Variational Inference: CAVI applied to Bayesian mixture of Gaussians

Caterina Fabbri

3rd February, 2021

Contents

1 Bayesian mixture of Gaussians

In this section we discuss the CAVI in the context of a mixture of univariate Gaussian, following the example reported in Blei et al. (2017).

Assume to dispose of $x_{1:n}$ observations, and that each x_i is a sample from one of K mixture components, each distributed as a univariate Gaussian with unkown mean $\mu_{1:K}$, and unit variance. The cluster means μ_k are drawn independently from a common prior $p(\mu_k)$, assumed to be a Gaussian $\mathcal{N}(0,\ \sigma^2)$, in which σ^2 is a hyper-parameter. The latent cluster, to which each observation belongs to, is assumed to be sampled from a categorical distribution over $\{1, ..., K\}$, and it is marked by the parameter $c_{1:n}$. *In the following calculations c_i is encoded as a K dimensional vector of zeros except for a single one in the position of the x_i's cluster.*

The model can be expressed hierarchically as:

$$\mu_k \sim \mathcal{N}(0,\ \sigma^2)\ ,\ k = 1, ..., K, \tag{1}$$

$$c_i \sim Categorical(1/K,\ \ldots,\ 1/K)\ ,\ i = 1, ..., n, \tag{2}$$

$$x_i | c_i, \boldsymbol{\mu} \sim \mathcal{N}(c_i^T \boldsymbol{\mu},\ 1)\ ,\ i = 1, ..., n \tag{3}$$

As we can see from the graphical representation in Figure 1, to generate a sample x_i we draw K choices of $\boldsymbol{\mu}$ from a Gaussian distribution, as in Distribution (1). Then, we sample, as in Distribution (2), a cluster assignment c_i to decide which mean among $\mu_{1:K}$ is responsible for generating the i-th datapoint. Thus, the x_i are samples from the corresponding Gaussian as in Distribution (3).

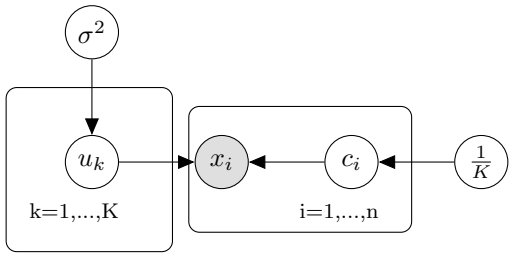

Figure 1: Graphical representation of Bayesian Gaussian Mixture model

1.1 Problem statement

In this setting the latent variables are the cluster mean μ_k, and the cluster assignation c_i. At this point, we would like to compute the exact posterior distribution $p(\boldsymbol{\mu}, \mathbf{c}|\mathbf{x})$ that it is needed to perform statistical inference.

Is the posterior distribution known in closed form? And, in particular, is the normalizing constant of the posterior $p(\mathbf{x})$ analytically tractable?
To answer those questions, let's state the joint probability distribution of the latent and the observed variables for a sample of size n:

$$p(\boldsymbol{\mu},\ \mathbf{c},\ \mathbf{x}) = p(\boldsymbol{\mu})p(\mathbf{c},\ \mathbf{x}|\boldsymbol{\mu}) = p(\boldsymbol{\mu})\prod_{i=1}^{n} p(c_i)p(x_i|c_i,\ \boldsymbol{\mu}) \quad (\textit{follows from Figure 1}) \tag{4}$$

And, we obtain the evidence:

$$\begin{aligned} p(\mathbf{x}) &= \int p(\boldsymbol{\mu}) \prod_{i=1}^{n} \sum_{ci} p(c_i)p(x_i|c_i,\ \boldsymbol{\mu}) d\boldsymbol{\mu} \\ &= \sum_{c} p(c) \int p(\boldsymbol{\mu}) \prod_{i=1}^{n} p(x_i|c_i,\ \boldsymbol{\mu}) d\boldsymbol{\mu} \end{aligned}$$

From the above equation, each single integral is available in closed form thanks to the conjugacy between Gaussian prior-Gaussian likelihood. However, the evidence is a sum over all the possible cluster assignations, resulting in K^n integrals. So, to answer the questions posed above, the posterior is not analytically tractable since the evaluation of the evidence requires a time complexity of K^n.

