MNRAS 000, 1–17 (2021) Preprint 1 April 2023 Compiled using MNRAS LATEX style file v3.0

The Pendulum Project: An exploration of experimental physics

Utkarsh Mali[1]★
[1]Department of Physics, University of Toronto, 60 St. George Street, Toronto ON M5S 1C6, Canada

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT

The simple pendulum is a common system studied in most undergraduate physics labs. It's demonstrates many faucets of physics which students can study and learn from. Studying its motion improves one's understand of experimental science and the scientific method. This paper demonstrates the steps taken in building, recording and analysing a homemade pendulum. The purpose of this experiment was to test the theoretical models against the values found from data collection. The parameters studied include; the system symmetry, decay constant, time period, damping coefficient and acceleration due to gravity. This was done by varying the mass M, length L and release angle θ_0, recording the motion and then analysing the recorded data. In most cases studied, the data collected matches the values predicted by the theoretical model (a more in-dept version of this can be found in the introduction). The few mismatches between theory and experiment were explainable. The results and error propagation demonstrate the theoretical model's robust predictions given the imperfections of the pendulum system. The analysis offered insights into reliable pendulum design using homemade materials. These insights can be used in developing new experiments which will ultimately lead to more reliable measurements of the simple pendulum and analogous experiments in more specialized fields of physics.

Key words: time – reference systems – instabilities

1 INTRODUCTION

The pendulum is an object which is free to move held from a pivot position. Its first documentation was in the *Han Dynasty* with more modern developments made in the 1600's namely by Galileo and Huygens Fulcher & Davis (1976). Today, an idealization of this pendulum is commonly studied in undergraduate labs. The purpose of this paper is to demonstrate the methods and results in building a simple homemade pendulum and the study of its motion, namely damped harmonic motion. While the purpose is quite general the following will be studied: The time period of the pendulum, the gravity associated with the characteristic time period, the damping coefficient of the system as a function of the variations of mass M, length L and release angle θ_0.

In the first experimental setup which varies the masses, the time period for oscillations was independent of the masses $T = 2.08s \pm 0.05$, this was within 1% of both the theoretical models which are going to be introduced below. The damping coefficient was found to be independent of mass (as will be demonstrated in the theoretical background below), with a mean value of 1.48.

In the second experiment, the release angle θ_0 was varied. Both the time period and decay parameter were constant with $T_{angles} = 1.935 \pm 0.04$ and $\gamma = \pm 77 \pm 4$. The length of the pendulum was changed between setups, hence the time period from this does not match the time period in the first experiment.

In the final experiment, the length was varied. The decay parameter remained constant $\gamma = 111 \pm 6$, as expect. The acceleration due to gravity was computed from the differences in the time period, this was found to be $g = 9.98\pm, 0.3$ which is within 2% of the theoretical model. In the following section, the equations being used to compute these parameters will be highlighted. The values being measured will be clearly mentioned, as will the parameters used in the fitting.

2 THEORETICAL BACKGROUND

2.1 The Wilson Model

This is the model used by Professor Brian Wilson who is then under the Department of Physics at the University of Toronto. His model assumes underdamped harmonic motion.

$$\theta(t) = \theta_0 e^{-t/\tau} \cos\left(2\pi \frac{t}{T} + \phi_0\right)$$

Furthermore, his model predicts that the final time period of the pendulum will not depend on the time period of oscillations T not the decay constant γ.

$$T_o = 2(L + D)^{1/2}$$

The first goal of this experiment is to verify his results (or prove them wrong in our case). The subsequent goals of the experiment are to demonstrate the accuracy the model that is introduced in the following sections.

★ E-mail: utkarsh.mali@utoronto.ca

2.2 Equations of Motion

The set for the equations has been deliberately made different from recent literature (PHY324 Guidelines) as its follows underdamped motion more accurately Aggarwal et al. (2005). The motion occurring in the simple homemade pendulum models a damped harmonic oscillator. Using Newton's 2nd Law, we can arrive at its differential equation Aggarwal et al. (2005). The setup of the **ODE has been intentionally included** because it provides additional insights into the forces involved and assumptions made (like the drag being linear).

$$\frac{d^2\theta}{dt^2} + \frac{c}{m}\frac{d\theta}{dt} + \frac{g}{l}\sin\theta = 0$$

Here c represents the the linear drag coefficient, g represents the acceleration due to gravity and L represents the length of the pendulum (from its center of mass to the pivot point). The differential equation can be solved to give the angular equation of motion.

$$\theta(t) = e^{-\gamma t} A\cos(\omega_1 t - \alpha)$$

We have defined $\omega_1 = \sqrt{\omega_o^2 - \gamma^2}$ where ω_1 is the angular frequency of the under-damped motion, ω_o is the angular frequency of the motion given that there was no damping. The decay constant is denoted by γ, the phase shift is written as α. Note that we are working under the small angle approximation $\sin\theta \approx \theta$. Hence we can re-write this equation as a function of the x-position (using trigonometry).

$$x(t) = L\sin(\theta(t)) \approx \theta(t) = e^{-\gamma t}\tilde{A}\cos(\omega_1 t - \alpha)$$

We have used $\tilde{A} = AL$ with L representing the length of the string. This is the equation that will be used in order to derive most of the desired quantities and parameters. It will also be used in the `curve_fit()` function to build a theoretical model and goodness of fit parameter.

2.3 Studying the Pendulum Symmetry

Before the detailed goals of the experiment are computed, the symmetry of the pendulum needs to be checked. If the pendulum is not symmetric, then we cannot trust the remaining goals checked in the remaining sections of the experiments. This will be determined by computing the difference between the peaks of the positive amplitudes and negative amplitudes respectively. The first intuition was to use the reduced chi squared χ^2_{red}. However, this does not work since the values less than machine precision from 0 will end up causing a divide by zero error. Hence a new system needed to be generated in order to determine the symmetry of the system. From this the symmetry parameter was introduced.

$$\text{Symmetry Parameter} = \sum_{t=0}^{\texttt{max(t)}} |d_t|$$

Here t represents the time step of measurement, we expect this to be very small (one per time period of the pendulum). The value of the difference between the positive and negative peak at this time step is denoted by d_t. **The symmetry parameter represents the sum of all the differences in peaks** at each moment in the pendulums motion. If the value of the symmetry parameter is below 0.05 for all oscillations and 0.02 and small oscillations, then the pendulum is said to be symmetric.

2.4 Determining the Damping Coefficient

In our derivation for the equations of motion, we have assumed that there is linear drag taking place with its coefficient dependent on the masses and the decay constant of the curveAggarwal et al. (2005).

$$F_{damp} = -cv$$

$$c = 2m\gamma$$

The goal in determining the decay constant will be to first determine if the decay is considered to be exponential, this is done by applying `curve_fit()`. After this the decay constants calculated should all be equal to one another. The final goal of the section is to verify this. It is also possible to apply this technique to determine the variability of the decay constant as the release angle θ_0 is change.

2.5 Time Period of an Underdamped Gravity Pendulum

The time period of the pendulum representing under-damped motion is given by.

$$T_1 = \frac{2\pi}{\omega_1}$$

In order to derive the non-damped time period, we must convert our angular frequency to that of the non-damped motion. This value should equal the theoretical derivation for the time period of a simple gravity pendulum.

$$T_o = \frac{2\pi}{\sqrt{\omega_o^2 + \gamma^2}} = 2\pi\sqrt{\frac{L}{g}}$$

Note that this equation can be inverted to solve for the value of the acceleration due to gravity

$$L(\omega_o^2 + \gamma^2) = \frac{4\pi^2 L}{T_o^2} = g$$

According to this equation, the time period is expected to be independent of both the mass M and the angle of release θ_0.

2.6 Measuring Acceleration due to gravity

As an extension of the equation shown above, we can use this to compute the acceleration due to gravity by varying the length of the pendulum and measuring its time period. From the equation above:

$$T_o = \sqrt{\frac{4\pi^2 L}{g}} = a\sqrt{L}$$

The constant a represents the `popt` parameter that will be used in the `curve_fit()` function. The graph we will plot is T vs L, we expect a sqaure root relationship, we will apply the curve fit using that relationship to find the value of a. This in turn will give the value of g.

3 MATERIALS AND PREPARATION

3.1 Materials

All of the materials used were either present in the household in which the experiment took place, or bought at an inexpensive price from Dollarama which is a local retail store. PlayDoh was used as the variable masses, this was chosen since it was homogeneous, and

Figure 1. Size comparison of the masses used to measure the oscillations of the homemade pendulum. The mass on the left is approximately 5-10g while the mass on the right is in excess of 150g.

its moment of inertia can be easily controlled. Green garden string was chosen as the massless string, this was chosen because it was lightweight. An *iPhone 8* was used as the camera's measurement device. A camera was preferred to human measurements as video footage enables software tracking and is reproducible. Finally, the variables masses were stored in a ziplock bag which was tied with the massless string. Both medium and small sizes were used to prevent avoidable drag.

3.2 Preparation of Masses

The PlayDoh being used was extracted from four different tubs each containing 113g. The PlayDoh was separated into different weights in an attempt to generate a double order of magnitude difference in the mass. At the time, a mass balance was not present so the data was taken without initial knowledge of the mass. The masses were rolled into a cylindrical ellipsoid shape. This was chosen because its moments of inertia have principle axis around the vertical axis with the horizontal planar axis free to rotate without changing motion. This mitigates some of the spinning effect that may be observed due to the string. The color choices of the mass was deliberately chosen in order to give the tracking software optimum conditions.

