Assignment 2

Neel Bhalla

October 2021

1 Intro

All code was done in MATLAB r2021a. Data is stored in files `train100.mat`, `train1000.mat`, `train10000.mat`, and `validate.mat` for each program to load.

2 Question 1

2.1 ERM from true pdf

Recall the special case ratio test for a 2 class system is:

$$\frac{p(x|L=1)}{P(x|L=0)} \lessgtr \gamma$$

for some threshold γ. When the problem is parametrized by a loss matrix, γ can be expressed as

$$\frac{p(x|L=1)}{P(x|L=0)} \lessgtr \frac{(\lambda_{10}-\lambda_{00})p(L=0)}{(\lambda_{01}-\lambda_{11})P(L=1)}$$

In order to minimize probability of error, we can use the 0-1 loss matrix:

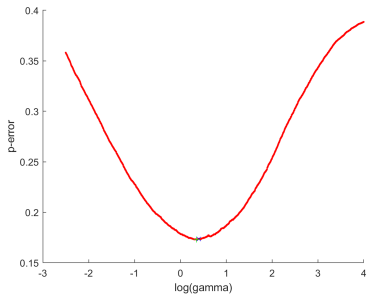
$$\lambda = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Substituting values into the original equation yields:

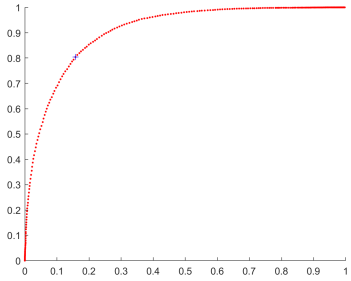
$$\frac{p(x|L=1)}{P(x|L=0)} \lessgtr \frac{p(L=0)}{P(L=1)}$$

$$\frac{p(x|L=1)}{P(x|L=0)} \lessgtr \frac{0.6}{0.4}$$

where $p(x \mid L=0)$ is $\frac{1}{2}g(x, m_{01}, c_{01}) + \frac{1}{2}g(x, m_{02}, c_{02})$
and $p(x \mid L=1) = g(x, m_1, c_1)$



(a) Minimum p-error was achieved at $\gamma = 1.41 \simeq 1.5$ with p-error of 0.173, the theoretical optimal.



(b) ROC curve of the classifier. There is a large AUC, demonstrating that the optimal classifier does perform well for the classification problem at hand.

Figure 1: ROC for classifier and p-error wrt threshold charts to understand performance wrt threshold.

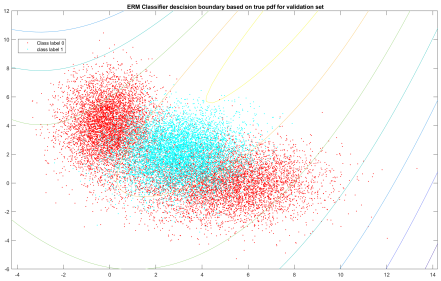


Figure 2: Scatter of samples and contours of the descision function. View this Figure in higher defintion in `Images1/Truepdfclassifier1.png`

It sufifces to implement a classifier simmilar to A1 which yields the following results:

Furthermore we can visualize the descision boundary. Figure 2 depicts the points and the countours of the descision function. One example of a threshold that seems effective is the orange contour.

2.2 EM Gaussian Mixture param estimation

Estimating Class labels 0 consists of determining the weights of the gaussian mixtures (α_1, α_2) alongside their means and covariances. (μ_1, μ_2, Σ_1, Σ_2). Since the two components form the full pdf, we know that $\alpha_1 + \alpha_2 = 1$.

The EM algorithm aims to iteratively improve Θ, the estimation of model parameters, by finding the argmax of the likelyhood given the prior estimation.

Formally

$$\underset{\Theta}{\operatorname{argmax}} \sum_{l=1}^{M}\sum_{i=1}^{N} \ln \alpha_i p(l \mid x_i, \theta^g) + \sum_{l=1}^{M}\sum_{i=1}^{N} \ln \alpha_i p(l \mid x_i, \theta^g) \ln p_l(x_i \mid \theta_l)$$

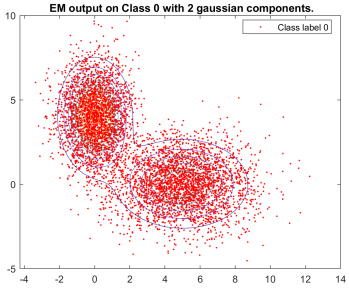
where θ^g was the last estimate of the parameters, and θ_i are the model parameters of gaussian i. It should be noted that Θ is the vector containing $(\alpha_1, \alpha_2, \cdots, \alpha_m, \theta_1, \theta_2, \cdots, \theta_m)$, ie all parameters to describe the distribution.

