Why preferring EM to infer GMM parameters rather than GD ?

Implementation details : Why_EM.ipynb

12 avril 2021

1 Gaussian Mixture model

We consider the following K-Gaussian mixture model (GMM) for multivariate data :

$$\begin{aligned} X_0 \sim & \quad \sum_{k=1}^{K} \pi_k \mathcal{N}(\mu_k, \Sigma_k) \\ \text{s.t.} & \quad \sum_{k=1}^{K} \pi_k = 1 \\ & \quad 0 \leq \pi_k \leq 1 \end{aligned} \tag{1}$$

Let us denote $(x_1, ..., x_n)$ realizations of this random variable. We aim at estimating the weights and the normal component parameters of this mixture $\theta = (\pi_j, \mu_j, \Sigma_j)_{1 \leq j \leq K}$. A standard way to do so is based on Expectation Maximization algorithm but we could also rely on a Gradient Descent method.

2 Gradient Descent

2.1 Idea

The goal is to minimize an objective that can be rewritten as a function of a certain random variable $X \in \mathcal{X}$:

$$\min_{\theta \in \Theta} J_X(\theta) = j(\theta, X)$$

To perform such a minimization, we could come back to a deterministic approach by computing $J_X(\theta)$ and $\nabla_\theta J_X(\theta)$, at each descent iteration, considering X to be fixed. Indeed, the idea is to generate $(x_1, ..., x_n)$ iid realizations of X and to perform updates on θ using simulated instances of X. Many approaches can be used : Newton Method, Gradient Descent, etc.

2.2 On GMM

The function $j : (\theta, X) \mapsto j(\theta, X)$ is set to be the opposite of the log likelihood. We only dispose of a finite number of iid realizations of the random variable X :

$$\begin{aligned} j(\theta, X) = j(\theta, (x_1, ..., x_n)) &= -\mathcal{L}_{(x_1,...,x_n)}(\theta) \\ &= -\sum_{i=1}^{n} \log[\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i; \mu_k, \Sigma_k)] \\ &= -\sum_{i=1}^{n} \log[\sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi}^d} \frac{1}{\sqrt{|\det \Sigma_k|}} e^{-\frac{1}{2}(x_i - \mu_k)^\top \Sigma_k^{-1}(x_i - \mu_k)}] \end{aligned}$$

In the case of a single gaussian, $K = 1$, there exists a closed form to the optimal solution. When $K > 1$, Gradient Descent with projections (to ensure the weights to be in the probability simplex and the covariance matrix to be SDP) is performed. An example in dimension 2 is given in 1.

When increasing the dimension while keeping a fixed number of samples, the estimation error increases, see 2. Because of some numerical instabilities, I could not go further a certain dimension.

