Computational Statistics : Assignment 2

Raphaël Bernas

November 2024

1 Exercice 1

1.1 Q1.

Let $n \in \mathbb{N}^*$ and let $X \in X^n = \{x_1, \cdots, x_n\}$ a random variable such that $\mathbb{P}(X = x_i) = p_i > 0$. We want to simulate X.
Assume we know how to simulate the variable $U \sim \mathbb{U}(0,1)$.
Let us recall the following results :

Theorem 1 ***(Inversion Theorem) :*** *For $U \sim \mathbb{U}(0,1)$ and F_X^- the quantil function of a random variable X, then $F_X^-(U) \sim \mathcal{L}aw(X)$*

Therefore, to simulate X we will need to compute F_X^-. First let us assume without loss of generality that $x_1 \leq x_2 \leq \cdots \leq x_n$. Then we can remark that for $k \in \mathbb{N}$ and $x \in [x_k, x_{k+1})$ $F_X(x) = \sum_{i=1}^k p_i$.

Thus we can conclude that for $u \in [\sum_{i=1}^k p_i, \sum_{i=1}^{k+1} p_i)$, then $F_X^-(u) = x_k$. Which can be rewritten in :

$$F_X^-(u) = max\{x_k \in X \mid \sum_{i=1}^{k} p_i \leq u\} \tag{InvFunc}$$

This ultimately give us the following algorithm :

```
Algorithm 1: Inversion sampling method algorithm
    Input: Probabilities list $(p_i)_{i\in\{1,\cdots n\}}$, $X$ ordered values $(x_i)_{i\in\{1,\cdots n\}}$
    Sample $U \sim \mathcal{U}(0,1)$
    $k \leftarrow \underset{i\in\{1,\cdots,n\}}{argmax} \left(\sum_{j=1}^{i} p_j \leq U\right)$
    Return $x_k$
```

1.2 Q2.

You can find all implemented algorithm on this GitHub in the file *TP2.py*:

$$https://github.com/Raphael-Bernas/$$

1.3 Q3.

See Figure 1

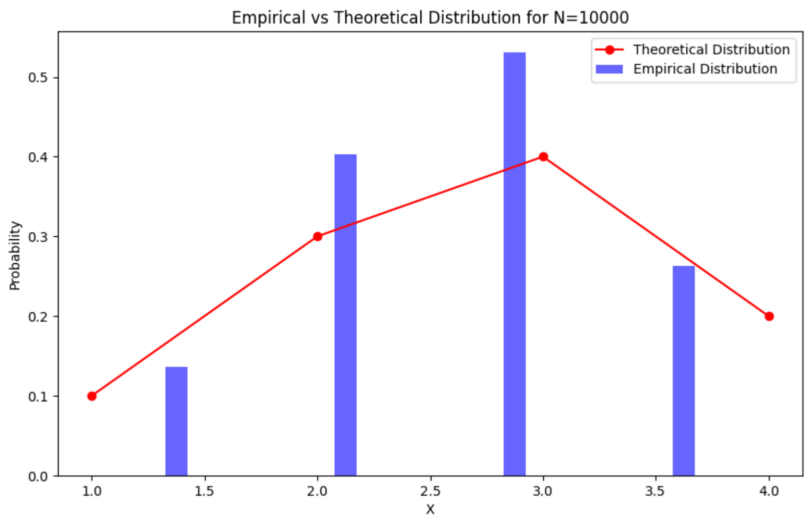


Figure 1: Q3. Empirical distribution (obtained with inversion) VS Theoretical one.

2 Exercise 2

Let us set some more context to this exercise :
We study a sample of random variable $(X_i)_{i\in\{1\cdots n\}}$ taken in $\mathbb{R}^d$.
We denote by $\mathcal{N}(\mu,\Sigma)$ the gaussian law around μ with variance Σ. Such a law as for density $f_{\mathcal{N}(\mu,\Sigma)}(x) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}\exp(\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))\,,\ \forall x\in\mathbb{R}^d$.

