Problem Set 7 Statistical Methods

Jake Muff

4/11/20

1 Question 1

1. A poisson distributed variable has pmf

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Where λ is the expectation value. In our case $\lambda = v$ so we have

$$f(n, v) = \frac{v^n e^{-v}}{n!}$$

For m observations we have (log = natural logarithm here)

$$\log L(v) = \sum_{i=1}^{m} \log f(n, v)$$

$$= \sum_{i=1}^{m} \log(v)n_i - mv - \log(n_1!)$$

For 1 observation we will have

$$\log L(v) = \log(v)n - v - \log(n!)$$

We find the maximum by taking the dervative w.r.t v

$$\frac{\partial}{\partial v} \log L(v) = \frac{n}{v} - 1 = 0$$

$$\hat{v} = n$$

2. To find if $\hat{v}$ is unbiased we need to show that the expectation value of $\hat{v}$ is equal to $\hat{v}$

$$E[\hat{v}] = E[n] = n$$

The variance is

$$E[\hat{v}^2] - E[\hat{v}]^2 = n^2 - n^2 = 0$$

3. The RCF bound is (V[] is variance)

$$V[\hat{v}] \geq (1 + \frac{\partial b}{\partial v})^2 / E[-\frac{\partial^2 logL(v)}{\partial v^2}]$$

Where b is

$$b = E[\hat{v}] - v$$

$$b = n - v \rightarrow \frac{\partial b}{\partial v} = -1$$

So

$$(1 + \frac{\partial b}{\partial v})^2 = (1 + (-1))^2 = 0$$

Top partial of the fraction is 0 so RCF is 0.

$$RCF\ V[\hat{v}] = 0$$

There is equality so $\hat{v}$ is efficient.

4. In the case of m observations we have $n_1, n_2 \ldots n_m$ independent and identicially distributed poisson variables, so we have

$$\log L(v) = \sum_{i=1}^{m} (n_i \log(v) - v - log(n_i!))$$

$$= \log v \sum_{i=1}^{m} n_i - mv - \sum_{i=1}^{m} \log(n_i!)$$

Maximising by taking the derivative w.r.t v and setting equal to 0

$$\frac{\partial}{\partial v} \log L(v) = \frac{1}{v} \sum_{i=1}^{m} n_i - m = 0$$

So $\hat{v}$ is

$$\hat{v}_m = \frac{1}{m} \sum_{i=1}^{m} n_i$$

Which is essentially the mean so

$$\hat{v}_m = mean(n) = \overline{n}$$

Question2

November 4, 2020

1 Question 2

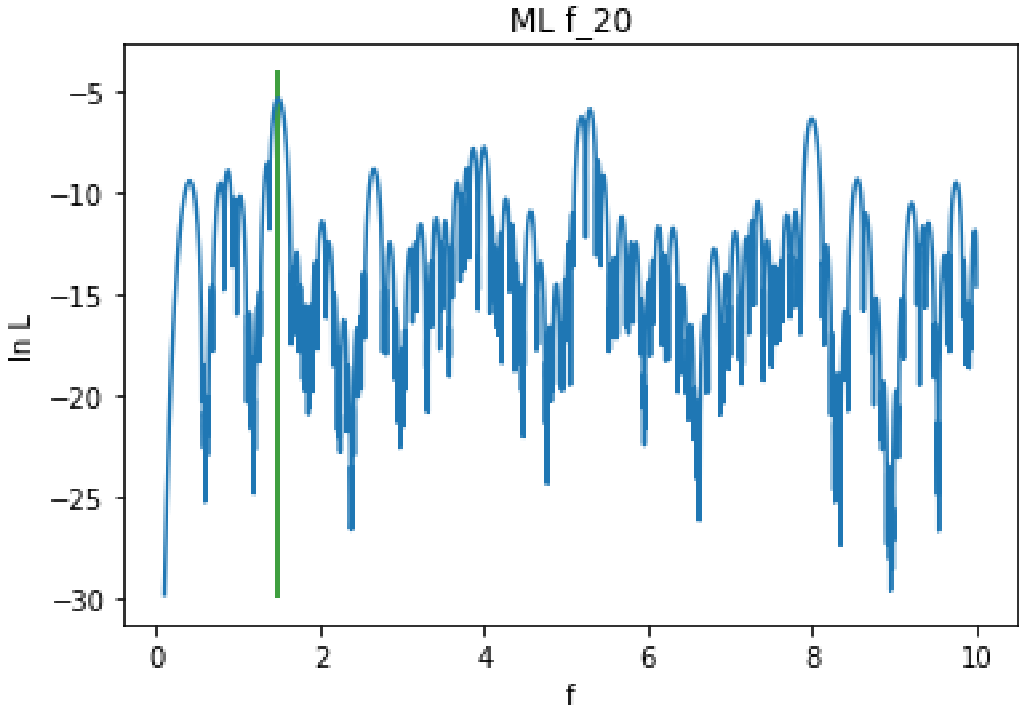
[76]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import find_peaks

