Python Assignment -3

Swapnil Basak

EE11B122, IIT Madras

Abstract

We will concentrate on reading in data, analyzing them, manipulating them and studying the effects of varying levels of noise added to signals.

Program

We include the shebang and import out libraries

Code

1a ⟨*code* 1a⟩≡

```
#! usr/bin/python

from scipy import *
import sys
import matplotlib.pyplot as graph
import scipy.special as sp
from scipy.linalg import lstsq
import numpy as np
```

Now, we set our constants and load the fitting.dat file. we read in the data column by column

1b ⟨*program* 1b⟩≡

```
A = 1
B = -0.02546303
# 2. loading file and reshaping
L = loadtxt('fitting.dat'.unpack=True)
```

We then declare the computing function to see the amount of noise added.

1c ⟨*function* 1c⟩≡

```
# 4. Declare the function g and compute
def g(t,a,b):
        g = a*sp.jn(2,t) + b*t
        return g;
real_values = g(L[0],A,B).reshape(101,-1)
```

We then plot the graph denoting the various levels of noise.

1d ⟨*noise* 1d⟩≡

```
# 3. Plotting with varying levels of noise
graph.figure(1)
graph.xlabel('$t$')
graph.ylabel('$F_\sigma$')
for i in range(1,9,2):
    graph.plot(L[0],L[i],'+')
    graph.plot(L[0],L[i+1],'.')
graph.plot(L[0],real_values,'k')
graph.legend(('$log(\sigma) = -1$','$log(\sigma) = -0.5$','$log(\sigma) = 0$',
'$log(\sigma) = 0.5$','$log(\sigma) = 1$','$log(\sigma) = 1.5$','$log(\sigma) = 2$',
'$log(\sigma) = 2.5$','$log(\sigma) = 3$','Actual function'),loc=0)
```

Now we write another method to calculate the value of G through matrices. We then output true if the matrix and the function result are the same.

2a ⟨*matrix* 2a⟩≡

```
# 5. Compute G with the matrix method
j = sp.jn(2,L[0])
M = c_[j,L[0]]
AB = [[A],[B]]
g_alt = dot(M,AB)
if all(g_alt)==all(real_values):
    print "Matrix and Algebraic methods both give equal values"
```

Now we calculate Epsilon($\varepsilon\,[i]\,[j]$), the mean squared error over an uniformly distributed interval of A and B.

2b ⟨*error* 2b⟩≡

```
AA=arange(0,2.1,0.1)
BB=arange(-.05,0.0025,0.0025)
epsilon = np.zeros((len(AA),len(BB)))
sum1 = 0

# 6. Computing the mean squared error
for a in range(len(AA)):
        for b in range(len(BB)):
                epsilon[a][b]=sum((L[1] - g(L[0],AA[a],BB[b]))**2)
epsilon/=101
```

Once we have the Epsilon plot, we plot it's contour plot to analyze it with A and B.

2c ⟨*contour* 2c⟩≡

```
# 7. Plotting the Countour
graph.figure(2)
graph.contour(BB,AA,epsilon)
graph.xlabel('$A$')
graph.ylabel('$B$')
graph.title('Contour plot of A and B')
p=[]
errors =[]
```

We now estimate the error using the method of least squares. We use scipy's inbuilt *lstsq* function for this. We then store the error in an error array. We then go ahead and plot and error estimate both on a normal axis and a loglog axis. We see that both these plots are best fit lines.

2d ⟨*estimate* 2d⟩≡

```
# 8., 9. Estimating error
for x in range(1,10):
    least=lstsq(M,L[x].reshape(101,-1))[0]
    p.append(least)
    err = sqrt(dot(transpose(dot(M,least) - L[x].reshape(101,-1)),dot(M,least) - L[x].reshape(101,-1)))
    errors=append(errors,err[0][0])
p=array(p)
sigma = logspace(-1,-3,9)

# 9. Plotting normal error
graph.figure(3)
graph.plot(sigma,errors)
graph.plot(sigma,errors,'ro')
graph.xlabel(r"noise $\sigma$")
graph.ylabel('error')
graph.title('Plot of error in estimate of A and B vs noise')

# 10. Logarithmic error
graph.figure(4)
graph.loglog(sigma,errors)
graph.loglog(sigma,errors,'ro')
graph.xlabel(r"log($\sigma$)")
graph.ylabel('log(error)')
graph.title('Plot of logarithmic error in estimate of A and B vs noise')
```

