EE2703: Applied Programming Lab
Assignment 3: Linear Least Squares Fitting

Soham Roy
EE20B130

February 14, 2022

1 Introduction

For this assignment we use the Bessel function and Gaussian noise to study the effect of changing standard deviation on the linear fitting of the data.

$$f(t) = AJ_2(t) + Bt + n(t)$$

Where, $A = 1.05$, $B = -0.105$, J_2 = Bessel function, $n(t)$ = Noise function. Our aim is to relate the error in estimating A, B to the standard deviation of the Gaussian noise.

2 Subquestions

2.1 Generate the Data

On running the python script `generate_data.py`, the data is written to the file `fitting.dat`. The `scipy` library has been used to calculate the Bessel function. The `numpy.array t` contains 101 equally spaced numbers between 0 and 10, which are fed into the Bessel function and added with noise to generate the data.

2.2 Load the Data

The data from `fitting.dat` is loaded using `numpy.loadtxt()`. The first column of `raw_data` are the time values, and the subsequent columns are the corresponding noisy data values.

```
raw_data = loadtxt(DATAFILE)

Time = raw_data[:, 0]
F = raw_data[:, 1:]
```

2.3 The Function and the Noise

The true values are generated using `F_true = g(Time)`, where the function `g(t,A,B)` defined as:

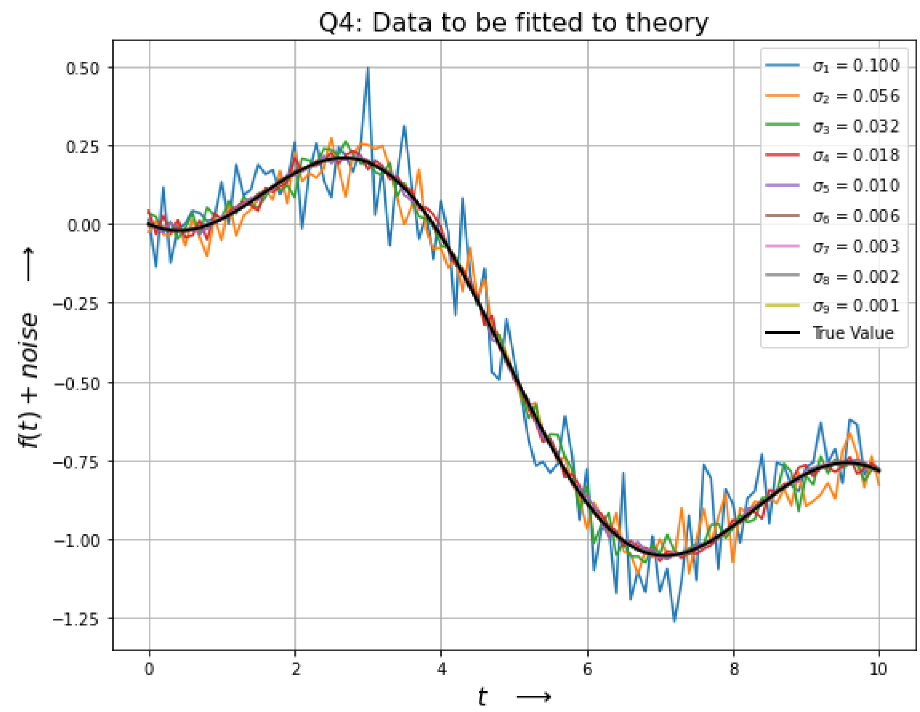
```
def g(t, A=A_true, B=B_true):
    return A * jn(2, t) + B * t
```

where `A_true, B_true = 1.05, -0.105` are the true values of A and B. The noise is generated using `Sigma = logspace(-1, -3, K)`, where `K = 9` is the number of curves to be plotted.

2.4 Plot the Data

We plot the data along with the function `g(t,A,B)` for `A=1.05`, `B=-0.105`.