3.3 Experimental Setup

The final, refined version of the experimental setup has been shown. This is the outcome of an iterative redesign initiative which allowed for more accurate measurements to occur. First I had only used one string, but found that the motion was co-planar resulting in unwanted circular motion. Additionally, I improved the shapes of the masses to give the same moments of inertia.

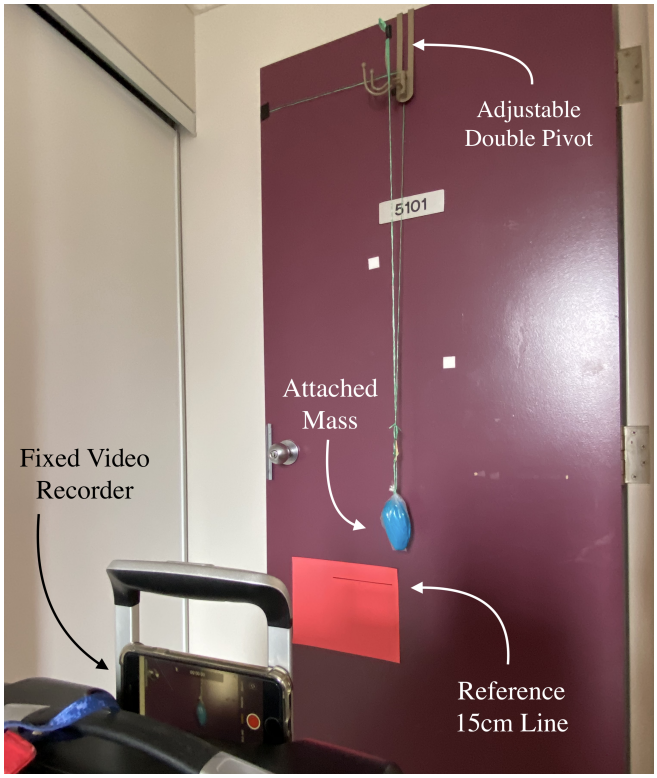

Figure 2. The experimental setup of the homemade pendulum. The massless string is attached to the adjustable pivot point. This point is then attached to a hook which allowed for the quick interchanging of masses, it is also attached to a **double pivot to stop the co-planar (circular) motion of the pendulum**. The video recorded is fixed in place using the baggage handle and a measured 15cm line has been placed in order to convert the tracked pixels to meters. Fine angle markings were done but cannot be seen in the diagram. The lengths of the string are the same since the pivot is adjustable.

3.4 Control Variables and Minimizing Uncertainties

In this section, the methods used in minimizing the uncertainties will be shown. The first control variable in all experiments is the **angle of video recording**. At all times the camera was placed perpendicular to the rest center of mass of the pendulum. This ensured that the tracking software would calculate the true position.

The lengths of all masses were calibrated in order to **maintain the same length of the pendulum**. The reader can see this in Figure 1 in which two lines were drawn with the lengths of the strings varied to equal one another while keeping the position of the center of mass the same.

When the masses were being prepared, all the **additional elements of the zip-lock bags were taped in order to minimize the frictional force** from oscillations. This could not be fully done for the smaller masses. The smallest mass in Figure 1 has additional surface area which could cause friction. As a result, we expect the smaller masses to have additional friction. Additionally, extra caution was taken to **ensure that the shape of the mass remained constant**. This will ensure that its moment of inertia stays the same.

When the experiment was being recorded, all heating and **air conditioning was switched off to reduce unobserved wind channels** of air that would skew the results. When the experiment was first done, this effect caused a noticeable anti-symmetry in the pendulum.

When the experiment was first done, the circular motion of the pendulum distorted the single dimensional motion. In the redesigned setup of the experiment, two strings with two pivots were used instead of one. This **eliminated circular planar motion of the pendulum**. The length of both pivots were the same distance from the center of mass.

Finally, when a specific setup of the experiment was done, the elements that were varied in the other setups were kept constant. For example, if the length was being changed, then the mass and angle of release were kept constant.

4 EXPERIMENTAL PROCEDURE

This section is broken down into two parts, procedure and experimental notes. The experimental notes will highlight possible sources of error.

4.1 Methodology

(i) The experiment was set up according to Figure 2 and the masses were prepared according to Figure 1 in according with the *Preparation of Masses* criteria. When the mass was attached to the string, it was ensured that the mass did not spin and were vertically oriented. The paper was ensured to be level to the horizontal place. Even a small change in its angle will result in inaccurate values alter on.

(ii) A central black spot was drawn 109cm from the top of the pivot point. This served as the reference point for which all masses needed to be adjusted too. The length of the string was adjusted so that the center of mass was parallel (along the vertical axis) to the black dot drawn.

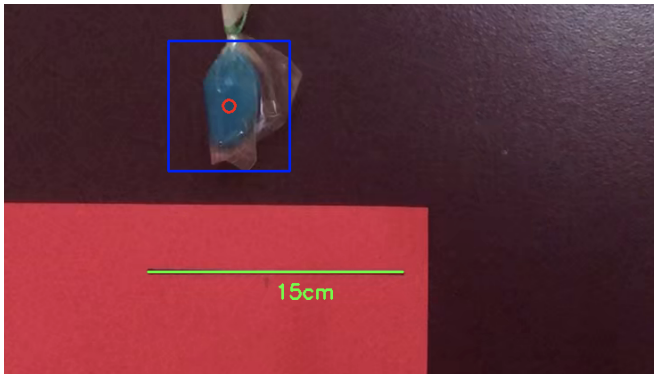

Figure 3. Video capture analysis software. The box represents the item being tracked, the red circle represents the centroid of the item from which the position will be calculated. The green line represents what 15cm would be in real life.

(iii) The mass was drawn back and set at a specific θ_0 defined from the pivot point. This became the reference point for all future masses to be swung from.

(iv) The video recorded was started, and the mass was set free to swing ensuring that its motion in the y direction (towards and away from the camera) was zero. All air conditioning and heating was turned off to ensure a turbulent-free environment. Once the mass's oscillations became visibility negligible (< 0.5cm). The recording was stopped.

(v) The steps above were repeated for each mass element from M1 to M6, with 3 trails taken for each data-set, the average of all data-sets were plotted.

(vi) The entire procedure above was repeated with both the angle of release θ_0, and the length L varied separately. For experiments in which the mass was not changed, the mass was fixed to Mass 3 (M3).

(vii) The video was processed through the Python software and the symmetry of the pendulum was checked before taking subsequent trials.

4.2 Experimental Notes

- I noticed that if the lengths of the ziplock bags tied were not symmetric, then the ziplock bag would start to spin due to the torsional energy stored in the spring.
- The object would settle down in to a forward and backward motion, towards and away from the camera. This could have been due to the natural frequency of oscillations of the door and hang-mount. In order to eliminate this, a second string was added to restrict the motion to a simple xz planar surface.
- Due to the attached clip, the mass did not completely line up with the string attached to the pivot point.
- The string was observed to be very slightly elastic. It would stretch by around 1cm when pulled firmly. The errors in L should demonstrate this.

4.3 Custom Video Tracker: *Data Preprocessing*

Viewing the following video (2 mins, uploaded by me) is **highly recommended for the reader** as it demonstrates the video analysis section of the methodology in action.
https://www.youtube.com/watch?v=f92H_bB6VAs The initial idea was give by Adrian Rosebrock and Corey M. Schafer Larsen et al.

Mass Number	Mass ($\pm 1g$)
1	10
2	25
3	60
4	90
5	110
6	160

Table 1. Table representing the correlation between the mass number and the weights of the masses. It was observed that as the mass number increased, the mass value also increased. The scale being used was only accurate to 1g.

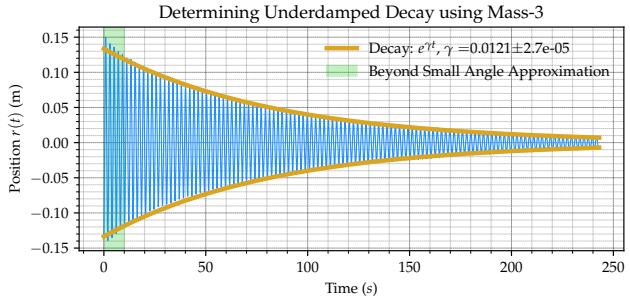

Figure 4. An example of what the plot of the decay coefficient looks like. There is characteristic exponential decay in the maximum amplitudes of the oscillations. The region at the beginning was purposely set beyond the small angle approximation to demonstrate that the theory does not fit perfectly in this region.

(2016); Goyal et al. (2017); Villán (2019), this was done under their MIT License which permits copying, using and distributing their code for private and commercial use. The software was then specialized and made useful to the recorded video by me.

5 RESULTS

After all the data was collected the masses were measured with their values listed in the table below. These values should not play a role in both the damping coefficient and the time period of oscillations.

5.1 Determining Damping Coefficient using Exponential Decay

In this section, we analyse the exponential decay of the oscillations. In the sample plot below, the oscillations followed exponential decay through oscillations. This was not the case for all masses. Using the `curve_fit()` function, the value and error for γ were able to be computed. **The mass was purposely swung beyond the small angle approximation**.

5.1.1 Comparison of Damping Coefficients

$c = 1.479 \pm 0.3$(units)

Since this is the same pendulum setup with different masses, we expect the damping coefficient to stay the same. This was observed for most masses (except Mass M4). The uncertainty in the damping coefficient increases as the decay constant γ gets smaller. Hence the uncertainties are very large for large masses. **The full error propagation for this is attached in the appendix on the recommendation by Professor Wilson.**

Number	Mass	Gamma	Damp Coeff	Damping Coeff Error
1	10	0.0326	0.651	0.0294
2	25	0.0325	1.626	0.1990
3	60	0.0121	1.449	0.1328
4	90	0.0094	1.696	0.1626
5	110	0.0069	1.527	0.5110
6	160	0.0060	1.927	0.5746

Table 2. This table represents the correlation between the values of the masses and the damping coefficients of motion. As the mass increased, the value of gamma was observed to decrease, however, the damping coefficient stayed relatively stable baring the exception of the first value of mass. The errors in the damping coefficient increases with mass.