In order to update the parameters, the following updates are used
$\alpha_l = \frac{1}{N}\sum_{i=1}^{N} p(l \mid x_i, \Theta^g)$
$\mu_l = \frac{\sum_{i=1}^{N} x_i p(l \mid x_i, \Theta^g}{\sum_{i=1}^{N} p(l \mid x_i, \Theta^g)}$
$\Sigma_l = \frac{\sum_{i=1}^{N} p(l \mid x_i, \Theta^g)(x_i - \mu_l)(x_i - \mu_l)^T}{\sum_{i=1}^{N} p(l \mid x_i, \Theta^g)}$

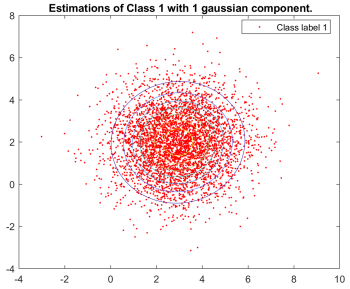
For class label 1, there is only 1 gaussian component, thus standard sample average and sample covariance formulas will suffice: $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$ $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T$

2.2.1 Results 10k train

Using matlabs `fitgmdist`, we can fit a set of data to a given number of gaussian components via EM. The results in Figures 3a and 3b depict the EM algorithm was run on roughly 6k samples (of the 10k dataset).



(a) Class 0 fitting

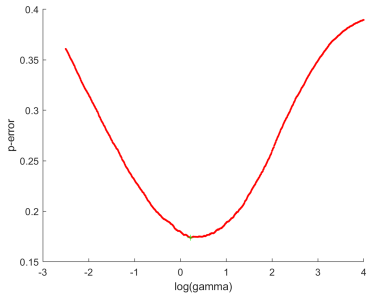


(b) Class 1 fitting.

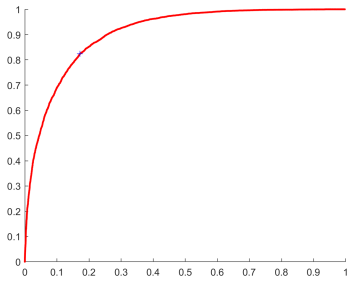
Figure 3: Estimations of model parameters. For Class 0 was classified into a distrbution with weight 0.51 with $\mu = \langle 4.97, 0.02 \rangle$ and cov $\begin{bmatrix} 3.96 & -0.01 \\ -0.01 & 1.95 \end{bmatrix}$ and distrbution with weight 0.49 with $\mu = \langle 0.02, 4.02 \rangle$ and cov $\begin{bmatrix} 1 & -0.02 \\ -0.02 & 2.72 \end{bmatrix}$

Upon estimating the parameters of the gaussian mixtures of each class label, the pdf's were constructed, and then a min-perror ERM binary classifier was trained. Figure 4 deipcts results for ROC and p-error and figure 5 depicts the

boundaries created by the ERM with the new pdfs.



(a) Minimum p-error was achieved at $\gamma = 1.246$ with p-error of 0.173



(b) ROC of the classifier. Operating threshold marked on ROC.

Figure 4: Data about new classifier

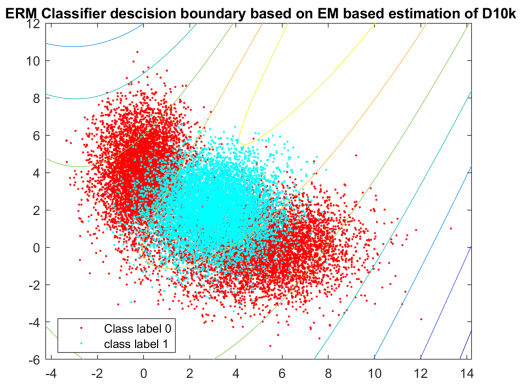
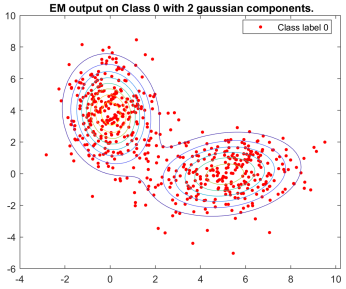


