Assignment No 3

M Srinivas Reddy

February 11, 2020

1 Abstract

In engineering, we try to acquire data from various sensors and devices and analyse the data. But the data produced is almost never perfect and there is always noise associated with it. So we try to model the data and get a function for easier calculations. We are going to do some calculations with coefficients and errors of functions involved and plot various graphs.

2 Introduction

In this assignment we are going to try to estimate the coefficients of a function when noise is added to it. Different coefficients are tested and the coefficients with the lowest error is chosen.
If we know the basic functions present in the data, we have to estimate the coefficients of the functions and chose the coefficient which is closest to actual data. Here the main function is a first order Bessel function.

$$G(t) = 1.05J(t) - 0.105t \tag{1}$$

$$F_1(t) = G(t) + n(t) \tag{2}$$

3 Extracting Data and Identifying Noise

3.1 Extracting Data

We need to extract data from 'fitting.dat' file. It consists of 10 columns. The first column corresponds to time and the rest are samples of data. To extract data we can use 'for loops' and store them in arrays.

```
for i in range(0,N):
    dat[i]=dat[i].split()
    for j in range(0,k):
        vect[i][j]=float(dat[i][j])
```

3.2 Noise Identification

These data samples are random amount of noise added to them. Now we plot the data using **plot** function. The noise in different samples of data is

Figure 1: Functions with different amount of noise

different and we can observe that in the above figure. Now taking one of data set we plot an errobar on top of the known function 'F(t)'.

```
stdev=std(G(t)-F(t))
plot(t,g(t,A,B),color='black',label='f(t)')
errorbar(t[::5],y[0][::5],yerr=stdev[0],fmt='ro',label='errorbar')
```

Errorbar gives us an idea about how much the generated function is deviating from G(t).

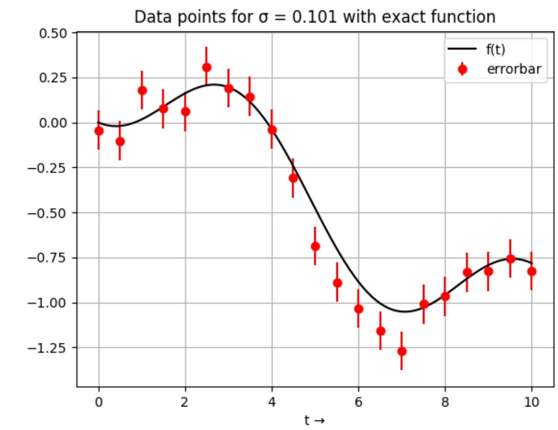

Figure 2: Data points for $\sigma = 0.101$ with exact function

4 Identifying Coefficients

The actual function can be in terms of J(t) and t as:

$$G(t) = \begin{bmatrix} J(t_0) & t_0 \\ ... & ... \\ J(t_m) & t_m \end{bmatrix} \begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \tag{3}$$

Similarly we need to multiply J(t),t with different Coefficients A and B and varying A and B linearly.Let M and p be defined as follows:

$$M = \begin{bmatrix} J(t_0) & t_0 \\ ... & ... \\ J(t_m) & t_m \end{bmatrix}, p = \begin{bmatrix} A \\ B \end{bmatrix} \tag{4}$$

To confirm that M multiplied by A_0 and B_0 gives G(t) we can compare M.p_0 with G(t) obtained from first equation.

```
print("dot(M,[1.05,-0.105]) == G(t)")
```

After constructing the matrix, find out the mean squared error between G(t) and and $F_k(t) = M.[A, B]$.

$$A_i = 0, 0.1, ..., 2, B_j = -0.2, -0.19, ..., 0 \tag{5}$$

$$\epsilon_i j = \frac{1}{101} \sum_{k=0}^{101} (f_k - g(t_k, A_i, B_j)^2) \tag{6}$$

Plot contour of $\epsilon_i j$ with A along x-axis and B along y=axis.

```
cont=contour(a,b,error)
colorbar(cont)
```

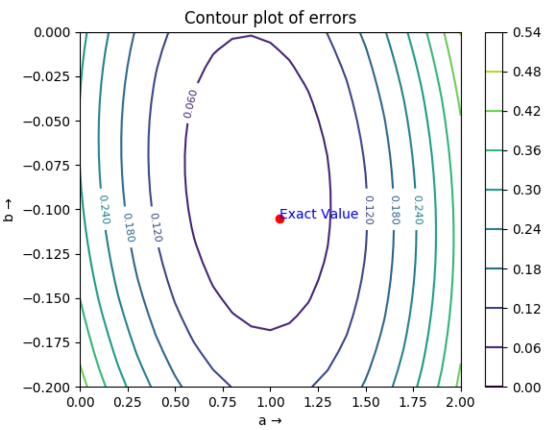

Figure 3: Contour Plot

We can observe from the contour plot that there exists a minimum value of $\epsilon_i j$ for a particular A,B.

5 Finding The Best Estimate

In the previous section we obtained different values of errors by using different coefficients A and B. With an inbuilt function in python called **lstsq** we can find the best estimate of A and B for which the error $\epsilon_i j$ is minimized.

```
p,resid,rank,sig=lstsq(func,y[i])
```

lstsq finds the error between the functions and finds A and B for which derivative of error with respect to p matrix is minimum.
We then plot a figure showing the variation of error in A and B with variying amounts of noise.

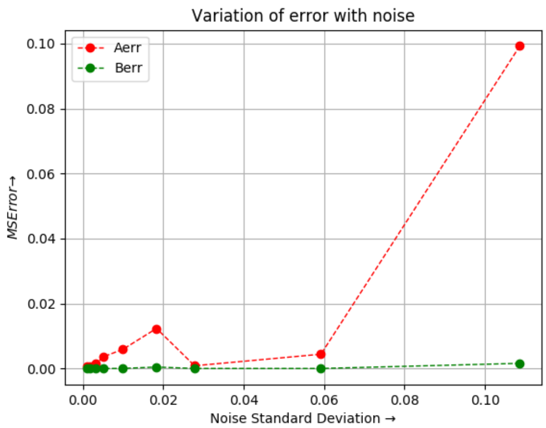

Figure 4: Error in A and B

We can say that the plot is not linearly increasing.
Similarly we plot another curve with x-axis and y-axis in log scale.

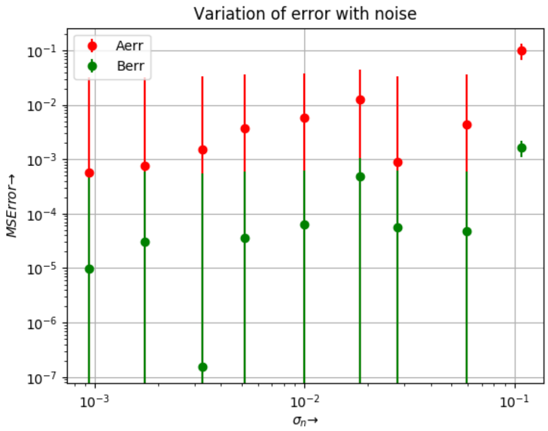

Figure 5: Log Error in A and B

We can see that errorbar lengths are linearly increasing in log scale and so we can say that error in estimation increases with standard deviation of noise when both are measured in log scale.

6 Observation

From this experiment we learnt that we will be able to approximate a function if noise is present if we know the underlying main functions involved. All we have to do is try different coefficients in a reasonable range and then find the best estimate using functions available in python.