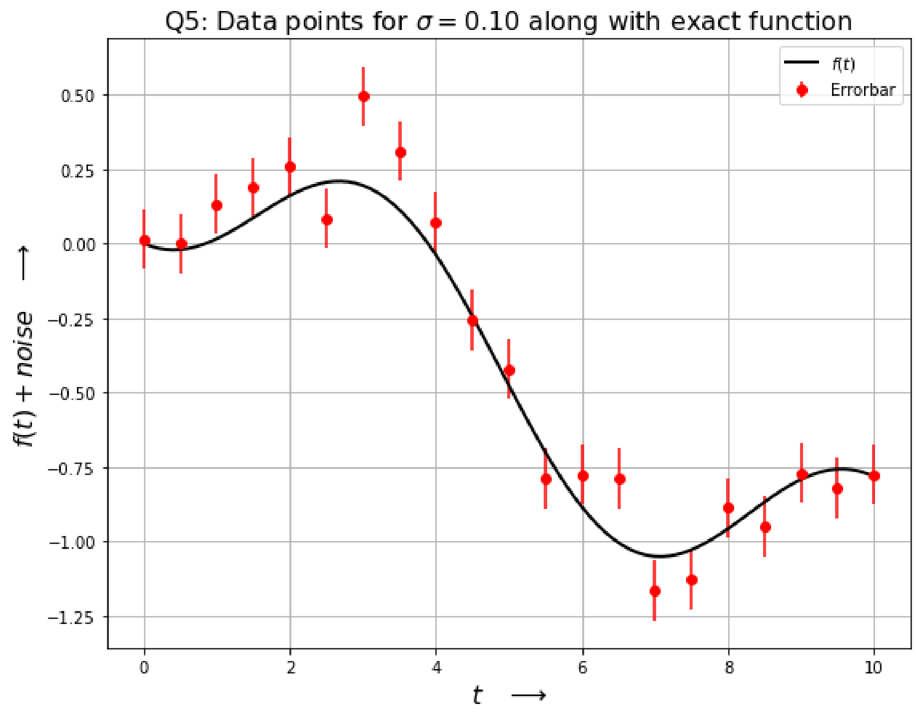
```
plot(Time, F)
plot(Time, F_true, color='black', lw=2)
```



2.5 Plot with Error Bars

A plot of the first column of data with error bars has been generated, with every 5^{th} data item plotted for readability. The exact curve has also been plotted to see how much the data diverges.

```
errorbar(Time[::5], F[::5, 0], Sigma[0], fmt="ro")
plot(Time, F_true, color='black', lw=2)
```



2.6 Equate the Vectors

$$g(t, A, B) = \begin{pmatrix} J_2(t_1) & t_1 \\ \cdots & \cdots \\ J_2(t_m) & t_m \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} \equiv M \cdot P \tag{1}$$

`F_true.reshape(N, 1)` is $g(t, A, B)$, the vector of the true values.
`M = c_[jn(2, Time), Time]` generates M, which is multiplied by `[[A_true], [B_true]]`, i.e. P, to obtain the RHS vector.
`assert` ensures that the two vectors are equal by evaluating `numpy.allclose()`, as we cannot reliably equate floats.

2.7 Mean Squared Error

The mean squared error between the data (f_k) and the assumed model has been calcuated for every combination of A and B, where A and B range from 0 to 1 and -0.2 to 0 respectively.
The following formula has been implemented:

