AST325: LAB 1

Basic Photon Statistics with Python

Anntara Khan

October 3, 2022

Question 1

Using the Measured Photon Count Rates (= Number of incoming photons per second) the following plot can be created:

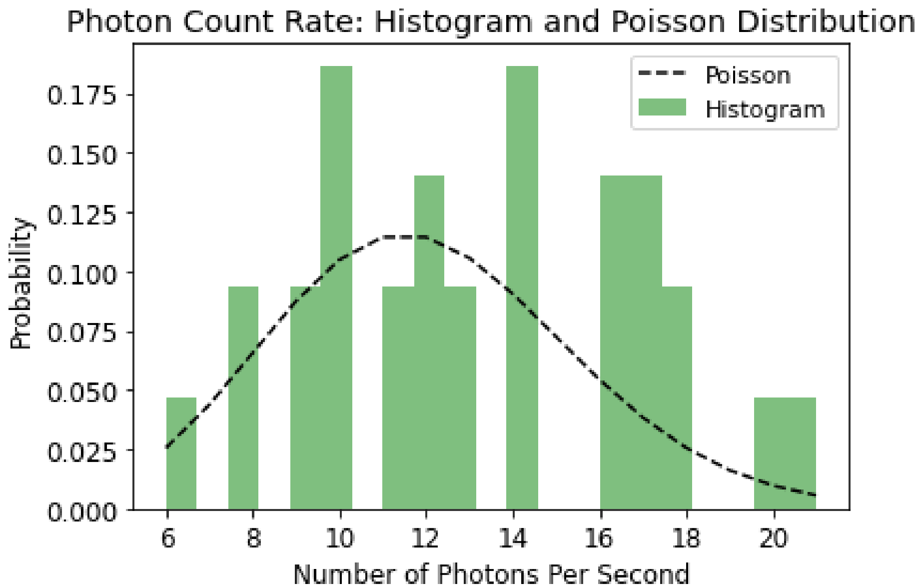


Figure 1: Reproducing Figure 3 using my own Python code

We can observe that the plot in Figure 1 is similar to the Plot in Figure 3 of the Lab 1 Assignment Sheet.

See Appendix A.1 for the corresponding code.

Question 2 and 3

Reading the two files into Python code, we calculate the mean and standard deviation of the measurements recorded in each file:

Small

$STDDEV = 2.113$

$Mean = 4.445$

Large

$STDDEV = 31.385$

$Mean = 977.347$

**rounded to the 3rd decimal place*

For Poisson distribution we know that, $STDDEV = \sqrt{(Mean)}$
So for Poisson distribution we could calculate the Standard Deviation to be;

$STDDEV_S = 2.108$

$STDDEV_L = 31.262$

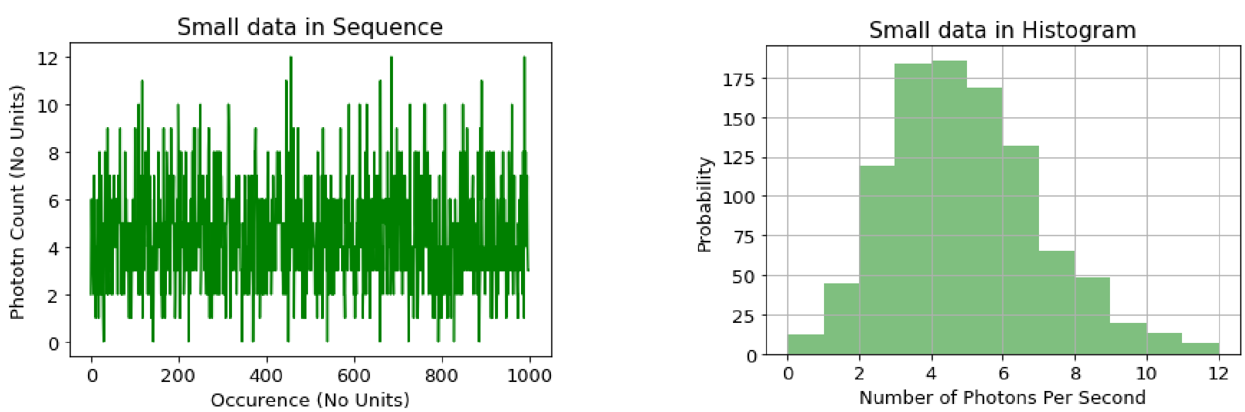
Which seems pretty similar to the calculated value of the Standard Deviation. Thus we can say that the calculated mean and standard deviation are consistent with what we expect from a Poisson distribution.