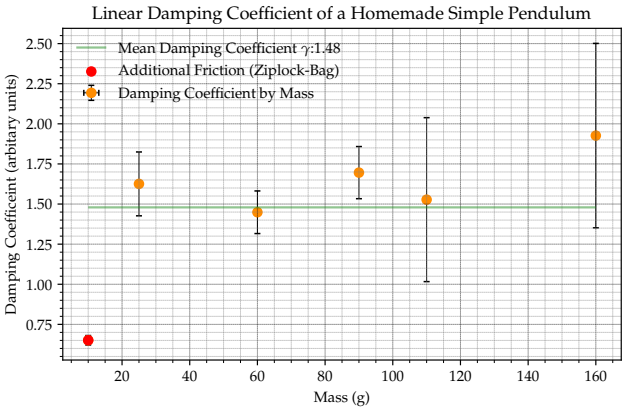

Figure 5. The values of the damping coefficient in arbitrary units is plotted as a function of the mass of the pendulum. As the mass increased, the value of the damping coefficient stayed relatively stable. The value of the first damping coefficient was very different from the rest of the values.

5.2 Verifying Pendulum Symmetry

Before further in-depth analysis was done to the pendulum system, its symmetry was verified. The scientific python function `scipy.signal.find_peaks()` was used in order to determine these peaks.

$$\text{Symmetry Parameter} = \sum_{t=0}^{\texttt{max(t)}} |d_t| \tag{1}$$

$$= \texttt{np.sum(abs(differences))} \tag{2}$$

$$= 0.016730 \tag{3}$$

Where d_t is the difference at time step t. This means that the absolute sum of all the differences is less than 0.02. As a result, it can be concluded that the pendulum is indeed symmetric.

This verification was done for all further sets of experiments. If the symmetry parameter was **below 0.02 for small angle oscillations, and 0.05 for all oscillations** and there was no skew in the difference **then the pendulum was said to be symmetric**. The individual plots have been omitted from the report since the

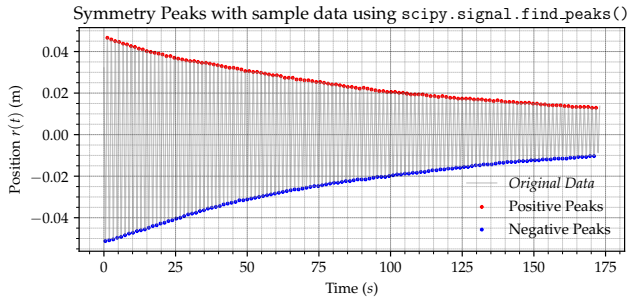

Figure 6. Plot representing the original data with the positive and negative peaks. The original data followed characteristic underdamped motion while the positive and negative peaks followed positive and negative exponential decay, respectively.

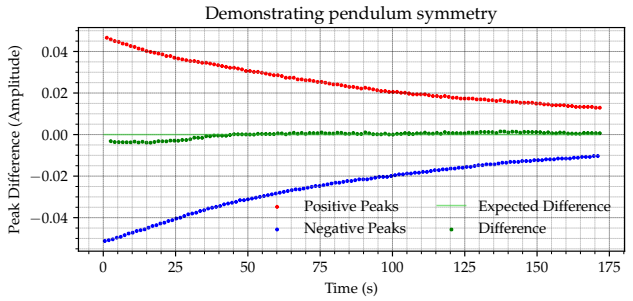

Figure 7. This plot represents a refinement Figure 6 shown above. Positive and negative peaks are shown with their difference. The difference is almost always near zero but varies slightly. The expected difference for all oscillations in a perfect pendulum is zero.

difference was very close to zero. If any skew was found, then the experiment was repeated.

5.3 Measuring the Period

The data was measuring using the custom video tracker highlighted in the section above. The goal of this section is to show how the values for time period were calculated and to compare them with the theoretical value for the time period. Firstly the useful data was extracted from the tracking footage. In the second *Sample Errors for Time Period* the methods for calculating the errors represented in the plot will be demonstrated. Once the useful data was extracted from the motion detector. The `curve_fit()` function was applied from the `scipy.optimize` package. Following this, the time period of each mass was predicted using the parameter and error values computed from the curve fit. Attached in Appendix A is a sample of what was done for every plot. The χ^2 values were computed for each mass setting using `scipy.stats.chisquare()` and represented in the table.

5.3.1 Comparison of Time Periods

Using the plotted values of the time period as well as their errors calculated. The final plot demonstrating the time period for each individual mass was drawn. The value of the time period according

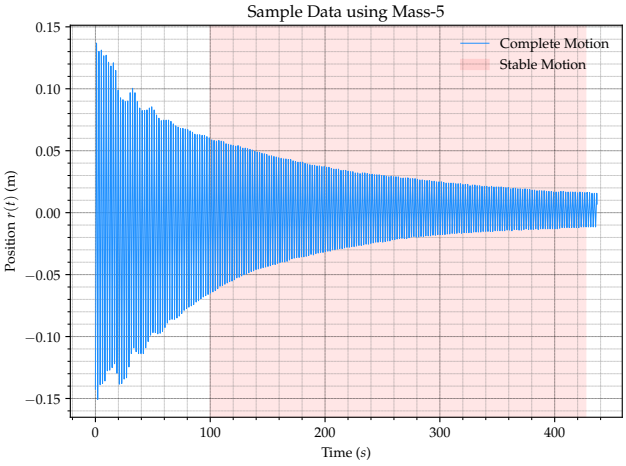

Figure 8. Data representing the stable state motion of the M5 mass. The oscillations at the start did not represent exponential decay due to the mass spinning. **Only stable motion data is used to calculate the time period.**

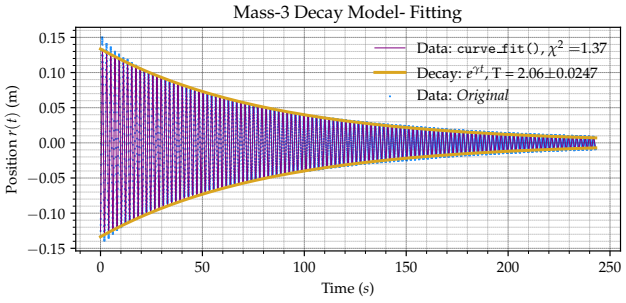

Figure 9. Figure representing the model fit of the `curve_fit()` function as the pendulum oscillates for the M2 mass. The exponential decay factor is plotted as an overlay. The decay curve fits the measured values perfectly.

Mass Number	Time Period (s)	Time Period Error ($\pm s$)	χ^2
1	1.99	3.16e-02	-2.84
2	2.10	1.05e-01	-45.86
3	2.06	2.47e-02	1.37
4	2.08	2.86e-02	6.98
5	2.09	7.36e-02	139.33
6	2.08	3.38e-02	-13.32

Table 3. Here I have included the tabulated results of all the data included. This includes the sample data shown in the pages above. All the periods of oscillation lie between 2.0 to 2.10 seconds. It was also observed that as the mass increased, the time period also increased. Values with higher absolute χ^2 values were found to have a greater error in time. Hence for large values of M, the data was not fully trusted. More measurements would be required.

to an idealized theoretical assumption was found to be 2.094s, with the mean value of the computed time periods being $2.077s \pm 0.05$. The length L was measured to be 1.09m with an uncertainty of 5cm due to the worn down measuring tape.

$$T_{theory} = \frac{2\pi}{L} = \frac{2\pi}{1.09} = 2.0944s$$

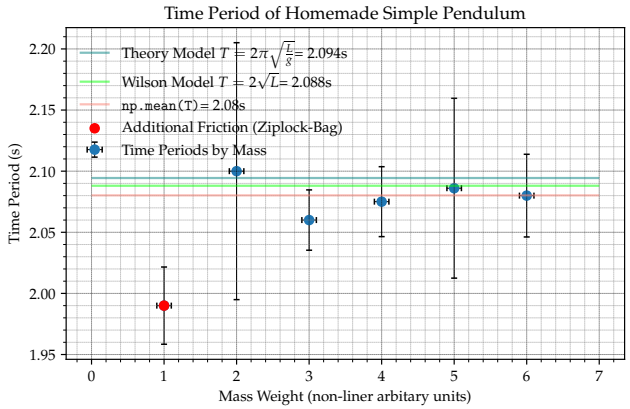

Figure 10. All the time periods measured for each individual mass is plotted as a function of mass. The theoretical mean is plotted as a horizontal line with the uncertainties individually calculated as a sum of tracking uncertainty and `curve_fit()` uncertainty.

$$T_{theory\_err} = T_{theory}\sqrt{\left(\frac{L_{err}}{L}\right)^2} \tag{4}$$

$$= 2.0944\sqrt{\left(\frac{0.05}{1.09}\right)^2} \tag{5}$$

$$= \pm 0.09607 \tag{6}$$

$$T_{exp} = \frac{1}{6}\left(T_{M1} + T_{M2} + T_{M3} + T_{M4} + T_{M5} + T_{M6}\right) \tag{7}$$

$$= 2.0768s \pm 0.05s \tag{8}$$

5.3.2 Comparison with The Wilson Model

By empirical observations the motion measured by the custom tracker matches that described by the initial equation given in the Wilson Model. Additionally, the time period is given by $2(\tilde{L} + D)^{1/2}$. In our calculations we have included the length of $\tilde{L} + D$ in our value for L, hence we arrive at $T = 2\sqrt{L}$. Since L is measured to be 1.09 ± 0.05 we arrive at our time period according to Professor Brian Wilson.

$$T_{Wilson} = 2\sqrt{L} = \sqrt{1.09} = 2.08806\text{s}$$

Errors in quadrature are also used to compute the error in the time period.

$$T_{err} = T_o\sqrt{\frac{L_{err}^2}{L}^2} = 2.08806\sqrt{\frac{0.05^2}{1.09^2}} = \pm 0.0958\text{s}$$

The value for the time period computed is within the uncertainty range of the experimental time period $T_{exp} = 2.0768s \pm 0.05$s and the more accurate theoretical time period $T_{theory} = 2.0944 \pm 0.09607$s.