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

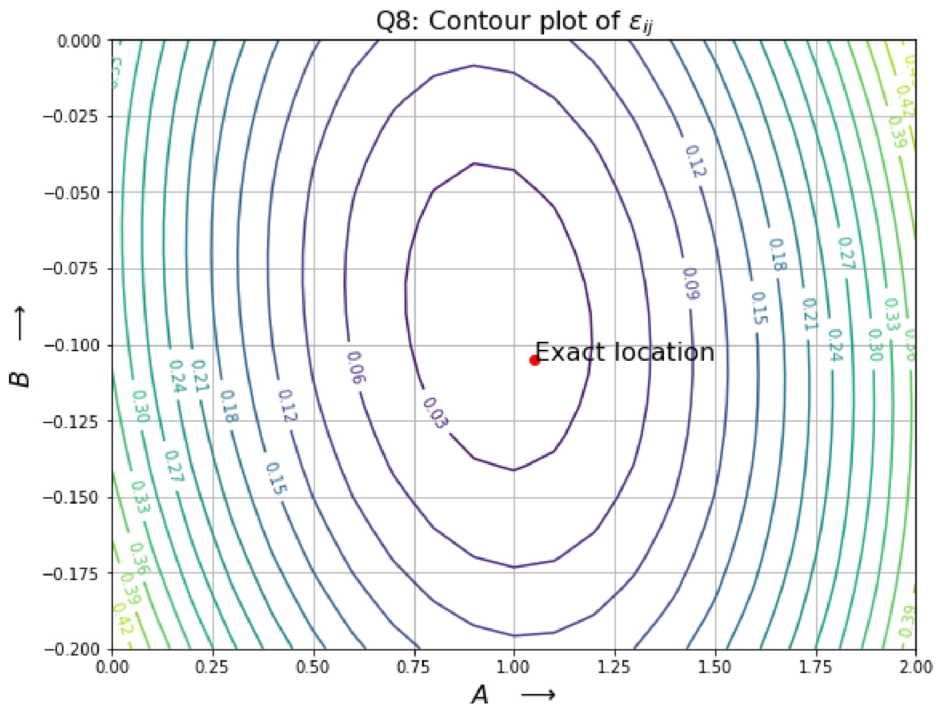
by looping over the following line of code:

```
eps[i][j] = mean((F[:, 0] - g(Time, A[i], B[j])) ** 2)
```

2.8 Plot the MSE

The contour plot has been generated by `contour`, and labeled using `clabel`. Further, the exact location of `(A_true, B_true)` has been plotted and annotated.

```
clabel(contour(A, B, eps, 15))
plot([A_true], [B_true], "ro")
annotate("Exact location", xy=(A_true, B_true), size=16)
```



2.9 Best Estimate for A and B

The matrix M, defined in Equation 1, has been used to find the best estimate of A and B for the first column of data. This was done by computing the least-squares solution for it using `scipy.linalg.lstsq()` to print:

```
"Best estimate:   A = {}, B = {}".format(*lstsq(M, F[:, 0])[0])
```