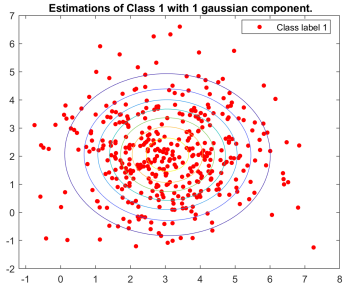
Figure 5: Data plotted with ERM log likelyhood function contours on top. There are the contours in the figure that decide the labels effectively. To view in higher quality take a look at `Images1-2/10k/10kerm.png`

2.2.2 Results 1k train

The results in Figures 6a and 6b depict the EM algorithm was run on roughly 600 samples (of the 1k dataset). Upon estimating the parameters of the gaussian mixtures of each class label, the pdf's were constructed, and then a min-perror ERM binary classifier was trained. Figure 7 deipcts results for ROC and p-error and figure 8 depicts the boundaries created by the ERM with the new pdfs.

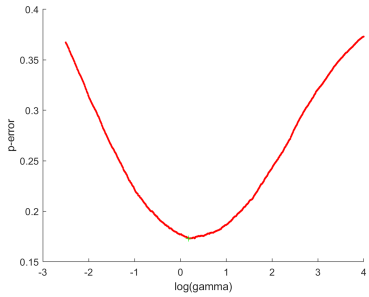


(a) Class 0 fitting

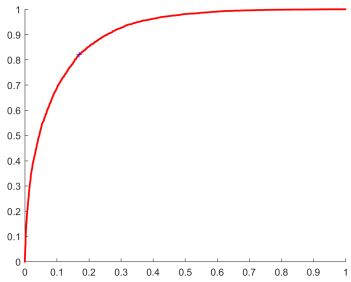


(b) Class 1 fitting.

Figure 6: Estimations of model parameters. For Class 0 was classified into a distribution with weight 0.49 with $\mu = \langle 5.08, 0.02 \rangle$ and cov $\begin{bmatrix} 2.97 & 0.4 \\ 0.4 & 1.8 \end{bmatrix}$ and distribution with weight 0.51 with $\mu = \langle -0.03, 3.79 \rangle$ and cov $\begin{bmatrix} 1.05 & -0.16 \\ -0.16 & 3.30 \end{bmatrix}$



(a) Minimum p-error was achieved at $\gamma = 1.1972$ with p-error of 0.173.



(b) ROC of the classifier. Operating threshold marked on ROC.

Figure 7: Data about 1k classifier. It is interesting that the model still achieved similar performance to ERM 10k while having significantly different parameters

2.3 Results 100 train

The results in Figures 9a and 9b depict the EM algorithm was run on roughly 60 samples (of the 100 sample dataset). The model fits were rough as there was very little data to infer via. Upon estimating the parameters of the gaussian mixtures of each class label, the pdf's were constructed, and then a min-perror ERM binary classifier was trained. Figure 10 deipcts results for ROC and p-error and figure 11 depicts the boundaries created by the ERM with the new pdfs.

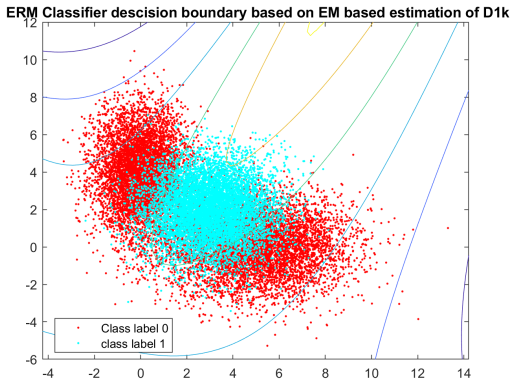
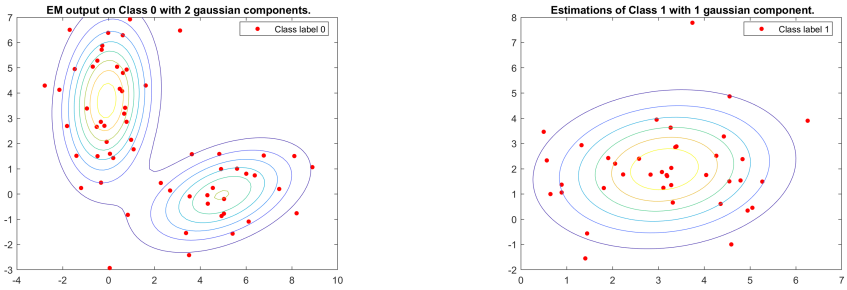