See Appendix A.2 for the corresponding code.

Question 4

Plotting the measurements in sequence and in Histogram we get the following plots:

Small



(a) Small data plotted in Sequence

(b) Small data in Histogram

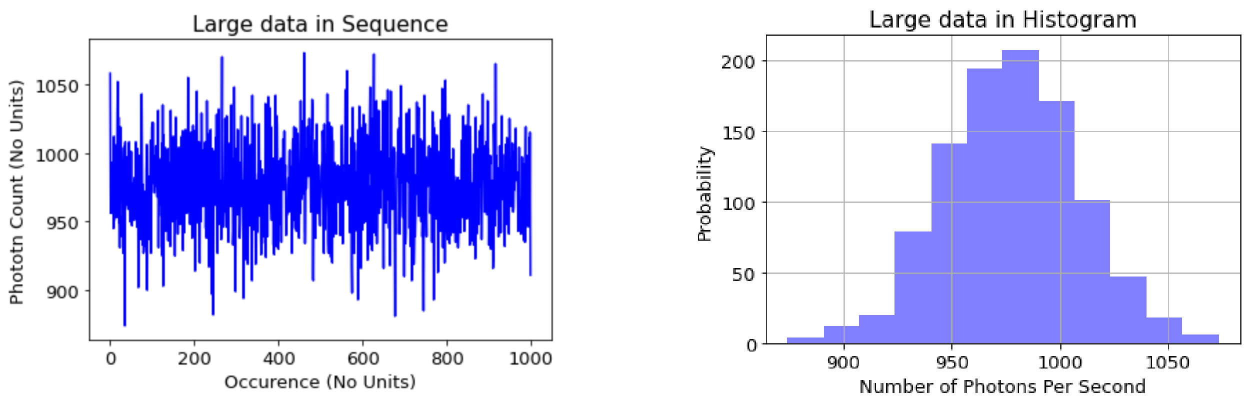
Figure 2: Sequence and Histogram of Small Data

Looking at the Histogram plot we can roughly estimate the mean to be approximately 5, but for the standard Deviation we would need to use the following equation:

$$\sigma = \sqrt{\frac{\sum (x_i - \mu)^2}{N}} \tag{1}$$

since we know the photon count rate, size of the observation and the mean, we estimate the standard deviation to be 4.44. which is similar to the calculation output we got from python.

Large



(a) Large data plotted in Sequence

(b) Large data in Histogram

Figure 3: Sequence and Histogram of Large Data

Looking at the Histogram plot we can roughly estimate the mean to be approximately 975, and for the standard Deviation we would need to use the following equation:

$$\sigma = \sqrt{\frac{\sum (x_i - \mu)^2}{N}} \tag{2}$$

since we know the photon count rate, size of the observation and the mean, we estimate the standard deviation to be 976.4. which is similar to the calculation output we got from python.

We take the Histogram for the small data and Plot a Poisson distribution over it.

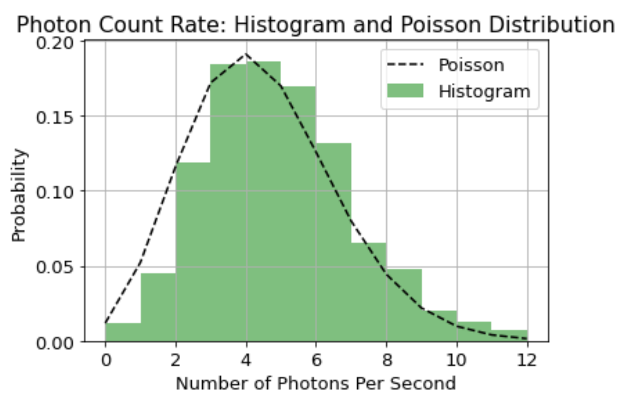


Figure 4: Plotting the Poisson Distribution over the Histogram of Small data

We observe from this plot that the data distribution does follow Poisson Distribution.

See Appendix A.3 for the corresponding code.

Question 5

We take the Histogram for the large data and Plot a Poisson distribution over it.

