Assignment 4

Computational Intelligence, SS2017

Team Members		
Last name	First name	Matriculation Number
Alié	Félix	01644819
Sourmpis	Christos	01644560

Computational Intelligence - 04

Maximum Likehood

Félix Alié, Christos Sourmpis

02.06.2017

Contents

Introduction

In this fourth assignment, we are going to use try to understand how to use the maximum likelihood estimation to solve a positioning problem. We have four anchors and one agent that we need to localize. Given noisy measurement of the approximate distance between the agent and each anchor, the goal is to try to estimate with the more possible precision the correct position of the agent.

1 Scenario and model

2 Maximum Likelihood Estimation of Model Parameters

2.1 Question 1

Out of the four anchors, three measures following a Gaussian distribution, and one following an exponential distribution. But we don't know which one. To do so, we just compute the mean of the difference between the true distance and the input data. We find that 3 of these means are around 0, while one (the first one) is around 0.8. We can then deduce that this is the exponential one.

2.2 Question 2

Analytic solution of maximum likelihood for the exponential anchor. For the rest of this question we will set $x_i = (r_i - d(p))$. With i being the index of the measurement and n the number of measurements.

$$\begin{aligned} L &= \prod_{i=1}^{n} \lambda_i e^{-\lambda_i x_i} \\ &= \lambda^n \cdot e_i^{n} {}^{-\lambda_i x i} \\ &= \lambda^n \cdot e^{-\lambda \sum x_i} \\ ln(L) &= nln(\lambda) - \lambda \sum x_i \\ \frac{\partial ln(L)}{\partial \lambda} &= n\frac{1}{\lambda} - \sum x_i \\ \frac{\partial ln(L)}{\partial \lambda} &= 0 \Leftrightarrow \hat{\lambda} = \frac{n}{\sum x_i} \end{aligned}$$

To see if this is indeed the optimal solution we compute the second derivative:

$$\frac{\partial^2 ln(L)}{\partial \lambda} = \frac{-n}{\lambda^2}$$

Which is negative so we indeed found the optimal solution.

2.3 Question 3

We now want to estimate the parameters of the models in the three scenarii, we obtain the following results:
First scenario:

$$\sigma_1^2 = 0.091 \quad \sigma_2^2 = 0.092 \quad \sigma_3^2 = 0.097 \quad \sigma_4^2 = 0.085$$

Second scenario:

$$\sigma_2^2 = 0.093 \quad \sigma_3^2 = 0.090 \quad \sigma_3^2 = 0.086 \quad \lambda_1 = 1.112$$

Third scenario:

$$\lambda_1 = 1.116 \quad \lambda_2 = 0.090 \quad \lambda_3 = 1.130 \quad \lambda_4 = 1.100$$

3 Least-Squares Estimation of the Position

3.1 Question 1

We now want to show that the maximum likehood estimation is equivalent to the least-square error estimation. So we process analytically:

$$\begin{aligned}
\hat{p}_{ML} &= \underset{p}{argmax}\ p(r|p)) && (1)\\
&= \underset{p}{argmax} \prod_{i=1}^{N_A} p(r|p)) && (2)\\
&= \underset{p}{argmax}\ log \prod_{i=1}^{N_A} (\frac{1}{\sqrt{2n\sigma_i^2}}) \cdot exp(\frac{-\lambda_i (ri_{di}(p))^2}{2\sigma_i^2}) && (3)
\end{aligned}$$

with the assumption that $\lambda_i = \lambda$ and $\sigma_i = \sigma$

$$(4)$$

$$\begin{aligned}
&= \underset{p}{argmax}[log(\frac{1}{\sqrt{2n\sigma_i^2}})^{N_A} - \frac{\lambda}{2\sigma_i^2} \sum_{i=1}^{N_A} (r_i - d_i(p))^2] && (5)\\
&= [\frac{\lambda}{\sigma_i^2} \sum_{i=1}^{N_A} (r_i - d_i(p))^2] && (6)\\
&= argmin||\underline{r} - \underline{d}(p)||^2 && (7)\\
&= \hat{p}_{LS} && (8)\\
& && (9)
\end{aligned}$$

3.2 Question 2

To compute the least square estimation of the position, we implement the Gauss-Newton algorithm with the following parameters:

- **tolerance:** 0.01
- **max iterations:** 4

Note: These parameters were chosen after running some small experiments to test the different values. These values give consistent result in a really short time and we thought there were good ones.
The implementation of the of the function LeastSquaresGN is in the code file.