2.1 Q1.

In such a case a gaussian mixture model, the parameters are not only the $(\alpha_j)_j$ but also each parameters of each gaussian law $(\mathcal{N}(\mu_j,\Sigma_j))_j$. Thus

$$\theta = (\alpha_1,\cdots,\alpha_m,\mu_1,\Sigma_1,\cdots,\mu_m,\Sigma_m)$$

Now, assume that we have an outcome $(x_i)_{i\in\{1,\cdots n\}}$ of the sample $(X_i)_{i\in\{1\cdots n\}}$. Let us compute the likelihood of θ given this sample :

$$\mathcal{L}(x_1, \cdots, x_n; \theta) = \prod_{i=1}^{n} f_\theta(x_i)$$

$$= \prod_{i=1}^{n} \sum_{j=1}^{m} \mathbb{P}(Z_i = j \,|\, \theta) f_{\mathcal{N}(\mu_j, \Sigma_j)}(x_i)$$

$$= \prod_{i=1}^{n} \sum_{j=1}^{m} \alpha_j f_{\mathcal{N}(\mu_j, \Sigma_j)}(x_i)$$

$$= \sum_{j=1}^{m} \prod_{i=1}^{n} \alpha_j f_{\mathcal{N}(\mu_j, \Sigma_j)}(x_i)$$

$$= \sum_{j=1}^{m} (\alpha_j)^n \prod_{i=1}^{n} \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_j|^{\frac{1}{2}}} \exp(\frac{1}{2}(x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j))$$

$$= \sum_{j=1}^{m} \left(\frac{\alpha_j}{(2\pi)^{\frac{d}{2}} |\Sigma_j|^{\frac{1}{2}}} \right)^n \prod_{i=1}^{n} \exp(\frac{1}{2}(x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j))$$

$$= \sum_{j=1}^{m} \left(\frac{\alpha_j}{(2\pi)^{\frac{d}{2}} |\Sigma_j|^{\frac{1}{2}}} \right)^n \exp(\frac{1}{2} \sum_{i=1}^{n} (x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j))$$

2.2 Q2.

See Figure 2

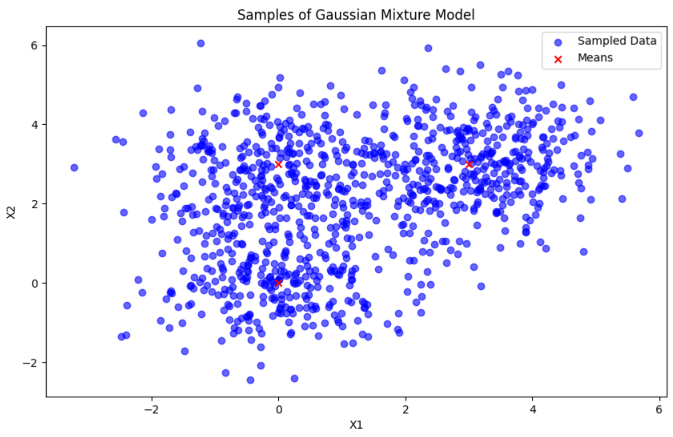


Figure 2: Q2. Gaussian Mixture Model Sample.

2.3 Q3.

See Algorithm2

Algorithm 2: Gaussian Mixture variant of Expectation-Maximization Algorithm

```
Input: $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$, number of components $m$, precision $\epsilon$
Initialize:
Parameters $\theta^{(0)}$ ( coefficients $\alpha^{(0)}$, means $\mu^{(0)}$, covariances $\Sigma^{(0)}$)
$t \leftarrow 0$
while $|\mathcal{L}(x_1, \cdots, x_n; \theta^{(t+1)}) - \mathcal{L}(x_1, \cdots, x_n; \theta^{(t)})| < \epsilon$ do
    E-step: Calculate the responsibilities
        $$\gamma_{ij}^{(t)} = \frac{\alpha_j^{(t)} f_{\mathcal{N}(\mu_j^{(t)}, \Sigma_j^{(t)})}(x_i)}{\sum_{k=1}^{m} \alpha_k^{(t)} f_{\mathcal{N}(\mu_k^{(t)}, \Sigma_k^{(t)})}(x_i)}$$
    M-step: Update the parameters
        $$\alpha_j^{(t+1)} = \frac{1}{n} \sum_{i=1}^{n} \gamma_{ij}^{(t)}$$
        $$\mu_j^{(t+1)} = \frac{\sum_{i=1}^{n} \gamma_{ij}^{(t)} x_i}{\sum_{i=1}^{n} \gamma_{ij}^{(t)}}$$
        $$\Sigma_j^{(t+1)} = \frac{\sum_{i=1}^{n} \gamma_{ij}^{(t)} (x_i - \mu_j^{(t+1)})(x_i - \mu_j^{(t+1)})^T}{\sum_{i=1}^{n} \gamma_{ij}^{(t)}}$$
    Increment: $t = t + 1$
Output: Estimated parameters $\theta$
```

For the log likelihood see Figure 3

We observe that indeed, the likelihood converge implying we have reached a point near our origninal parameters (which should increase the likelihood).