5.4 Determining the Equation of Motion

Using the `scipy.optimize.curve_fit()` the values of the coefficients in the equation can be determined. Recall that the theoretical model for the equation of motion is described by a underdamped harmonic oscillation.

$$x(t) = e^{-\gamma t}A\cos\left(\omega_1 t - \alpha\right)$$

#	ω_1	$\omega_1\pm$	γ	$\gamma\pm$	A	$A\pm$	α	$\alpha\pm$
1	3.3	1.7e-04	0.04	1.7e-04	0.2	6.0e-04	0.2	2.9e-03
2	3.3	1.1e-04	0.02	1.1e-04	0.2	5.3e-04	0.3	2.7e-03
3	3.3	8.8e-05	0.02	8.8e-05	0.2	7.3e-04	0.3	3.5e-03
4	3.2	8.5e-05	0.01	8.4e-05	0.2	1.0e-03	0.1	4.2e-03
5	3.2	5.6e-05	0.01	5.6e-05	0.2	8.7e-04	0.1	3.7e-03
6	3.2	2.5e-05	0.01	2.5e-05	0.2	3.4e-04	0.9	1.8e-03

Table 4. Table representing the values of the equation of motion for all the masses used using the `scipy.curve_fit()` function. The oscillation frequency was observed to decrease, the damping coefficient was also observed to decrease, the amplitude stayed relatively stable and the value for alpha varied.

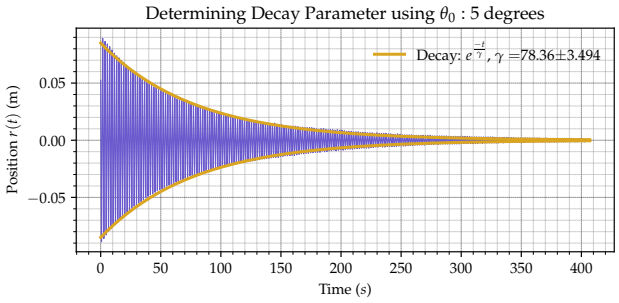

Figure 11. Motion represented by the underdamped motion of the pendulum with an initial release of $\theta_0 = 5^o$. The entire motion followed exponential decay. The amplitude of oscillation was small, so was the uncertainty on the decay parameter γ. The value of γ was found to be 0.0128.

The full equation will be shown for the Mass 3 (M3) with the other values tabulated in the table below. According to the `curve_fit()` function we found that $\gamma = 0.017$, $A = 0.207$, $\omega_1 = 3.248$, and $\alpha = 0.25$. When this is plugged back in, the resulting equation is found.

$$x(t) = e^{-0.017t}0.207\cos\left(3.248t - 0.25\right)$$

Note that for this section of the experiment, the oscillations began at different starting locations θ_0, hence α is expected to vary. Additionally the amplitudes A were not controlled, so we expect them to also vary. Tabulating these result in a table now.

5.5 Variable Angle, Computing the Decay Constant

In this section, the relationship between the decay constant and the variable release angle θ_0 will be studied. According to theory highlighted in the introduction. Since the decay constant depends only on the mass, we expect it to remain constant throughout the experiment. **The length L was adjusted to 0.95m when conducing this experiment, so we expect a different time period.** First, sample data will be shown, then the plot showing the difference in the decay coefficients will be shown highlighting the motion which violates simple harmonic motion. The mass has been kept constant as Mass 3 ($M3 = 0.6kg$) through out the experiment. The uncertainty on the angle was taken to be 1 degrees (due to an old protractor) and the uncertainty on the decay constant was given by the `curve_fit()` function.

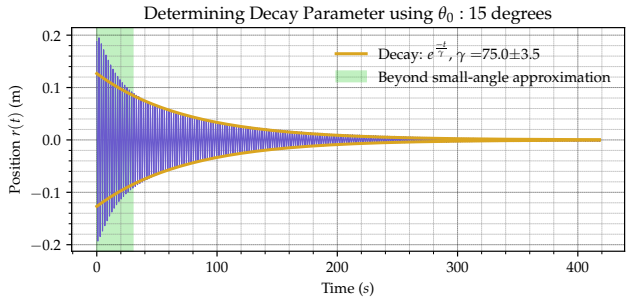

Figure 12. Motion represented by the underdamped motion of the pendulum with an initial release of $\theta_0 = 15^o$. The entire motion followed exponential decay. **The initial oscillations violated the small angle approximation they were not ignored, unlike in the previous section**. The amplitude of oscillation was small, so was the uncertainty on the decay parameter γ. The value of γ was found to be 0.0133.

θ_0	γ	$\Delta\gamma$
5	78.36	3.49
10	78.08	3.54
15	75.00	3.50
20	79.50	4.69

Table 5. Table representing the differences in the decay parameter with the release angle. The parameter was found to remain relatively stable as the angle changed, so were the uncertainties.

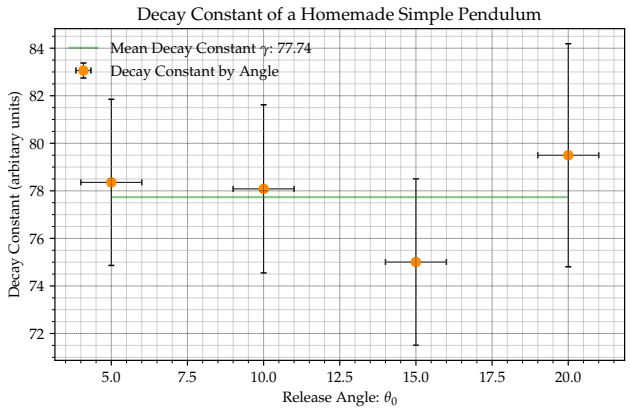

Figure 13. Graph representing the decay parameter for each release angle. The values for the decay parameter stayed relatively constant within the given uncertainties. Additionally, the errors on the decay constant, each computed through `curve_fit()` represent a reasonable degree of error, increasing with the angle as expected.The value for the mean error was computed according to Appendix A.

5.6 Relationship between Release Angle and Time Period

Using the same experimental setup as before, the time period should also remain independent of the release angle. As before the model was curve fit, the parameter for time period computed as a result and then plotted as a function of the angle of release θ_0. Below you will find an example of the `curve_fit()` the the variations in time period.

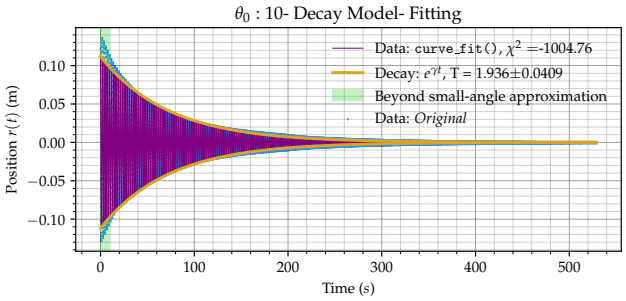

Figure 14. Graph representing the fit between the decay model and the resulting function from curve fit. As before, **values that did not obey the small angle approximation were not ignored.**

θ_0	Period T	ΔT	χ^2
5	1.934	0.0299	77.82
10	1.936	0.0409	-1004.76
15	1.934	0.0229	-0.33
20	1.937	0.0419	389.74

Table 6. Table representing the variation between the time period and the release angle of the pendulum. The time period remains relatively constant. The goodness of fit values vary greatly, with slightly larger errors in values with bad fit representing this.

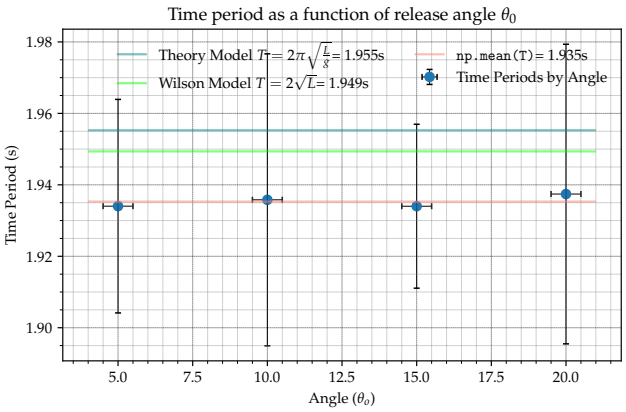

Figure 15. Graph representing the fit between the decay model and the resulting function from curve fit. As before, **values that did not obey the small angle approximation were not ignored.** The value for the mean error was computed according to Appendix A.

The data for all these values are tabulated into both tables and graphs.

5.7 Determining the decay constant for varied length

The goal of this section is to determine the decay constant. The method of analysis are exactly the same as those in the previous two computations of the decay constant. As a result of this, the analysis graphs are omitted from the report.

Length (±0.03)	γ	$\Delta\gamma$
0.40	113.12	8.16
0.50	106.06	7.20
0.60	117.98	6.77
0.70	106.81	4.61
0.80	111.58	5.24
0.90	116.07	5.66
1.00	109.06	5.49

Table 7. Table representing the variation between the length of the string and its decay constant. The length remained constant within the range of uncertainties given.

