EE2703: Assignment # 3

Sourav Sahoo

February 10, 2019

Part 1

Run the following code to generate the table.

```
import numpy as np
sigma = np.logspace(-1,-3,9)
print(sigma)
```

Sl. No.	σ
1	0.1000
2	0.0562
3	0.0316
4	0.0178
5	0.0100
6	0.0056
7	0.0032
8	0.0018
9	0.0010

Table 1: Values of σ

The function `g(t,A,B)` is defined and the graph of $f(t)$ with noise due to different values of σ are plotted as described in the code below.

```
def g(t,A,B):
    """
        Return the A*sp.jn(2,t) + B*t
    """
    return A*sp.jn(2,t) + B*t
gold = g(t,A0,B0)
N,k = 101,9
t = np.linspace(0,10,N)
y = 1.05*sp.jn(2,t)-0.105*t # f(t) vector
```

```
Y = np.meshgrid(y,np.ones(k),indexing='ij')[0] # make k copies
n = np.dot(np.random.randn(N,k),np.diag(sigma))
yy = Y+n
plt.grid()
plt.plot(t,yy)
plt.plot(t,gold,color='k')
plt.xlabel(r'$t$');plt.ylabel(r'$f(t)+error$')
plt.title(r'Data to be fitted to theory')
plt.legend(['0.1000','0.0562','0.0316','0.0178','0.0100','0.0056','0.0032',
            '0.0018','0.0010','gold'])
plt.show()
```

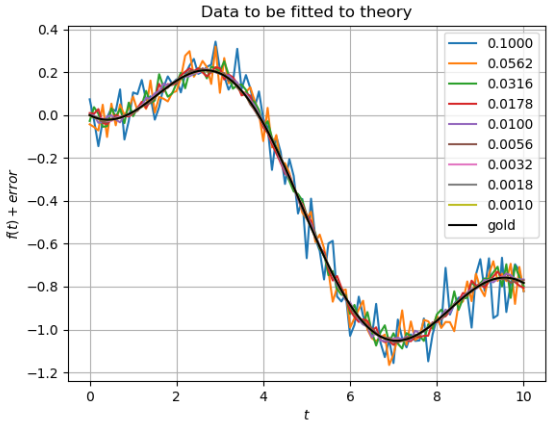

Figure 1: $f(t)$ with noise for different σ

Part 2

Then the error bars corresponding to $\sigma = 0.1$ are generated according to the code.

```
plt.grid()
plt.plot(t,gold)
plt.errorbar(t[::5],f[::5],yerr=0.1,fmt = '.',capsize=3)
plt.title('Error bars for sigma = 0.1')
plt.xlabel('t')
plt.ylabel('f(t) + error')
plt.legend(['gold','error_bar'])
plt.show()
```

The graphs with error bars is given below.

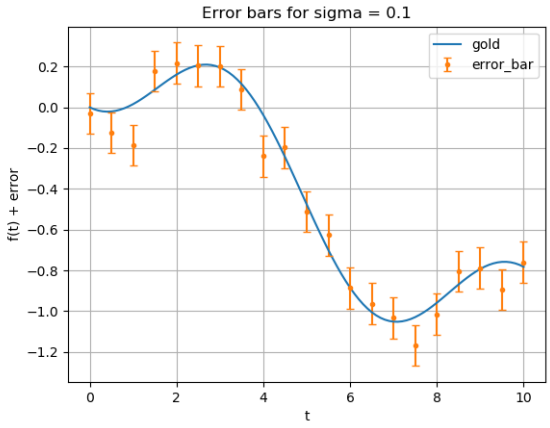

Figure 2: Error bars for $\sigma = 0.1$

Part 3

The contour plot has a single minima. For $\sigma = 0.1000$ the minima occurs at $A = 1.1051$ and $B = -0.1063$. The mean squared error of $f(t)$ at that particular value of A and B is 0.0082. Then the contour plot is generated.

```
plt.grid()
plt.text(A0,B0,'exact location')
cp = plt.contour(A,B,error,20)
plt.clabel(cp, inline=True,fontsize=10)
A,B = np.meshgrid(A,B)
plt.title('Contour Plot for sigma = 0.1')
plt.xlabel('A')
plt.ylabel('B')
plt.show()
```

Part 4

The error matrix is then calculated.

```
M = np.vstack((sp.jn(2,t),t)).T
A = np.linspace(0,2,21)
B = np.linspace(-0.2,0,21)
error = np.zeros((len(A),len(B)))
mean_error = np.zeros(9)
a,b = [],[]
for k in range(1,10):
```

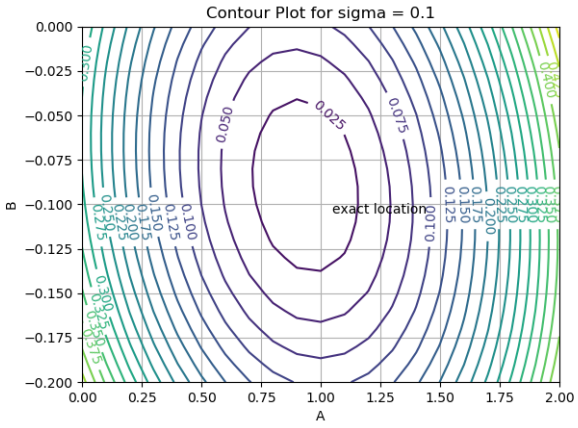

Figure 3: Contour Plot for $\sigma = 0.1$

```
if k == 1:
    for i in range(len(A)):
        for j in range(len(B)):
            error[i,j] = np.mean(np.square(data[:,k] - g(t,A[i],B[j])))
X,_,_,_ = np.linalg.lstsq(M,data[:,k],rcond=-1)
a.append(X[0])
b.append(X[1])
mean_error[k-1] = np.mean(np.square((M.dot(X) - data[:,k])))
```