def F(f,t):
    return np.abs(np.sin(2*np.pi *f *t))

def L(f,t):
    return np.log(np.prod(F(f,t)))

ml_sample_1 = np.loadtxt("ml_sample_1.txt")
ml_sample_2 = np.loadtxt("ml_sample_2.txt")


x = np.linspace(0.1,10,10000)
l1 = [L(f,ml_sample_1) for f in x]

plt.figure()
plt.plot(x,l1)
plt.xlabel('f')
plt.ylabel('ln L')
plt.title('ML f_20')

plt.vlines(x[np.where(l1==np.max(l1))], -30, -4, color='green',␣
 ↪linestyle='solid')

print('ln L_max_20 = %f' %np.max(l1))
print('f_20 = %f' %x[np.where(l1==np.max(l1))])

sigma_f20 = np.max(l1) - 0.5
print('ln L_max_20 - 0.5 = %f' %sigma_f20)

y_f20 = np.linspace(1.4,1.6,10000)
y1_f20 = [L(f,ml_sample_1) for f in y_f20]
```

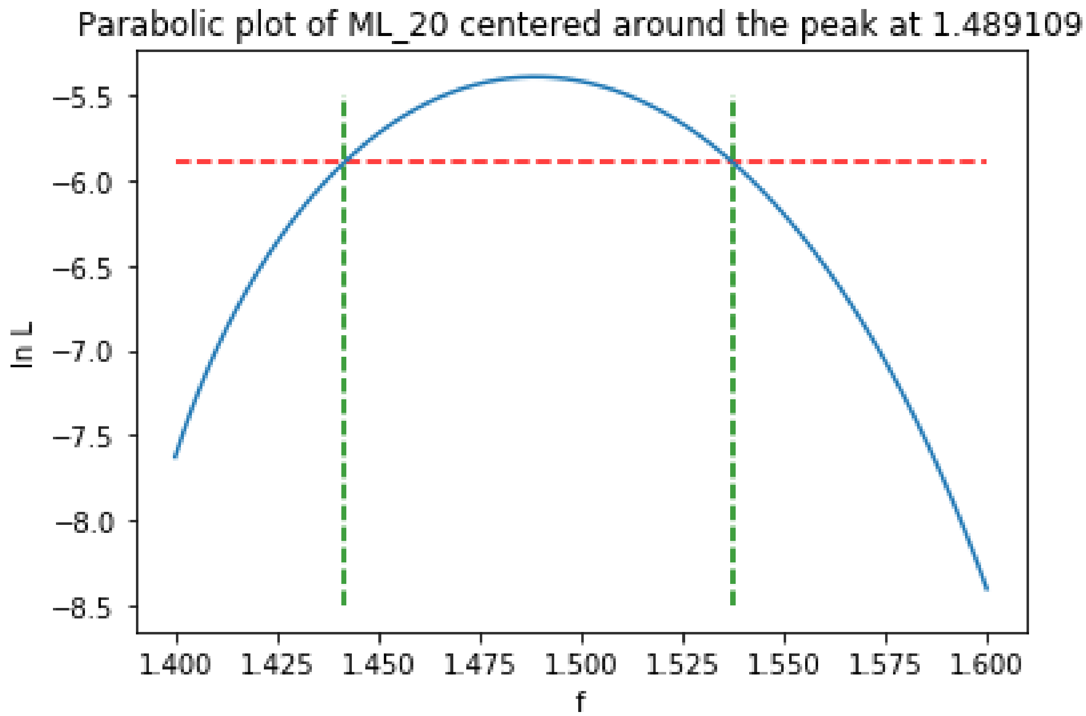
```
plt.figure()
plt.plot(y_f20,y1_f20)
plt.xlabel('f')
plt.ylabel('ln L')
plt.title('Parabolic plot of ML_20 centered around the peak at %f' %x[np.
 ↪where(l1==np.max(l1))])

