EE2703 : Applied Programming Lab
Assignment - 3

Jawhar S
EE20B049

February 18, 2022

Obtaining And Visualizing The Data

The data in the file *fitting.dat* is obtained by running the python script *generate.py*.

The data obtained has 10 columns. The first column corresponds to time and the remaining nine columns correspond to the function,

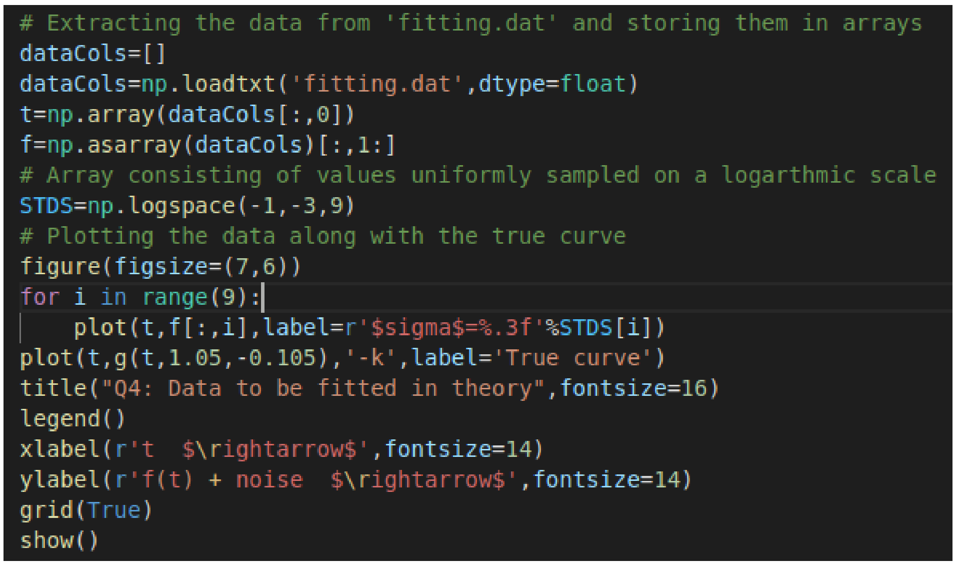
$$f(t) = 1.05J_2(t) - 0.105t + n(t)$$

riddled with different amount of noise, with standard deviation uniformly sampled on a logarithmic scale, ranging from 10^{-3} to 10^{-1}.

The noise is given to be normally distributed, i.e., its probability distribution is given by

$$Pr(n(t)|\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp \frac{-n(t)^2}{2\sigma^2}$$

Also, the noise is assumed to be non-correlated over time.



```
# Extracting the data from 'fitting.dat' and storing them in arrays
dataCols=[]
dataCols=np.loadtxt('fitting.dat',dtype=float)
t=np.array(dataCols[:,0])
f=np.asarray(dataCols)[:,1:]
# Array consisting of values uniformly sampled on a logarthmic scale
STDS=np.logspace(-1,-3,9)
# Plotting the data along with the true curve
figure(figsize=(7,6))
for i in range(9):
    plot(t,f[:,i],label=r'$sigma$=%.3f'%STDS[i])
plot(t,g(t,1.05,-0.105),'-k',label='True curve')
title("Q4: Data to be fitted in theory",fontsize=16)
legend()
xlabel(r't  $\rightarrow$',fontsize=14)
ylabel(r'f(t) + noise  $\rightarrow$',fontsize=14)
grid(True)
show()
```

