Single Photon Interference

Gerod Dunn
*Modern Physics Laboratory, Department of Physics, University of North Texas, 1155 Union Circle, #311427, Denton, Texas 76203

Submitted 25 02 2021

Summary

In this experiment, students will explore one of the most major breakthroughs in science in the early 20th century. Prior to this experiment, physics was still ruled by a Newtonian view in that light is a vave. 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) distance away through a separation (d) that is small. This grating of 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. Students will use 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. Students will then use equation 2 to determine the accuracy of the wavelength measurement.

$$m\lambda = d\sin\theta \qquad (1)$$

$$\frac{m\lambda L}{d} = D \qquad (2)$$

Data

In table 1, the second column contains the peak index or interference index (M) as an integer. The Next column the position of the peak along the x axis(mm) there is the average node that had a peak in the data set was $5.36 \pm 1.68 mm$along the x axis. The next column is the distance (D) in millimeters each peak is away from the principal peak. The average distance between the principal peak and the node that had a peak is .036 $\pm .036 mm$from the principal peak.The peak value is listed at the end of the table. On average, the peaks are 3643 ± 22.812photons. This information is important as this information, with the help of equation (2) allows the student to be able to determine slope of graph 2, leading to the experimental accepted value for the wavelength. With generally low standard deviations, it is safe to say that the data is precise as it has a small distance from the mean. Table 2 shows the value for the wavelength, where each position is the result of using equation (2) with the information held under the distance D(mm) column and the integer count (M) column. The standard deviation for each element is listed next to the element. The average value for the wavelength is 0005524499 $\pm 1.5120468 \cdot 10^{-6}$ mm. There was an average error percent for each value of about .97339%. With small error percentages, it is safe to say that the data is accurate and precise. For all measurements in table 2 as searching through the lab manual and the pre lecture video proved to be fruitless in trying to determine the distance between the split (L in mm) or the distance between the splits (d in mm). This proved to be a long task, but thankfully the sound bit of being them or how to get them was found. The data analysis for this lab was primarily done in a python 3 IDE, the code will be posted at the end for reference. **Table 1** shows the index position of the peak as an integer, the position along the x axis that the peak occurred in mm, the distance from the peak to the principal peak in mm, and the number of photons at that the peak(#of photons):

	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	-0.75	4716
	0	5.9	0	5224
	1	6.75	0.85	4601
	2	7.5	1.6	3254
	3	8.25	2.35	1963
Gaussian average	0	5.935	0.0357	3643
stdev.s	2.160246899	1.678	1.678	1270.5
running average	0	5.935	0.0357	3643
running std	0	0.9208	0.0714	22.8129

Table 2: Is a sumarzed set of Data from the experiment that shows the value for the wavelength to be roughly $0.000552449 \pm 1.5120468 \bullet 10^{-6}$mm or about 552nm. There was, on average, a percent error for each wavelength of .9733$\pm$ 0.127586 % . With such a small error and an even smaller percent error, it is safe to say that the date is both precise and accurate.

	wavelength
	5.49E-04
	5.49E-04
	5.58E-04
	5.56E-04
	5.50E-04
	5.45E-04
	5.42E-04
	5.46E-04
averages	5.49E-04
Percent Error	9.73E-01
standard deviation	1.51E-06
expected error from D	1.93E-06
systematic error	0.7836167374

Calculations and Graphs

A brief example on how to find lambda:

recalling equation 2: $\frac{m\lambda L}{d} = D$

knowing that L=500mm, d=0.3556mm and D at index 1 is .85, and considering that an example index is say 1, the way to determine the wavelength is to first write out what is given and them rearrange to solve for the variable, then plug numbers in.

$$\frac{m\lambda L}{d} = D$$

$$\lambda = \frac{Dd}{mL} = \frac{.85 \bullet .3556}{1 \bullet 500} = .060452\text{mm}$$

For an example of error analysis:

Take the previous example, .060452mm and compare it to the average wavelength for percent error

$\frac{|.06452 - 0.000549402|}{.06452} * 100$=99% error. While the writer is unsure how this has happened, they did use a computer language to automate some of the tasks.

