Assignment 3 : Fitting Data to Models

Rohit Kumar (EE20b110)

February 18, 2022

Abstract

In this assignment we aim to :

- Extract and observe the data from a file and make visualization plots of the same.
- Observe the error in fitting the Least Error Function to any given function.
- Find the relation between the error observed and the noise in the data.

1 Generating the data

The data required is generated by running the script `"generate_code.py"` script file. This generated a file `"fitting.dat"`. This file consists of 10 columns each having 100 data points, in which the first column corresponded to time stamp and the remaining columns are data of the noisy functions $f(t) = 1.05J_2(t) - 0.105t + n_\sigma(t)$, where the noise signal $n_\sigma(t)$ is *normally distributed* and its probability distribution is given by:

$$P(n(t)|\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{n(t)^2}{2\sigma^2}} \tag{1}$$

2 Analyse the data

We know the actual function to be fitted which is given by,

$$f(t) = 1.05J_2(t) - .105t \tag{2}$$

where $J_2(t)$ is the *Bessel Function of the first kind of Order 2*. We can plot it's graph also using the `pyplot` library by knowing the data points over a specific period of time. We define a python function $g(t, A, B)$ defined as follows:

```
def g(t, A, B):
    return A*sp.jn(2,t) + B*t
```

We keep the time period same for all the plots and it is defined by:
`t = linspace(0,10,101)`

On plotting the true function's value along with all the 9 noise added values, the following plot was generated:

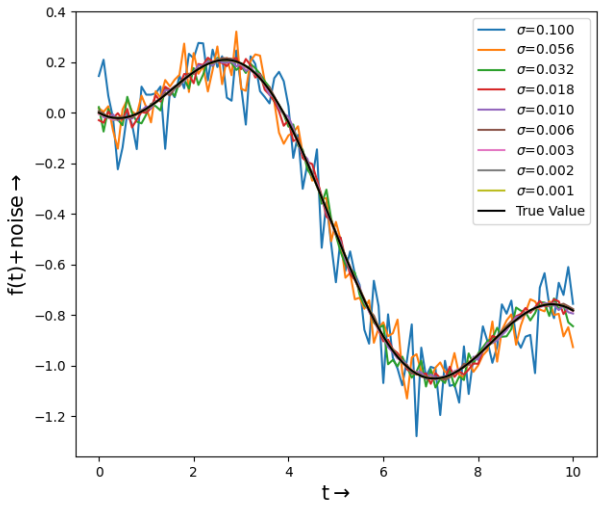

Figure 1: Noisy Data with True Data

As we can see from Figure 1, the "noisiness" of the data increases with increasing value of σ.

Another view of how the noise affects the data can be seen below:

The blue lines (error bar) indicate the deviation of the noisy data from the original data, at that value of t. It is plotted at every 5th point to make the plot readable else the plot would have been clumsy.

3 The Matrix equation

The function which we need to approximate can also be computed by means of a matrix equation. We obtain the function $g(t, A, B)$ as a column vector using the following matrix equation:

$$g(t, A, B) = M.p \tag{3}$$

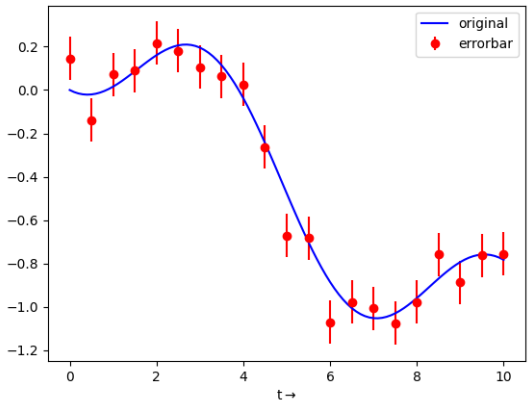

Figure 2: Noisy Data with Errorbar

where the matrix

$$M = \begin{bmatrix} J_2(t_1) & t_1 \\ ... & ... \\ J_2(t_m) & t_m \end{bmatrix}, p = \begin{bmatrix} A \\ B \end{bmatrix} \tag{4}$$

While performing the computation of the function we generate the matrix M and compute $M.p$ and get the function output.