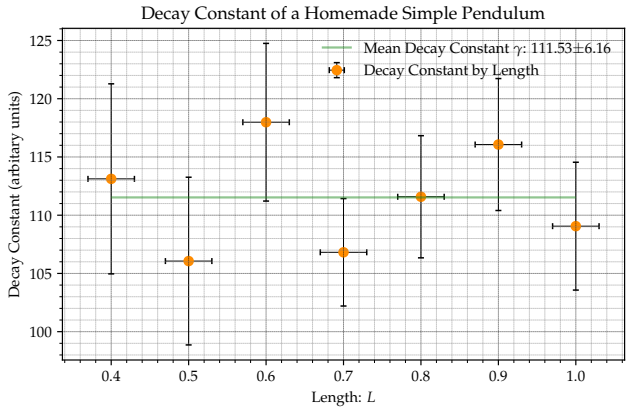

Figure 16. This is the plot representing the data shown above. The decay constant parameters are shown with their mean value. All computed parameters with their uncertainty fell within the mean value shown in the plot.

Lengths	Time Period	Time Period Error	Chi-Squared
1.00	1.953	0.0486	4.93
0.90	1.852	0.0467	4.33
0.80	1.741	0.0457	3.49
0.70	1.613	0.0429	0.64
0.60	1.476	0.0476	-13.60
0.50	1.344	0.0583	-26.61
0.40	1.187	0.0475	24.85

Table 8. Table representing the relationship between the length of the pendulum and its time period computed from `curve_fit()`.

5.8 Measuring acceleration due to gravity: g (Length vs Time)

The fitting to compute the value of the acceleration due to gravity was computed as per the equation highlighted in the *acceleration due to gravity* section of the theoretical background. This section shows the results of that fitting along with its uncertainty.
The calculation of the acceleration due to gravity has been moved to Appendix A on the recommendation of Professor Wilson.

6 DISCUSSION

The following section will collect all the results in a textual format. Its analysis will be discussed followed by the errors involved with the experiment.

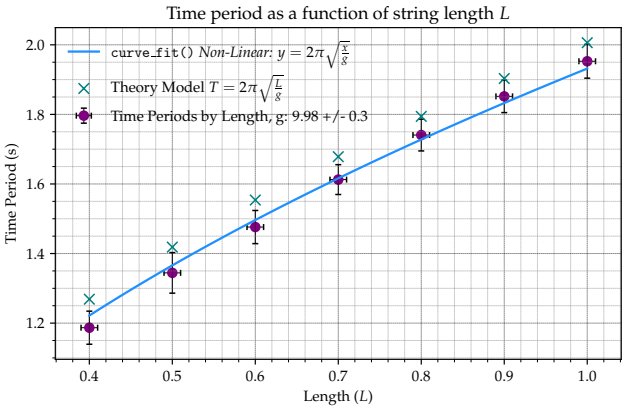

Figure 17. Plot representing the relationship between the length of the pendulum and its time period. As the length was increased, the time period also increased following the characteristic $2\pi\sqrt{x/g}$ relationship highlighted in the `curve_fit()` model.

6.1 Analysis

In the first experimental setup, in which the mass of the pendulum was varied, the oscillations were first allowed to fall below the small angle approximation. The values from the results indicate that the oscillations were exponentially decaying. The decay constant γ decreased as the mass was increased, the damping coefficient c remained constant with a mean value of $c = 1.479 \pm 0.3$ (units). The errors for the damping coefficient increased with mass.

Following exponential decay, the symmetry of the pendulum was computed. The pendulum was determined to be symmetric by taking the difference of the positive and negative peaks. If the sum of the absolute difference was below 0.02 for small angles oscillations, and 0.05 for all oscillations the pendulum was said to be symmetric. The sample value computed was 0.017 which was well smaller than the threshold set for symmetry. This was repeated in every single case and **the pendulum was found to be symmetric every time**. Since over 500 values of difference were taken, the errors associated with this difference were orders of magnitudes below the threshold.

The time period of the varying mass was now computed. The time period was found to be $T_{exp} = 2.0768s \pm 0.05$ which matched both the theoretical model proposed by Professor Wilson, and myself. T_{exp} was also found to agree with the theoretical model in remaining constant, the only exception to this was mass $M1$. This was due to the non-negligible friction and the mass of the string. The goodness of fit χ^2 for the `curve_fit()` varies by model, the largest being mass M5 and the smallest being mass M3. This variation was also evident in the error analysis. Large values of χ^2 reduce the reliability of the result, hence the data collected from mass $M5$ must be trusted less than that from mass $M3$. This did not affect the verification of the theoretical model.

An auxiliary goal of the experiment was to determine the equations of motion. The results were as expected, the damped angular frequency increased with mass, the decay constant decreased with mass (due to the damping coefficient) and the amplitude stayed constant. Alpha α, was found to vary greatly. This is not surprising as it was not set as a control variable in this particular experiment. Taking 5-10 trials instead of 3 will reduce the uncertainty here.

Now lets consider the analysis of the second experimental setup in which the release angle θ_0 was varied. The motion once again followed underdamped oscillations, the value of the decay constant once again remained constant with a mean value of $\gamma = 77.74 \pm 4$, this once again confirmed the theoretical predictions. The time period for this experimental setup also remained constant with a mean value of $T_{angles} = 1.935 \pm 0.04$. Once again, this is within the values of both my and Professor Wilson's models. The uncertainty in time period was found to be very large. This was due to the lack of a good fit (due to higher χ^2). Hence, this result should not be fully trusted without further investigation. Looking at this holistically, the values of the decay parameter and time period were expected to change, since we assume a non-ideal environment. But they did not change. The fitting function forced the equation into underdamped motion giving the time period which matched the theoretical model. This means the large, non-linear oscillations were ignored resulting in idealized results which was what was observed, hence constant time period and decay parameter. As a result **the result should not be fully trusted**. The fitting `curve_fit()` used the theoretical model to fit the data to match the theoretical model. While the fitting was successful, it does not fully represent the data.

In the final experimental setup the length of the string was varied. It was found that changing the length of the string did not alter the value of the decay constant which stayed within its uncertainties at $\gamma = 111.53 \pm 6.16$. This once again matched the theoretical model. The errors in the decay constant, although large, stayed within reasonable bounds. Following this, the change in time period was not constant. This was used to compute the acceleration due to gravity g. This was found to be 9.98 ± 0.3 which is within 2% of the known value of gravity in Toronto. The errors in the time period remained relatively uniform at around ± 0.04, values of lengths smaller than $L < 0.5$ had large values of χ^2 reducing the reliability in their respective time periods. In general, their uncertainties stayed low, so the data was still considered to be reliable. In each section, there were many potential sources of errors.

6.2 Error Discussion

The experiment consisted of three different setups, one with the mass varied, another with the angle of release varied, and a final one with the length varied. Each setup exposed a different set of potential errors. This section is dedicated to studying those sources of errors.

The first possible source of error would be the spinning of the mass held inside the ziplock bag. The spinning would create torsional energy which would then be converted into kinetic energy as the object oscillated. This can be seen in Figure 6 in which the first potion of the oscillations has varying amplitudes which did not match the exponential decay curve. The two solutions to dealing with this was widen the string grip holding the masses, and to wait for the spinning to stop before the video analysis took place.

A second possible source of error could have been the non-negligible mass of the paperclip. This clip has a non-negligible mass which adds to the total mass of the pendulum while altering its moment of inertia. Since the mass of the clip was much less than that masses, this did not alter the results.

Another possible source of error was the elasticity of the string. At a fully length of 1.09m the string would be able to stretch up to an additional 1cm given enough downward force. This would increase the length of the string thus increase the time periods of the larger masses. It is likely that the masses were not able to apply enough force for this effect to occur. In either case, the error in the string length was increased due to this phenomenon, **this explains the larger errors in the time period analysis**.

One final source of error is in the assumptions of the theoretical model. The models make multiple assumptions which are nonphysical. For example, both theoretical models assume that the string is massless. For values of small mass, the mass of the string would play a larger role in skewing the results. These assumptions would add variability between the computed result and the theory. It was surprising to see that most values feel within the given range of uncertainties.

7 CONCLUSION

To conclude, the goal of this experiment was to build a simple homemade pendulum and study its motion by changing different parameters such as m, L and θ_o. The pendulum was built with a custom designed tracker to determine the motion. First, the periodicity of the pendulum was confirmed using a custom difference algorithm. Following this, other parameters were studied.

For the first experimental setup, the mass was varied. The time period of the oscillations were found to be independent of mass with a value of $2.0768s \pm 0.05s$ this was within 1% of the value from the first theoretical model: $T_{myself} = 2.094 \pm 0.09s$ and less than 1% from the second theoretical model: $T_{Wilson} = 2.088 \pm 0.09$. In addition to this, the damping coefficient was found to be independent of mass with a mean value of 1.48. Most values fell within their uncertainty range of this number sans the lightest mass.

When directly comparing the varied release angles $\theta_0 \pm 0.3$, time period and decay parameter. Both the decay parameter and time period were found to be constant $T_{angles} = 1.935 \pm 0.04$, $\gamma = 78 \pm 4$. This matched the theoretical predictions for this model. Due to the fitting model assuming the theoretical models during fitting, these results cannot be fully trusted.

The final experimental setup consisted of varying the lengths of the pendulum. The decay constant was found to remain constant, as expected. The decay constant was computed to be $\gamma = 111 \pm 6$, the time period varied. A non-linear fitting was applied to the change in the time period which was used to calculate the acceleration due to gravity g. This was computed to be $g = 9.98 \pm 0.03$ which is within 2% of the known value.