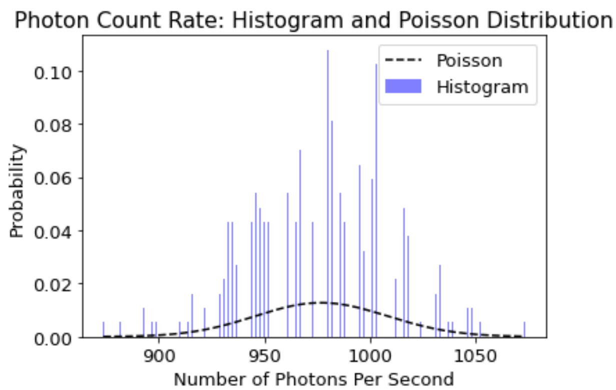


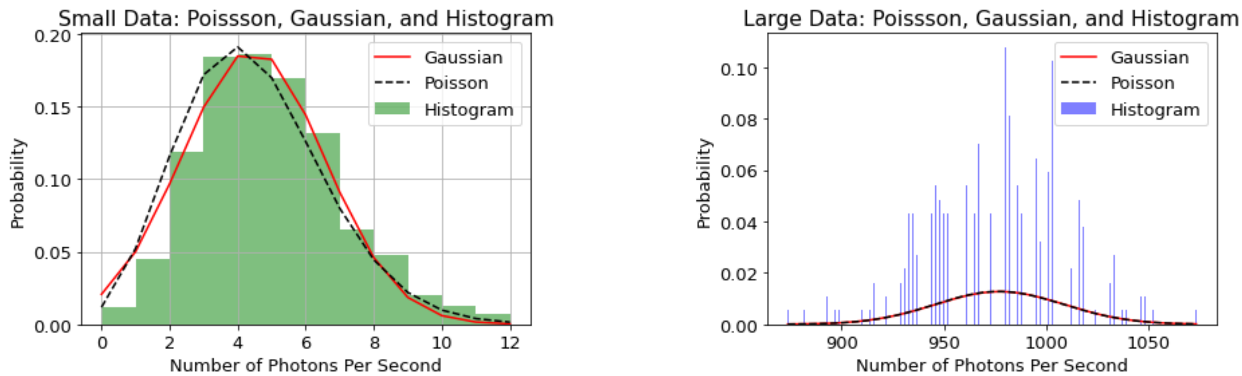
Figure 5: Plotting the Poisson Distribution over the Histogram of Large data

We observe from this plot that the data distribution looks much closer to a Gaussian distribution than a Poisson distribution. And this makes sense because we learned in class that when the expected average is large, a Poisson distribution becomes very similar to a Gaussian distribution

Comparing Figure 4 and Figure 5 we see that compared to the small data Poisson distribution of the large data looks more like a like a Gaussian distribution.

We take the Mean and Standard Deviation Calculated in Question 3 and used that to plot a Gaussian Distribution on top of Figure 4 and Figure 5, as shown in Figure 6.

We see that for the Small set of data where the average is small the Poisson Distribution and Gaussian Distribution are different (Figure 6(a): red solid line which is Gaussian and black dashed line which is Poisson are not aligned). But for the large set of data where the Average is large, the Poisson Distribution and the Gaussian Distribution are the same (Figure 6(b): red solid line which is Gaussian and black dashed line which is Poisson are aligned).



(a) Photon Count Rate: Histogram, Gaussian and Poisson Distribution for Small Data

(b) Photon Count Rate: Histogram, Gaussian and Poisson Distribution for Large Data

Figure 6: Comparison of Poisson and Gaussian Distribution for Large and Small data

See Appendix A.4 for the corresponding code.

Question 6

Using the example Python script, the distances and velocities of the 13 galaxies, We plot the Least Square Fit. From the Slope of the fitted data we can obtain the Hubble constant.

After loading the data onto python we convert the distance in km to obtain the following Hubble constant:

$$H = 2.49e - 18 \pm 2.82e - 20\ s^-1 \tag{3}$$

We also get the fitted plot:

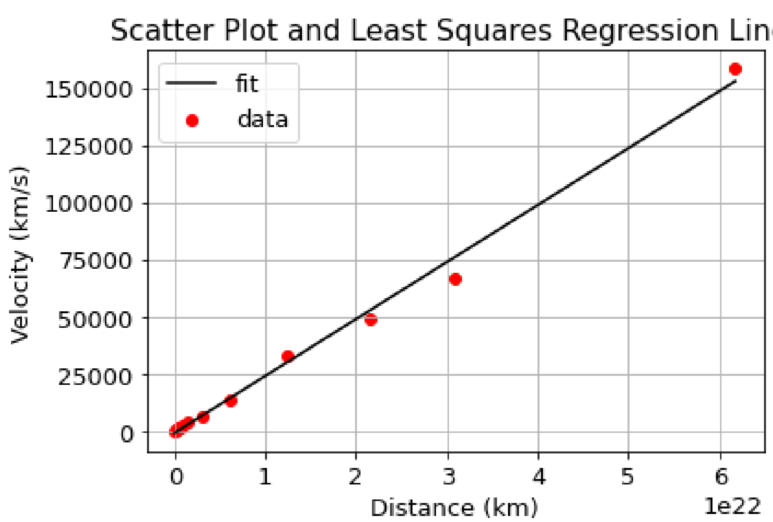


Figure 7: Plotting the Distance and the Velocity of the 13 Galaxies, and their Least Square Fit. the slope of the line gives us the Hubble constant.

See Appendix A.5 for the corresponding code.

References

- Saint-Antoine, M. (2022, April 13). Comp_Bio_Tutorials/fit_distributions.py. GitHub. https://github.com/mikesaint-antoine/Comp_Bio_Tutorials/blob/main/parameter_estimation/discrete_distributions/fit_distributions.py

Appendix A

Appendix A includes all codes used in Lab 1.

Appendix A.1

Code for Question 1

```
#Create a list of Photon Counts
PC = [13, 17, 18, 14, 11, 8, 21, 18, 9, 12,
 9, 17, 14, 6, 10, 16, 16, 11, 10, 12,
 8, 20, 14, 10, 14, 17, 13, 16, 12, 10]

#define the probability function
def probability_func(params, data):

    '''this function will check how far off the model is the data, the result
    will minimize the probaility function to show how far of the data and
    model are.'''

    deviation = 0

    for i in range(len(data)):

        '''Log likelihood method: How Likely it is so see a data point
        Given the Distribution.
        We take the probability of occurrence and take the natural
        log of that probability after running it through the Poisson
        Distribution in this case'''
        '''
        reference:https://github.com/mikesaint-antoine/Comp_Bio_Tutorials/blob/main/
        parameter_estimation/discrete_distributions/fit_distributions.py
        '''
        probability = scipy.stats.poisson.logpmf(data[i], mu)
```

```
            Change = -probability

            deviation += Change

        return(deviation)

#initial guess for mu
mu = 12

#find the paramter value that minimizes how far off the model is
minimum = scipy.optimize.fmin(probability_func, mu, args=(PC,))

#fitted value of mu
mu_fit = minimum[0]

#Sort the photon count list into an ascending order
x = list(range(int(np.min(PC)), int(np.max(PC))+1))

'''plotting the poisson distribution using the fitted value of mu and the
sorted list of Photon Counts'''

#PLOTS
plt.title("Photon Count Rate: Histogram and Poisson Distribution")
plt.rcParams['font.size'] = 11
plt.hist(PC, bins=int(np.max(PC)), density=True, alpha=0.5, color='green', label='Histogram')
plt.plot(x, scipy.stats.poisson.pmf(x, mu_fit), color="Black", linestyle='dashed', label='Poisson')
plt.legend()
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()
```

Appendix A.2

Code for Question 2 and 3

```
#load text
Small = np.loadtxt('Khan-khanannt-Small.txt', unpack=True)
Large = np.loadtxt('Khan-khanannt-Large.txt', unpack=True)

#Calculate Standard Deviation, Mean, and consistency with Poisson Distribution for Small
Small_STDV = np.std(Small)
Small_Mean = np.mean(Small)
SmallP = np.sqrt(Small_Mean)
```

```python

#Calculate Standard Deviation, Mean, and consistency with Poisson Distribution for Small
Large_STDV = np.std(Large)
Large_Mean = np.mean(Large)
LargeP = np.sqrt(Large_Mean)

#Print out the values
print (Small_STDV)
print (Small_Mean)
print(SmallP)
print (Large_STDV)
print (Large_Mean)
print(LargeP)
```