plt.hlines(sigma_f20, 1.4,1.6, color='red', linestyle='dashed')
#sigmaplus = y[np.where(y1>)]
#print(sigmaplus)
idx_f20 = np.argwhere(np.diff(np.sign(y1_f20 - sigma_f20))).flatten()
#print(idx_f20)
#print(y[2092])
#print(y[6860])

plt.vlines(y_f20[2092], -8.5, -5.5, color='green', linestyle='dashed')
plt.vlines(y_f20[6860], -8.5, -5.5, color='green', linestyle='dashed')

deltaf_plus_f20 = x[np.where(l1==np.max(l1))] - y_f20[2092]
deltaf_minus_f20 = y_f20[6860]- x[np.where(l1==np.max(l1))]

var_f20 = (deltaf_plus_f20 + deltaf_minus_f20)/2.0
print('Uncertainty on f_20 = %f ' %var_f20)

###############################################
l2 = [L(f,ml_sample_2) for f in x]

plt.figure()
plt.plot(x,l2)
plt.xlabel('f')
plt.ylabel('ln L')
plt.title('ML f_100')

print('ln L_max_100 = %f' %np.max(l2))
print('f_100 = %f' %x[np.where(l2==np.max(l2))])

sigma_f100 = np.max(l2) - 0.5
print('ln L_max_100 - 0.5 = %f' %sigma_f100)

y_f100 = np.linspace(1.465,1.525,10000)
y1_f100 = [L(f,ml_sample_2) for f in y_f100]

plt.figure()
plt.plot(y_f100,y1_f100)
plt.xlabel('f')
plt.ylabel('ln L')
```

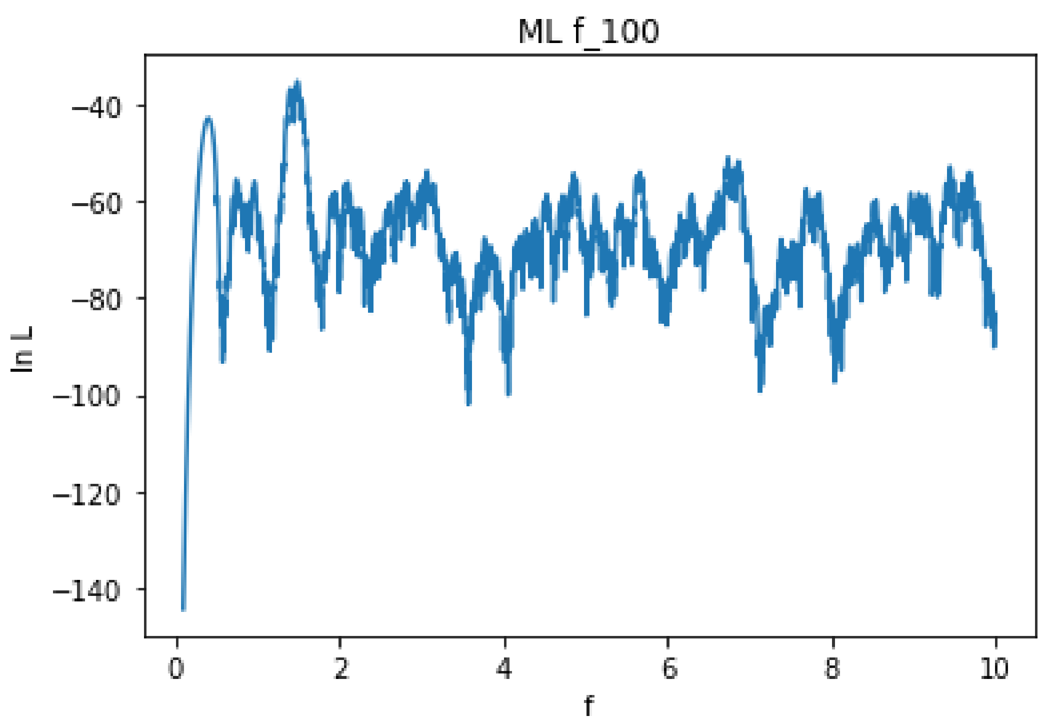
```python
plt.title('Parabolic plot of ML_100 centered around the peak at %f' %x[np.
 ↪where(l2==np.max(l2))])