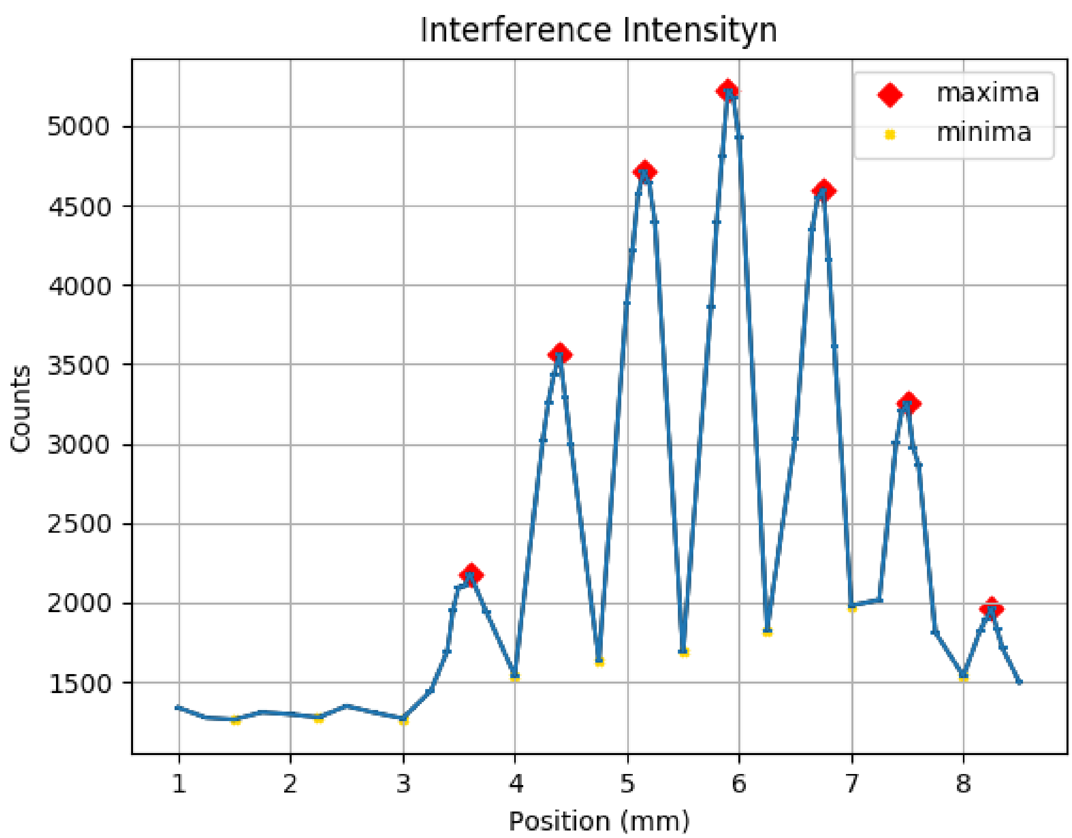

Figure 1: Graph of the counts of Photons against the position that they landed. There are 7 peaks labeled in read, these are the peaks that were involved in the calculations. There are error bars, but they are too small to see.

