DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Homework Assignment No. 3:

HW No. 3: Maximum Likelihood VS. Bayesian Estimation

submitted to

Professor Joseph Picone
ECE 8527: Introduction to Pattern Recognition and Machine Learning
Temple University
College of Engineering
1947 North 12th Street
Philadelphia, Pennsylvania 19122

February 3, 2024

prepared by:

Leo Berman
Email: leo.berman@temple.edu

A. GENERATE 11 INDEPENDENT SETS OF DATA

First, 11 one dimensional vectors with 10^6 points, Variance = 1, and the followings means were generated:

0.90, 0.92, 0.94, 0.96, 0.98, 1.00, 1.02, 1.04, 1.06, 1.08, 1.10

These were generated using the following python snippet:

```
# generate the points for each set
  vectors = []
  for i in range(11):
      vectors.append(numpy.random.normal(loc = .9+(i*.02),scale = 1,size = 10**6))
```

The Maximum Likelihood Estimation was calculated for all the independent vecotrs by calculating the mean with respect to the number of points factored in with the following snippet:

```
# create a dictionary to hold each set's means
  mean_plot_points = {}

  # iterate through all sets
  for i,x in enumerate(vectors):

      # keep track of the mean of that set
      mysum = 0

      # Add the set to the dictionary
      mean_plot_points[round(.90+(.02*i),2)] = []
      for j,y in enumerate(x):
          # keep track of the mean and append each mean point
          mysum+=y
          mean_plot_points[round(.90+(.02*i),2)].append(mysum/(j+1))
```

All the plots using MLE use the following function to create the plots:

```
def cascading_arrow(data,ylower,yhigher):

  # generate x_axis
  x_axis = numpy.linspace(1,10**6,10**6)

  # create subplots
  fig,ax = plt.subplots()

  # plot the data
  ax.plot(x_axis,data)

  # set the x-axis to a logarithmic scale
  plt.xscale("log")

  # set bounds
  plt.ylim(ylower,yhigher)
  plt.xlim(0,10**7)

  #iterate through 1,5,10,50,100,500...5*10^5,10**6
  index = 1
  for i in range(13):

    # scientific notation to be efficient with space
    if i > 10:
      pltstr = '('+"{:.0e}".format(round(x_axis[index-1],0))+','+str(round(data[index-1],2
                                          ))+')'
    else:
      pltstr = '('+str(round(x_axis[index-1],0))+','+str(round(data[index-1],2))+')'

    # annotate stacked cascading arrows with text centered
    ha = 'center'
    if i == 0:
      xloc = x_axis[index-1]+2
    else:
      xloc = x_axis[index-1]

    if (i%2) == 0:
      ax.annotate(pltstr,xy=(x_axis[index-1],data[index-1]),xytext=(xloc,ylower+((yhigher-
                                          ylower)/14)+(i*((yhigher-ylower)/42))),
                                          arrowprops=dict(facecolor='black', shrink=.05)
                                          , horizontalalignment=ha)
      index*=5
    else:
      ax.annotate(pltstr,xy=(x_axis[index-1],data[index-1]),xytext=(xloc,yhigher-((yhigher
                                          -ylower)/14)-(i*((yhigher-ylower)/42))),
                                          arrowprops=dict(facecolor='black', shrink=.2),
                                           horizontalalignment=ha)
      index*=2
```

B. MLE FOR MEAN=1.00 AND MEANS=(0.90, 0.92, . . . , 1.00)

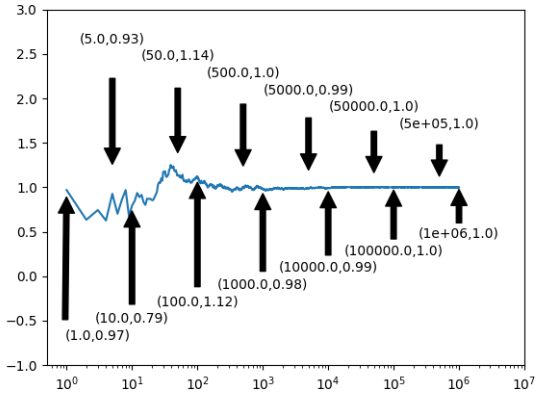

Figure 1: (Mean = 1.00)

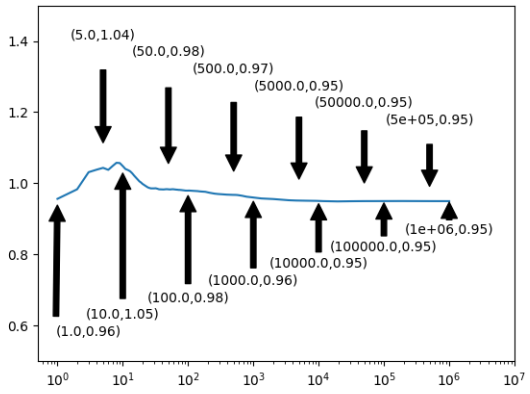

Figure 2: (Means = 0.90, 0.92, . . . , 0.98, 1.00)