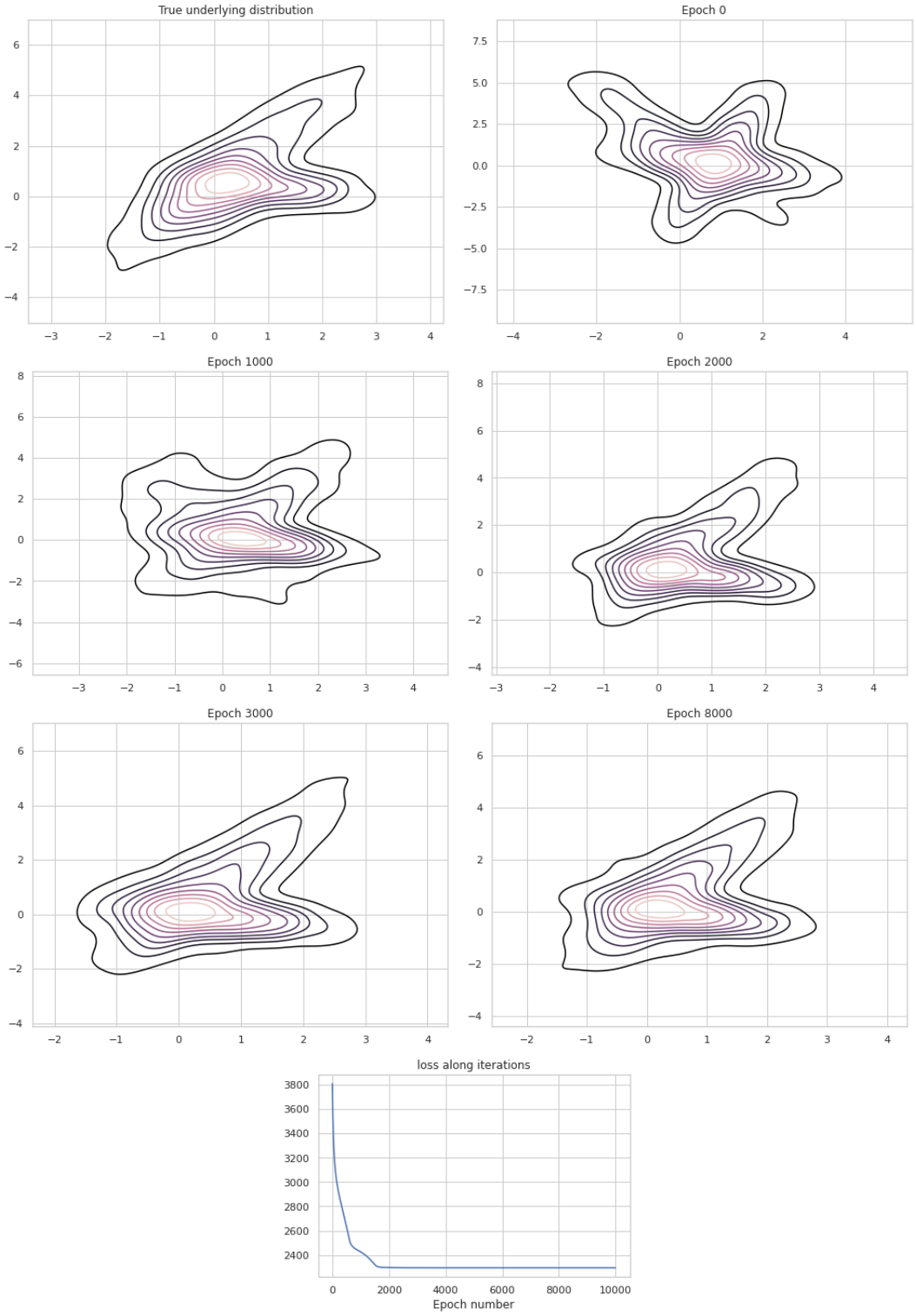

Figure 1: 1000 samples, K=3, learning rate=1e-3, Adam Optimizer

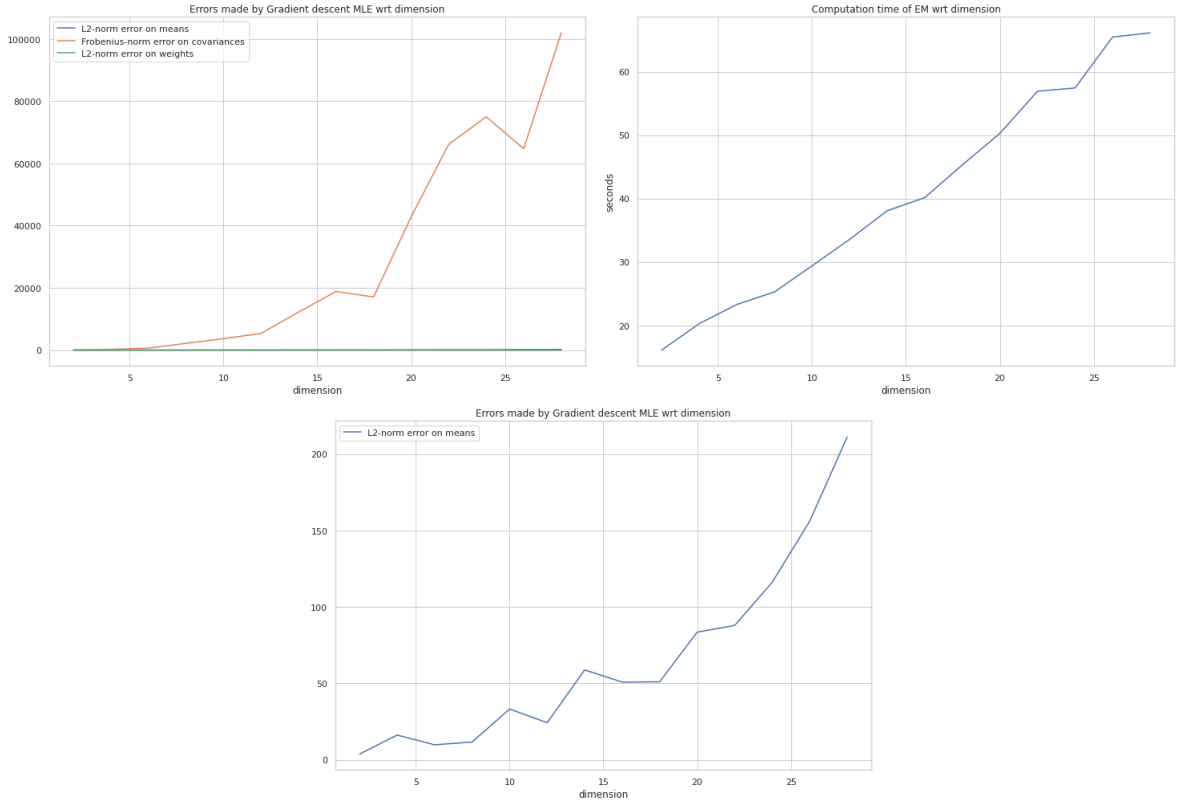

Figure 2: 1000 samples of increasing dimension, K=10, learning rate=1e-3, Adam Optimizer

3 Expectation Maximization

3.1 Idea

Recalling the objective :

$$\theta^* = \arg\min_{\theta\in\Theta} j(\theta, X)$$

Commonly, $j(\theta, X) = -q(x;\theta)$, denoting the log likelihood of parameters θ given fixed observations x. The idea is to introduce a latent variable Z, whose joint distribution with X can be explicitly given. The objective then becomes :

$$\theta^* = \arg\max_{\theta\in\Theta} \mathbb{E}_Z(q(x, Z;\theta))$$

Instead of maximizing this quantity, the standard way to go is to maximize iteratively a lower bound given by Jensen's inequality.

At each step t,

$$\theta_{t+1} = \arg\max_{\theta\in\Theta} Q(\theta|\theta_t)$$
$$\text{where} \quad Q(\theta|\theta_t) = \mathbb{E}_{Z|X,\theta_t}(q(Z, x;\theta_t))$$

3.2 On GMM

3.2.1 Log-likelihood of the complete data sample

