Table of Contents

[1 Introduction 2](#_Toc155132348)

[1.1 Disclaimer 2](#_Toc155132349)

[1.2 Inspiration 2](#_Toc155132350)

[1.3 Purpose of this project 2](#_Toc155132351)

[1.4 Contents of this project 2](#_Toc155132352)

[1.5 Usage 3](#_Toc155132353)

[1.5.1 How to run the code and what is needed 3](#_Toc155132354)

[1.5.2 Intended use of project 3](#_Toc155132355)

[1.5.3 Limitations of project 3](#_Toc155132356)

[2 Packages and techniques used 4](#_Toc155132357)

[2.1 Packages/libraries used 4](#_Toc155132358)

[2.2 Techniques used 4](#_Toc155132359)

[3 Code details 4](#_Toc155132360)

[3.1 first.py 4](#_Toc155132361)

[3.2 second.py 5](#_Toc155132362)

[4 Background 6](#_Toc155132363)

[4.1 The experiment 6](#_Toc155132364)

[4.1.1 Assumptions made 7](#_Toc155132365)

[4.1.2 The math involved 7](#_Toc155132366)

[4.2 The equipment 8](#_Toc155132367)

[5 Results 8](#_Toc155132368)

[5.1 Results of first.py 8](#_Toc155132369)

[5.1.1 first.py graph and discussion 8](#_Toc155132370)

[5.1.2 challenges from first.py 9](#_Toc155132371)

[5.1.3 Discussion of statistical results 9](#_Toc155132372)

[5.2 Results of second.py 10](#_Toc155132373)

[5.2.1 A brief note about the statistical results, code and potential future updates 10](#_Toc155132374)

[5.2.2 second.py graph and discussion 10](#_Toc155132375)

[5.2.3 challenges from second.py 11](#_Toc155132376)

[5.2.4 Discussion of statistical results 11](#_Toc155132377)

# Introduction

In this section, I will be:

* Giving a disclaimer as I was granted permission to use data provided from a university and my old work from classes.
* Discussing the inspiration behind the project and the experiment.
* Discussing and detailing the purpose of this of the project and its results
* The contents of this project
* How to run the code for this project
* What is needed to run the code for this project
* The intended usage of this project
* The limitations of this project

## Disclaimer

\*\*\*From spring 2020 to fall 2021, UNT implemented a virtual classroom approach, due to the pandemic. During this time, I took Modern Physics Laboratory (PHYS 3030) in spring 2021. Before I continue, I must say I have gained permission from the UNT Department of Physics to use the data that was given to us, my original lab report, and my code.\*\*\*

## Inspiration

One of most major breakthroughs in science in the early 20th century regards proving that a photon could act like a particle. Prior to this experiment, physics was still ruled by a Newtonian view that light is a wave. This experiment showed that light can behave like a particle as when it is filtered through tiny slits, the interference will cause the count for the photons in a region, similar to how a particle behaves.

## Purpose of this project

The purpose of the application is to analyze experimentally generated data to demonstrate that light can behave like a particle as well as a wave. The application demonstrates the use of several Python packages to create data visualizations that graph the count (or intensity) of photons at each index vs the position along the back film opposite of a slit(s). Those results will then be processed to graph the distances between peaks (gained from the first graph) and the principal index and the distances between the peaks. The result of this is a linear graph with a slope equal to that of the wavelength of the photons shot through the slit(s).

## Contents of this project

This project contains the following:

1. original data that was experimentally gained in a CSV file (Single\_Photon\_Interference.csv)
2. 2 Python3 code files
   * first.py
   * second.py
3. this readme file
4. the original lab report submitted.

## Usage

### How to run the code and what is needed

This section will discuss how to run the code for this project. There are several packages that both rely on to run, as well as there is an outside csv file that is required to run one of them that must be in the same file folder/directory as it. In the following 3 sections, I will discuss what is needed for both to run, then I will discuss the specifics for first.py to be run, and then I will end with discussing what is needed to run second.py.

#### Both executables

The project contains two executables, first.py and second.py. To run both, you will need to run the code in an environment that can read in csv files and display data visualizations, you will also need to ensure that up-to-date versions of each of the following packages are installed:

* + - 1. Numpy-this is used for the numpy arrays mainly, as other packages are built on top of Numpy.
      2. Pandas-this is used to manage and manipulate data frames by cleaning data and moving data to more easily workable structures as well as reading in files.
      3. Matplotlib-This is mainly used for data visualization by graphing the experimentally gained data.
      4. Statistics-This is used to preform statistical analysis at various points in the lab analysis
      5. Scipy- This was used to sure up some of the statistical analysis in the lab, and was a main stay when programming my other labs.