plt.hlines(sigma_f100, 1.465,1.525, color='red', linestyle='dashed')
#sigmaplus = y[np.where(y1>)]
#print(sigmaplus)
idx_f100 = np.argwhere(np.diff(np.sign(y1_f100 - sigma_f100))).flatten()
#print(idx_f100)
#[2170 5894]
#print(y_f100[2170])
#print(y_f100[5894])

plt.vlines(y_f100[2170], -48, -35, color='green', linestyle='dashed')
plt.vlines(y_f100[5894], -48, -35, color='green', linestyle='dashed')

deltaf_plus_f100 = x[np.where(l2==np.max(l2))] - y_f100[2170]
deltaf_minus_f100 = y_f100[5894]- x[np.where(l1==np.max(l1))]

var_f100 = (deltaf_plus_f100 + deltaf_minus_f100)/2.0
print('Uncertainty on f_100 = %f ' %var_f100)

l3 = np.log10(x)

plt.figure()
plt.plot(l3,l1)
plt.xlabel('Log10 f')
plt.ylabel('ln L')
plt.title('ln L vs log10 f for ML20')


plt.figure()
plt.plot(l3,l2)
plt.xlabel('Log10 f')
plt.ylabel('ln L')
plt.title('ln L vs log10 f for ML100')

print('Ratio of uncertainty between f_20 and f_100 is %f' %(var_f20/var_f100))
print('The ratio is ~4x of which one sample is 20 and the other 100 which is␣
 ↪about ~5x, so it is consistent')
```

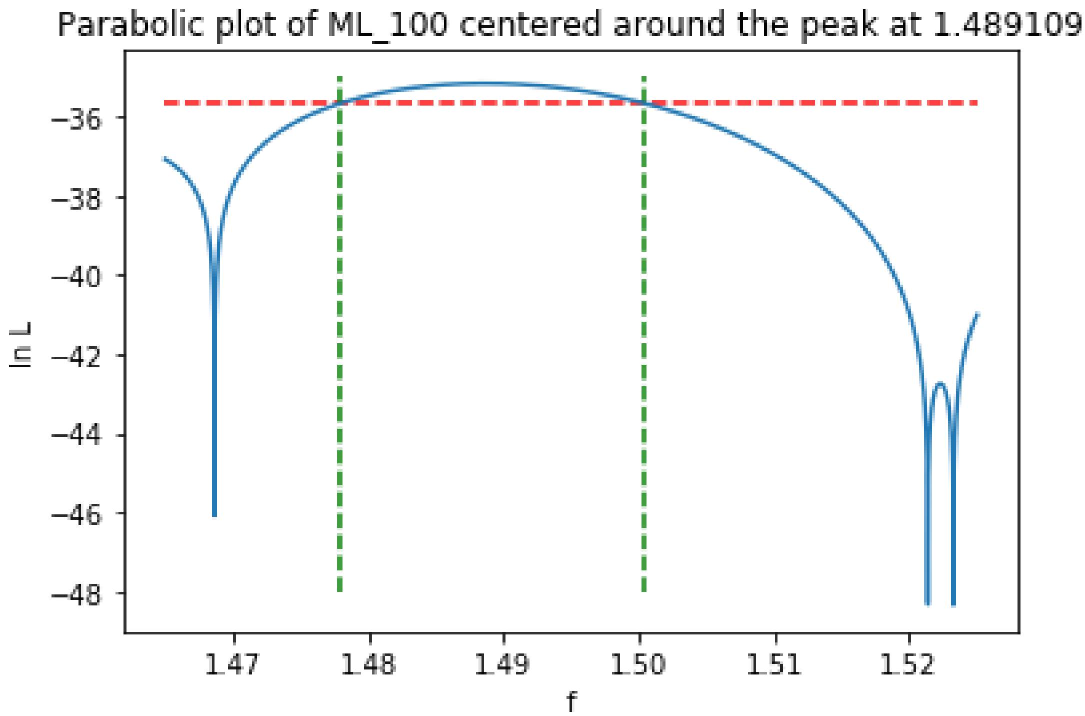
```
ln L_max_20 = -5.388091
f_20 = 1.489109
ln L_max_20 - 0.5 = -5.888091
Uncertainty on f_20 = 0.047685
ln L_max_100 = -35.166601
f_100 = 1.489109