The first estimate that uses the one dimensional vector centered around one is clearly biased towards one and clearly converges to 1. However, the second estimate isn't biased towards one and clearly converges towards 0.95 which makes sense as:

$$\frac{\sum_{n=0}^{5} 0.90 + (.02 * n)}{6} = .95$$

C. MLE FOR MEANS = (0.90, 0.92, . . . , 1.10)

Plot the maximum likelihood estimation by plotting the means of each individual vector as a function of $\frac{TotalPoints}{TotalVectors}$. In practice, this looks like taking the mean of Figure 3 and plotting it with a correlating x axis.

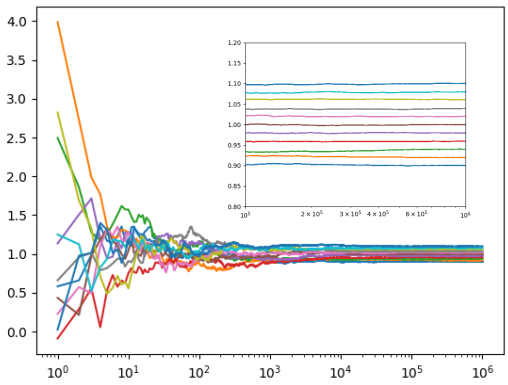

Figure 3: Means = (0.90, 0.92, . . . , 1.10)

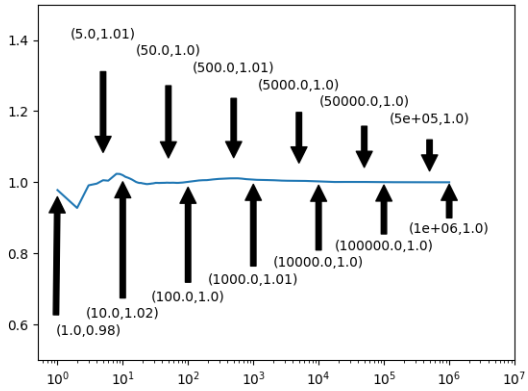

Figure 4: Mean of Means = (0.90, 0.92, . . . , 1.10)

When we compare the plot in Figure 4 to the plot of Figure 3, we can see that both plots converge to a mean estimation of 1.00, but not only does Figure 4 converge faster, it also has less noise. The reason that this happens is because there are more point surrounding the mean of 1.00 for Figure 4. Not only does it have access to all the points that Figure 1 has it also has access to the 10 other sets that have a combined mean of means equal to 1.00 as well.

One way to show this is to compare the points in each algorithm when they have process 110 data points. For Figure 4 this happens at x = 10 and for figure 1 this happens at x = 110. For Figures 1 and 4, their estimations after 110 data points were processed were 1.12 and 1.02 respectively. When the "noise" shown in Figure 1 is taken into account, the difference of these numbers show anecdotal evidence that the speed of convergence is linearly coorelated with the number of points independent of whether they come from a one dimensional vector or a ten dimensional vector as long as the mean = 1.

D. BAYESIAN ESTIMATION WITH INITIAL GUESS = 2

The next step is to plot a Bayesian Estimation of the mean while having an initial guess of a mean of 2. Essentially, this means that the hypothesis is that the mean won't be very far from 2. In reality, the mean is one. As a result, one of the the main differences between Bayesian Estimation and the MLE shown in Figure 1 and Figure 4 is how fast it approaches a precise mean. Despite having access to more information than Figure 1 and equal access to information as Figure 4, it still converges slower than the other two algorithms.

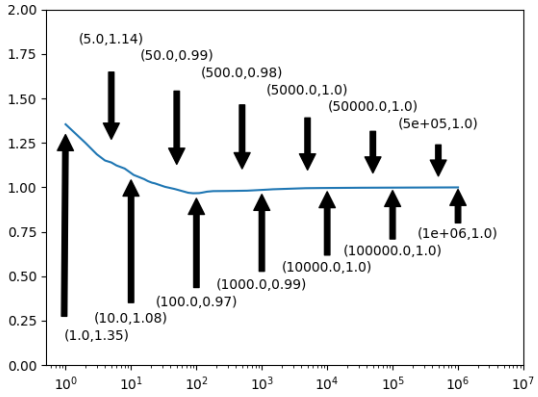

Figure 5: Bayesian Estimation of Mean with Known Variance

E. SUMMARY

In order for the Bayesian Estimation to overcome an initial bad guess, it takes around 10 to 1000 points. At around 1000 points, the initial guess is weighted much less than the prior mean with variance taken into account. Bayesian Estimation is a really good tactic for large data sets or if there is a known mean or relatively close guess to what the mean is.