#### First.py

In addition to the above, to run first.py, you will need to ensure following:

* + - 1. Ensure that the csv file (Single Photon Interference) is in the same directory/folder and immediately accessible to the code and is a csv file, not a xls file.

#### Second.py

There are no other specifications to run second.py. The executable is self-contained, notwithstanding the above mentioned required specifications and dependencies.

### Intended use of project

This project is meant to display the skill of Gerod as a python programmer. This project is intended to show graphically the results of a single photon interference lab and solve for the wavelength of the photons emitted by a bulb. This project is not meant to be a simulation, as such; to run this project, a csv file will need to be provided with column A having position in mm from the principal peak and column B having the count of photon interaction that occurred at the position.

### Limitations of project

This project is limited to Python 3, and this specific experiment. It is not resilient to other physics experiments nor will it be able to properly manage data if the data being given to it is altered beyond updating the data itself. This project may become obsolete and may need to be updated if some of the methods used from the installed packages and modules become outdated.

# Packages and techniques used

In this section, I will be discussing the packages and libraries used throughout this project as well as the techniques and outside knowledge that I used in coding this project.

## Packages/libraries used

For this project, I used the following Python3 packages/libraries:

* Numpy-this is used for the numpy arrays mainly, as other packages are built on top of Numpy.
* Pandas-this is used to manage and manipulate data frames by cleaning data and moving data to more easily workable structures as well as reading in files.
* Matplotlib-This is mainly used for data visualization by graphing the experimentally gained data.
* Scipy-This was used for this find\_peaks method found in the signals module to find the peaks of the data for the first graph as well as the stats module for some of the statistical analysis of the second.py file.
* Statistics-This was used to sure up some of the statistical analysis in the lab, and was a main stay when programming my other labs.

csv-this was used to assist in reading in the initial csv file at the start of the project.

## Techniques used

**For this project, I used the following techniques and outside information to complete this project:**

* Python 3: Used for the power of the language and the plethora of trusted libraries
* Signal module – The signal module from the Scipy package contains a built in method called find\_peaks that allows the user to find the peaks and minima of data when graphed based on user parameters
* I ran Poisson distribution – partially because that was part of the parameters that we were told in class, but also partially because we needed to find the distribution of instances within a fixed interval of time
* Ran gaussian Variation – for the statistical analysis at the end of the lab to determine if the experiment was successful or not as well as how accurate and precise the data was.
* Higher levels of calculus and partial derivatives were used to derive equations to measure what my class called “smear” or the uncertainty in our measurements per instance of measurement.

# Code details

In this section, I will be giving a line by line explanation of the code and what it does for both programs. I will be making note of when I use an outside package and why I am using it. I will also be describing the purpose of each of the programs written.

## first.py

The purpose of first.py is to graph the count of photons at each index vs the Position in mm. The results of this graph are used to form the second graph. I use Pandas and CSV to read in and manage data frames from a CSV file. I use the Numpy package to make use of the Numpy arrays, whereas other packages make use of them. I use Matplotlib to graph the data. I use the statistics package for the statistical analysis at the end of the file.

The results of the first graph result in something that looks like a mixed signal, where I make use of the Scipy package to find the peaks.

The first section of first.py deals with reading data from a CSV file into a data frame, making use of the pandas module and library and organizing the information to be easier to work with.

First.py first creates a list called column\_names so that columns would have names. It then opens the CSV file called Single\_Photon\_Interference.csv and then drops all columns after the first 2 due to them being empty. First.py then drops all NaN (not a number) entries in the data frame and then creates a new one based on the columns being deleted and dropping of NaN entries.

First.py then moves all values in Position(mm) and Counts to an array and type casts them as floats for both Position and Counts.

The second section of first.py deals with graphing the first graph, and finding the peaks and troughs using Matplotlib and the find\_peaks method from the signal module in the Scipy package.

First.py sets x-values to the array created from the position values and y-values to the Counts values. Then first.py has peaks as a dictionary, where; after experimentally tuning the function inputs, returns the peaks of the first graph using the find\_peaks method from the signal module in the Scipy package. First.py then uses the find\_peaks method to find the minima (troughs), after some fine-tuning of the function inputs again. First.py then makes a line graph of all the plots given and highlights the peaks as read and the minima as gold.

The third section of first.py deals with the statistical analysis of the information given.