Appendix A.3

Code for Question 4

```python
'''Using the same data used in Question 2 & 3, using the results
for mean and standard deviation from Question 3 as well'''
#-------------------------------------------------------#
#Sequence Plot
  #Small
plt.rcParams['font.size'] = 13
plt.title('Small data in Sequence')
plt.xlabel("Occurence (No Units)")
plt.ylabel("Phototn Count (No Units)")
plt.plot(Small, color = 'green')
plt.show()
  #Large
plt.rcParams['font.size'] = 13
plt.title('Large data in Sequence')
plt.xlabel("Occurence (No Units)")
plt.ylabel("Phototn Count (No Units)")
plt.plot(Large, color = 'Blue')
plt.show()
#-------------------------------------------------------#
#Histogram
  #Small
plt.title('Small data in Histogram')
plt.grid()
plt.rcParams['font.size'] = 13
plt.hist(Small, bins=int(np.max(Small)), alpha=0.5, color='green', label='Histogram')
plt.xlabel("Number of Photons Per Second")
```

```
plt.ylabel("Probability")
plt.show()
  #Large
plt.title('Large data in Histogram')
plt.grid()
plt.rcParams['font.size'] = 13
plt.hist(Large, bins=int(np.max(Small)), alpha=0.5, color='blue', label='Histogram')
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()
#----------------------------------------------------#
##plotting Poisson over Histogram(Small data)##

'''Using the same data used in Question 2 & 3, using the results
for mean and standard deviation from Question 3 as well'''

 #define the probability function for Small
def probability_func_S(params, data):

    '''this function will check how far off the model is the data, the result
    will minimize the probability function to show how far of the data and
    model are.'''

    deviation = 0

    for i in range(len(data)):
        '''Log likelihood method: How Likely it is so see a data point
        Given the Distribution.
        We take the probability of occurrence and take the natural
        log of that probability after running it through the Poisson
        Distribution in this case'''
        '''
        reference:https://github.com/mikesaint-antoine/Comp_Bio_Tutorials/blob/main/
        parameter_estimation/discrete_distributions/fit_distributions.py
        '''
        probability = scipy.stats.poisson.logpmf(data[i], mu1)

        Change = -probability

        deviation += Change

    return(deviation)

#intital guess for mu
mu1 = Small_Mean
```

```

#find the paramter value that minimizes how far off the model is
minimum1 = scipy.optimize.fmin(probability_func_S, mu1, args=(Small,))

#fitted value of mu
mu_fit1 = minimum1[0]

#Sort the photon count list into an ascending order
x1 = list(range(int(np.min(Small)), int(np.max(Small))+1))

'''plotting the Poisson distribution using the fitted value of mu and the
sorted list of Photon Counts'''

#PLOTS small
plt.title("Photon Count Rate: Histogram and Poisson Distribution")
plt.grid()
plt.rcParams['font.size'] = 13
plt.hist(Small, bins=int(np.max(Small)), density=True, alpha=0.5, color='green', label='Histogram')
plt.plot(x1, scipy.stats.poisson.pmf(x1, mu_fit1), color="Black", linestyle='dashed', label='Poisson')
plt.legend()
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()
```

