ASST03

February 7, 2022

1 Assignment 03

1.1 Koidala SURYA Prakash

1.2 EE18BTECH11026

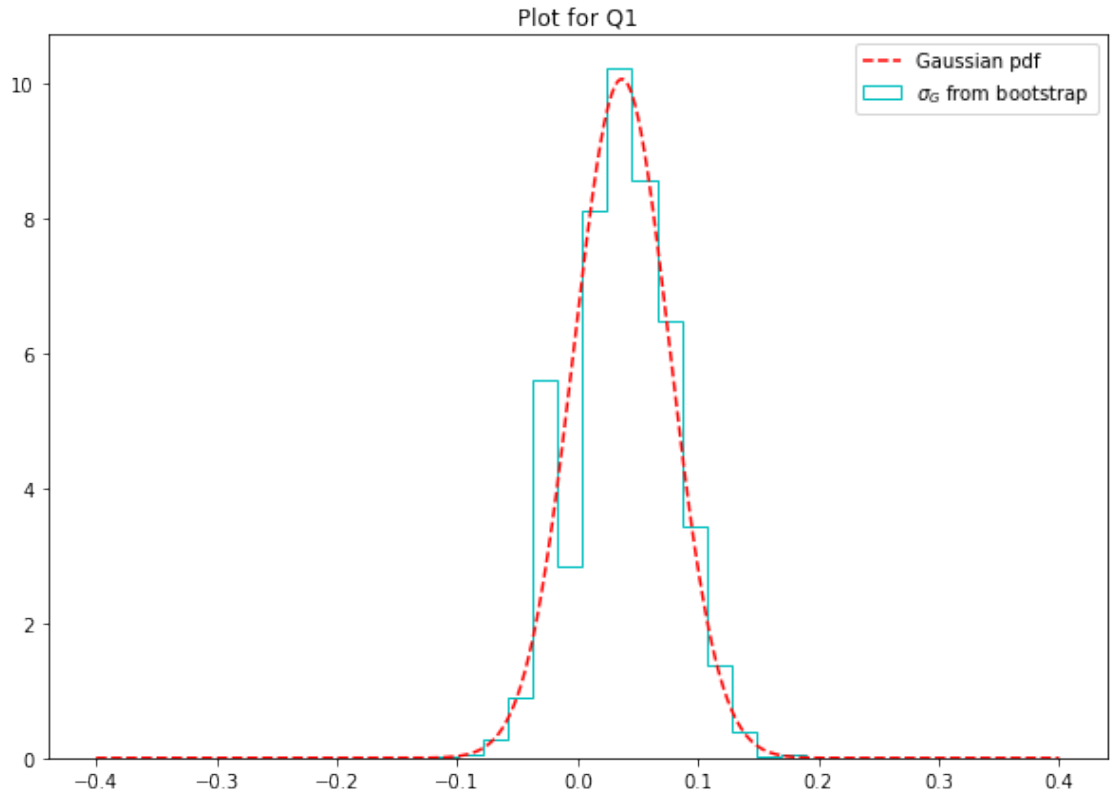
[2]:
```
import scipy as sp
from scipy import optimize
from scipy import stats
import numpy as np
import matplotlib.pyplot as plt
import math
from astroML.resample import bootstrap
from astroML.stats import median_sigmaG
```

2 Q1

[31]:
```
#### Seeding
np.random.seed(1)

m,n = 1000, 10000 ## no. of samples , no. of bootstrapped dataset

dataset = stats.norm(0,1).rvs(m) ## contains 1000 samples drawn from gauss

## list of 10000 medians --- median of each bootstrapped dataset
median,_ = bootstrap(dataset, n, median_sigmaG, kwargs= {'axis' : 1})

print(len(median))

mean_of_med = np.mean(median)
sig_of_med = np.sqrt(np.pi/(2*m) )

print(mean_of_med, sig_of_med)

dist = stats.norm(mean_of_med,sig_of_med)
```

```
x = np.linspace(-0.4, 0.4, 2000)

plt.figure(figsize = (10,7))
plt.plot(x, dist.pdf(x), 'r--', label = 'Gaussian pdf')
plt.hist(median, bins = 15, density = True, histtype='step', color = 'c', label␣
 ↪= r'$\sigma_G$ from bootstrap')
plt.title('Plot for Q1')
plt.legend()
plt.show()
```

```
10000
0.036647754373429826 0.03963327297606011
```