3.3 Question 3

In this question we try to estimate the wanted position using the distances from the experiments and the least squares Gauss-Newton algorithm as it is explained in the exercise. After, we do that for all our measurements and for all the scenarios.

3.3.1

Table 1: The means and the variances

	Mean [x, y]	Variances [x, y]
Scenario 1	[1.98, -3.99]	[0.044, 0.061]
Scenario 2	[1.79, -4.54]	[0.073, 0.325]
Scenario 3	[2.26, -4.36]	[0.887, 0.896]

3.3.2

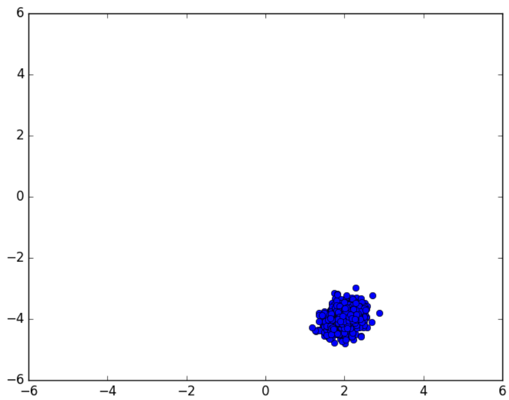

Figure 1: Final Points after running the algorithm for scenario 1

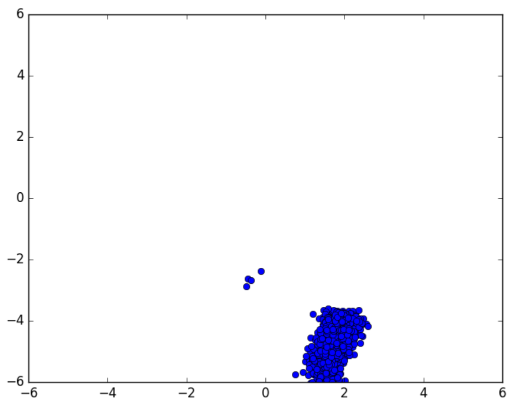

Figure 2: Final Points after running the algorithm for scenario 2

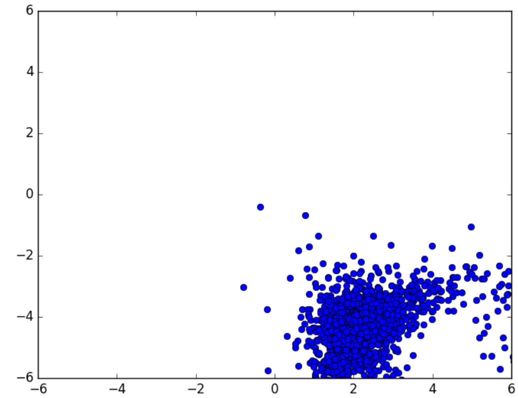

Figure 3: Final Points after running the algorithm for scenario 3

From the above plots we can clearly see that the first scenario follows the gaussian distribution, for the second scenario one could say that it also follows the gaussian but we can see that some points are off and finally the third scenario one can see only from the points that it doesn't follow a gaussian distribution since the points are sparsely distributed.
From the contour plots someone can not derive much since there are gauss contour plots but can see for sure that the density in the middle from the first scenario is the biggest then the second follows and the lowest density in the middle has the scenario 3. Which is what we expected since the gaussian model creates higher density around the mean value.