Appendix A.4

Code for Question 5

```
##plotting Poisson over Histogram(Large data)##

'''Using the same data used in Question 2 & 3, using the results
for mean and standard deviation from Question 3 as well'''

#define the probability function for Large
def probability_func_L(params, data):

    '''this function will check how far off the model is the data, the result
    will minimize the probability function to show how far of the data and
    model are.'''

    deviation = 0

    for i in range(len(data)):
        '''Log likelihood method: How Likely it is so see a data point
```

```python
        Given the Distribution.
        We take the probability of occurrence and take the natural
        log of that probability after running it through the Poisson
        Distribution in this case'''
        '''
        reference:https://github.com/mikesaint-antoine/Comp_Bio_Tutorials/blob/main/
        parameter_estimation/discrete_distributions/fit_distributions.py
        '''
        probability = scipy.stats.poisson.logpmf(data[i], mu2)

        Change = -probability

        deviation += Change

    return(deviation)

#initial guess for mu
mu2 = Large_Mean

#find the paramter value that minimizes how far off the model is
minimum2 = scipy.optimize.fmin(probability_func_L, mu2, args=(Large,))

#fitted value of mu
mu_fit2 = minimum2[0]

#Sort the photon count list into an ascending order
x2 = list(range(int(np.min(Large)), int(np.max(Large))+1))

'''plotting the poisson distribution using the fitted value of mu and the
sorted list of Photon Counts'''

#PLOTS large
plt.title("Photon Count Rate: Histogram and Poisson Distribution")
plt.rcParams['font.size'] = 13
plt.hist(Large, bins=int(np.max(Large)), density=True, alpha=0.5, color='blue', label='Histogram')
plt.plot(x2, scipy.stats.poisson.pmf(x2, mu_fit2), color="Black", linestyle='dashed', label='Poisson')
plt.legend()
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()
#--------------------------------------------------------#
##plotting Gaussian over the Poisson and Histogram of Small and Large data##

'''Using the same data used in Question 2 & 3, using the results
for mean and standard deviation from Question 3,
```

```python
and the Probability function used in question 4 and 5'''

#Normalize the y values using the Mean and Standard Deviation
y1 = scipy.stats.norm(Small_Mean, Small_STDV)
y2 = scipy.stats.norm(Large_Mean, Large_STDV)

'''
Gaussian: using the sorted list of Photon Counts and
the normalized y values
Poisson:fitted value of mu and the sorted list of Photon Counts
'''
#PLOTS small
plt.title("Small Data: Poissson, Gaussian, and Histogram")
plt.grid()
plt.rcParams['font.size'] = 13
plt.hist(Small, bins=int(np.max(Small)), density=True, alpha=0.5, color='green', label='Histogram')
plt.plot(x1, y1.pdf(x1),color='red', label='Gaussian')
plt.plot(x1, scipy.stats.poisson.pmf(x1, mu_fit1), color="Black", linestyle='dashed', label='Poisson')
plt.legend()
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()

#PLOTS large
plt.title("Large Data: Poissson, Gaussian, and Histogram")
plt.rcParams['font.size'] = 13
plt.hist(Large, bins=int(np.max(Large)), density=True, alpha=0.5, color='blue', label='Histogram')
plt.plot(x2, y2.pdf(x2), color='red', label='Gaussian')
plt.plot(x2, scipy.stats.poisson.pmf(x2, mu_fit2), color="Black", linestyle='dashed', label='Poisson')
plt.legend()
plt.xlabel("Number of Photons Per Second")
plt.ylabel("Probability")
plt.show()

```

Appendix A.5

Code for Question 6

```python
x, y = np.loadtxt('Khan-khanannt-Hubble.txt', unpack=True)
nx = len(x) # Number of data points
x = x * 3.086e+19 #converting Meagaparsect to km

#Modified from Python Example give on the Assignment sheet
```

```
# Construct the matrices
ma = np.array([ [np.sum(x**2), np.sum(x)],[np.sum(x), nx ] ] )
mc = np.array([ [np.sum(x*y)],[np.sum(y)]])
# Compute the gradient and intercept
mai = np.linalg.inv(ma)
print ('Test matrix inversion gives identity',np.dot(mai,ma))
md = np.dot(mai,mc) # matrix multiply is dot
print (md)

mfit = md[0,0]
cfit = md[1,0]
y_fit = mfit*x + cfit

#Uncertainitiy
m = mfit
c = cfit
sum_sq = 0
sumx2 = 0
sumx = 0
sumx_sq = 0
sum_all_den1 = 0
besel_corr = 1/(nx-2)
for i in range(len(x)):
    x2 = (x[i]**2)
    sumx2 += x2
    sumx += x[i]
    sumx_sq = sumx**2
    sum_all_den = sumx2 - sumx_sq
    sum_all_den1 += (x2 - sumx_sq)
    mx = (x[i]*m)
    mxc = mx-c
    ymxc = (y[i]-mxc)
    sq_ymxc = ymxc**2
    sum_sq += sq_ymxc

sigma2 = besel_corr * sum_sq
sigma = np.sqrt(sigma2)
sigmac2 = (sigma2)/(sum_all_den1)
sigmac = np.sqrt(abs(sigmac2))
print (sigmac)

print ('m = ',mfit,'+/-', sigmac)

#print (sigmac)
'''
```

```
the standard deviation of the slope equation require that we subtract the SQUARE OF SUMS
from the value OF Xi, and then taking the sum of it. but the sum value is Negative.
Which means that sigma_c square is Negative and i cannot take the square root of a negative
number np.sum(x**2) - np.sum(x)**2 = -14825810.0. In order to find uncertainty which is the
square root of sigmac square value, I am taking the absolute value of it'''

plt.rcParams['font.size'] = 13
plt.grid()
plt.title("Scatter Plot and Least Squares Regression Line ")
plt.scatter(x, y, color='red', label='data')
plt.plot(x, y_fit, color='black', label ='fit')
plt.legend()
plt.xlabel("Distance (km)")
plt.ylabel("Velocity (km/s)")
plt.show()
```