y_i - \sum x_i \sum (x_i y_i)}{D} \qquad a_2 = \frac{N \sum x_i y_i - \sum x_i \sum y_i}{D}$$

where

$$D = N \sum x_i^2 - (\sum x_i)^2.$$

All sums are $i = 1$ to N. The *standard error* estimates Δa_1 and Δa_2 in a_1 and a_2 respectively are:

$$\Delta a_1 = \sigma_y \sqrt{\frac{\sum x_i^2}{D}} \quad \text{and} \quad \Delta a_2 = \sigma_y \sqrt{\frac{N}{D}}$$

and σ_y is defined by

$$\sigma_y = \sqrt{\frac{\sum h_i^2}{N-2}}$$

where h_i is the deviation of the measured y_i from the fitted line,

$$h_i = y_i - (a_1 + a_2 x_i).$$

The results of a least-squares fit are quoted in the form $a_1 \pm \Delta a_1$ and $a_2 \pm \Delta a_2$. The errors have the normal meaning of a probable error in that there is a 68% probability of finding the true result within the quoted range. [2] [3]

Write a Python function, called `mylinreg`, to perform linear regression on the given data set, and provide estimates for $a1$, $a2$, $\Delta a1$ and $\Delta a2$. The nominal values for a_1 and a_2 are 5.0 and 2.0 (arb. units), respectively. The function should also produce a plot of the data points, given above, and the line of best fit.

[2] x=[1 2 3 4 5 6 7 8 9 10]

[3] y=[7.371 8.292 11.296 12.805 15.550 16.340 18.789 20.493 23.127 25.104]

x	1	2	3	4	5	6	7	8	9	10
y	7.371	8.292	11.296	12.805	15.550	16.340	18.789	20.493	23.127	25.104

Problem 6 *The non-linear mass-spring*

The EOM[4] for the linear mass-spring system

[4] Equation of motion

$$\ddot{x} + \omega_n^2 x = 0. \tag{2}$$

has well-known sinusoidal solutions

$$x(t) = A\cos(\omega_n t + \phi). \tag{3}$$

Not to labour this point, but most physics and engineering textbooks study systems which have analytical solutions for obvious reasons. However, we now have access to high quality computational resources that allow us to examine systems that do not have analytical solutions. In many cases, these are more interesting and we can use numerical simulations to extend our intuition about mechanical systems.

However, if a small amount of non-linearity is introduced to our system the well-known and familiar solution no longer describes our system. For example, we could introduce a second-order term to the restoring force

$$F(x) = -\left(k_1 x + k_2 x^2\right) \tag{4}$$

and the equation of motion would then become

$$\ddot{x} + \omega_n^2 x + \frac{k_2}{m} x^2 = 0. \tag{5}$$

We will continue to use ω_n. It is the natural frequency of the system when $k_2 = 0$. How does the presence of the second-order term k_2 affect the motion of the mass? This problem is designed to allow you to investigate what happens when non-linearity is present.

What are the dimensions of k_2? Are they the same as the dimensions of k_1?

This is the form of the restoring force introduced by French in Chapter 1. We will continue to use it here.

The accompanying Jupyter notebook has some Python code that calculates the response of the non-linear mass-spring system for you. It does not matter, at this point, how this is done. We will look at that later.

2016-a1-nonlinearmassspring

In the notebook the linear spring constant k_1 = 40 N/m and the mass is 0.1 kg. You might ask, why these values? Well, a mass of 100 g is something that we can all relate to. The iPhone 5S has a mass of 112 g. Why not find out how much your phone weighs? The spring constant of 40 N/m was specially chosen because the place we enounter springs most often is in ball-point pens. The small spring that is used to retract the nib has a spring constant in the range 20-40 N/m. So I chose this value. Chosing parameters that we have a physical feel for develops our intuition. The notebook calculates ω_n for this system, but you may want calculate f_n.

Also note that the sign of k_2 has the same sense as the sign of k_1. A positive value of k_1 translates to a harder spring; the restoring force is larger for a given displacement. When k_2 is negative, the total spring constant will be softer, the restoring force will be smaller.

The initial condition I have used for $x_\circ$ may seem a little unrealistic because it is 1.0 m. However, I wanted to sample a large part of the force curve. An alternative approach would be to make $x_\circ$ smaller and k_2 larger.

a Set k_2 to +15 n/m^2 and explain the changes you see in $x(t)$, $v(t)$ and v versus x.

b Set k_2 to -15 n/m^2 and explain the changes you see in $x(t)$, $v(t)$ and v versus x.

For both of the above, your solution should provide graphs of the forces and the associated system responses. You can generate these by uncommenting lines in the notebook that generate output.