Almost all elements of the experiment were considered to be a success with the exception of the forceful `curve_fit()` mentioned above. This was because the experiment was being constantly improved on. The success of the data collection was due to multiple repeats of redesigning the pendulum. There were mistakes in each iteration which were not previously noticed. Only the final version of the pendulum is being presented in this paper.

If the experiment were to be repeated, additional trials should be taken to reduce the standard deviation of uncertainties. Overall,

the experiment was a success with key concept of an underdamped pendulum tested.

ACKNOWLEDGEMENTS

I wish to acknowledge this land on which the University of Toronto operates. For thousands of years it has been the traditional land of the Huron-Wendat, the Seneca, and most recently, the Mississauga's of the Credit River. Today, this meeting place is still the home to many Indigenous people from across Turtle Island and we are grateful to have the opportunity to work on this land.

I also wish to acknowledge all the repositories of information and code that were made available under the open-source commons GitHub license. In particular some work also abides by the MIT Public License. All such work will be duly credited.

Finally I wish to acknowledge Professor Brian Wilson for assigning us this project. Building and fine tuning the apparatus has been one of the highlights of the year, it has reminded me why I enjoy (and dislike) experimental physics. I hope this project becomes a mainstay in the PHY324 syllabus. It gives us enough time to engage a style of thinking that I have only experienced at this intensity in my previous summer research internships.

DATA AVAILABILITY

The data used is proprietary and limited to the measurements taken by the author. All the analysis code used is listed in the appendix.

REFERENCES

Aggarwal N., Verma N., Arun P., 2005, European journal of physics, 26, 517

Fulcher L. P., Davis B. F., 1976, American Journal of Physics, 44, 51

Goyal K., Agarwal K., Kumar R., 2017, in 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA). pp 474–478

Larsen K., Pankratz C., Giles B., Kokkonen K., Putnam B., Schafer C., Baker D., Burch J., 2016, in EGU General Assembly Conference Abstracts. pp EPSC2016–10061

Villán A. F., 2019, Mastering OpenCV 4 with Python: a practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7. Packt Publishing Ltd

APPENDIX A: APPENDIX A: SAMPLE ERROR CALCULATIONS

A1 Sample Time Period Calculations with Errors

Here the formula for error quadrature is used to determine the uncertainty in time for each specific mass. **I will show the example for the time period and errors in Mass M3.**

$$T_{M3} = \frac{2\pi}{\sqrt{\omega_1^2 + \gamma^2}} = \frac{2\pi}{\sqrt{3.0502^2 + 0.012073^2}} = 2.0599s$$

$$S_{err} = \frac{1}{\sqrt{w_o^2 + \gamma^2}}\sqrt{w_{err}^2 + \gamma_{err}^2}$$

$$T_{err} = T\left(\sqrt{\frac{w_0}{2}}S_{err}\right) \tag{A1}$$

$$= 2.06\left(\sqrt{\frac{3.05}{2}}\frac{1}{\sqrt{3.06}}\sqrt{1.62\times10^{-4} + 6.4\times10^{-7}}\right) \tag{A2}$$

$$= 0.02467s \tag{A3}$$

Combining these we get out value for the time period of mass $M3$, $T_{M3} = 2.06s \pm 0.025s$ as shown in the plot.

A2 Sample Damping Coefficient Calculations with Errors

The formula for error quadrature along with the formula for the damping coefficient was used to calculate the error in the damping coefficient. **I will show a sample calculation for Mass M3.** The gamma error for M3 was $\gamma_{err} = 2.7\times10^{-5}$ obtained from `curve_fit()`. **Unit conversions have been ignored** since gamma is a unitless number.

$$c = 2m\gamma = 2(60)(0.01233) = 1.4796$$

$$c_{err} = c\sqrt{\left(\frac{M_{err}}{M}\right)^2 + \left(\frac{\gamma_{err}}{\gamma}\right)^2} \tag{A4}$$

$$= 1.4496\sqrt{\left(\frac{0.01}{60}\right)^2 + \left(\frac{2.7\times10^{-5}}{0.01208}\right)^2} = 0.3277 \tag{A5}$$

A3 Sample Errors: Acceleration due to Gravity

After measuring the time periods in both the theoretical models and the experimental setup, these values can be used to compute the acceleration due to gravity g. By doing this, more elements can be found which will ultimately give the constants highlighted in the equations of motion.

$$g = \frac{4\pi^2 L}{T_{exp}^2} = \frac{4\pi^2(1.09)}{2.0768^2} = 9.97792ms^{-2}$$

Once again, using errors in quadrature, we compute the uncertainty in g. Notice we are computing the uncertainty in the square value of T.

$$g_{err} = g\sqrt{\left(\frac{L_{err}}{L}\right)^2 + \left((T_{exp}/2)\frac{T_{exp-err}}{T_{exp}}\right)^2} \tag{A6}$$

$$= 9.97792\sqrt{\left(\frac{0.05}{1.09}\right)^2 + \left((2.0768/2)\frac{0.05}{2.0768}\right)^2} = \pm0.321 \tag{A7}$$

Using this the experimental value of the acceleration due to gravity $g = 9.98 \pm 0.3ms^{-2}$