2.4 Q4.

The likelihood plot is not enough to verify that we do converge to the exact parameters. Thus we will plot the following norm for each iterations t:

$$\|\alpha - \alpha^{(t)}\|_\infty$$

$$\|\mu - \mu^{(t)}\|_\infty$$

$$\|\Sigma - \Sigma^{(t)}\|_\infty$$

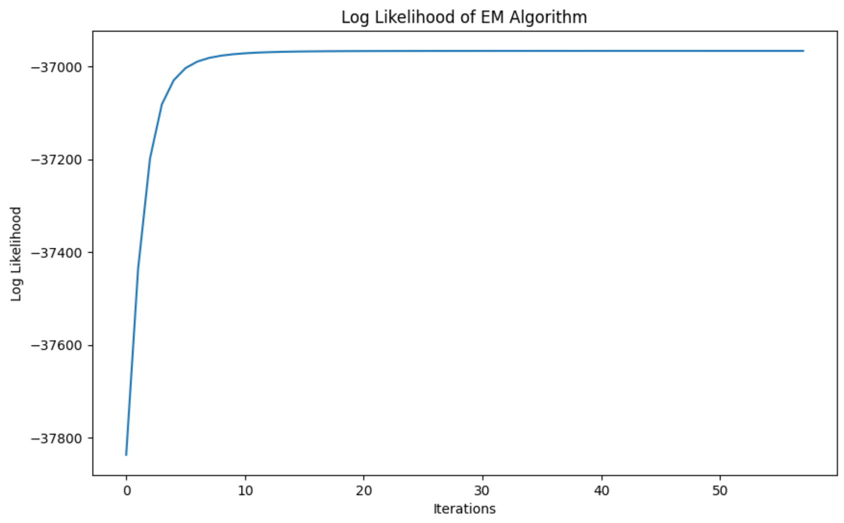


Figure 3: Q3. Log-likelihood of our model upon each EM iteration.

In order to plot them all together we are going to normalised them by the highest value that each norm has taken.

See Figure 4

To get a clearer view let us see the output obtained by our EM algorithm :

See Figure 5

We had chosen :

$$\alpha = (0.3\,,0.4\,,0.3)$$

$$\mu = \begin{pmatrix} 0 & 3 \\ 3 & 3 \\ 0 & 0 \end{pmatrix}$$

Thus the data output appears to be near the expected one (We can explain μ curve by the fact that we have normalized over an already small value).

2.5 Q5.

See Figure 6

It appears that a gaussian mixture model could simulate the data (if we don't take into account the ouliers).

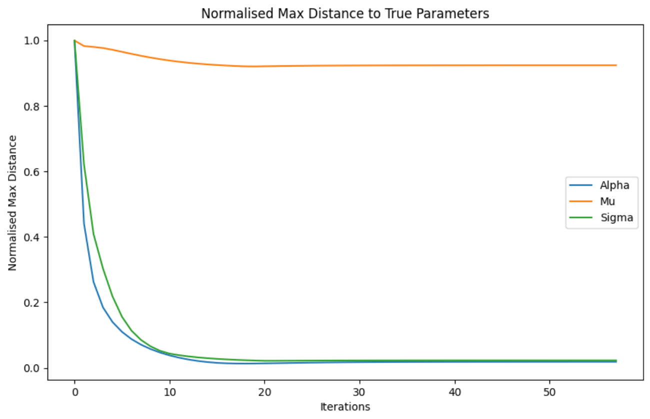
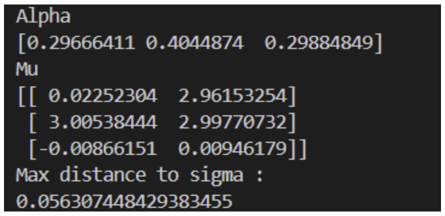


Figure 4: Q4. Normalised maximal distance between true and computed values.



```
Alpha
[0.29666411 0.4044874  0.29884849]
Mu
[[ 0.02252304  2.96153254]
 [ 3.00538444  2.99770732]
 [-0.00866151  0.00946179]]
Max distance to sigma :
0.056307448429383455
```

Figure 5: Q4. Output of the EM algorithm.