We aim at estimating the weights and the normal component parameters of this mixture $\theta = (\pi_j, \mu_j, \Sigma_j)_{1\leq j\leq K}$. Let us denote Z_i a multinomial random variable. $Z_{ij} = 1$ encodes the fact that observation i X_i is from component j. By construction, $Z_i \in \{0,1\}^K$ and $\sum_{j=1}^K Z_{ij} = 1$.

$$\forall i \in Z_i \sim \mathcal{M}(1, \pi) \text{ iid}$$

These realizations are independent, so we can split the log-likelihood of the complete sample $(x_0, ..., x_n)$ into a sum of log-likelihood evaluated on each realization.

$$\begin{aligned}
\log q((x_1, z_1), ..., (x_n, z_n)|\theta) = & \sum_{i=1}^n \log q((x_i, z_i)|\theta) \\
= & \sum_{i=1}^n \sum_{j=1}^K \log\left[q(x_i|z_i = j, \theta) * q(z_i = j|\theta)\right]\mathbb{1}_{z_i=j} \\
= & \sum_{i=1}^n \sum_{j=1}^K \log\left[\pi_j \mathcal{N}(x_i;\, \mu_j,\, \Sigma_j)\right]\mathbb{1}_{z_i=j} \\
= & \sum_{i=1}^n \sum_{j=1}^K [\log \pi_j + \log(\frac{1}{\sqrt{2\pi}^d} \frac{1}{\sqrt{|\det \Sigma_j|}} e^{-\frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j)})]\mathbb{1}_{z_i=j}
\end{aligned}$$

Recall that $x_i|z_i = j \sim \mathcal{N}(x_i;\, \mu_j,\, \Sigma_j)$ and that $q(z_i = j|\theta) = \pi_j$.