APPENDIX B: APPENDIX B: PYTHON CODE `PYTHON3.8`

B1 Time Period

```
print("[STATUS] Initializing...")
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
from scipy.optimize import curve_fit
from scipy.stats import chisquare
```

```
mpl.rcParams['legend.frameon'] = False
mpl.rcParams['figure.autolayout'] = True

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "sans-serif",
    "font.sans-serif": ["Helvetica"]})

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
})


def utkarshGrid():
    plt.minorticks_on()
    plt.grid(color='grey',
            which='minor',
            linestyle=":",
            linewidth='0.1',
            )
    plt.grid(color='black',
            which='major',
            linestyle=":",
            linewidth='0.1',
            )


def underdamped_oscillator(t, Amp, gamma, omega, alpha):
    exponential_factor = np.exp(-gamma * t)
    cos_factor = Amp * np.cos(omega * t + alpha)
    return exponential_factor * cos_factor


def centralizeX(_data):
    # centerX = (_data.x.max() + _data.x.min()) / 2
    # _data["x"] = centerX - _data.x
    _data["x"] = np.mean(_data.x) - _data.x
    return _data


def wrapUp():
    print("[STATUS] All Done!")


print('[STATUS] Stage 1...')

# Load Data
data1 = pd.read_csv('m1.csv')
data2 = pd.read_csv('m2.csv')
data3 = pd.read_csv('m3.csv')
data4 = pd.read_csv('m4.csv')
data5 = pd.read_csv('m5.csv')
data6 = pd.read_csv('m6.csv')

# Convert to real time
m1Time = 2 * 60 + 24
m2Time = 2 * 60 + 50
m3Time = 4 * 60 + 3
m4Time = 6 * 60 + 14
m5Time = 7 * 60 + 17
m6Time = 9 * 60 + 35

data1['time'] = np.linspace(0, m1Time, len(data1.x))
data2['time'] = np.linspace(0, m2Time, len(data2.x))
data3['time'] = np.linspace(0, m3Time, len(data3.x))
data4['time'] = np.linspace(0, m4Time, len(data4.x))
data5['time'] = np.linspace(0, m5Time, len(data5.x))
data6['time'] = np.linspace(0, m6Time, len(data6.x))

data1 = centralizeX(data1)
data2 = centralizeX(data2)
data3 = centralizeX(data3)
data4 = centralizeX(data4)
data5 = centralizeX(data5)
data6 = centralizeX(data6)

# Plot Figures
plt.figure()
utkarshGrid()
plt.plot(data5.time, data5.x, linewidth=0.5,
     label="Complete Motion", color="dodgerblue")
plt.ylabel("Position $r(t)$ (m)")
plt.xlabel(r"Time ($s$)")
xmin = 100
xmax = data5.time.max() - 10
plt.axvspan(xmin, xmax, label="Stable Motion",
     color="red", alpha=0.1)
plt.tight_layout()
plt.legend()
plt.title("Sample Data using Mass-5")
plt.savefig("fig1.pdf")

EOM = pd.DataFrame(columns=['Mass','Omega', "OmegaErr",
     'Gamma', "GammaErr", 'Amp', "AmpErr", 'Alpha',
     "AlphaErr"])
mass_list = [1,2,3,4,5,6]
omega_list = [3.384363748879389, 3.298038983067513,
     3.2482444198618436, 3.236717918650125,
     3.1968754066813188,
            3.175153037033223]
gamma_list = [0.039384731537753806, 0.022600942462282572,
     0.017004989469935578, 0.013809010292495013,
           0.010063699780439006, 0.007861198189241804]
Amp_list = [0.20770949541621692, 0.19107836037464626,
     0.2078927306662884, 0.24390768684178987,
     0.2372968740176148,
          0.19102950043602088]
Alpha_list = [0.20176968713558593, 0.313439547407809,
     0.25918052879539766, 0.1126303928487382,
     0.01326325726679072,
            0.8349842549813317]

omega_err_list = [0.00017122461801893305 ,
     0.00010943583477115274, 8.808102703353578e-05,
     8.448552618128108e-05, 5.5869172375435496e-05,
     2.465447755484106e-05]
gamma_err_list = [0.00016987079908491114 ,
     0.0001101461627217873, 8.781082616708 63e-05,
     8.432341457410364e-05, 5.581301683408848e-05,
     2.472540131864058e-05]
Amp_err_list = [0.0006013334594600011,
     0.000527078164400348, 0.0007294759393692631,
     0.0010132796175279298, 0.0008664533286610733,
     0.0003367855633287493]
Alpha_err_list = [0.0029330886221547404,
     0.0027344647170478254, 0.00353143405293673,
     0.004167191257674186, 0.0036554444205268385,
     0.0017524902055905621]

EOM["Mass"] = mass_list
EOM["Omega"] = omega_list
EOM["Gamma"] = gamma_list
EOM["Amp"] = Amp_list
```

```
EOM["Alpha"] = Alpha_list
EOM["OmegaErr"] = omega_err_list
EOM["GammaErr"] = gamma_err_list
EOM["AmpErr"] = Amp_err_list
EOM["AlphaErr"] = Alpha_err_list


def fig_and_chiSquared(_data, number, cqs_list):
    print(f"\n[STATUS] Starting Data Point {number}")
    guess = [0.06, 0.002, (2 * np.pi / 3), 2]

    popt, pcov = curve_fit(underdamped_oscillator,
                           xdata=_data.time,
                           ydata=_data.x,
                           p0=guess
                           )
    omeg = popt[2]
    gamma = popt[1]
    Amp = popt[0]
    phase = popt[3]
    # print("Omega: ",omeg,"\nGamma: ", gamma,"\nAmp: ",
        Amp,"\nAlpha: ", phase)
    theory = underdamped_oscillator(_data.time, Amp,
        gamma, omeg, phase)
    exp = Amp * np.exp(-gamma * _data.time)

    p_err = np.sqrt(np.diag(pcov))

    omeg_err = p_err[2]
    gamma_err = p_err[1]
    Amp_err = p_err[0]
    phase_err = p_err[3]
    print("OmegaErr: ", omeg_err, "\nGammaErr: ",
        gamma_err, "\nAmpErr: ", Amp_err, "\nAlphaErr:
        ", phase_err)

    summ = omeg ** 2 + gamma ** 2
    w0 = np.sqrt(summ)
    T = (2 * np.pi) / w0

    # Compute Chi Squared
    cqs, p = chisquare(_data.x, theory)

    # Tracking Error was within 40 pixels
    tracking_err = 0.02

    # Computing errors in quadrature
    omeg_sqaured_err = omeg * 2 * omeg_err
    gamma_squared_err = gamma * 2 * gamma_err
    sum_err = np.sqrt(omeg_sqaured_err * 2 +
        gamma_squared_err * 2)
    w0_err = w0 * 0.5 * sum_err
    quadrature_err = T * sum_err / summ
    T_err = quadrature_err * np.sqrt(abs(cqs)) +
        tracking_err
    # print(omeg_sqaured_err, gamma_squared_err)

    # Curve Fit
    plt.figure(figsize=(6, 3))
    utkarshGrid()
    plt.plot(_data.time, theory, linewidth=0.5,
        label=r"Data: \texttt{curve\_fit()}, $\chi^2
        =$"f"{round(cqs, 2)}",
           color="purple")
    plt.scatter(_data.time, _data.x, label=r"Data:
        \textit{Original}", color="dodgerblue", s=0.15,
        marker="x")
    plt.plot(_data.time, exp, linewidth=2,
             label=r"Decay: $e^{\gamma t}$"f", T =
                 {round(T, 3)}"r"$\pm$"f"{round(T_err,
                 4)}", color="goldenrod")
    plt.plot(_data.time, -exp, linewidth=2,
         color="goldenrod")
    plt.ylabel("Position $r(t)$ (m)")
    plt.xlabel(r"Time ($s$)")
    plt.tight_layout()
    plt.legend()
    plt.title(f"Mass-{number} Decay Model- Fitting")
    plt.savefig(f"fig2m{number}.pdf")

    cqs_list.append(cqs)

    print(f"Mass{number}- Time Period: {round(T, 3)} +/-
        {round(T_err, 9)}")
    print(f"Mass{number}- Chi-Squared: {cqs}\n")
    return T, T_err


cq_list = []

t1, t1_err = fig_and_chiSquared(data1, 1, cq_list)
t2, t2_err = fig_and_chiSquared(data2, 2, cq_list)
t3, t3_err = fig_and_chiSquared(data3, 3, cq_list)
t4, t4_err = fig_and_chiSquared(data4, 4, cq_list)
t5, t5_err = fig_and_chiSquared(data5, 5, cq_list)
t6, t6_err = fig_and_chiSquared(data6, 6, cq_list)

xRanges = np.arange(1, 7, 1)
massTable = pd.DataFrame(columns=['Mass Number', 'Mass'])
massTable["Mass"] = xRanges
massTable["Mass Number"] = xRanges
# massTable["Chi-Squared"] = np.array(cq_list)

print('[STATUS] Completed Stage 1...')
print('[STATUS] Stage 2...')

T_arr = np.array([t1, t2, t3, t4, t5, t6])
T_arr_err = np.array([t1_err, t2_err, t3_err, t4_err,
    t5_err, t6_err])

L = 1.09

plt.figure(figsize=(6, 4))
utkarshGrid()
T_mean = float(np.mean(T_arr[1:]))
T_mean_err = np.mean(T_arr_err)
print(f"Mean Time Period: {T_mean}s +/- {T_mean_err}s")
theoryT = 2 * np.pi * np.sqrt(L / 9.81)
print(f"Theoretical Time Period: {theoryT}")
plt.hlines(theoryT, min(xRanges) - 1, max(xRanges) + 1,
         label=r"Theory Model $T =
             2\pi\sqrt{\frac{L}{g}}$=
             "f"{round(theoryT, 3)}s", color="teal",
         alpha=0.5, zorder=3)
plt.hlines(2*np.sqrt(L), min(xRanges) - 1, max(xRanges) +
    1,
         label=r"Wilson Model $T = 2\sqrt{L}$=
             "f"{round(2*np.sqrt(L), 3)}s",
             color="lime",
         alpha=0.5, zorder=3)
plt.hlines(T_mean, min(xRanges) - 1, max(xRanges) + 1,
         label=r"\texttt{np.mean(T)}= "f"{round(T_mean,
             3)}s", color="salmon",
         alpha=0.5, zorder=3)
plt.errorbar(x=xRanges,
           y=T_arr,
```

```
            yerr=T_arr_err,
            xerr=0.1,
            elinewidth=0.5,
            capsize=2,
            ecolor='black',
            fmt='o',
            label="Time Periods by Mass",
            zorder=1
            )

plt.scatter(1, t1, color="red", label="Additional
    Friction (Ziplock-Bag)", zorder=2)
plt.xlabel("Mass Weight (non-liner arbitary units)")
plt.ylabel("Time Period (s)")
plt.legend()
plt.title("Time Period of Homemade Simple Pendulum")
plt.savefig("fig3.pdf")

tTable = pd.DataFrame(columns=['Mass Number', 'Time
    Period', "Time Period Error", "Chi-Squared"])
tTable["Mass Number"] = xRanges
tTable["Chi-Squared"] = np.array(cq_list)
tTable["Time Period"] = T_arr
tTable["Time Period Error"] = T_arr_err

from tabulate import tabulate

print(tabulate(tTable, tablefmt="latex",
            floatfmt=(".0f", ".2f", ".2e", ".2f", ".2e",
                ".2f", ".2e", ".2f", ".2e"),
            headers=list(tTable.columns),
            showindex=False
            ))

wrapUp()
```