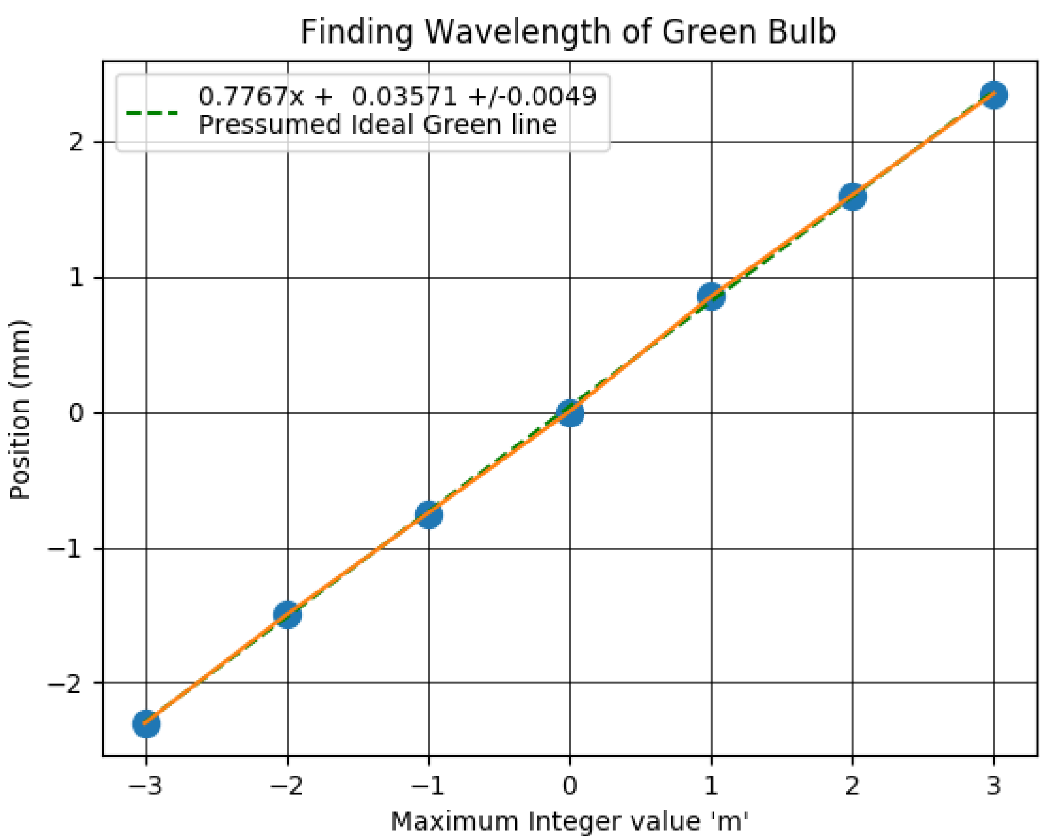

Figure 2: Using equation 2, it was found that there the wavelength is about 0.000549 $\pm$ 0.0049872. There are error bars, they are just too small to see. With a tight fitting line and a

low standard deviation, it is no surprise that the confidence value is 2.0696756787724537e-10, and with such a low number, there is a very high chance that there is a strong correlation.

Discussion & Error Analysis

There were no unusual problems that occurred, other than not being able to find where some of the information was, but that is a minor event. With a low confidence test, low standard deviation and variance where the error bars are unable to be seen, it is this writer's opinion that the data is accurate and precise. What would be an interesting test is to see if the individual property affects the grating of light and if so, how much. There was an average percent error of 0.973399014778314. Qualitatively, there was a systematic error ratio of 0.7836167374. It is this writer's opinion that with this high of accuracy with just one variable D, it may prove informational to determine the total impact of uncertainty that all the possible variables would have.

File one contains the code for the first plot and most of the code for determining if a point is a peak:

Python Code appendix:

```
from numpy import *
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import scipy
from scipy import signal
from scipy.signal import find_peaks
from scipy import constants
from scipy import stats
import statistics


column_names = ["Position (mm)", "Counts"]
df = pd.read_csv('Single_Photon_Interference_Data.csv',names=column_names)
#print (df)

new_df = df.dropna()
#print(new_df.to_string())


pos = new_df['Position (mm)'].values
posit=np.array(pos)
position=posit.astype('f')
#print(position)

cnt=new_df['Counts'].values
cnts=np.array(cnt)
counts=cnts.astype('f')
#print(counts)


##graphing
xpoints=position
ypoints=counts
#   Peaks

peaks = find_peaks(counts, height = 1500, threshold = 1, distance = 1)
height = peaks[1]['peak_heights'] #list containing the height of the peaks
```

```
peak_pos = position[peaks[0]]   #list containing the positions of the peaks
#   Minimums
y2=ypoints*-1

minima = find_peaks(y2, threshold = 1, distance = 1)
min_pos = position[minima[0]]   #list containing the positions of the minima
min_height = y2[minima[0]]   #list containing the height of the minima

#   Plotting the main imported data
plt.plot(xpoints, ypoints)
plt.subplots()
#   Plotting the maxima and minima
plt.scatter(peak_pos, height, color = 'r', s = 40, marker = 'D', label = 'maxima')
plt.scatter(min_pos, min_height*-1, color = 'gold', s = 10, marker = 'X', label = 'minima')



##statistical analysis

#poison standard deviation for
sums=0.0
for i in range(len(position)):
  sums=sums+position[i]
avpos=sums/len(position)
xsqare=(avpos**.5)/len(position)

print(peak_pos)
print(height)

xerror=xsqare
yerror =6.85944579

for i in range(len(peaks)):
  sigma=sigma+((((((peaks[i]-avpos))**2))/len(peaks))**.5)#x error bars
#print(sigma)



total=0



plt.title("Interference Intensityn", loc = 'center')
plt.xlabel("Position (mm)")
plt.ylabel("Counts")
plt.plot(xpoints, ypoints, color = 'k')
```

```
plt.errorbar(xpoints, ypoints, xerr=xerror, yerr=yerror, errorevery=1, markeredgewidth=10)


plt.legend()
plt.grid()
plt.show()
```