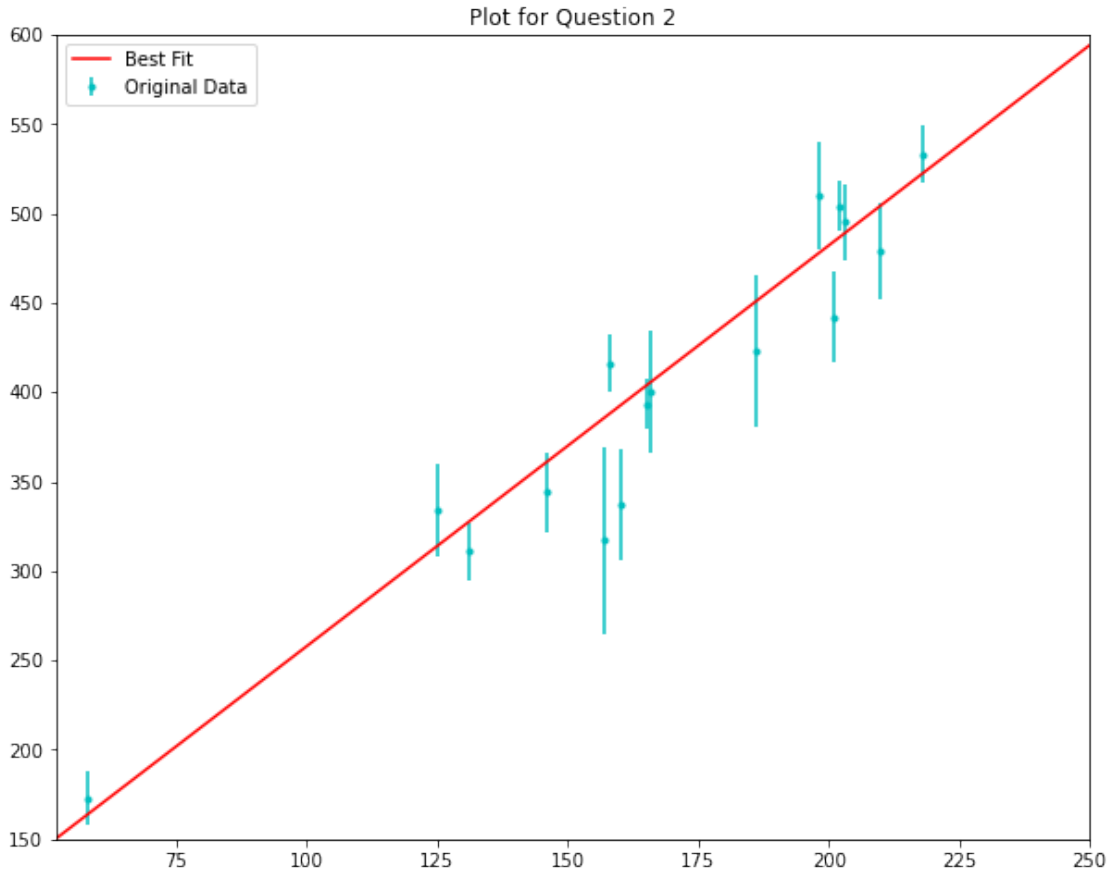


3 Question 02

[6]:
```
## loading data
data = np.genfromtxt('DATA_Q2.dat', skip_header = 0, skip_footer = 0, names =␣
 ↪True, dtype = None)
```

```
# func
def func(x,m,b):
    return m*x+b

X,Y,sigma = [],[],[]

for _,x,y,s in data:
    X.append(x)
    Y.append(y)
    sigma.append(s)

#fit curve
popt, pcov = optimize.curve_fit(func, X, Y, sigma=sigma, absolute_sigma = True)
perr = np.sqrt(np.diag(pcov))

print("Opt val of  m = ", popt[0])
print("Opt val of  b = ", popt[1])
print("Uncertainity in  m = ", perr[0])
print("Uncertanity in b = ", perr[1])


#plot original data
plt.figure(figsize=(10,8))
plt.errorbar(X, Y, yerr = sigma, fmt=".", color = 'c',label = 'Original Data')

#plot best fit
plt.plot(np.linspace(50, 250, 1000), func(np.linspace(50, 250, 1000), *popt),␣
 ↪color = 'r', label='Best Fit')
plt.xlim([52, 250])
plt.ylim([150, 600])
plt.legend()
plt.title("Plot for Question 2")
plt.show()
```

```
Opt val of  m =  2.2399208553933314
Opt val of  b =  34.047723577096654
Uncertainity in  m =  0.10778046998831296
Uncertanity in b =  18.246165249246637
```



4 Question 03

[7]:
```
def PVal(dof , chi2dof):
    return 1-(stats.chi2(dof).cdf(chi2dof*dof))

n = 50 ## no. of samples
m = 1 ## no. of free param
dof = n-m

print('p-value for plot with correct correct err {}'.format(PVal(dof, 0.96)))
print('p-value for plot with correct overestimated err {}'.format(PVal(dof, 0.
 ↪24)))
print('p-value for plot with correct underestimated err {}'.format(PVal(dof, 3.
 ↪84)))
print('p-value for plot with correct incorrect model :  {}'.format(PVal(dof, 2.
 ↪85)))
```

```
p-value for plot with correct correct err 0.5529264339960218
p-value for plot with correct overestimated err 0.9999999917009567
p-value for plot with correct underestimated err 0.0
p-value for plot with correct incorrect model :  1.2107292945984227e-10
```

5 THE END