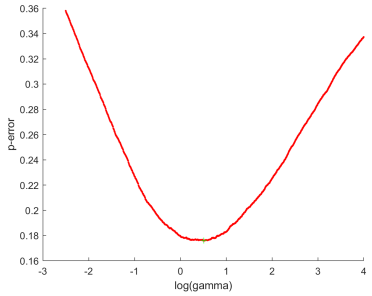
Figure 8: Data plotted with ERM log likelyhood function contours on top. There are the contours in the figure that decide the labels effectively. To view in higher quality take a look at `Images1-2/1k/1kerm.png`



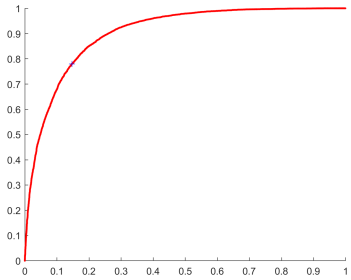
(a) Class 0 fitting

(b) Class 1 fitting.

Figure 9: Estimations of model parameters. For Class 0 was classified into a distribution with weight 0.41 with $\mu = \langle 4.94, -0.08 \rangle$ and cov $\begin{bmatrix} 4.19 & 1.08 \\ 1.08 & 1.428 \end{bmatrix}$ and distribution with weight 0.58 with $\mu = \langle -0.04, 3.70 \rangle$ and cov $\begin{bmatrix} 1.32 & 0.28 \\ 0.28 & 3.594 \end{bmatrix}$



(a) Minimum p-error was achieved at $\gamma = 1.6653$ with p-error of 0.176.



(b) ROC of the classifier. Operating threshold marked on ROC.

Figure 10: Data about the 100 sample classifier. It performed slightly worse than the other 2, but it is interesting that it was still able to achieve a low p-error classifier with very poor estimations of the true pdfs.

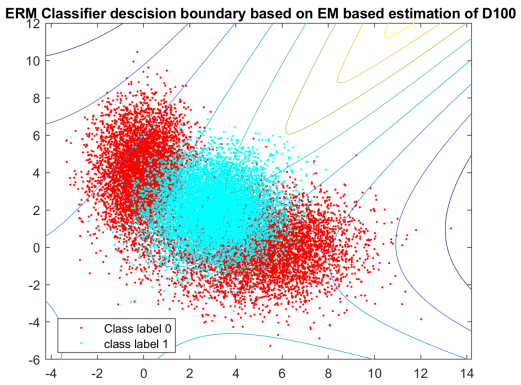


Figure 11: Data plotted with ERM log likelyhood function contours on top. There are the contours in the figure that decide the labels effectively. To view in higher quality take a look at `Images1-2/100/100erm.png`

2.4 LLF Classifiers

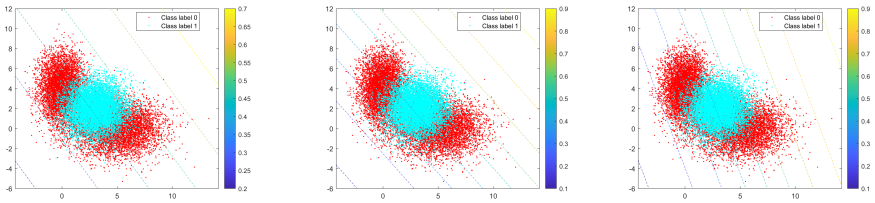
Next LLF classifiers were trained, meaning each input feature vector x was transformed(via b) into $\langle 1, x_1, x_2 \rangle$, and weights were trained such that

$$h(x; \theta) = \frac{1}{1 + e^{-w^T b(x)}}$$

was 1 if the label is 1, and 0 if the true label is zero.

Since LLF produces a linear boundary, it follows that no line will achieve any performance close to the nonlinear theoretical optimal boundary required.

For the 10k training samples, the best classifier can be seen on Figure 12a with a p-error of 0.3966 at a threshold of 0.62. For the 1k training samples, the best classifier can be seen on 12b with a p-error of 0.3966 at a threshold of 0.7600. For the 100 training samples, the best classifier can be seen on 12c with a p-error of 0.3966 at a threshold of 0.842.