4 Best fit approximation to the noisy data

From the data given to us we see that, we must approximate the noisy data to a function of the form:

$$g(t, A, B) = AJ_2(t) + Bt \tag{5}$$

where we must find the values of the constants A and B by using the least square estimation. For this purpose we take the value of A = 0,0.1,0.2,...,2 and B = -0.2,-0.19,...,0 and iterate through each of the above values of A and B, and finally get the mean squared error between the given noisy data and the approximated function by taking the each pair of above values of A and B.

The error is computed by means of the following equation:

$$\epsilon_{ij} = \frac{1}{101} \sum_{k=0}^{101} (f(t_k) - g(t_k, A_i, B_j))^2 \tag{6}$$

where ϵ_{ij} is the error for (A_i, B_j).

After computing the mean squared error we plot a contour plot for the same with the values of A and B on the x-axis and y-axis respectively. The plot is as shown below:

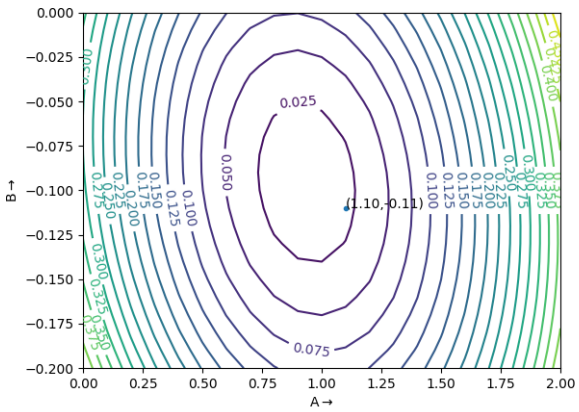

Figure 3: Contour Plot of ϵ_{ij}

5 Performing the Least error fit

We are required to find the best measure estimate for the constants A and B in the equation (3) for the noisy data. We do this by applying the method of least square estimation. This is implemented in python by using the `lstsq()` function from the `scipy` package. We pass the matrix M and the noisy data as the input to the `lstsq()` function and we get the best fit values of A and B. We perform this for all the 9 random noise added data.

We then compute the mean squared error in the coefficients A and B to their actual value of `A0 = 1.05` and `B0 = -0.105` and plot the same against the standard deviation of the random noise. The plot is shown below:

We aren't able to seek in much information to get a relation between $\sigma and\ \epsilon$ from the above plot. If we plot the same in `log-log` scale along with their error bar, the plot looks like:

From the above plot we can infer that there is a linear relation between σ_n and ϵ, which is the required result to be produced.

6 Conclusion

We have used the given noisy data and computed the best possible for the underlying model parameters by minimizing the mean squared error. By plotting the mean squared error between the estimated model parameters

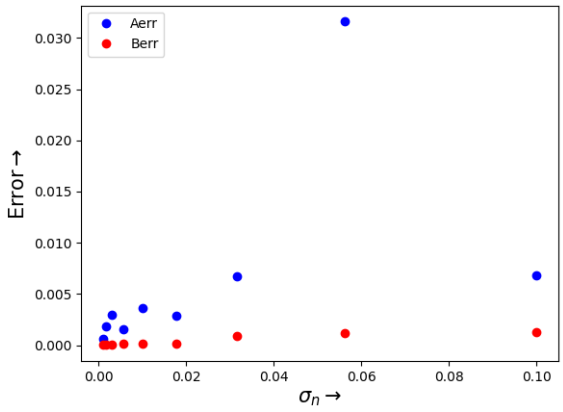

Figure 4: Mean squared error vs Noise Standard deviation

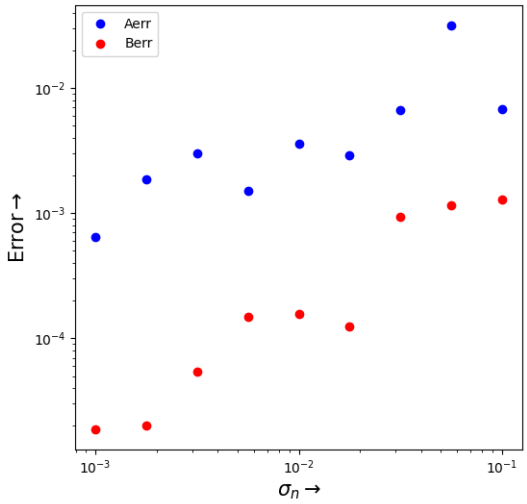

Figure 5: Mean squared error vs Noise Standard deviation log-log plot

A and B and the true value function's parameters A0 and B0, we see that this error is varying linearly with the standard deviation of the noise added to true value.