Hence the complete data log-likelihood is :

$$\begin{aligned}
\log q((x_1, z_1), ..., (x_n, z_n)|\theta) = & \sum_{i=1}^n \sum_{j=1}^K \log\left[\pi_j \mathcal{N}(x_i;\, \mu_j,\, \Sigma_j)\right]\mathbb{1}_{z_i=j} \\
= & \sum_{i=1}^n \sum_{j=1}^K [\log \pi_j + \log(\frac{1}{\sqrt{2\pi}^d} \frac{1}{\sqrt{|\det \Sigma_j|}} e^{-\frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j)})]\mathbb{1}_{z_i=j}
\end{aligned}$$

3.2.2 EM algorithm

This Expectation-Maximization algorithm aims to forecast the parameters of this mixture $\theta = (\pi_j, \mu_j, \Sigma_j)_{1\leq j\leq K}$, iteratively, from step t to step $t+1$, from $\theta^t = (\pi_j^t, \mu_j^t, \Sigma_j^t)_{1\leq j\leq K}$ to $\theta^{t+1} = (\pi_j^{t+1}, \mu_j^{t+1}, \Sigma_j^{t+1})_{1\leq j\leq K}$

— **Phase E :** Compute the expectation of the complete data log-likelihood (X_i, Z_i), which are independent, under the posterior distribution $Z_i|X_i, \theta^t$:

$$\begin{aligned} Q(\theta|\theta_t) = &\quad \mathbb{E}_{Z|X,\theta_t}[\log q((x_1,z_1),...,(x_n,z_n)|\theta_t)] \\ = &\quad \sum_{i=1}^{n} \mathbb{E}_{Z_i|x_i,\theta_t}[\log q((x_i,z_i)|\theta_t)] \\ = &\quad \sum_{i=1}^{n}\sum_{j=1}^{K} \mathbb{E}_{Z_i|x_i,\theta_t}(\mathbb{1}_{Z_i=j})[\log \pi_j + \log(\frac{1}{\sqrt{2\pi}^d}\frac{1}{\sqrt{|\det \Sigma_j|}} e^{-\frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j)})] \end{aligned}$$

Now, denoting $\tau_{ij}^t = \mathbb{P}(Z_i = j|x_i, \theta_t) = \mathbb{E}_{Z_i|x_i,\theta_t}(\mathbb{1}_{Z_i=j})$, the function Q becomes :

$$Q(\theta|\theta_t) = \quad \sum_{i=1}^{n}\sum_{j=1}^{K} \tau_{ij}^t[\log \pi_j + \log(\frac{1}{\sqrt{2\pi}^d}) - \frac{1}{2}\log|\det \Sigma_j| - \frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j)] \tag{2}$$

— **Phase M :** Maximize this expectation of the complete data log-likelihood with respect to the parameters θ^t. Q is a convex differentiable function with respect to parameters $(\mu_j^t, \Sigma_j^t)_{1\leq j\leq K}$ so the associated maximum arguments will cancel the partial derivatives of Q with respect to these variables :

— over μ_k^t :

$$\begin{aligned} \nabla_{\mu_k^t} Q = &\quad \frac{\partial Q}{\mu_k^t}(\theta|\theta_t) \\ = &\quad \sum_{i=1}^{n} \tau_{ik}^t \Sigma_k^{-1}(x_i - \mu_k) \end{aligned}$$