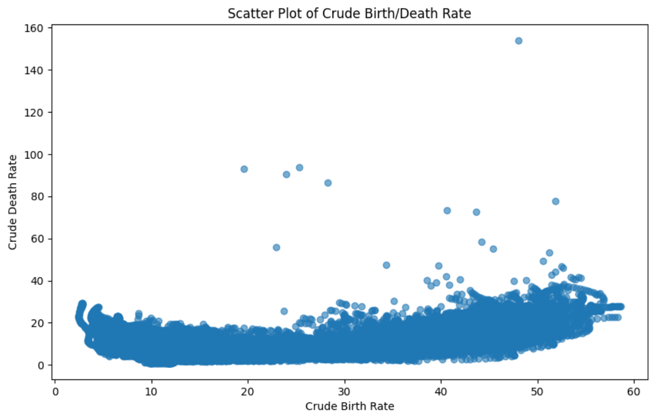


Figure 6: Q5. Scattered data of Crude Birth/Death Rate.

2.6 Q6.

We will be computing different model over our data and we will need to find a way to dicriminate between them. Let us define the Bayesian Information Criterion (BIC), given by:

$$\mathrm{BIC} = -2\ln(\mathcal{L}(\text{data};\theta)) + \mathit{df}(m)\ln(n) \tag{1}$$

where:

$$\begin{aligned} \mathit{df}(m) &= \text{the number of parameters in the model} \\ n &= \text{the number of data points} \end{aligned}$$

Recall that m is the most important number here to differentiate each model (Indeed, m is the number of Gaussian cluster we assume there is in our data). Thus :

- $\mathbf{m-1}$: number of α parameters (one is dependant of all others).
- $\mathbf{m \times d}$: number of means to compute.
- $\mathbf{m \times \frac{d(d+1)}{2}}$: number of parameters for each covariances matrices (they are symetric).

$$\mathit{df}(m) = m\left(1 + d + \frac{d(d+1)}{2}\right) - 1$$

We get the following results :

m =	**BIC**	**loglikelihood**	**EM Iterations**
1	1149716.76	-574830.04	2
2	1037606.37	-518740.84	125
3	1007983.23	-503895.27	237
4	999811.99	-499775.63	425
5	994133.39	-496902.32	254

Table 1: BIC results for each models of GMM.

Thus we select the model with $\hat{m} = 5$ and plot it : See Figure 7.

3 Exercise 3

3.1 Q1.

See Algorithm3

See Figure8

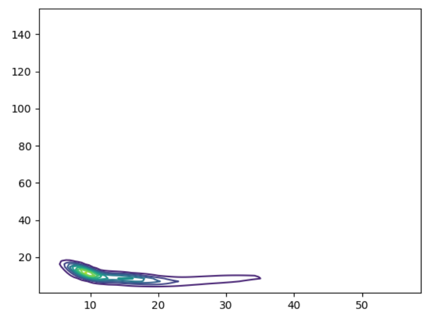


Figure 7: Q6. Plot of the Gaussian Mixture Model contour for Crude Birth/Death Rate data.

Algorithm 3: Importance Sampling Algorithm

```
Input: Number of samples $N$, target density $p(x)$, proposal density
  $q(x)$, function $f(x)$
Initialize:
Set $S_w \leftarrow 0$ and $S_f \leftarrow 0$
for $i = 1$ to $N$ do
    Sample $x_i$ from the proposal density $q(x)$
    if $x_i \geq 0$ then
        Ensure $x_i$ is within the support of $p(x)$
        Compute weight $w_i = \frac{p(x_i)}{q(x_i)}$
        $S_w \leftarrow S_w + w_i$
        $S_f \leftarrow S_f + f(x_i) \cdot w_i$
Compute the estimated average: $M(f) = \frac{S_f}{S_w}$
Output: Estimated average of $f(x)$
```

3.2 Q2.

See Figure9

$$Average \approx 0.77$$

$$Variance \approx 1.57$$

3.3 Q3.

Now we are going to test ”moving” the gaussian law. We change the law such that the mean becomes $\mu = 6$. Obviously we expect such a law to be inproper

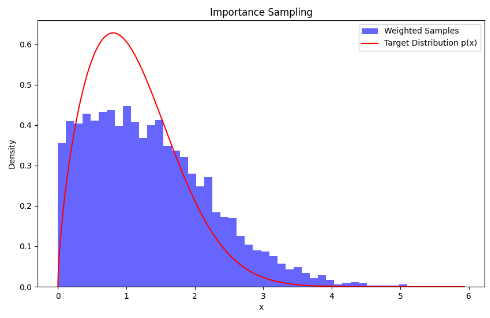


Figure 8: Q1. Weight value comparison with expected density.