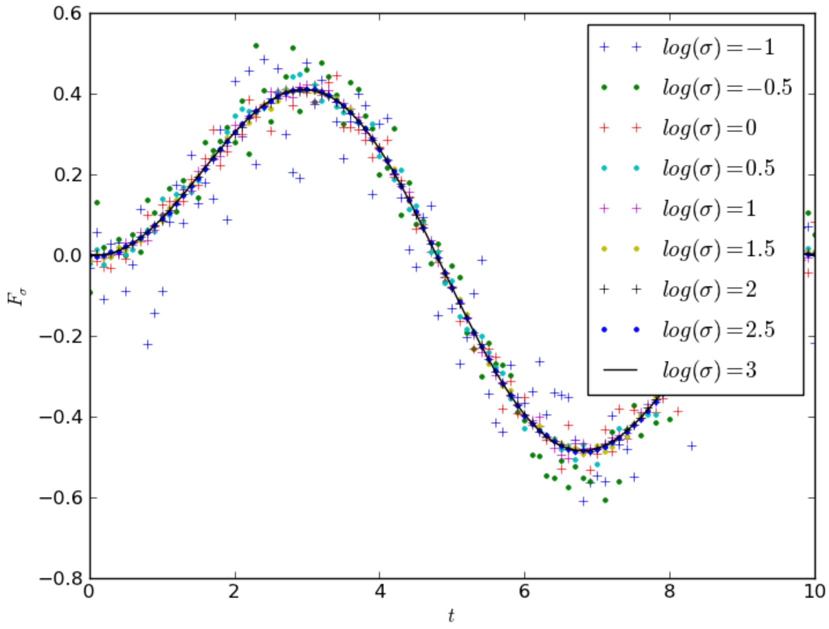


Figure 1: Noise plot

We now go to the exponents that will optimize the expressions *abcde*.

3 ⟨*alphabeta* 3⟩≡

```
# 11. Finding the exponents
noise=log(sigma).reshape((len(log(sigma)),1))
sigma=hstack((ones((len(log(sigma)),1)),noise))
solution=lstsq(sigma,log(errors))
print 'Alpha=%f'%pow(e,solution[0][0])
print 'Beta=%f'%solution[0][1]
graph.show()
```