B2 Decay Constant

```
print("[STATUS] Initializing...")
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
from scipy.optimize import curve_fit
from scipy.stats import chisquare

mpl.rcParams['legend.frameon'] = False
mpl.rcParams['figure.autolayout'] = True

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "sans-serif",
    "font.sans-serif": ["Helvetica"]})

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
})


def utkarshGrid():
    plt.minorticks_on()
    plt.grid(color='grey',
            which='minor',
            linestyle=":",
            linewidth='0.1',
            )
    plt.grid(color='black',
            which='major',
            linestyle=":",
            linewidth='0.1',
            )


def underdamped_oscillator(t, Amp, gamma, omega, alpha):
    exponential_factor = np.exp(-gamma * t)
    cos_factor = Amp * np.cos(omega * t + alpha)
    return exponential_factor * cos_factor


def centralizeX(_data):
    # centerX = (_data.x.max() + _data.x.min()) / 2
    # _data["x"] = centerX - _data.x
    _data["x"] = np.mean(_data.x) - _data.x
    return _data


def wrapUp():
    print("[STATUS] All Done!")


print('[STATUS] Stage 1...')

# Load Data
data1 = pd.read_csv('m1.csv')
data2 = pd.read_csv('m2.csv')
data3 = pd.read_csv('m3.csv')
data4 = pd.read_csv('m4.csv')
data5 = pd.read_csv('m5.csv')
data6 = pd.read_csv('m6.csv')

# Convert to real time
m1Time = 2 * 60 + 24
m2Time = 2 * 60 + 50
m3Time = 4 * 60 + 3
m4Time = 6 * 60 + 14
m5Time = 7 * 60 + 17
m6Time = 9 * 60 + 35

data1['time'] = np.linspace(0, m1Time, len(data1.x))
data2['time'] = np.linspace(0, m2Time, len(data2.x))
data3['time'] = np.linspace(0, m3Time, len(data3.x))
data4['time'] = np.linspace(0, m4Time, len(data4.x))
data5['time'] = np.linspace(0, m5Time, len(data5.x))
data6['time'] = np.linspace(0, m6Time, len(data6.x))

data1 = centralizeX(data1)
data2 = centralizeX(data2)
data3 = centralizeX(data3)
data4 = centralizeX(data4)
data5 = centralizeX(data5)
data6 = centralizeX(data6)

data_list = [data1, data2, data3, data4, data5, data6]
number_arr = [1, 2, 3, 4, 5, 6]
t_violation_arr = [0, 0, 10, 0, 0, 0]
tau_list = []
gamma_list = []
gamma_err_list = []

for i in range(len(t_violation_arr)):
    _data = data_list[i]
```

```
    number = number_arr[i]
    t_violation = t_violation_arr[i]

    guess = [0.06, 0.002, (2 * np.pi / 3), 2]

    popt, pcov = curve_fit(underdamped_oscillator,
                        xdata=_data.time,
                        ydata=_data.x,
                        p0=guess
                        )
    omeg = popt[2]
    gamma = popt[1]
    Amp = popt[0]
    phase = popt[3]

    p_err = np.sqrt(np.diag(pcov))

    omeg_err = p_err[2]
    gamma_err = p_err[1]
    Amp_err = p_err[0]
    phase_err = p_err[3]

    theory = underdamped_oscillator(_data.time, Amp,
         gamma, omeg, phase)
    exp = Amp * np.exp(-gamma * _data.time)
    tau = 1 / gamma
    tau_list.append(tau)
    gamma_list.append(gamma)
    gamma_err_list.append(gamma_err)

    # Plot Figures
    plt.figure(figsize=(6,3))
    utkarshGrid()
    plt.plot(_data.time, _data.x, linewidth=0.75,
         color="dodgerblue")
    plt.plot(_data.time, exp, linewidth=3,
         color="goldenrod",
            label=r"Decay: $e^{\gamma t}$,
                 "r"$\gamma=$"f"{round(gamma,
                 4)}"r"$\pm$"f"{round(gamma_err, 6)}")
    plt.plot(_data.time, -exp, linewidth=3,
         color="goldenrod")
    plt.axvspan(0, t_violation, label="Beyond Small Angle
         Approximation", color="limegreen", alpha=0.3)
    plt.ylabel("Position $r(t)$ (m)")
    plt.xlabel(r"Time ($s$)")
    plt.tight_layout()
    plt.legend()
    plt.title(f"Determining Underdamped Decay using
         Mass-{number}")
    plt.savefig(f"fig4m{number}.pdf")

    # print(f"Mass{number}- Decay Constant: {round(tau,
         3)} +/- {round(0, 9)}")

print("[STATUS] Computing Damping Coefficient")

masses = np.array([10, 25, 60, 90, 110, 160]) # g
mass_err = np.ones(len(masses)) * 1 # g

gamma_list = np.array(gamma_list)
gamma_err_list = np.array(gamma_err_list)


c = 2 * masses * gamma_list
c_err = masses * np.sqrt((mass_err / masses) ** 2 +
     (gamma_err_list / gamma_list) ** 2)

dTable = pd.DataFrame(columns=['Mass Number', 'Mass',
     "Gamma","Damping Coefficient", "Damping Coefficient
     Error"])
dTable['Mass Number'] = np.arange(1, 7, 1)
dTable['Mass'] = masses
dTable["Gamma"] = gamma_list
dTable['Damping Coefficient'] = c
dTable['Damping Coefficient Error'] = c_err

plt.figure(figsize=(6, 4))
utkarshGrid()
c_mean = float(np.mean(c))
plt.hlines(c_mean, min(dTable["Mass"]),
     max(dTable["Mass"]),
          label=r"Mean Damping Coefficient
               $\gamma$:"f"{round(c_mean, 2)}",
               color="forestgreen",
          alpha=0.5, zorder=3)
plt.errorbar(x=dTable["Mass"],
            y=dTable["Damping Coefficient"],
            yerr=dTable["Damping Coefficient Error"],
            xerr=mass_err,
            elinewidth=0.5,
            capsize=2,
            ecolor='black',
            fmt='o',
            label="Damping Coefficient by Mass",
            color = "darkorange",
            zorder=1
            )

plt.scatter(dTable["Mass"][0], dTable["Damping
     Coefficient"][0],
          color="red", label="Additional Friction
               (Ziplock-Bag)", zorder=2)
plt.xlabel("Mass (g)")
plt.ylabel("Damping Coefficeint (arbitary units)")
plt.legend()
plt.title("Linear Damping Coefficient of a Homemade
     Simple Pendulum")
plt.savefig("fig5.pdf")

from tabulate import tabulate

print(tabulate(dTable, tablefmt="latex",
             floatfmt=(".0f", ".0f", ".4f", ".3f", ".4f"),
             headers=list(dTable.columns),
             showindex=False
             ))
```

B3 Symmetry

```
print("[STATUS] Initializing...")
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
from scipy.optimize import curve_fit
from scipy.stats import chisquare
from scipy.signal import find_peaks

mpl.rcParams['legend.frameon'] = False
mpl.rcParams['figure.autolayout'] = True

plt.rcParams.update({
```

```
    "text.usetex": True,
    "font.family": "sans-serif",
    "font.sans-serif": ["Helvetica"]})

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
})


def utkarshGrid():
    plt.minorticks_on()
    plt.grid(color='grey',
            which='minor',
            linestyle=":",
            linewidth='0.1',
            )
    plt.grid(color='black',
            which='major',
            linestyle=":",
            linewidth='0.1',
            )


def underdamped_oscillator(t, Amp, gamma, omega, alpha):
    exponential_factor = np.exp(-gamma * t)
    cos_factor = Amp * np.cos(omega * t + alpha)
    return exponential_factor * cos_factor


def centralizeX(_data):
    # centerX = (_data.x.max() + _data.x.min()) / 2
    # _data["x"] = centerX - _data.x
    _data["x"] = np.mean(_data.x) - _data.x
    return _data


def wrapUp():
    print("[STATUS] All Done!")


print('[STATUS] Stage 1...')

data90 = pd.read_csv('l90.csv')
l90Time = 2 * 60 + 53
data90['time'] = np.linspace(0, l90Time, len(data90.x))
data90 = centralizeX(data90)

_data = data90
number = 90
t_violation = 0

guess = [-5.23360498e+03, 2.24496234e-01, 5.28912890e+00,
     -1.05371727e+02]

if t_violation > 0:
    fitData = _data
    for i in range(len(_data.time)):
        if abs(_data.time[i] - t_violation) < 0.3:
            fitData = _data[i:]
else:
    fitData = _data

popt, pcov = curve_fit(underdamped_oscillator,
                    xdata=fitData.time,
                    ydata=fitData.x,
                    p0=guess,
                    maxfev=10000
                    )
omeg = popt[2]
gamma = popt[1]
Amp = popt[0]
phase = popt[3]

p_err = np.sqrt(np.diag(pcov))

omeg_err = p_err[2]
gamma_err = p_err[1]
Amp_err = p_err[0]
phase_err = p_err[3]

theory = underdamped_oscillator(_data.time, Amp, gamma,
     omeg, phase)
exp = Amp * np.exp(-gamma * _data.time)
tau = 1 / gamma

tau_err = tau * np.sqrt(gamma_err / gamma)

# fitData = fitData[:int(len(fitData)/1.5)]

# Find Peaks
peaks_ind = find_peaks(fitData.x, height=0.01)[0]
peaksData = fitData.iloc[peaks_ind]
peaks_time = peaksData.time
peaks_x = peaksData.x

negpeaks_ind = find_peaks(-fitData.x, height=0.01)[0]
negpeaksData = fitData.iloc[negpeaks_ind]
negpeaks_time = negpeaksData.time
negpeaks_x = negpeaksData.x

# Plot Figures
exp_color = "goldenrod"
plt.figure(figsize=(6, 3))
utkarshGrid()
plt.plot(_data.time, _data.x, linewidth=0.75,
     color="grey", label=r"\textit{Original Data}",
     alpha=0.5)
plt.scatter(peaks_time, peaks_x, label=r"Positive Peaks",
     s=3, marker="x", color="red")
plt.scatter(negpeaks_time, negpeaks_x, label="Negative
     Peaks", s=3, marker="x", color="blue")
# plt.plot(_data.time, exp, linewidth=2,
     color="goldenrod",
#         label=r"Decay: $e^{\frac{-t}{\gamma}}$,
     "r"$\gamma=$"f"{round(tau,
     2)}"r"$\pm$"f"{round(tau_err, 3)}")
# plt.plot(_data.time, -exp, linewidth=2,
     color="goldenrod")
if t_violation > 0:
    plt.axvspan(0, t_violation, label="Beyond small-angle
         approximation", color="limegreen", alpha=0.3)
plt.ylabel("Position $r(t)$ (m)")
plt.xlabel(r"Time ($s$)")
plt.tight_layout()
plt.legend()
plt.title(r"Symmetry Peaks with sample data using
     \texttt{scipy.signal.find\_peaks()}")
plt.savefig(f"fig30.pdf")

difference = peaks_x[1:].reset_index() +
     negpeaks_x.reset_index()

realDifference = difference.x
```

```
plt.figure(figsize = (6,3))
utkarshGrid()
plt.scatter(peaks_time, peaks_x, label=r"Positive Peaks",
     s=3, marker="x", color="red")
plt.scatter(negpeaks_time, negpeaks_x, label="Negative
     Peaks", s=3, marker="x", color="blue")
plt.hlines(0, xmin=0, xmax=max(peaks_time),
     color="limegreen", linewidth=1, alpha=0.6, label =
     "Expected Difference")
plt.scatter(peaks_time[1:], realDifference,
     label=r"Difference", s=3, marker="x", color="green")
plt.xlabel("Time (s)")
plt.ylabel("Peak Difference (Amplitude)")
plt.title("Demonstrating pendulum symmetry")
plt.legend(ncol=2)

print(chisquare(realDifference,
     np.zeros(len(realDifference))))

plt.savefig("fig31.pdf")
wrapUp()
```

This paper has been typeset from a TEX/LATEX file prepared by the author.