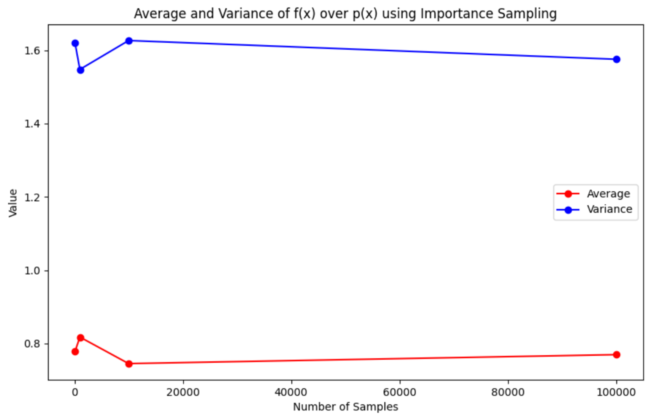


Figure 9: Q2. Average and variance computed for different value of samples using importance sampling.

in order to simulate the density p.

See Figure10

We do observe that our simulated distribution is far from recreating the expected distribution. Thus any average computed would be improper. Indeed the average and variance we get now are :

$$Average \approx -0.37$$

$$Variance \approx 0.64$$

If we compare both Figure 8 and 10 we see that the issues here is not only about sampling wrong values with q but also that we multiply by p computed on a dataset for which it is almost constant. Thus it simplifies when we normalize[1].

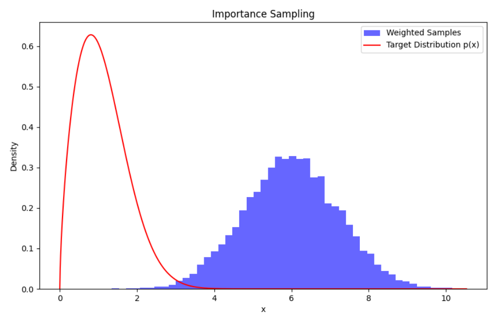


Figure 10: Q3. Weight value comparison with expected density for an incorrect proposal distribution.

3.4 Q4.

When we use the EM algorithm to find an optimal gaussian mixture model what we search to solve is :

$$\theta_1^* \in \underset{\theta=(\alpha_1,\cdots,\alpha_m,\mu_1,\Sigma_1,\cdots,\mu_m,\Sigma_m)}{argmin} \left(\sum_{i=1}^{n} log \left(\sum_{j=1}^{m} \alpha_j f_{\mathcal{N}(\mu_j,\Sigma_j)}(x_i) \right) \right)$$

Here in the Population Monte Carlo Algorithm, at step (iii) we have obtained weights $(\tilde{\omega}_i^{(0)})_{i\in\{1,\cdots,n\}}$ which are meant to simulate the distribution law. Therefore we search to solve :

$$\theta_2^* \in \underset{\theta=(\alpha_1,\cdots,\alpha_m,\mu_1,\Sigma_1,\cdots,\mu_m,\Sigma_m)}{argmin} \left(\sum_{i=1}^{n} \tilde{\omega}_i^{(0)} log \left(\sum_{j=1}^{m} \alpha_j f_{\mathcal{N}(\mu_j,\Sigma_j)}(x_i) \right) \right)$$

This implies that we need to rewrite the EM algorithm to solve our particular case. In the EM each $\gamma_{ij}^{(t)}$ is the responsabilities of x_i in the law $\mathcal{N}(\mu_j, \Sigma_j)$, thus it is an empirical approximation of $\mathbb{P}(Z_i = j|\theta^{(t)})$. Then when we compute the coefficient using those responsabilities we make a strong assumption : **Each data point x_i is as relevant as one another.** For example when we compute $\alpha_j^{(t)}$, the coefficient $\frac{1}{n}$ means that we give the same weight at each data :

$$\alpha_j^{(t+1)} = \sum_{i=1}^{n} \frac{\mathbf{1}}{\mathbf{n}} \gamma_{ij}^{(t)}$$

[1]This is due to python which never exactly return zero implying that when we do normalization both value will simplify. But we could also observe worse effect, for example all weight could be equal to zero because we would be on the function support.