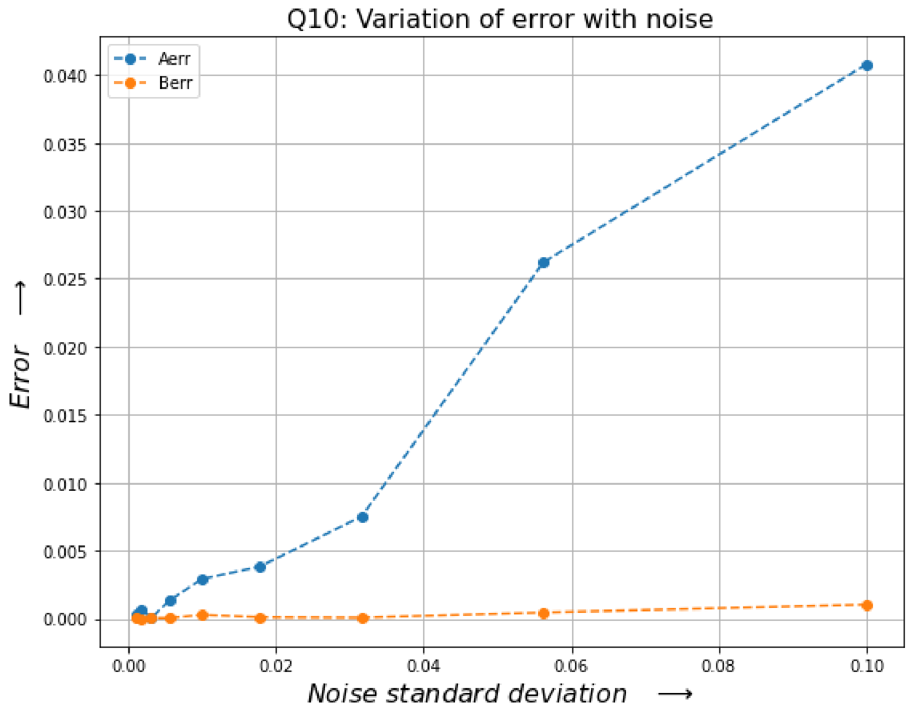
2.10 Plot the Errors in A, B

The errors in A and B have been calculated by subtracting the true values:

```
Aerr, Berr = abs(lstsq(M, F)[0] - [[A_true], [B_true]])
```

These have thus been plotted against the standard deviations of the data:

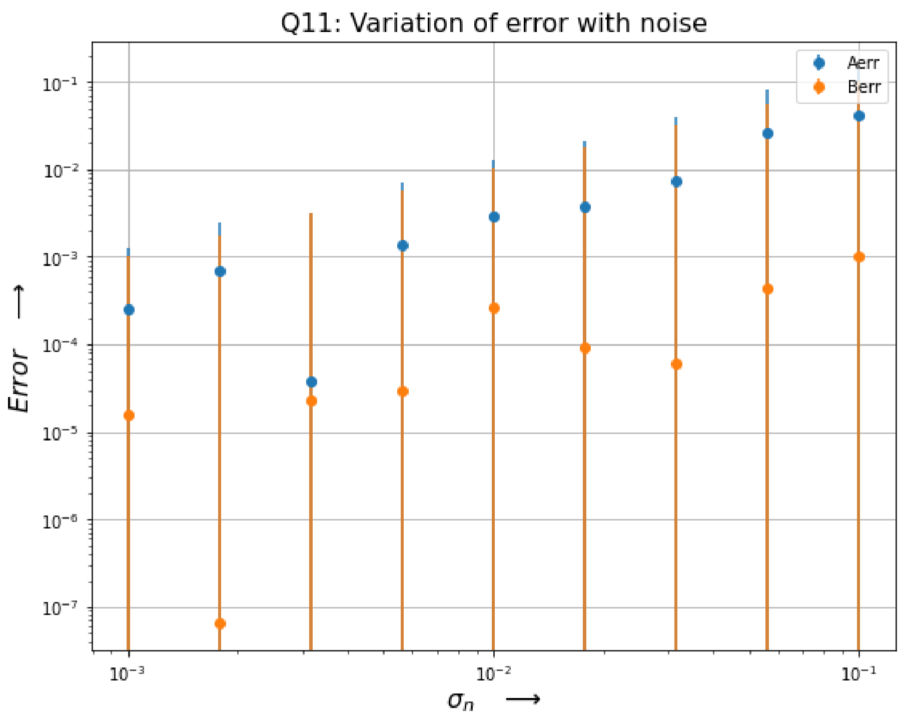
```
plot(Sigma, Aerr, 'o', linestyle="dashed")
plot(Sigma, Berr, 'o', linestyle="dashed")
```



2.11 Plot using log-log Scale

The scale of the graph has been changed to log-log, with an `errorbar()` plot:

```
xscale("log")
yscale("log")
errorbar(Sigma, Aerr, Sigma, fmt="o")
errorbar(Sigma, Berr, Sigma, fmt="o")
```



3 Conclusion

As we see from the plots, the error in estimated A and B increases with increase in the standard deviation of the Gaussian noise in the data. Further, we see that the increase is somewhat linear when plotted on a log-log scale.