Figure 1: Code for obtaining and plotting the data

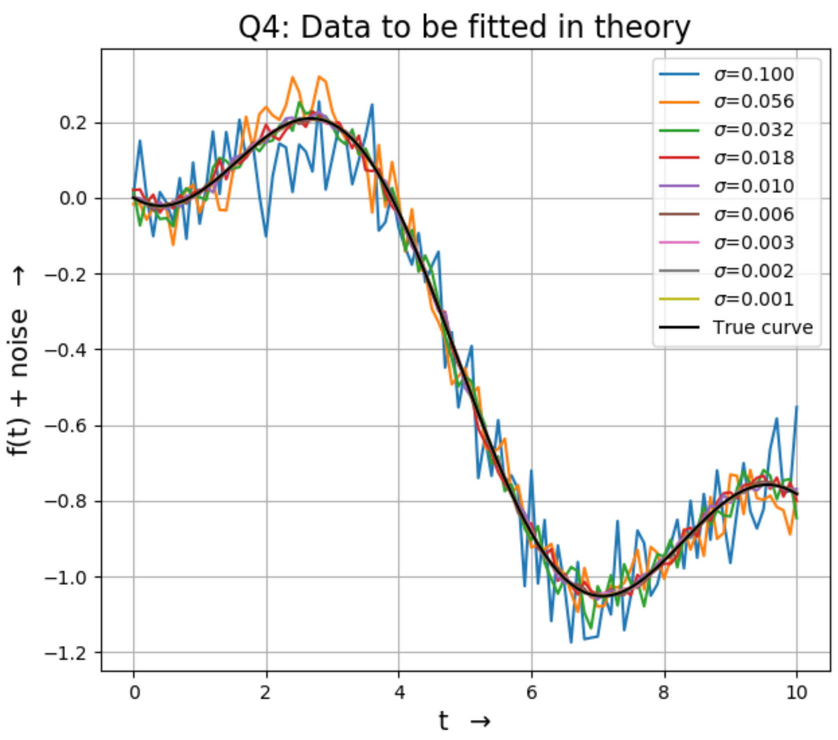
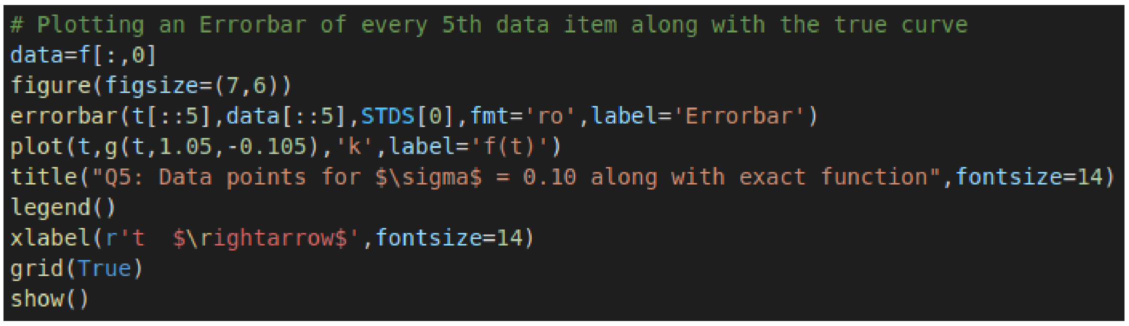


Figure 2: Plot depicting the data with different noise levels

Plotting The Errorbar



```
# Plotting an Errorbar of every 5th data item along with the true curve
data=f[:,0]
figure(figsize=(7,6))
errorbar(t[::5],data[::5],STDS[0],fmt='ro',label='Errorbar')
plot(t,g(t,1.05,-0.105),'k',label='f(t)')
title("Q5: Data points for $\sigma$ = 0.10 along with exact function",fontsize=14)
legend()
xlabel(r't  $\rightarrow$',fontsize=14)
grid(True)
show()
```

Figure 3: Code for plotting Errorbar

Errorbar is a convenient way of visualizing the noise in the data. The Errorbar for every 5th data point in the first data column ($\sigma = 0.1$), along with the exact function is plotted in the below figure.