In this section, first.py calculates the poison standard deviation and the error bars for the graph. It is important to note that since we were instructed to use the standard deviation of the points for the error bars, the error bars are there, but they are not visible. This is also due to how well the data was experimentally derived. Error bars exist for both the x and y directions for each point.

## second.py

The purpose of second.py is to graph the distances between peaks (gained from the first graph) and the principle index and the distances between the peaks. The result of this is a linear graph with a slope equal to that of the wavelength of the photons shot through the slit(s).

The results of this graph and statistics that were computed with Python are seen in the Lab report, showing low standard deviation and variance, it was shown that the data was accurate and precise, with an average percent error of less than one percent and a systematic error ratio of .78.

Second.py has a function called myfunc, where it returns an output (a y value) based on an x value that is manipulated by a variable named slope and a variable named intercept. Second.py then creates a first-degree polynomial fit based on the x and y values, and then runs a linear regression, returning the slope, intercept, r value, confidence value, and standard deviation.

Second.py then makes a list called mymodel from mapping x to the function myfunc returning y values. Second.py then makes the graph with labels based on the model and regression with a title, x, and y-axis, and a legend with error bars, but again the error bars are too small to be seen.

Second.py then prints the slope, intercept, r-value, confidence value, and standard deviation to the terminal. The next 4 lines deal with contestants that were given in class and algebraically solving for unknowns.

The next section of second.py is modular to save time on calculations. The functions return a wavelength list, the percent errors of a list, the average of a list, the sample standard deviation of a list, the expected errors for a list, and the average error of a list.

The next section of second.py is the main function of second.py. Main.py first makes a list of comparative lambdas and comparative errors based on the global variables. From there, python has the average wavelength and the average percent error. Python repeats a similar process for standard deviation and expected error in wavelengths. Main.py then repeats the whole process for the distance from the principal peak. Python then will print the results of the calculations and the results of a test to the terminal.

# Background

Since this project is based on a physics lab, this section will focus on discussing the following:

* Background of the lab
* The physics involved in the experiment.
* Any assumptions that were made in either the experiment or the analysis and calculations.
* Main overview of calculations involved in the experiment as well as derivations.
* Equipment used in the experiment.

## The experiment

One of most major breakthroughs in science in the early 20th century regards proving that a photon could act like a particle. Prior to this experiment, physics was still ruled by a Newtonian view that light is a wave. This experiment showed that light can behave like a particle as when it is filtered through tiny slits, the interference will cause the count for the photons in a region, similar to how a particle behaves.

When a photon is shot through a double slit onto a screen, what is expected, is two definite groups. However, what is seen is an interference pattern with a central area with high intensity/ getting struck by a large number of photons with adjacent areas with fewer number of photons striking. What this is called is an interference pattern. This is what a single photon would do. If the counts are graphed vs the position along the screen, a graph like that from first.py will be shown. We can then measure the angle from the central maxima to the diffraction order that is being examined (the peak in question). The equation that allows us to relate the wavelength (lambda) to the diffraction order (m) to the distance between the slits (d) and the angle between the central maxima and the diffraction order (theta) is equation 1. This is the diffraction equation that is used and confirmed in the lab to determine the wavelength of photons.

Consider the central maxima (peak, principal) as m=0, and the maxima to either side of it being either m=+1 or m=-1 such that m is symmetric about m=0.

It is difficult to directly measure the angle theta, so what is done is that the geometry of the system is rewritten in terms of things that can be measured.

This experiment showed that light can behave like a particle as when it is filtered through tiny slits, the interference will cause the count for the photons in that region, similar to how a particle behaves. These distances can be modeled using equation 1 and a simplified version of it, i.e. equation 2. The experiment works by having a light source shine light from (L) a distance away through a separation (d) that is small. This grating the will cause photon activity to be higher in some places. The places where the photon activity is the highest, except for the principal peak, will be (m) integer number of wavelengths and will be (D) distance from the middle of the principal peak. I will find the peaks in the data tables given to them and plot the D distance each peak is from the center of the principal peak as a function of time. I will then use equation 2 to determine the accuracy of the wavelength measurement. Theta is the angle between the principal peak and the next peak.

|  |  |
| --- | --- |
| =dsin | (1) |
|  | (2) |

The way that the data was obtained was by a photon generator firing a beam through 2 slits and the photons landing on a wall behind it. What is seen is that contrary to what may intuitively be thought, instead of having 2 definite clusters of intensity on the wall, there are several peaks in intensity, or counts, with the central one occurring in between the 2 slits, and having the greatest count of photons, or intensity, decreasing with each peak the further index(peak number) is away from the central peak (principal). The peaks will be symmetric about the principal peak, increasing and decreasing in integer values. The file that shows this graph is in ***first.py***, highlighting the peaks and troughs.