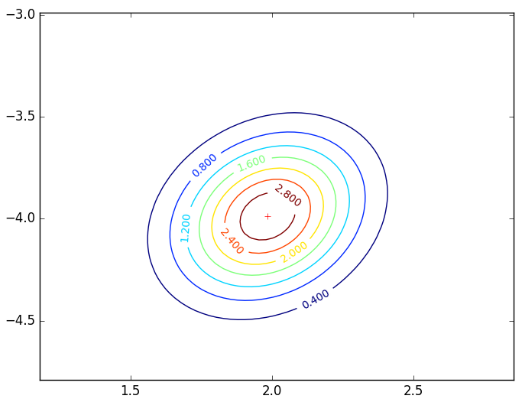

Figure 4: Contour Plot from Scenario 1

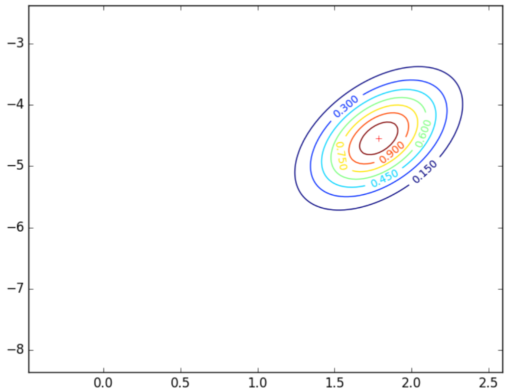

Figure 5: Contour Plot from Scenario 2

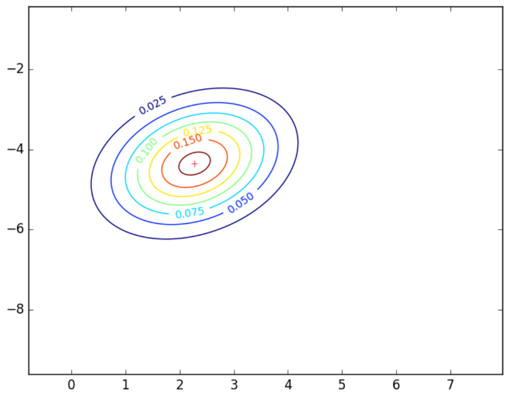

Figure 6: Contour Plot from Scenario 3

3.3.3

The cumulative density function of the error distribution is a very good measure to estimate the effectiveness of a scenario, since someone can see with a very simple way the probability of having big errors and also the mean error. This is observed also in our experiment in the following plots. Especially, we see that for the first scenario the errors are not bigger than 0.6 in contrast to the other case, where in the second scenario there are errors that go over 2 especially for the y axis and in the third scenario, where both axis go over 2.

The y axis is more affected than the x, because of the position of the first anchor with the exponential distribution. Especially, the x axis wasn't that affected because the 3rd and 4th anchor are between the true position and have the same height. But because they have the same height they cannot predict it and they need a third one and in our case the third is either the first anchor with exponential distribution or the second anchor, which is the farthest from the true position.