Hence :

$$\begin{aligned} \nabla_{\tilde{\mu}_k^t} Q = 0 \quad &\Leftrightarrow \quad \sum_{i=1}^{n} \tau_{ik}^t \Sigma_k^{-1} x_i = \sum_{i=1}^{n} \tau_{ik}^t \Sigma_k^{-1} \tilde{\mu}_k \\ &\Leftrightarrow \quad \tilde{\mu}_k^{t+1} = \frac{1}{\sum_{i=1}^n \tau_{ik}^t} \sum_{i=1}^{n} \tau_{ik}^t x_i \end{aligned}$$

— over Σ_k^t :

$$\begin{aligned} \nabla_{\Sigma_k^t} Q = &\quad \frac{\partial Q}{\Sigma_k^t}(\theta|\theta_t) \\ = &\quad \sum_{i=1}^{n} \tau_{ik}^t[-\frac{1}{2}\Sigma_k^{-1} + \frac{1}{2}\Sigma_k^{-1}(x_i-\mu_k)(x_i-\mu_k)^\top \Sigma_k^{-1}] \end{aligned}$$

Hence :

$$\begin{aligned} \nabla_{\tilde{\Sigma}_k^t} Q = 0 \quad &\Leftrightarrow \quad \sum_{i=1}^{n} \tau_{ik}^t = \sum_{i=1}^{n} \tau_{ik}^t \tilde{\Sigma}_k^{-1}(x_i-\mu_k)(x_i-\mu_j)^\top \\ &\Leftrightarrow \quad \tilde{\Sigma}_k^{t+1} = \frac{1}{\sum_{i=1}^n \tau_{ik}^t} \sum_{i=1}^{n} \tau_{ik}^t (x_i-\mu_k)(x_i-\mu_k)^\top \end{aligned}$$

Where we used :

$$\begin{aligned} \frac{\partial \log|\det X|}{\partial X} &= X^{-1} \\ \frac{\partial (a^T X^{-1} b)}{\partial X} &= -X^{-T} a b^T X^{-T} \end{aligned}$$

Note that covariance matrices are symmetric : $\Sigma^T = \Sigma$ and $\Sigma^{-T} = \Sigma^{-1}$.

— over π_k^t : We have a constrained maximization problem :

$$\begin{aligned} \max_{\pi} \quad & \sum_{i=1}^{n}\sum_{j=1}^{K} \tau_{ij}^t [\log \pi_j + \log(\frac{1}{\sqrt{2\pi}^d}) - \frac{1}{2}\log|\det \Sigma_j| - \frac{1}{2}(x_i-\mu_j)^\top \Sigma_j^{-1}(x_i-\mu_j)] \\ \text{s.t.} \quad & \mathbb{1}^T\pi = 1, \\ & \pi \preceq 1, \\ & 0 \preceq \pi \end{aligned} \tag{3}$$

First, we will relax these two last constraints and check at the end if the optimal solution verifies them. Computing the differentiable Lagrangian, and setting its derivative to 0 gives :

$$\mathcal{L}(\pi,\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{K} \tau_{ij}^t \log \pi_j + \lambda(1-\sum_{l=1}^{K}\pi_l)$$

$$\frac{\partial \mathcal{L}}{\partial \tilde{\pi}_k} = 0 \Leftrightarrow \tilde{\pi}_k\lambda = \sum_{i=1}^{n}\tau_{ik}^t$$

$$\Leftrightarrow \tilde{\pi}_k = \frac{\sum_{i=1}^{n}\tau_{ik}^t}{\lambda}$$

Using the constraint :

$$\begin{aligned} \sum_{j=1}^{N}\tilde{\pi}_j = 1 \quad &\Leftrightarrow \quad \sum_{j=1}^{K}\sum_{i=1}^{n}\tilde{\pi}_j = 1 \\ &\Leftrightarrow \quad \sum_{j=1}^{K}\frac{\sum_{i=1}^{n}\tau_{ij}^t}{\lambda} = 1 \\ &\Leftrightarrow \quad \frac{\sum_{i=1}^{n}\sum_{j=1}^{K}\tau_{ij}^t}{\lambda} = 1 \\ &\Leftrightarrow \quad \lambda = \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{K}\tau_{ij}^t} \\ &\Leftrightarrow \quad \tilde{\pi}_k^{t+1} = \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{K}\tau_{ij}^t}\sum_{i=1}^{n}\tau_{ik}^t \end{aligned}$$

— **Initialization** : It is set $\theta^0 = (\pi_j^0, \mu_j^0, \Sigma_j^0)_{1\leq j\leq K}$ using K-Means algorithm on the observations, which makes the algorithm much faster and more robust than with a random initialization.