The way the count is determined is by physically blocking one of the slits and restricting the second slit such that it is narrower, and then counting the number of photons using a pmt(photo-multiplier tube) that make it through the narrow slit, giving the count at that position on the back screen. After that, the slit is moved over some, and the process is repeated. Eventually, with good enough resolution, a graph can be made of the count (y-axis) of photons and the position (distance in mm) where the peaks show the index of the peaks.

### Assumptions made

When deriving the second equation, it is assumed that small angels are being worked with, as such it is assumed that sin(x)is approximately tan(x).

It is also assumed that photons and other quantum actions won’t change the results. This is because when the original experiment was done, many of these phenomena like observation changing outcomes were either just being discovered or have yet to be discovered.

### The math involved

Starting from equation 1, It is difficult to directly measure the angle theta, so what is done is that the geometry of the system is rewritten in terms of things that can be measured. Using one of the assumptions made above, we can assume that sin(a) is approximately tan(a). Equation 1 should now look like

The distance from the slits to the back screen is L, and the distance from peak to peak is D. Equation 1 can now be rewritten as:

From here, we can solve for D, giving equation 2.

L and d are constant. It is this derived equation 2 that is graphed in second.py. D is on the y-axis and m is on the x-axis. The remaining part of the line, the slope, is equal to . From there, set the slope of the line equal to the previously mentioned portion, and solve for lambda to find the wavelength.

The systematic error calculations were done by taking the partial derivatives of the variables that were directly measured in the experiment that were not integers (so the count did not have a partial derivative). We then plugged in the lowest tolerance for the remaining variables and normalized them. The sum of each of these instances lead to the total systematic error for the experiment.

## The equipment

* A device called a pmt(photo-multiplier tube) was used to count the photons at each location.
  + - * **The way that the pmt works is that it is effectively a series of very closely charged plates that have a high voltage applied to them (about 900V for our lab). What happens is that when a photon strikes the pmt, an electron is excited and hits the next plate behind it. When it hits the next plate, it will have so much energy that it will excite 2 electrons, which get excited and shot towards the next plate, resulting in in a cascading effect. This result takes a small signal from a single photon and then converts it to large signal that can be counted.**
* **A photon gun-shoots low intensity green light (individual photons)**
* **A laser to align the optics**
* **A checking screen to check if optics are aligned correctly**
* **An enclosure that holds all the devices**
* **A double slit assembly-a double slit on one side and a single slit on the other side that can be moved to block a slit. Can be moved using a micrometer; allowing for how light is passing through the double slit.**
* **One oscilloscope, used to determine the position that the double slit assembly needs to be moved to.**
* **One voltmeter to measure the voltage from the pmt.**

# Results

## Results of first.py

### first.py graph and discussion

The results of first.py are shown below. The results were obtained using the above packages and libraries.

*A graph with blue lines and red dots

Description automatically generated*

Figure 1: Interference intensity based on the count of photons interacting with the PMT.

The initial data visualization of the data gained from the experiment is shown in the graph. The red diamonds are shown at the peaks of each graph. These were gained from the find\_peaks() method from the signals module in the Scipy package. There are 7 peaks that the data found, with the central peak being found near the 6mm mark. There were also 9 local minima, marked by a yellow x, that were also found using the same method. The combination of the peaks and minima graphically visualize the interference pattern of the electron during the experiment. Error bars were done in the x direction due to the uncertainty of the measurements, however they are too small to be seen. The package that was used to graph was Matplotlib.

### challenges from first.py

The main challenge that was faced when writing the code for first.py was finding the appropriate package to find the peaks of the data. This was the first time that I had to look for something more niche, at least to me. This was solved by various google searches and learning how to use the Scipy package. During this lab, specifically for this part I gained a better understanding of code documentation for libraries.

The second biggest challenge that I faced when writing the code for first.py was finding a way to set the error bars equal to the standard deviation of each point measured. This was overcome with trail and error with plenty of look ups in the documentation for both Matplotlib and Scipy. Eventually, I figured out how generate the error for each point, append it to a list, and then pass that list to the function that shows the error bars.