The values of A and B are found out using `scipy.linalg.lstsq` in Python. Then the using the M matrix given in the problem the values of the function are calculated. $f = M \begin{bmatrix} A \\ B \end{bmatrix}$. The error in A and B from the original values of A_0 and B_0 is plotted below: Then the mean squared error values in $f(t)$ are calculated using the corresponding column

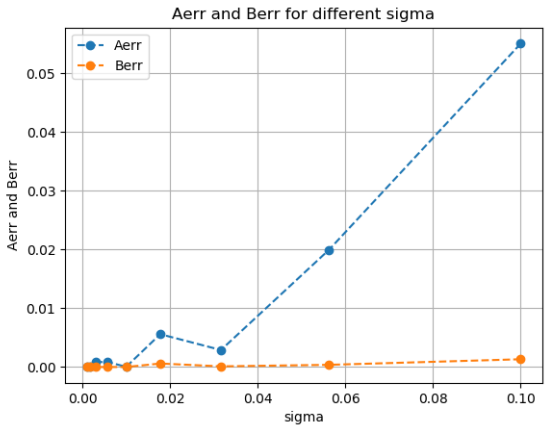

Figure 4: Error in the A and B

in `fitting.dat` and f.

```
Aerr = abs(A0-np.asarray(a))
Berr = abs(B0-np.asarray(b))
plt.grid()
plt.plot(sigma,Aerr,'o--')
plt.plot(sigma,Berr,'o--')
plt.title('Aerr and Berr for different sigma')
plt.xlabel('sigma')
plt.ylabel('Aerr and Berr')
plt.legend(['Aerr','Berr'])
plt.show()

plt.grid()
plt.loglog(sigma,Aerr,'o')
plt.loglog(sigma,Berr,'o')
plt.errorbar(sigma,Aerr,yerr=0.1,fmt = 'o')
plt.errorbar(sigma,Berr,yerr=0.1,fmt = 'o')
plt.title('Aerr and Berr for different sigma')
plt.xlabel('sigma')
plt.ylabel('Aerr and Berr in logscale')
plt.legend(['Aerr in logscale ','Berr in logscale'])
plt.show()
```

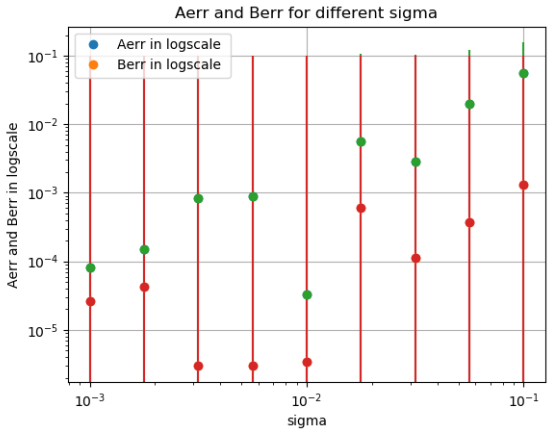

Figure 5: Log scale Error in the A and B

Then the mean squared error values in $f(t)$ are calculated using the corresponding column in `fitting.dat` and f.

```
plt.grid()
plt.plot(sigma,mean_error,'o--')
plt.title('MSError for f(t) different values of A and B')
plt.xlabel('sigma')
plt.ylabel('MSError')
```

```
plt.show()

plt.grid()
plt.loglog(sigma,mean_error,'o--')
plt.title('MSError of f(t) for different values of A and B')
plt.xlabel('sigma in logscale')
plt.ylabel('MSError in logscale')
plt.show()
```

Sl. No.	σ	A	B	MSError in $f(t)$
1	0.1000	1.1050	-0.1063	8.2465e-03
2	0.0562	1.0699	-0.1053	3.1010e-03
3	0.0316	1.0528	-0.1051	1.0081e-03
4	0.0178	1.0555	-0.1055	2.6623e-04
5	0.0100	1.0499	-0.1050	8.3921e-05
6	0.0056	1.0508	-0.1050	3.0413e-05
7	0.0032	1.0508	-0.1049	7.6048e-06
8	0.0018	1.0501	-0.1050	2.6296e-06
9	0.0010	1.0499	-0.1049	9.3969e-07

Table 2: Approximate A and B for different values of σ

Part 5

The final plots with error on y-axis and sigma on x-axis is plotted,first in linear scale and then in loglog scale.

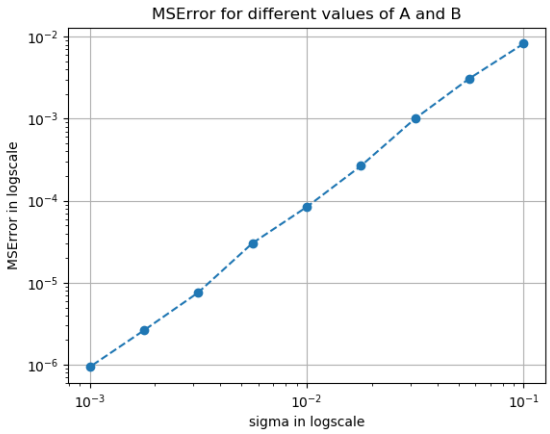

Figure 6: Log-Log Plot