3.3 Implementation

Based on sklearn packages, the curse of dimensionality is highlighted in what follows 3 :

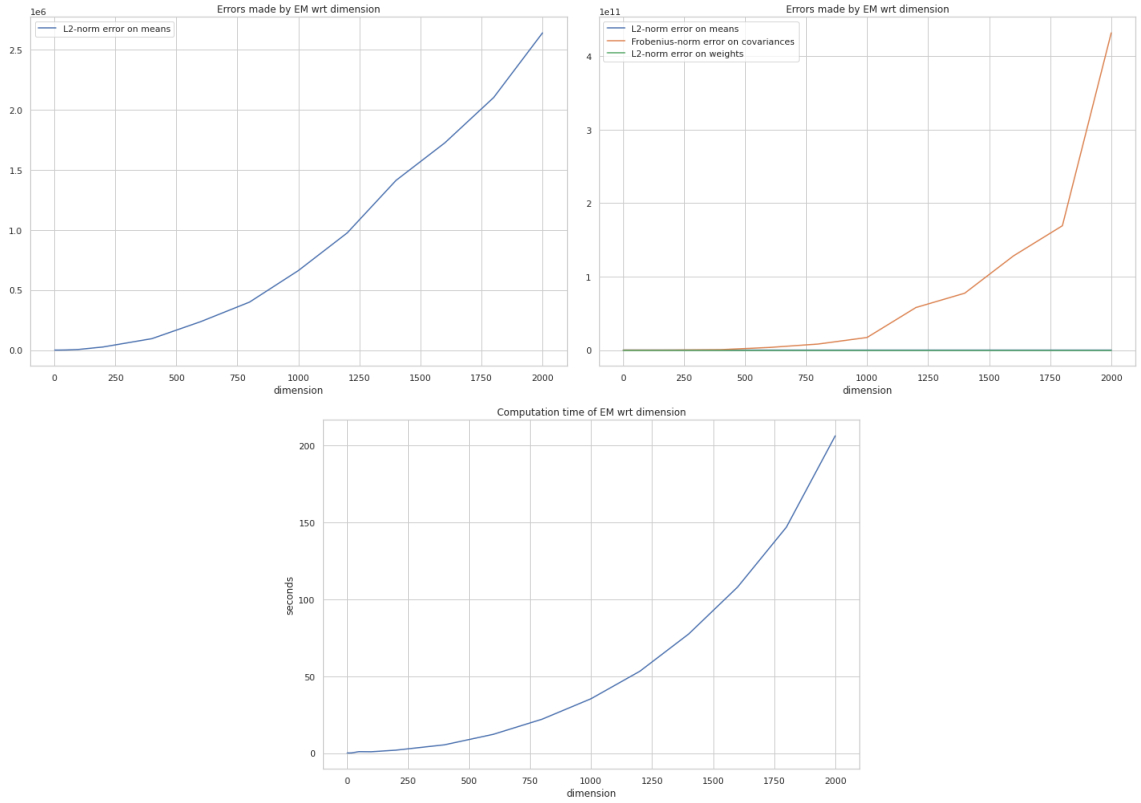

Figure 3: 1000 samples of increasing dimension, K=10, inference with EM algorithm

4 Conclusion

The GMM case is ideally to perform EM instead of GD because the maximization step can be solved efficiently using analytical expressions. Note that only convergence into a local optimum can be ensured under certain conditions. In practice, several runs are performed and the argument maximizing all the lower bounds is kept. Also, EM algorithm suffers from the curse of dimensionality.