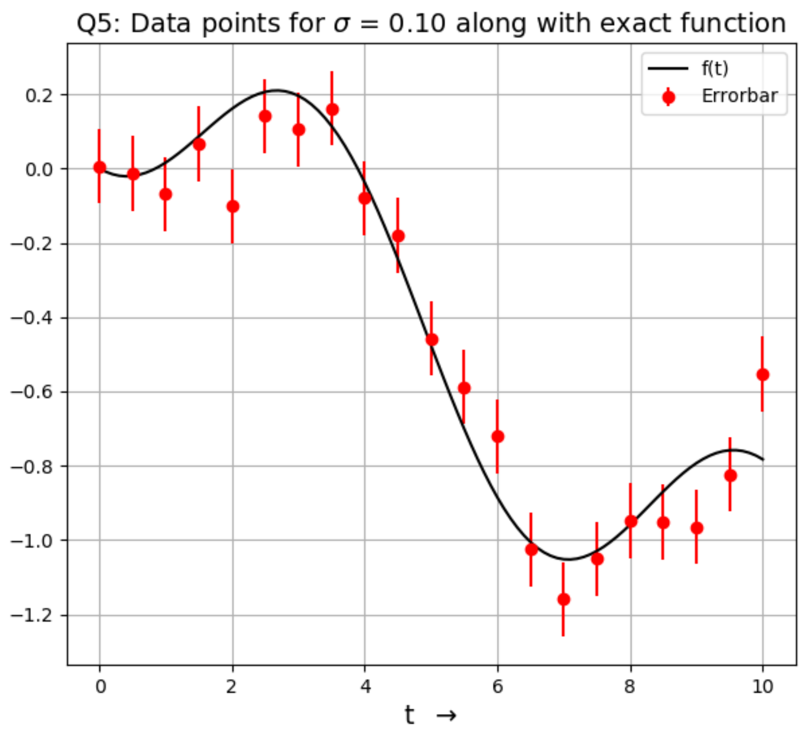
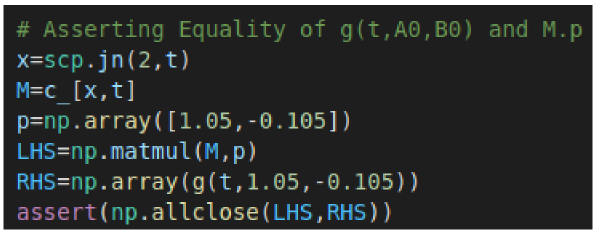


Figure 4: Errorbar plot

Testing Equality



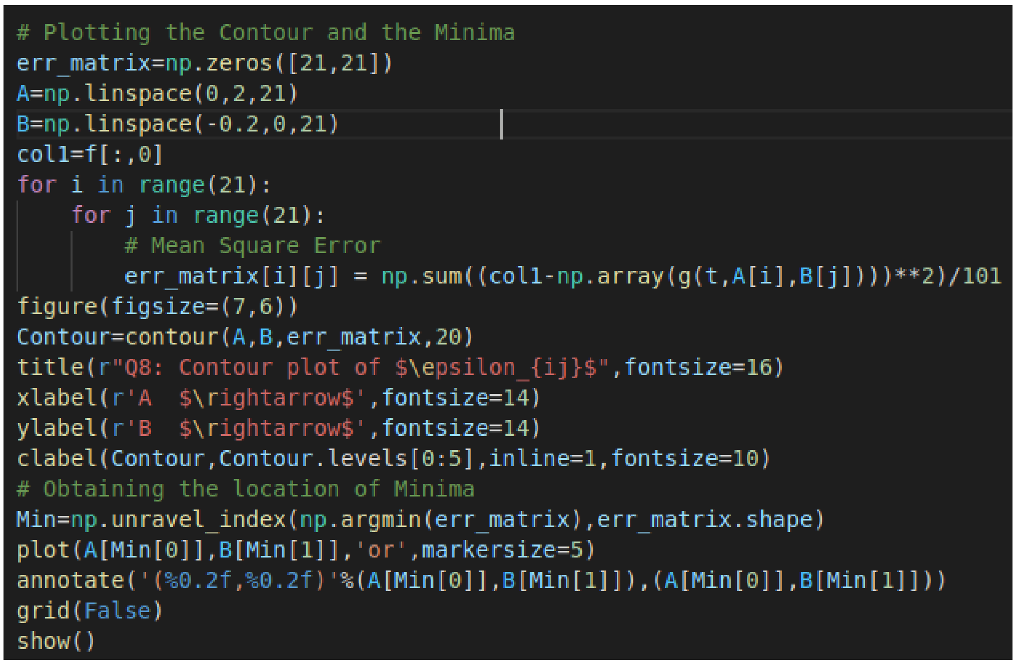
```
# Asserting Equality of g(t,A0,B0) and M.p
x=scp.jn(2,t)
M=c_[x,t]
p=np.array([1.05,-0.105])
LHS=np.matmul(M,p)
RHS=np.array(g(t,1.05,-0.105))
assert(np.allclose(LHS,RHS))
```

Figure 5: Code for testing equality of two vectors

The above code checks whether the vector obtained by multiplication of matrices M and p, and the vector obtained by the function $g(t, A, B)$ are equal.

$$g(t,A,B) = \begin{pmatrix} J_2(t_1) & t_1 \\ \dots & \dots \\ J_2(t_m) & t_m \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = M.p$$