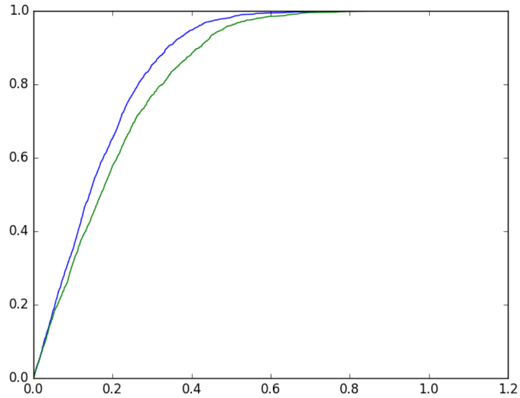

Figure 7: Cumulative Density Function of error distribution from scenario 1

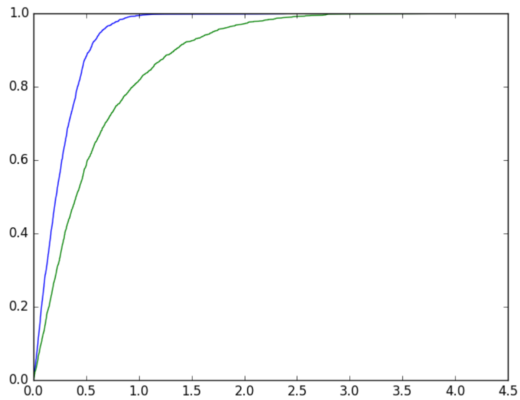

Figure 8: Cumulative Density Function of error distribution from scenario 2

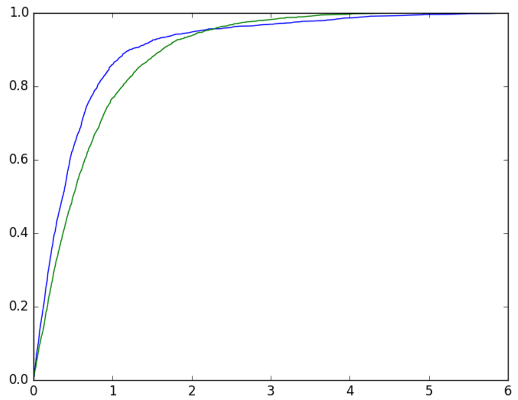

Figure 9: Cumulative Density Function of error distribution from scenario 3

Note: The blue line refers to x axis and the green line to y axis

3.4

In this case we see that the performance in this case is almost the same, even if we have removed one anchor. Especially, we see that the mean and the variance have almost the same divergence from the true value.

Table 2: The means and the variances

	Mean [x, y]	Variances [x, y]
Scenario 2	[1.79, -4.54]	[0.073, 0.325]
Scenario 2 w/t the 1st anchor	[2.13, -3.46]	[0.101, 0.299]

As we expected there is a increasing in the Gaussianity form of the results meaning that despite the fact that we have almost the same amount of big errors in y axis (>0.5) the biggest errors do not exceed a 2 value in contrast to the scenario 2.

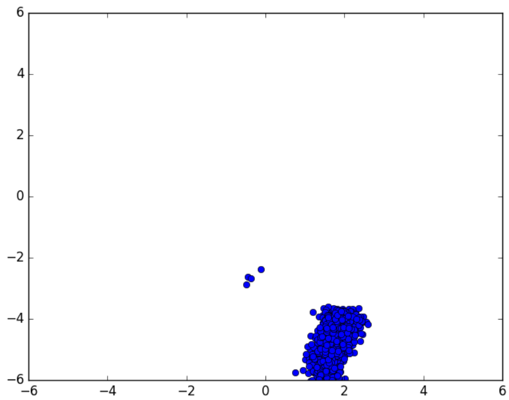

Figure 10: Final Points after running the algorithm for scenario 2

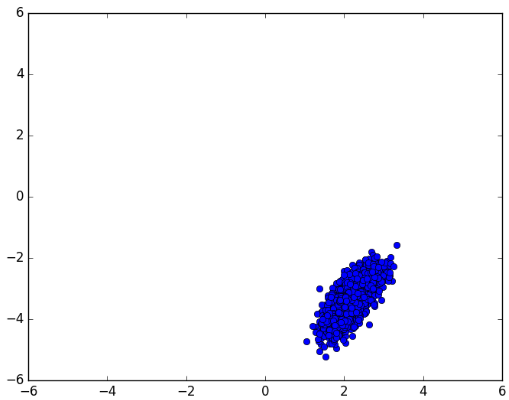

Figure 11: Final Points after running the algorithm for scenario 2 without the first anchor