### Discussion of statistical results

The statistical analysis shows that in first.py, the data is both accurate and precise. As was shown in the table in my lab report, now below this, a low sample standard deviation and a consistent gaussian average with a low running average and standard deviation showed that the data was reliable and statistically significant. The table below shows the Gaussian average, Sample standard deviation, running average, and running standard deviation for the position, distance to peak, and peak value. The index(integer) value calculations are not necessary, but it was faster to compute as adjacent columns.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Index (integer) | Position(mm) | Distance to peak(mm) | Peak Value |
|  | -3 | 3.6 | -2.3 | 2181 |
|  | -2 | 4.4 | -1.5 | 3562 |
|  | -1 | 5.15 | -.75 | 4716 |
|  | 0 | 5.9 | 0 | 5224 |
|  | 1 | 6.75 | .85 | 4601 |
|  | 2 | 7.5 | 1.6 | 3254 |
|  | 3 | 8.25 | 2.35 | 1963 |
| Gaussian average | 0 | 5.935 | .0357 | 3643 |
| Sample standard deviation | 2.160246899 | 1.678 | 1.678 | 1270.5 |
| Running average | 0 | 5.935 | .0357 | 3643 |
| Running standard deviation | 0 | .9208 | .0714 | 22.8129 |

## Results of second.py

### A brief note about the statistical results, code and potential future updates

**After submitting my lab report for class I have since found a typo in my second.py file that affected the systematic error results. I have sense then corrected the typo in the code. The systematic error changed from a total of .78 to 2.7. The results are still valid from the experiment.**

**Future updates to the code will include it being more modularized approach in both first.py and second.py.**

### second.py graph and discussion

**The results of second.py visualized below, along with the final statistical analysis of the lab:**

A graph with a line and dots

Description automatically generated

Figure 2: Finding wavelength of a green bulb

The data visualization above shows a high positive correlation indicating a r value close to 1. What is also shown is that a low average standard deviation. The graph shows the wavelengths for the index points that were solved for in the previous section. There are vertical error bars at each point, however they are to small to be seen. The packages that were used for the data analysis were the Scipy and statistics packages as well as various user defined functions. The package that was used for the data visualization was Matplotlib.

**The average wavelength(slope) is 0.7767857142857142 units.**

**The intercept 0.035714285714285775**

**The average variance is 0.9998969640044977.**

**The average confidence value is: 2.0696756787724537e-10.**

**The average standard deviation of the line is: 0.004987228587064592.**

**Average percent error 0.9733990147783143%**

**The average standard deviation of lambda 1.5120468742734175e-06**

**The average standard deviation of percent error 0.12758668412940277 per measurement**

**Average smear (Sigma) of lambda 5.49402e-07**

**Total/average systematic error 2.7521684927856422.**

### challenges from second.py

**There were no significant challenges to coding second.py. The most significant thing that I came across was that I found it faster to solve for some of the variable instances, like the wavelength using excel than I did Python mostly due to the fact that it takes less keystrokes to write the for loop or function to solve for them than make a function in excel.**

**One of the things that ended up making the second file easier to code than the first was modularizing my code with multiple functions that could be reused.**

### Discussion of statistical results

**As stated previously, the following is the print out (with some interpretation) of the analysis:**

* **The average wavelength(slope) is 0.7767857142857142. Which is what we would find to be normal in nature when measuring green light.**
* **The intercept 0.035714285714285775. The intercept is near 0, as it should be. What could be causing it to not be 0 is either the smear or lensing factor from the experiment or any background reactions that took place.**
* **The average variance is 0.9998969640044977. A low variance indicates a high accuracy.**
* **The average confidence value is: 2.0696756787724537e-10. With a confidence value like this we can be sure that our results are consistent have value of being statistically significant.**
* **The average standard deviation of the line is: 0.004987228587064592. With low standard deviation, we can say that the data is highly precise.**
* **Average percent error 0.9733990147783143% With a low percentage error, we can say that our data was can be trusted to be true to the true value.**
* **The average standard deviation of lambda 1.5120468742734175e-06. All this means is that our wavelengths didn’t spread their values much, if at all.**
* **The average standard deviation of percent error 0.12758668412940277 per measurement. This is reassurance that our analysis and data are highly accurate and precise.**
* **Average smear (Sigma) of lambda 5.49402e-07. This indicates that the experiment was done with little uncertainty.**
* **Total/average systematic error 2.7521684927856422. This is the compounded result of systematic error. Since there were 7 points for the error, this indicates that each point had little to no systematic error, most of which was probably caused by the thresholds of the measurements.**