Plotting The Contour



```python
# Plotting the Contour and the Minima
err_matrix=np.zeros([21,21])
A=np.linspace(0,2,21)
B=np.linspace(-0.2,0,21)
col1=f[:,0]
for i in range(21):
    for j in range(21):
        # Mean Square Error
        err_matrix[i][j] = np.sum((col1-np.array(g(t,A[i],B[j])))**2)/101
figure(figsize=(7,6))
Contour=contour(A,B,err_matrix,20)
title(r"Q8: Contour plot of $\epsilon_{ij}$",fontsize=16)
xlabel(r'A  $\rightarrow$',fontsize=14)
ylabel(r'B  $\rightarrow$',fontsize=14)
clabel(Contour,Contour.levels[0:5],inline=1,fontsize=10)
# Obtaining the location of Minima
Min=np.unravel_index(np.argmin(err_matrix),err_matrix.shape)
plot(A[Min[0]],B[Min[1]],'or',markersize=5)
annotate('(%0.2f,%0.2f)'%(A[Min[0]],B[Min[1]]),(A[Min[0]],B[Min[1]]))
grid(False)
show()
```

Figure 6: Code for finding the minima and plotting the contour

The above code finds the mean square error for the data in the first column ($\sigma = 0.1$). The true value is obtained by using the time data in the zeroth column.

$$\epsilon_{i,j} = \frac{1}{101}\sum_{k=0}^{101}(f_k - g(t_k, A_i, B_j))^2$$

The mean square error is calculated for different combinations in A = 0, 0.1,..., 2 and B = -0.2, -0.19,..., 0. Then the minima and contour plot are obtained.

