EE2703 : Applied Programming Lab
Experiment 3

Mukhesh Pugalendhi Sudha
EE18B114

February 10, 2020

Abstract

The problem is to read noisy data from a file and process it to determine the best fit and to plot various graphs.

Introduction

A piece of code that generates noisy data is given. The file that it produces is used for analysis. The data with noise is plotted using `matplotlib`. Errorbars are marked for noisy data of different standard deviations. A matrix is constructed using $J_2(t)$ and t for use in further questions. Fitting is done using `lstsq` function from `scipy.linalg`. The result is verified by several means.

Additional implementations

The question number is to be given as the argument while running the code to output specific parts of the program. Instead `'all'` can be sent as argument to display every result. A list of available arguments are displayed if no arguments are given.

Least squares estimation is also done using the formula. This output can be seen by entering `'9e'` as argument.

Results

1.Generating data

Data is generated by just copying the given code and running it. It generates a matrix of ten columns - the first with time values and the rest with error-prone data. Each column of this data contains noise of different standard deviations ranging from 10^{-1} to 10^{-3} on a logarithmic scale.

2.Reading data from file

The `loadtxt()` function is used to read and interpret the data as a numpy array. The resulting array is transposed so that indexing becomes easier.

3.Plotting data

The function `plot()` from `matplotlib` is used to plot the values read from the file. `legend()` is used to show labels.

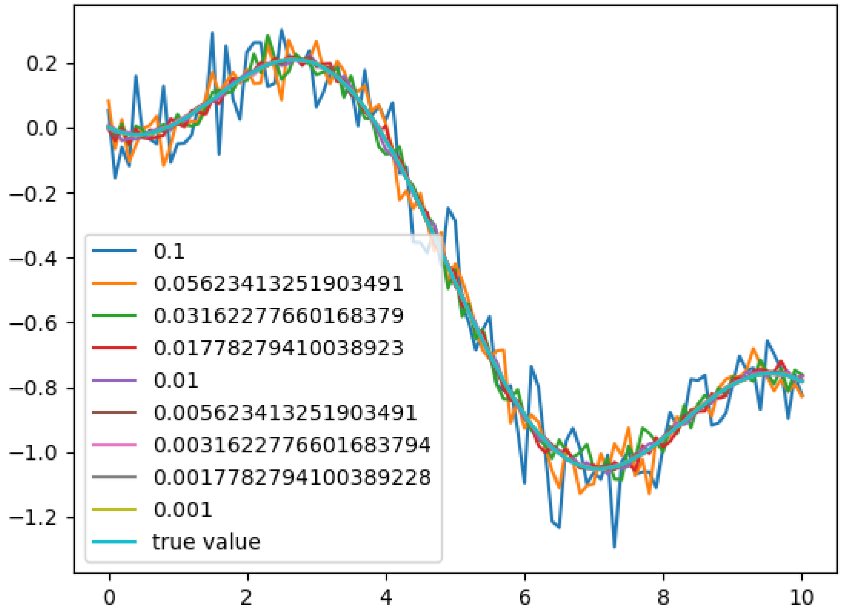

Figure 1: All values

4.Plotting true value

The true value is generated using the function below:

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

t varies from 1 to 100. A and B take the values 1.05 and -0.105 in order to produce the required data. $J_2(t)$ is the bessel function of the second kind. It is present in `scipy.special` as `jn(order, t)`

5.Plotting errorbars

The function `errorbars()` is used to show errorbars if the standard deviation is known. Errorbars are shown for every fifth data point.