(a) LLF classifier trained on a 10k dataset.

(b) LLF classifier trained on a 1k dataset.

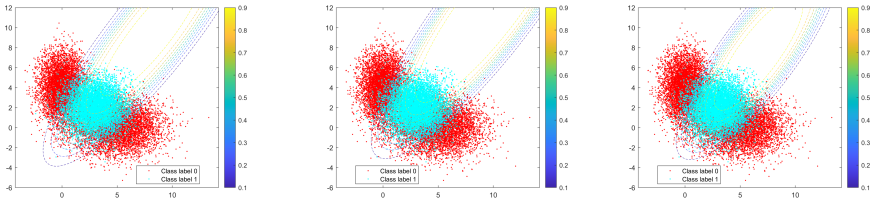
(c) LLF classifier trained on a 100 sample dataset.

Figure 12: LLF classifier results from different training sets. Higher quality images are located in `Images1-3/`

2.5 LQF Classifiers

Seeing that LLF Classifiers were a mismatch for the nonlinear problem required, LQF Classifiers are another approach. The input vector x is now transformed to $\langle 1, x_1, x_2, x_1^2, x_1 x_2, x_2^2 \rangle$, capturing all psuedo-polynomials of max degree 2. Since there are three nonlinear terms, the model is capable of producing quadratic boundaries that can better approximate the nonlinear boundaries from the true pdf.

For the 10k training samples, the best classifier can be seen on Figure 13a with a p-error of 0.173 at a threshold of 0.5330. For the 1k training samples, the best classifier can be seen on 13b with a p-error of 0.173 at a threshold of 0.5330. For the 100 training samples, the best classifier can be seen on 13c with a p-error of 0.1771 at a threshold of 0.656.



(a) LQF classifier trained on a 10k dataset.

(b) LQF classifier trained on a 1k dataset.

(c) LQF classifier trained on a 100 sample dataset.

Figure 13: LQF classifier results from different training sets. Higher quality images are located in `Images1-3/`

Below are the weights of the classifiers presented in figure 13:

	10k	1k	100
1	-0.8563	-0.8196	-0.9606
x_1	1.2111	1.1954	1.6434
x_2	0.0338	0.0125	-0.1508
x_1^2	-0.2762	-0.2749	-0.3518
x_1x_2	0.3402	0.3448	0.4122
x_2^2	-0.1417	-0.1397	-0.1754

It is interesting to note that the 10k and 1k datasets' solutions converged to the same local solution, but the 100 sample solution conerged to a simmilar, but possibly different local minimum.

3 Question 2

Problem formulation:

Let $\theta = \begin{bmatrix} X_T \\ Y_T \end{bmatrix}$ be the true position of the vehicle. We are provided a prior for θ which is

$$p(\theta) = (2\pi\sigma_x\sigma_y)^{-1} e^{-\frac{1}{2}\theta^T \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}^{-1} \theta}$$

Roughly an 2d gaussian pdf centered around (0,0) with cov matrix as follows. Next we define our dataset of pairs (x_i, y_i) as $x_i = [X_i, Y_i]$, the reference point of measurement, and $y_i = r_i + v_i$, the sensor observation, composed of the sum

of a true reading r_i between x_i and θ_T, and $v_i \sim N(0, \sigma_i^2)$. We will assume that for every sample x_i, we know the variance of the noise σ_i^2.

Observe that θ_{MAP} is obtained by looking at the maximum θ when applied to our log likelyhood function on input dataset D.

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}} \ln p(\theta \mid D)$$

Using bayes rule and log rules on $p(\theta \mid D)$ yields:

$$\underset{\theta}{\operatorname{argmax}} \ \ln p(D \mid \theta) + \ln p(\theta) - \ln P(D)$$

But $-\ln P(D)$ is a constant wrt θ, thus the argmax is not affected.
Secondly,

$$\ln P(D \mid \theta) = \ln \prod_{i=1}^{K} P(y_i, x_i \mid \theta)$$

$$= \sum_{i=1}^{K} \ln P(y_i, x_i \mid \theta)$$

Putting it all together, we get

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}} \ \sum_{i=1}^{K} \ln P(y_i, x_i \mid \theta) + \ln P(\theta)$$