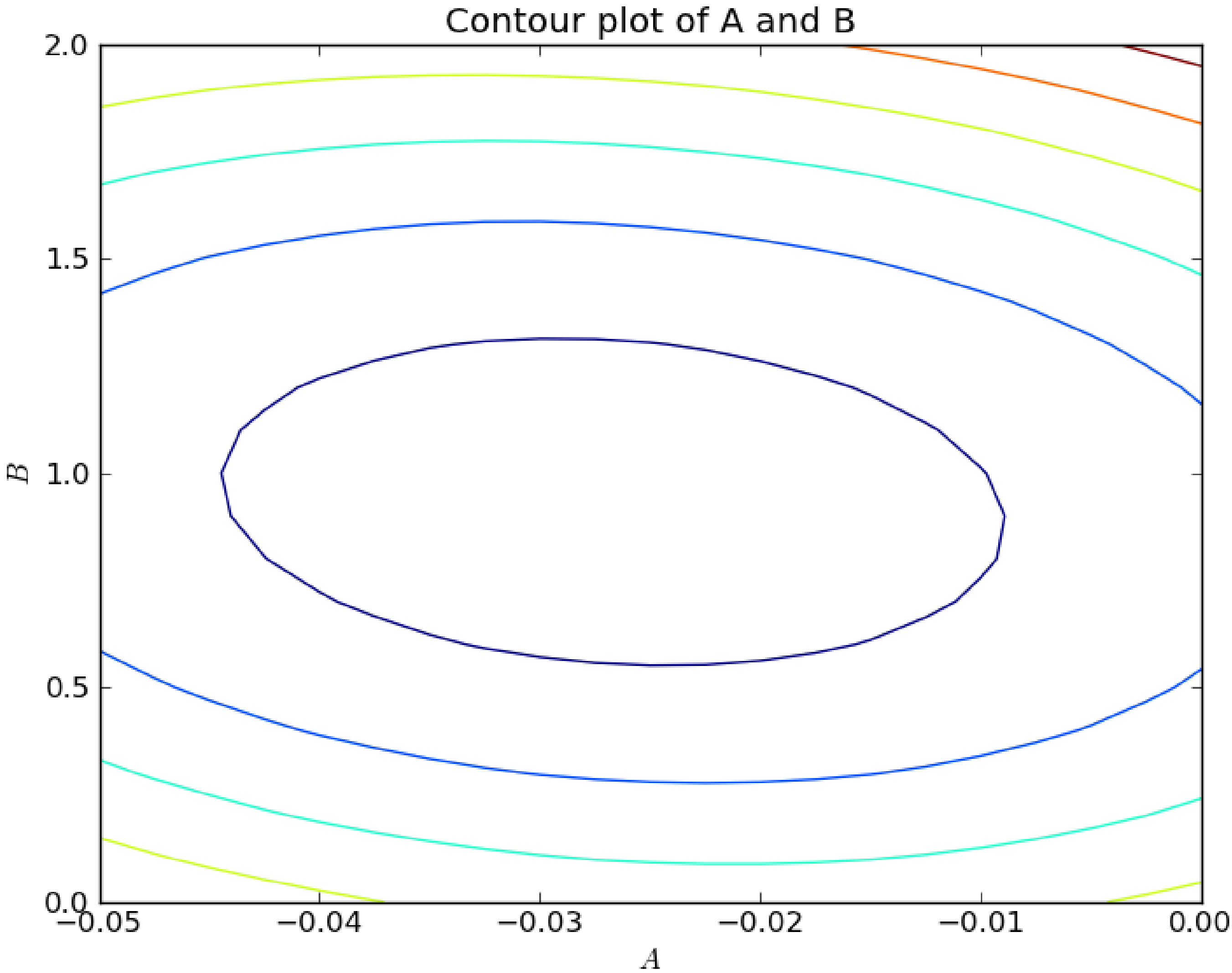
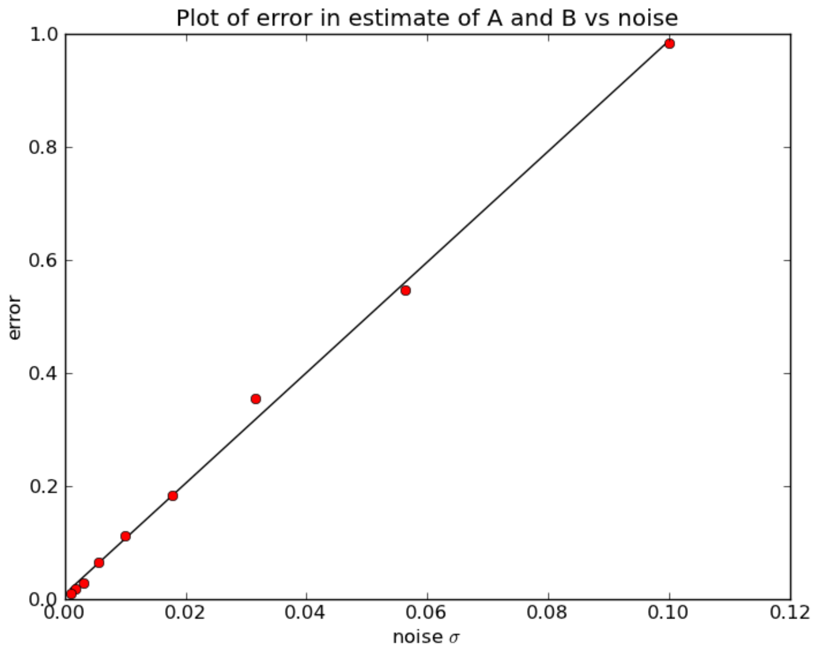
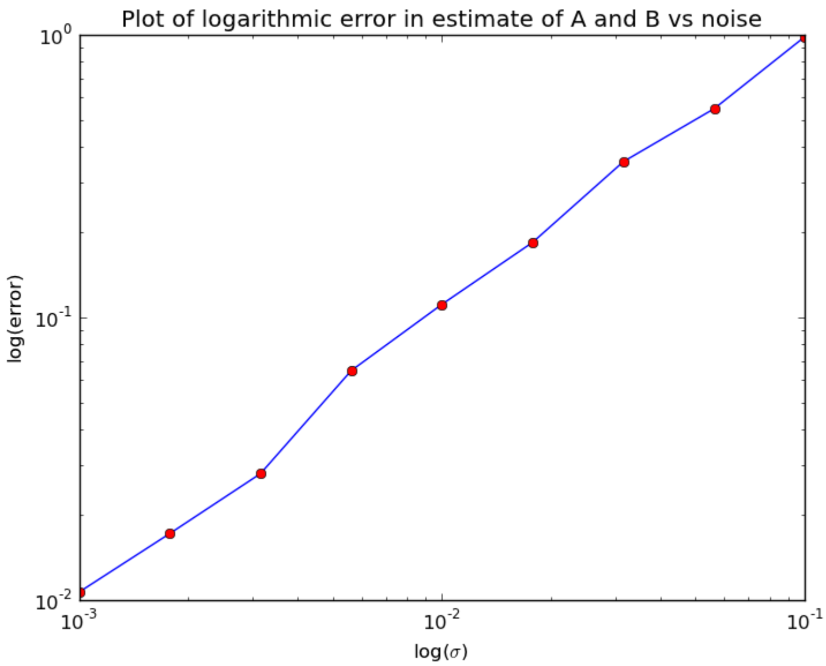


Figure 2: Electron Phase plots



Normal plot



LogLog plots

Figure 3: Normal and LogLog plots