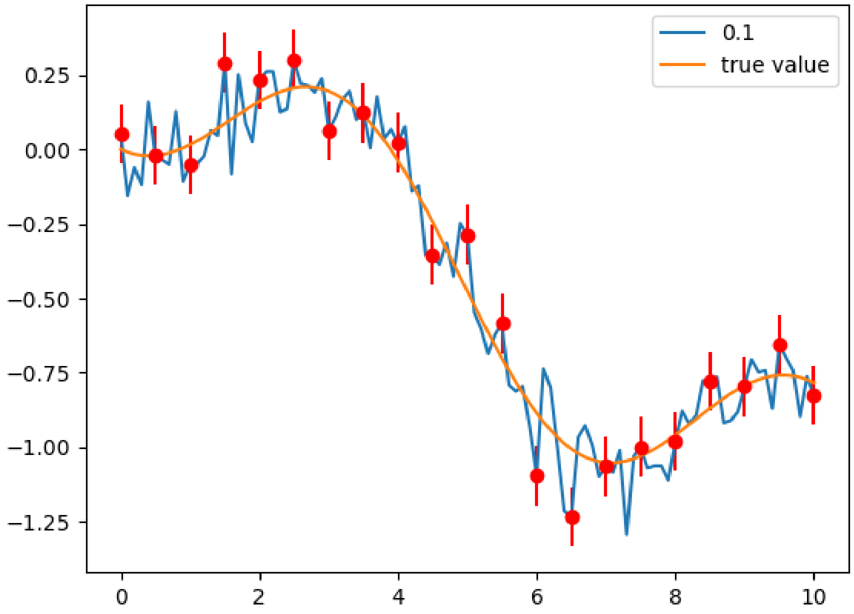

Figure 2: Error bars

6.Constructing matrix

The above-mentioned function is written as a linear combination of $J_2(t_n)$ and t_n. This gives the exact same values as the actual function for the same parameters A and B. This can be made sure by generating data using the above two methods and taking a sum of squared differences. If it turns out to be zero, then the data is same.

7.Mean square error matrix

A set of values for A and B are considered and output is computed for each combination of A and B. This is stored in a matrix ε whose elements are populated as follows.

$$\varepsilon_{ij} = \frac{1}{101}\sum_{k=0}^{100}(f_k - g(t_k, A_i, b_j))^2 \tag{1}$$

8.Contour plot

The matrix generated previously is used to make a contour plot to analyse the error produced by different values of A and B. The error is minimum for the values that result in the best fit.

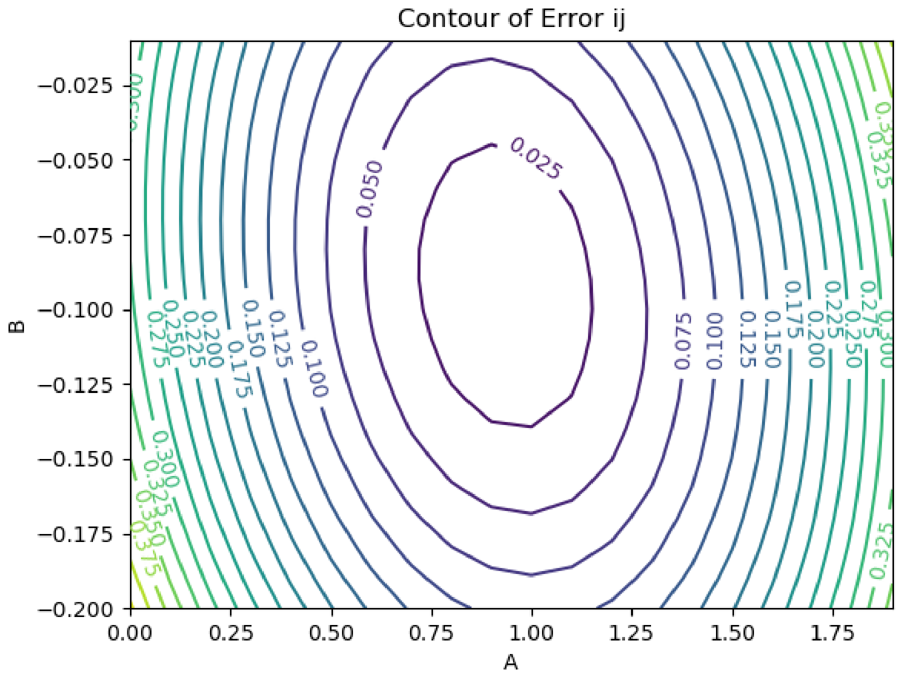

Figure 3: Contour plot

The graph has a minimum near $A = 1.05$ and $B = -0.105$. (This is not exact because the data used is noisy)

9.Finding the solution

The solution is computed using the `lstsq()` function from `scipy.linalg`. It computes the closest possible solution using the formula $(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}$

10.Plotting error in estimate

The solution is computed for data with noises of different standard deviations. Then the error in estimate of A and B is plotted as a function of the standard deviation.