Observe $P(y_i, x_i \mid \theta) = P(y_i \mid x_i, \theta) + P(x_i \mid \theta)$, but we shall assume that x_i is independent of θ, thus $P(x_i \mid \theta) = P(x_i)$. Since $P(x_i)$ is not dependent on θ, it can be ignored as well from the argmax.

$$= \underset{\theta}{\operatorname{argmax}} \ \sum_{i=1}^{K} \ln P(y_i \mid x_i, \theta) + \ln P(\theta)$$

Next we will multiply by $-\frac{1}{K}$

$$= \underset{\theta}{\operatorname{argmin}} \ \frac{-1}{K} \sum_{i=1}^{K} \ln P(y_i \mid x_i, \theta) - \frac{1}{K} \ln P(\theta)$$

We now need to find $\ln P(y_i \mid x_i, \theta)$. Recall $y_i \sim r_i + N(0, \sigma_i^2)$, and $r_i = \|x_i - \theta\|$, thus

$$p(y_i \mid x_i, \theta) = g(y_i, r_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{1}{2\sigma_i^2}(y_i - r_i)^2}$$

logging both sides yields

$$\ln p = -\frac{1}{2} \ln (2\pi\sigma_i^2) - \frac{1}{2\sigma_i^2}(y_i - r_i)^2$$

plugging back into our map eqn yields:

$$= \underset{\theta}{\text{argmin}} \ \frac{1}{2K}\sum_{i=1}^{K}(\ln(2\pi\sigma_i^2) + \frac{1}{\sigma_i^2}(y_i - r_i)^2) - \frac{1}{K}\ln P(\theta)$$

$$= \underset{\theta}{\text{argmin}} \ \frac{1}{2K}\sum_{i=1}^{K}\ln(2\pi\sigma_i^2) + \frac{1}{2K}\sum_{i=1}^{K}\frac{1}{\sigma_i^2}(y_i - r_i)^2 - \frac{1}{K}\ln P(\theta)$$

but the first term does not depend on θ, thus

$$= \underset{\theta}{\text{argmin}} \ \frac{1}{2K}\sum_{i=1}^{K}\sigma_i^{-2}(y_i - r_i)^2 - \frac{1}{K}\ln P(\theta)$$

(We cannot factor out σ_i^{-2} as each reference point has a possibly different noise variance).
Next we need to compute $\ln p(\theta)$.Using the equation of the prior, we get

$$\ln p(\theta) = -\ln 2\pi\sigma_x\sigma_y - \frac{1}{2}\theta^T \begin{bmatrix}\sigma_x^2 & 0\\ 0 & \sigma_y^2\end{bmatrix}^{-1}\theta$$

Putting it all together, we get

$$= \underset{\theta}{\text{argmin}} \ \frac{1}{2K}\sum_{i=1}^{K}\sigma_i^{-2}(y_i - r_i)^2 + \frac{1}{K}\ln 2\pi\sigma_x\sigma_y + \frac{1}{2K}\theta^T \begin{bmatrix}\sigma_x^2 & 0\\ 0 & \sigma_y^2\end{bmatrix}^{-1}\theta$$

The middlde term does not depend on θ yielding:

$$= \boxed{\underset{\theta}{\text{argmin}} \ \frac{1}{2K}\sum_{i=1}^{K}\sigma_i^{-2}(y_i - \|x_i - \theta\|)^2 + \frac{1}{2K}\theta^T \begin{bmatrix}\sigma_x^2 & 0\\ 0 & \sigma_y^2\end{bmatrix}^{-1}\theta}$$

3.1 Code

The situation in matlab was tested on $K = 1, 2, 3, 4$, and 40. True position was selected by a point satisfying $r \leq \frac{3}{4}$ and the reference were linearly spaced by angle on $r = 1$. Measurements at a noise sigma of 0.3 and were resampled if negative.

Figure 14 depicts the case of 1 reference point. It is clear that the objective function is primarily dependent on the prior. should there not have been any prior term, we could imagine that the classifer would equally weight any point r_i away from its reference point, but it is clear that this contour matches shape of the countour of the pdf of the prior, thus it took precedent (with a small pertubation from the reference point).