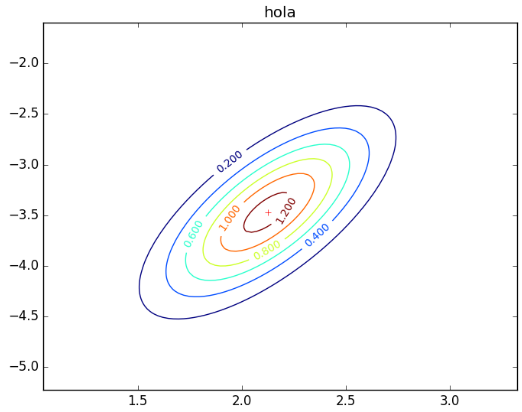

Figure 12: Contour Plot from Scenario 2

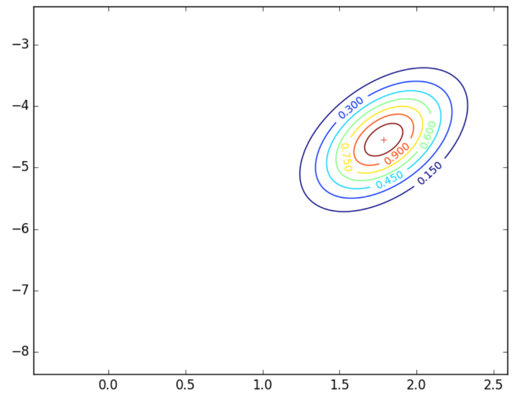

Figure 13: Contour Plot from Scenario 2 without the first anchor

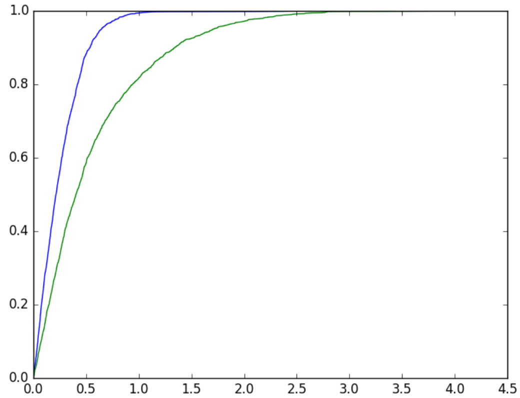

Figure 14: Cumulative Density Function of error distribution from scenario 2

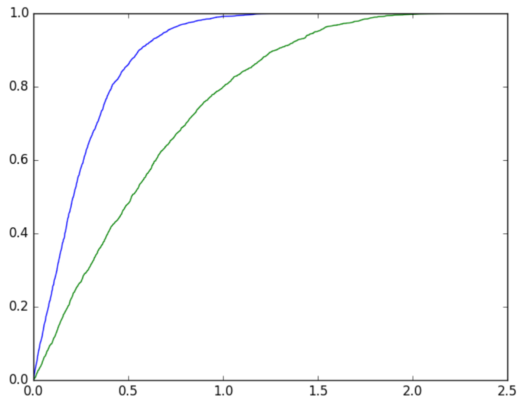

Figure 15: Cumulative Density Function of error distribution from scenario 2 without the first anchor

4 Numerical Maximum-Likelihood Estimation of the Position

4.1 Question 1

In this part we're gonna try to estimate the maximum likelihood by computing it directly from the measurements. We first build a grid of 200 * 200 points, we only consider the first measurement and we compute the joint likelihood for the agent being on this point for each point of the grid. By plotting the grid colored by probability, we can get a temperature map of the estimated position of the agent:

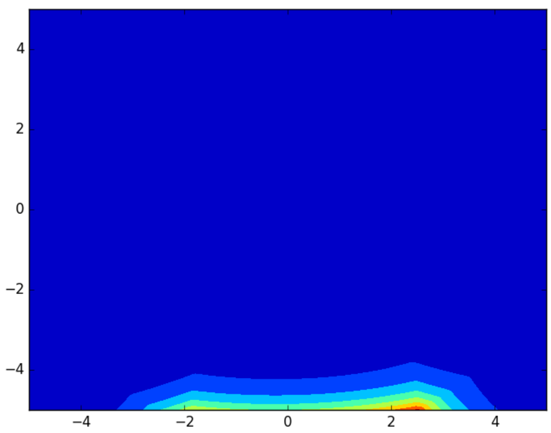

Figure 16: Temperature map of the likelihood for the agent's position

We can see by looking at the pictures that it might be hard to find an optimal solution by using gradient descent and a random starting point, since there are obviously different maxima, so such an algorithm might get stuck in a local maximum.