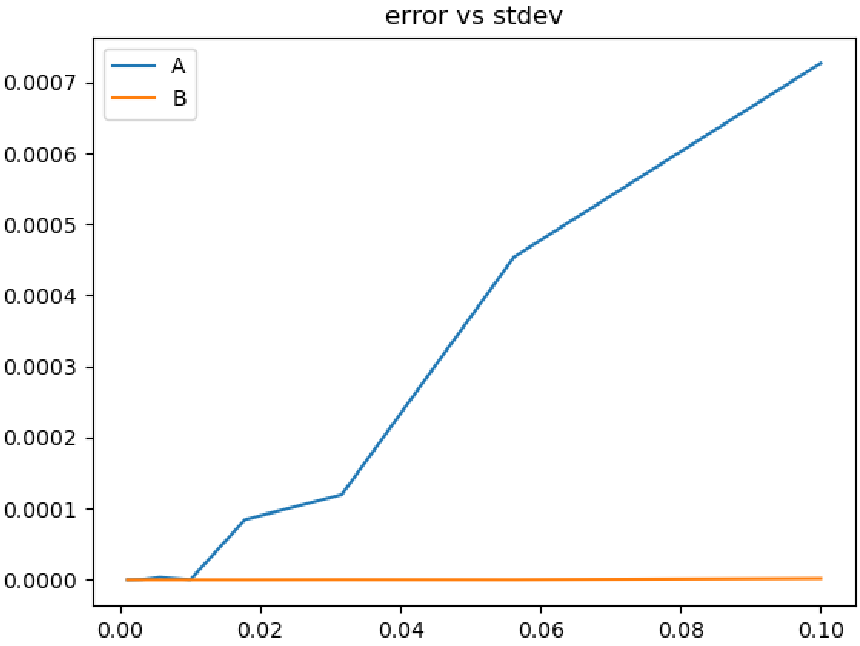

Figure 4: Error in estimate

The error in A seems to be growing linearly. The error in B is of much less magnitude because of its lower order.

11. Using loglog to plot

The same plot is redone using `loglog()` to dsiplay the plot in logscale.

Include all the plots asked in the question here.

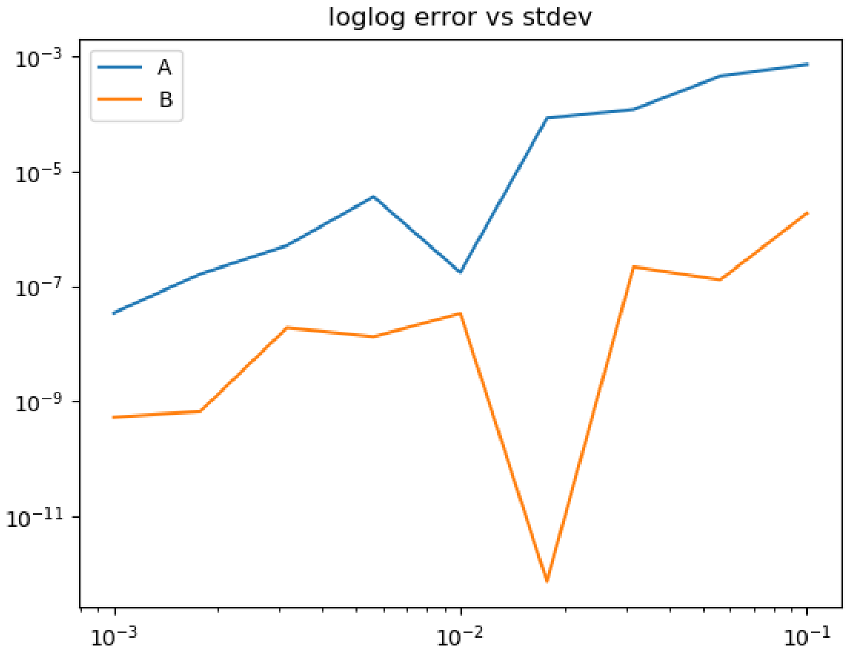

Figure 5: Error in estimate (logscale)

The error is approximately linear with respect to standard deviation except for a few outliers.

Conclusions

I have learnt how to plot data and contours using matplotlib. The least square estimate can be found using `lstsq()` function.