But in this new set up case we cannot do so. Indeed, each data point is generated with a degree of credibility using the importance sampling method. Thus we need to weight each responsabilities by the credibility of such γ_{ij}. This give us the following algorithm :

See Algorithm4

Algorithm 4: Weighted Gaussian Mixture variant of Expectation-Maximization Algorithm

```
Input: $\mathbf{X} = \{x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}\}$, weigh credibility of data
  $(\tilde{\omega}_i^{(0)})_{i \in \{1,\cdots,n\}}$, number of components $m$, precision $\epsilon$
Initialize:
Parameters $\theta^{(0)}$ ( coefficients $\alpha^{(0)}$, means $\mu^{(0)}$, covariances $\Sigma^{(0)}$)
$t \leftarrow 0$
while $|\mathcal{L}(x_1^{(0)}, \cdots, x_n^{(0)}; \theta^{(t+1)}) - \mathcal{L}(x_1^{(0)}, \cdots, x_n^{(0)}; \theta^{(t)})| < \epsilon$ do
    E-step: Calculate the responsibilities
        $\gamma_{ij}^{(t)} = \frac{\alpha_j^{(t)} f_{\mathcal{N}(\mu_j^{(t)}, \Sigma_j^{(t)})}(x_i^{(0)})}{\sum_{k=1}^{m} \alpha_k^{(t)} f_{\mathcal{N}(\mu_k^{(t)}, \Sigma_k^{(t)})}(x_i^{(0)})}$
    M-step: Update the parameters
        $\alpha_j^{(t+1)} = \sum_{i=1}^{n} \tilde{\omega}_i^{(0)} \gamma_{ij}^{(t)}$
        $\mu_j^{(t+1)} = \frac{\sum_{i=1}^{n} \tilde{\omega}_i^{(0)} \gamma_{ij}^{(t)} x_i^{(0)}}{\sum_{i=1}^{n} \tilde{\omega}_i^{(0)} \gamma_{ij}^{(t)}}$
        $\Sigma_j^{(t+1)} = \frac{\sum_{i=1}^{n} \tilde{\omega}_i^{(0)} \gamma_{ij}^{(t)} (x_i^{(0)} - \mu_j^{(t+1)})(x_i^{(0)} - \mu_j^{(t+1)})^T}{\sum_{i=1}^{n} \tilde{\omega}_i^{(0)} \gamma_{ij}^{(t)}}$
    Increment: $t = t + 1$
Output: Estimated parameters $\theta$
```

3.5 Q5.

Now we want to apply the Population Monte Carlo (PMC) Algorithm on a banana shaped density.
Firstly, we are going to generate the data using the true law. Then, we will apply EM algorithm on those data : See Figure 11.

This give us an approcimation of the expected result we would hope to get. And now, applying the population monte carlo algorithm given in the subject

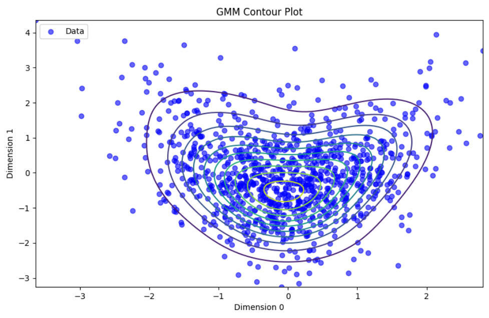


Figure 11: Q5. GMM obtained with EM algorithm on the true data for banana shaped density.

we obtain : See Figure 12

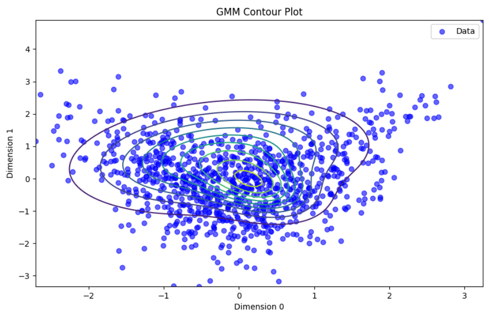


Figure 12: Q5. GMM obtained with PMC algorithm for a banana shaped density.

The PMC algotihm appears to be slow to converge and greatly time consuming but it does simulate around the main part of the density.