We find that the maximum here is located on [-5, 2.45], which is not exactly the real position. According to that, running the same algorithm on different measurements could be interesting.

4.2 Question 2

According to the conclusion of the previous question, we now try to compute the estimation based an ALL 2000 measurements. We find [2.37, -4.689] as the mean result after the 2000 iterations, which is not as good as what we find with the least square estimation, and in addition it takes way more time. That being said, we cannot really say the comparison is fair because in this question, we compute the estimated position with no informations but the measurements, when we use a cost function in the least square question. Using a cost function reduce a lot the computation time, since it orientates the search of an optimal solution.

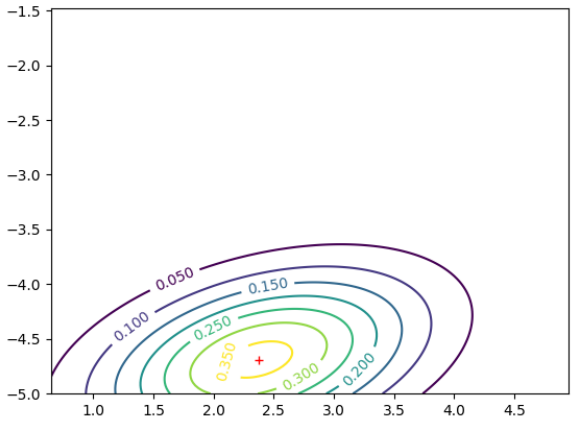

Figure 17: Gaussian Contour

4.3 Question 3

We now want to compute a Bayesian estimator (i.e using the probabilities *a priori*). Once again we use the previously computed grid: We run two times the algorithm, first time by computing the error using the position found in the previous question, and a second time by using the actual true distance of the agent. We find the following results:

- first run: [2.37375 -4.68825]
- second run: [2.274 -4.52875]

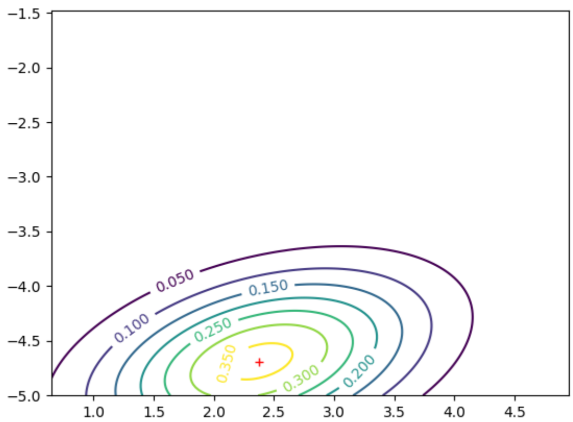

Figure 18: Gaussian Contour for first run

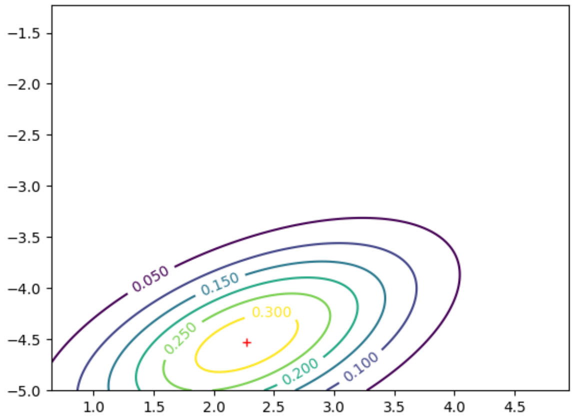

Figure 19: Gaussian Contour for second run

We see that the second one performs better, since the information added is a real value and not predicted.