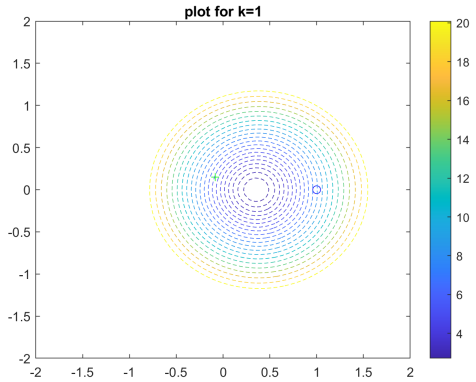


Figure 14: Case for $K = 1$ reference point. Solution is heavily dependent on prior, and is not close to the actual true location

3.2 Code Walkthrough

The code starts by initializing `postrue`, the true position of the vehicle based on polar coordinates with $r \leq \frac{3}{4}$.

Next for each `K` in 1,2,3,4 or 40:

- Compute $\theta_1, \theta_2, \cdots, \theta_k$ via linspace such that it is evenly spaced on the cirlce.
- For all $i \in 1, 2, 3, \cdots K$

  Initialize $\sigma_i = 0.3$

  Let $x_i = \langle \cos\theta_i, \sin\theta_i \rangle$

  Let $y_i = \|x_i - \texttt{truepos}\| + N(0, \sigma_i^2)$

  repeat the prior while $y_i < 0$
- Let $f(\theta)$ be our map equation derived.
- plot the contours of f with other landmarks.

Behaviors: From tinkering with the code, and the figures, it is clear that as we add more points, the readings get more certain. Initially with $K = 1, 2$ points, the objective function is heavily based on the prior of θ (true position), since there is not a lot of confidence in the actual samples (1 sample with sigma $= 0.3$), but as we add more samples, ex: $K = 40$, it is clear that the prior has little effect, and the true position is found within the minimum contour.

Looking at the equations, it follows as $K \to \infty$, the mAp estimate reaches the MLE estimator for the dataset.

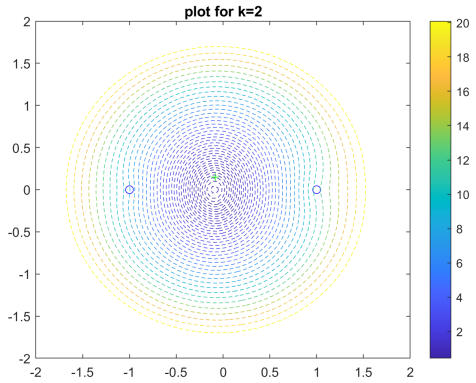
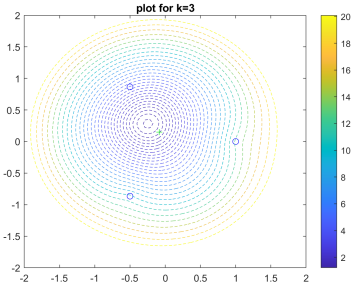
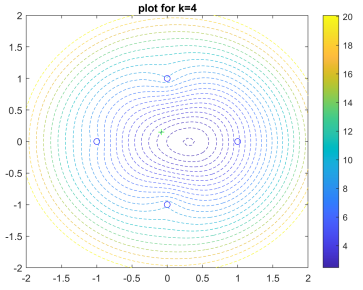


Figure 15: Case for $K = 2$ reference point. Solution is still heavily dependent on prior, but is improving as it has more points. Solution matches very close due to the prior prioritizing points closer to the origin and the reference points are symmetric around the origin.



(a) $K = 3$ case.



(b) $K = 4$ case.

Figure 16: Cases 3 and 4 (not contours are not the same level between figures, fcontour was chopping of ranges for some reason)

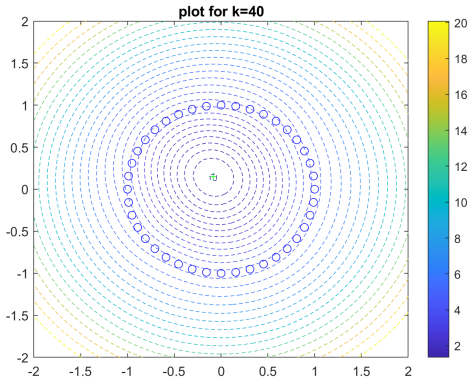


Figure 17: Case for $K = 40$ reference points. Solution is still fairly accurate, finds the true position of the vehicle, even with the noise. Prior has minimal effect due to the number of points contributing to the log likelihood function.

4 Question 3

Given:

The loss applied from deciding i when the true label was j: 0 if $i = j$, λ_r if $i = c + 1$(reject), and λ_s otherwise (mismatch).

Next we can compute $R(D = i \mid x)$ for $i = 1, \cdots, c$ and $i = c + 1$ (two cases).