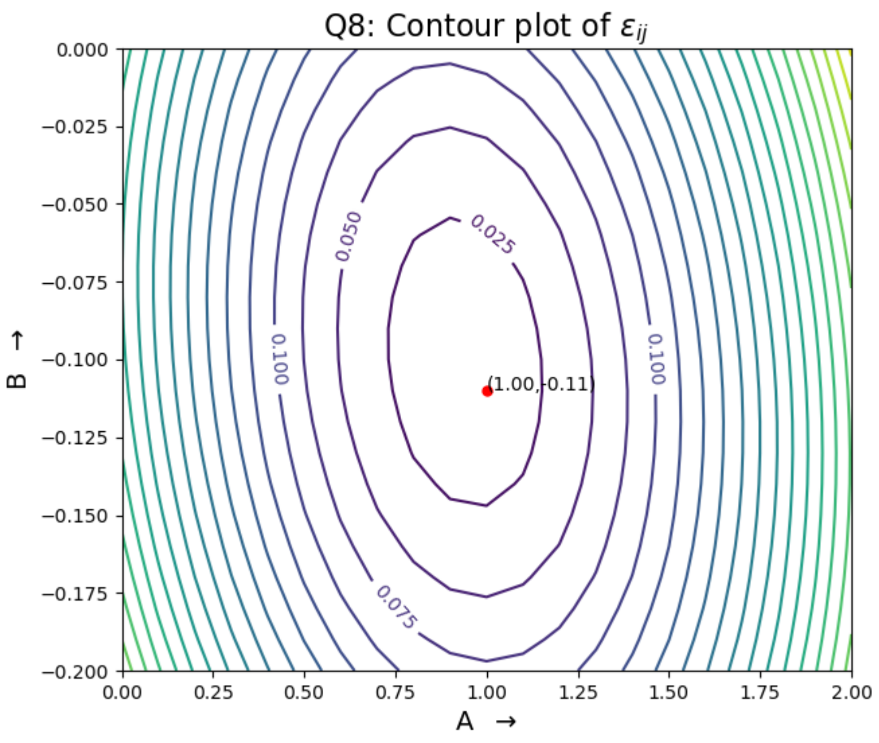


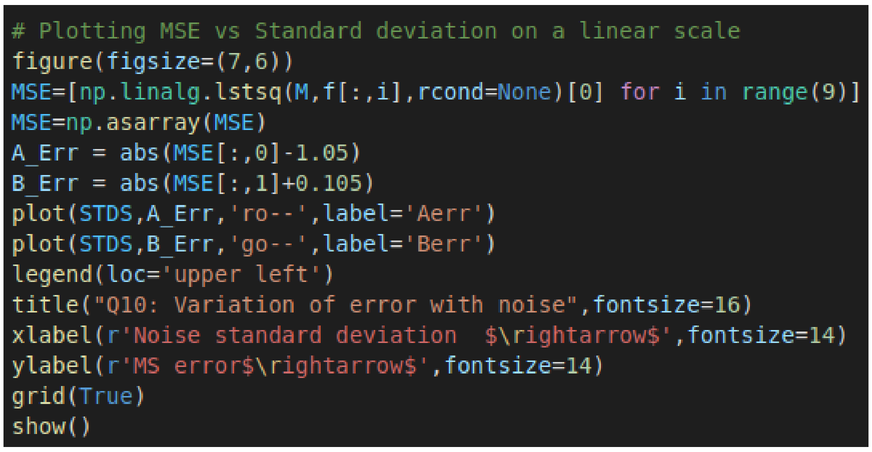
Figure 7: Contour plot along with the mininma

$$g(t, A, B) = AJ_2(t) + Bt$$

The red dot represents the minima which is the best approximate of A and B used to fit the data in the first column ($\sigma = 0.1$). The above graph has only one minima at 1.00 and -0.11 i.e at $A = 1.00$ and $B = -0.11$.

Error In The Estimate Of A And B

The values of A and B, which result in the least mean square error are calculated by using a built-in function in numpy. This is done for all nine data columns. Then the absolute error between the obtained values of A and B and the values given in the problem statement is plotted against the standard deviation of the noise in the nine data columns.



```
# Plotting MSE vs Standard deviation on a linear scale
figure(figsize=(7,6))
MSE=[np.linalg.lstsq(M,f[:,i],rcond=None)[0] for i in range(9)]
MSE=np.asarray(MSE)
A_Err = abs(MSE[:,0]-1.05)
B_Err = abs(MSE[:,1]+0.105)
plot(STDS,A_Err,'ro--',label='Aerr')
plot(STDS,B_Err,'go--',label='Berr')
legend(loc='upper left')
title("Q10: Variation of error with noise",fontsize=16)
xlabel(r'Noise standard deviation  $\rightarrow$',fontsize=14)
ylabel(r'MS error$\rightarrow$',fontsize=14)
grid(True)
show()
```

Figure 8: Code for plotting the error on a linear scale

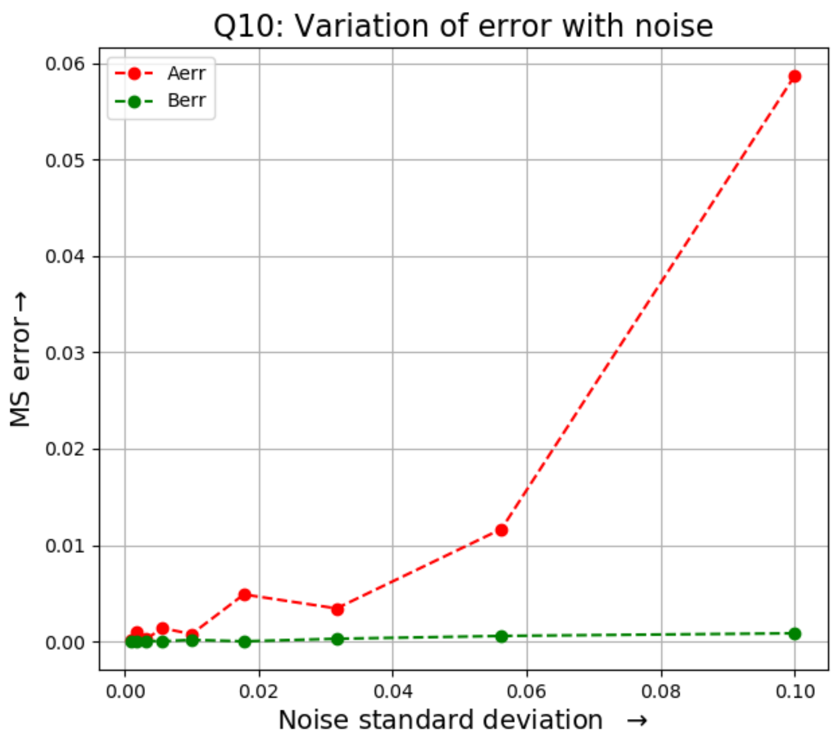
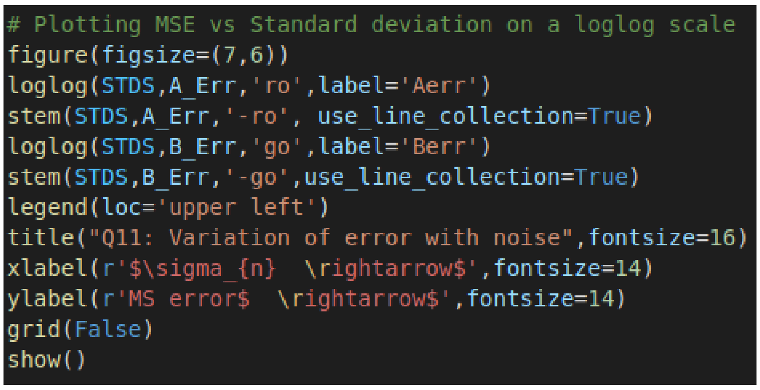