File 2 contains the code for the other graph and the statistics calculations

```
from numpy import *
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import scipy
from scipy import signal
from scipy.signal import find_peaks
from scipy import constants
from scipy import stats
import statistics

y = [-2.3,-1.5,-.75,0,.85,1.6,2.35]# D istances from principle index
x = [-3.0,-2.0,-1.0,0.0,1.0,2.0,3.0]# m values

def myfunc(x):
  return slope * x + intercept


z = np.polynomial.polynomial.polyfit(x, y, 1) #polynomial reagression with order 1

slope, intercept, r, p, std_err = stats.linregress(x, y)
#getting the slope, y-intercept, r


    # forming the slope
```

```
mymodel = list(map(myfunc, x))

plt.plot(x, y, 'o',ms = 10)
plt.plot(x, mymodel,"g--",label='0.7767x +  0.03571 +/-0.0049\nPressumed Ideal Green line')


plt.title("Finding Wavelength of Green Bulb",loc = 'center')
plt.xlabel("Maximum Integer value 'm'")
plt.ylabel("Position (mm)")

plt.grid(color = 'k', linestyle = '-', linewidth = .5)
plt.errorbar(x, y, xerr=0, yerr=std_err, errorevery=1, markeredgewidth=10)

plt.legend()
plt.show()

print("slope is", slope)
print("intercept is", intercept)
print("variance is", r)
print("Confidence value is: ",p)
print("Standard deviation of the line is:", std_err)
```

```
L=float(500)
d=float(25.4*(14/1000))

real_lmda=(d*slope)/L

lmda_real=((slope*d)*L)

def lmda(): #I had to hack this together with sheets

lmda_list=[0.000549402,0.0005492834667,0.0005575808,0.0005561584,0.0005504688,0.0005447792,
0.0005415788,0.0005459645333]
  return lmda_list

def percent_er(num,num_listL):
  percent_errors=[]
  for i in range(len(num_listL)):
    EE = ((num_listL[i]-num)/num)*100
    if EE<0:
      EE=EE*-1
```

```
    percent_errors.append(EE)
   if EE>0:
    percent_errors.append(EE)
  return percent_errors

def avgs(lists_here):
  runsum=0
  for i in range(len(lists_here)):
   runsum=runsum+lists_here[i]
  return runsum/len(lists_here)

def stdevS(num_list):
  stls=[]
  av=avgs(num_list)
  for i in range(len(num_list)):
   s=(((num_list[i]-av)**2)/(len(num_list)-1))**.5
   stls.append(s)
   s+=(((num_list[i]-av)**2)/(len(num_list)-1))**.5
  return stls

def dsig(lists_coming):
  lenzing=.001
  d_elems=[]
  intdex=0
  for i in range(len(lists_coming)):
   index=(((lists_coming[i]*.001)**2)**.5)
   d_elems.append(intdex)
   intdex+=(((lists_coming[i]*.001)**2)**.5)
  return d_elems

def vars(listings):
  for i in range(len(listings)):
   listings[i]=listings[i]**2
  return listings



def main():
  lamLists=lmda()#made list of comparitive lambdas
  errros=percent_er(real_lmda,lamLists)#made a list of comparitive errors

  #print(errros)

  avL=avgs(lamLists)#average lambda
  av_errros=avgs(errros)#average percent error
```

```
stedevesL=stdevS(lamLists)#standard deviation of lambda
stedevesE=stdevS(errros)#standard deviation of errors

#print(stedevesL)
#print(stedevesE)

avgstedL=avgs(stedevesL)
avgstedE=avgs(stedevesE)

sigL=dsig(lamLists)#exected error in lambda
sigE=dsig(errros)#expected error in errors

#print(sigL)
#print(sigE)


sigsL=avgs(sigL)#average error expected from D
#print(sigsL)
sigsE=avgs(sigE)

varL=vars(lamLists)#variance in lambda
#print(varL)
varE=vars(errros)#variance in error
#print(varE)

varsL=avgs(varL)
#print(varsL)
varsE=avgs(varE)
#print(varsE)

print("Comparing in main",real_lmda," ",avL)
#print("Lambdas ",lamLists)
#print("Percent Error: ",errros)
print("Average percent error ",av_errros)

print("Standard deviation of lambda ", avgstedL)
print("Standard deviation of percent error ", avgstedE)
print("Sigmda lamda ",sigsL)
#print("Sigmda error ",sigsE)
#print("Average variance in lambda ",varsL )
#print("Average variance in error ",varsE )

print("Systemetatic error ",avgstedL/sigsL)
```

```
  print("Sucsess")

if __name__ == '__main__':
    main()
```