ln L_max_100 - 0.5 = -35.666601
```

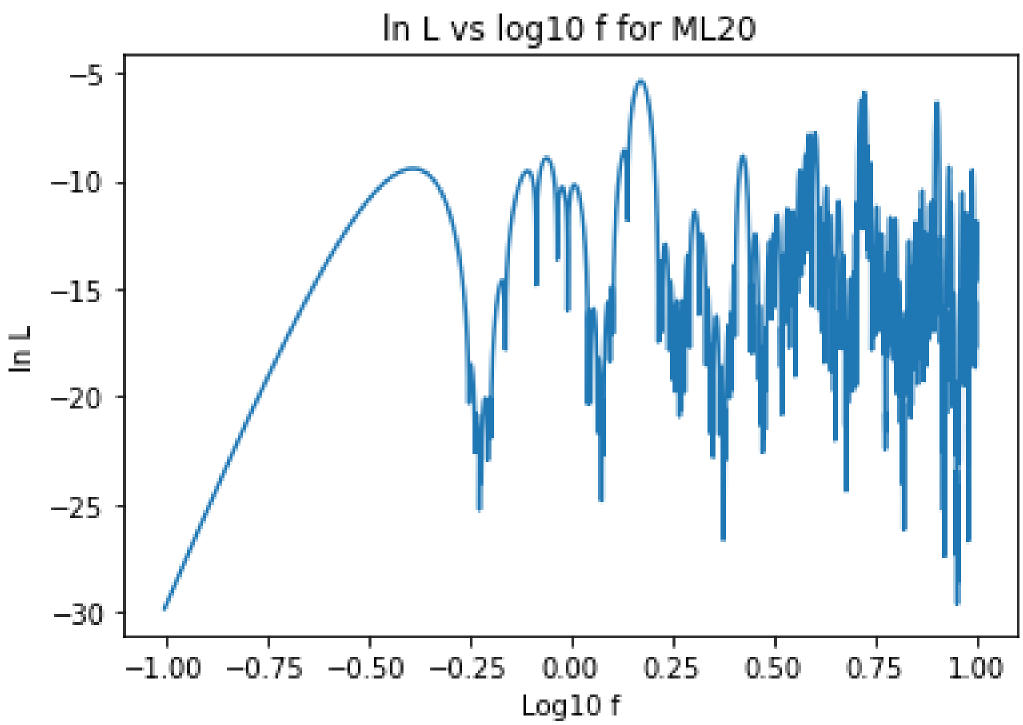
```
Uncertainty on f_100 = 0.011173
Ratio of uncertainty between f_20 and f_100 is 4.267812
The ratio is ~4x of which one sample is 20 and the other 100 which is about ~5x,
so it is consistent
```

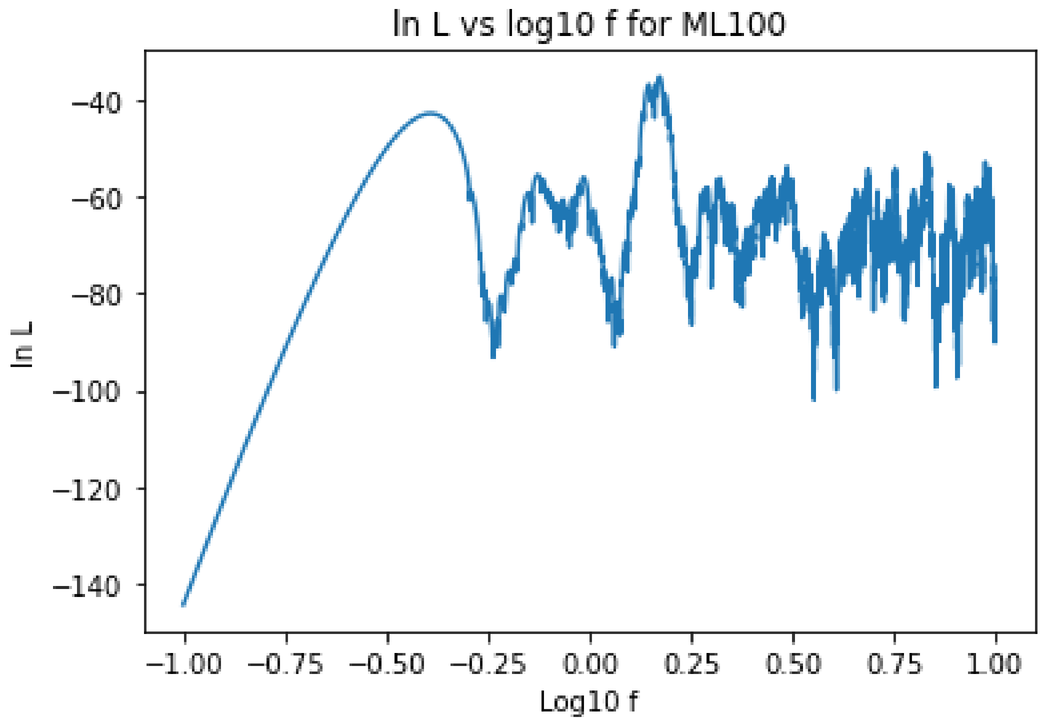



Parabolic plot of ML_20 centered around the peak at 1.489109
−5.5
−6.0
−6.5
−7.0
−7.5
−8.0
−8.5
ln L
1.400
1.425
1.450
1.475
1.500
1.525
1.550
1.575
1.600
f


ML f_100
−40
−60
−80
−100
−120
−140
ln L
0
2
4
6
8
10
f


Parabolic plot of ML_100 centered around the peak at 1.489109
ln L
f
−36
−38
−40
−42
−44
−46
−48
1.47
1.48
1.49
1.50
1.51
1.52


ln L vs log10 f for ML20
ln L
Log10 f
−5
−10
−15
−20
−25
−30
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00



In conclusion: For the data set with 20 occurances

$$lnL_{max,20} = -5.388091$$

$$\hat{f}_{20} = 1.489109$$

$$\sigma_{\hat{f}20} = 0.047685$$

For the data set with 100 occurances

$$lnL_{max,100} = -35.166601$$

$$\hat{f}_{100} = 1.489109$$

$$\sigma_{\hat{f}100} = 0.011173$$

The ratio between $\hat{f}_2 0$ and $\hat{f}_1 00$ is 4.267812 which is pretty close to the expected number 5 as there are 5 times more values in the second sample when compared to the first sample, so the ratio is consistent with the different sizes.

2 Method and comments

To determine the ML estimate for the data samples I plotted 10000 linearly spaced numbers between $0.1 - 10Hz$ against the equation below

$$ln\prod_{i=1}^{n} f(x_i, \theta)$$

Where f was the pdf in which we assumed the data set would follow. I then found the peak value on this plot to which the y-value would correspond to lnL_{max} and the x-value would correspond to the ML estimator. This was repeated for both $\hat{f}_{20}$ and $\hat{f}_{100}$ for both (i) and (iii).

For the uncertainty I used the graphical method and 'shrunk' the original plots down so that it only covered the max peak. I then found the y value using

$$lnL_{max} - \frac{1}{2}$$

And found the corresponding value on the x axis to which there were two. One would be the ML estimator + the variance and the other would be the ML estimator - the variance, so this could be rearranged easily to find the uncertainty by taking the mean (or dividing by 2)

The ratio between the uncertainty of $\hat{f}_{20}$ and $\hat{f}_{100}$ was done by dividing one by the other.

As we can see the ML estimators are equal but with uncertainty ~5x greater in the data with 100 occurances than with 20 as expected.

[ ]:

```

```