Figure 9: Plot of error in A and B on a linear scale

As we can observe, the first few data points are clustered together. So, now we plot the same data on a loglog scale.



```
# Plotting MSE vs Standard deviation on a loglog scale
figure(figsize=(7,6))
loglog(STDS,A_Err,'ro',label='Aerr')
stem(STDS,A_Err,'-ro', use_line_collection=True)
loglog(STDS,B_Err,'go',label='Berr')
stem(STDS,B_Err,'-go',use_line_collection=True)
legend(loc='upper left')
title("Q11: Variation of error with noise",fontsize=16)
xlabel(r'$\sigma_{n}  \rightarrow$',fontsize=14)
ylabel(r'MS error$  \rightarrow$',fontsize=14)
grid(False)
show()
```

Figure 10: Code for plotting the error on a loglog scale

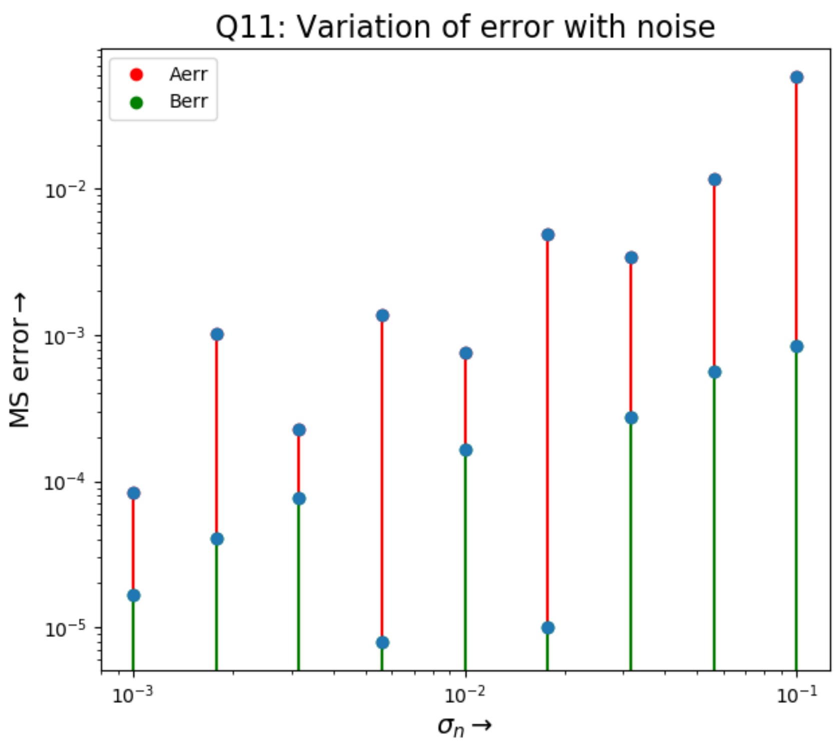


Figure 11: Plot of error in A and B on a loglog scale

As we can observe, the variation in the error in A and B against standard deviation is approximately linear and the error in A and B increases with increasing standard deviation of the noise in the data.

Conclusion

The coefficients A and B in the equation,

$$g(t, A, B) = AJ_2(t) + Bt$$

have been estimated by least squares method for nine different noisy data provided with different standard deviations.

Further, the absolute error in the estimate and the true value was found to be linearly increasing when plotted in a loglog scale against the standard deviation of the noise in the data.

Through this assignment, knowledge on various functions that are part of `numpy, matplotlib and scipy` was gained. Further, the various methods of visualising data like Errorbar, Contour and Stem plots became familiar. Finally, the math behind the function `numpy.linalg.lstsq()` was understood.