In the classical cases: $i \in 1, \cdots c,$:

$$R(D = i \mid x) = \sum_{l=1}^{c} \lambda(i \mid l) p(L = l|x)$$

Observe that $\sum_{l=1}^{c} P(L = l|x) = 1$, and $\lambda(i \mid l) = \lambda_s$ for $i \neq l$, thus

$$R(D = i \mid x) = \lambda_s(1 - P(L = i|x))$$

It suffices to find the minimum Risk over all classes $i \in 1, \cdots, c$

$$\min_{i \in 1, \cdots, c} \lambda_s(1 - P(L = i \mid x))$$

$$\min_{i \in 1, \cdots, c} (1 - P(L = i \mid x))$$

$$\max_{i \in 1, \cdots, c} P(L = i \mid x)$$

Part 1: Let such maximum argument (argmax) be y_i, the min-risk label should we have to decide between any of the classes 1 to c, as $\forall j, P(L = y_i \,|\, x) > P(L = j \mid x)$

Next we can calculate the risk with deciding $c + 1$:

$$R(D = c + 1 \mid x) = \sum_{l=1}^{c} \lambda(c + 1 \mid l) p(L = l \mid x)$$

but $\forall l, \lambda(c + 1 \mid l) = \lambda_r$ and $\sum_{l=1}^{c} P(L = l|x) = 1$,

$$= \lambda_r * 1 = \lambda_r$$

Finally it suffices to compute the decision boundary when $R(D = y_i \mid x) > R(D = c + 1 \mid x)$

$$\lambda_s(1 - P(L = i \mid x)) > \lambda_r$$

$$-P(L = y_i \mid x) > \frac{\lambda_r}{\lambda_s} - 1$$

$$\boxed{P(L = y_i \mid x) < 1 - \frac{\lambda_r}{\lambda_s}} \textbf{ rejection criteria}$$

as desired.

Part 3: If the penalty $\lambda_r = 0$, intuitively, there is a zero penalty for rejection, which means there is no reason to risk mismatch, and the classifier will always reject (choose $L = c + 1$).
Mathematically,

$$P(L = y_i \mid x) < 1 - \frac{\lambda_r}{\lambda_s}$$

$$P(L = y_i \mid x) < 1$$

which is always true by definition of probabilities, affirming our intuition.
There is one case in which rejection isnt needed which is if $P(L = y_i \mid x) = 1$, which means that the classifier is 100% certain of deciding such label(in which rejecting or deciding will give 0 penalty). In all practical cases, there is always some nonzero probability for deciding each class, which could be of extremely small magnitude.

Part 4: If the mismatch penalty $\lambda_s > \lambda_r$, intuitively, it is always better to guess rather than reject. This is because by guessing the penalties are either 0 if correct, or λ_s if mismatch, but $\max(0, \lambda_s) < \lambda_r$, thus the classifier should always decide $L \in 1, \cdots, c$.
Mathematically:

$$P(L = y_i \mid x) < 1 - \frac{\lambda_r}{\lambda_s}$$

but

$$\frac{\lambda_r}{\lambda_s} > 1 \rightarrow 1 - \frac{\lambda_r}{\lambda_s} < 0$$

hence

$$P(L = y_i \mid x) < 1 - \frac{\lambda_r}{\lambda_s} < 0$$

but $P(L = y_i \mid x) \geq 0$ by defintion of probability, thus the rejection criteria will never be met.

5 Appendix

All the figures are included in the `Images*/` folder within the zip, but also can be produced in MATLAB . The data generation files will generate new data, but not save it in order to preserve replicable results in the other parts of the assignment.

file	purpose	Figures created
`q1datagen.m`	make data	none
`q1a.m`	true ERM Classifer	1a 1b 2
`q1_2a.m`	EM 10k	3a 3b 4a 4b 5
`q1_2b.m`	EM 1k	6a 6b 7a 7b 8
`q1_2c.m`	EM 100	9a 9b 10a 10b 11
`q1_3aa.m`	LLF 10k	12a
`q1_3ab.m`	LLF 1k	12b
`q1_3ac.m`	LLF 100	12c
`q1_3ba.m`	LQF 10k	13a
`q1_3bb.m`	LQF 1k	13b
`q1_3bc.m`	LQF 100	13c
`q2.m`	Vehicle Experiment	14 15 16a 16b 17

Thanks,
Neel.