1.2 Coordinate ascent algorithm for GMM

1.2.1 Mean field assumption

So, the aim is to approximate the posterior distribution $p(\boldsymbol{\mu}, \mathbf{c}|\mathbf{x})$ with a function whose distribution belongs to the mean-field variational family.

$$q(\boldsymbol{\mu},\ \mathbf{c}) = \prod_{k=1}^{K} q(\mu_k) \prod_{i=1}^{n} q(c_i) \tag{5}$$

Note from the above density that each latent variable is governed by its own variational factor. We are forcing each approximating distribution to be independent even if, as follows from the graphical representation in Figure 1, the cluster means and the cluster assignments are not independent since the data is observed.

At this point, there are no further limitations on the form of the variational factors. However, the following derivations will suggest to reproduce the initial ones [1], that for the sake of presentation we introduce here. Following Blei et al. (2017), they are designed the following way:

$$q(\mu_k) \sim \mathcal{N}(m_k,\ s_k^2)\ ,\ k = 1, ..., K, \tag{6}$$

$$q(c_i) \sim \text{Categorical}(\varphi_i)\ ,\ i = 1, ..., n \tag{7}$$

Distributions (6) and (7) define the parametric form of the variational factors: to the k-th mixture component's mean we assign a Gaussian distribution with mean m_k and variance s_k^2, whereas to the i-th observation's mixture assignment is appointed a Categorical distribution with assignment probability φ_i. Note that $\varphi_i \in \mathcal{R}_+^K$ and $\sum_{k=1}^{K} \varphi_{ik} = 1$.
So, m_k and s_k^2 are cluster-level variational parameters, while φ_i is an individual-level variational parameter, since it is specific to the i-th data point.

[1] Rather than a priori assumption, this results from belonging to the conditionally conjugate family (Section 1.4)

Therefore, the variational optimization problem results in finding the variational parameters that maximize the ELBO, that are in total $2K$ (cluster parameters) $+nK$ (individual parameters, since for each observation the probability of belonging to each cluster is computed).

1.2.2 ELBO derivation

Recall the form of the joint distribution between the latent and the observed variables from (4), and the variational family from Distributions (6), and (7).

Expressing the ELBO as $\mathrm{E}_q[\log p(\mathbf{x}, \boldsymbol{\mu}, \mathbf{c})] - \mathrm{E}_q[\log q(\boldsymbol{\mu}, \mathbf{c})]$, let us derive its form in the context of Bayesian mixture of Gaussians - in the below formulation the dependences on the variational parameters are made explicit:

$$ELBO(\mathbf{m},\ \mathbf{s}^2,\ \boldsymbol{\varphi}) = \sum_{k=1}^{K} \mathrm{E}[\log p(\mu_k); m_k,\ s_k^2] + \sum_{i=1}^{n} (\mathrm{E}[\log p(c_i); \varphi_i] + \mathrm{E}[\log p(x_i|c_i,\ \mu); \varphi_i,\ \mathbf{m},\ \mathbf{s}^2])$$

$$- \sum_{i=1}^{n} \mathrm{E}[\log q(c_i; \varphi_i)] - \sum_{k=1}^{K} \mathrm{E}[\log q(\mu_k; m_k,\ s_k^2)] \tag{8}$$

In Equation (8) the expectations are available in closed form [2]. Later on, we will use the ELBO for checking the convergence of the CAVI algorithm.

1.2.3 CAVI update for the cluster assignment (c_i)

In order to find the variational parameters that maximize the ELBO, we apply the CAVI algorithm that updates each parameter in turn. In this section, we report the explained derivation of the cluster assignments (c_i)'s update, following the classical CAVI update $q_i^* \propto \exp \mathbb{E}_{j \neq i}[\log p(\mathbf{x}, \mathbf{z})]$.

In the below equations, the expectation is computed with respect to the terms that differ from $q(c_i)$, i.e. $\boldsymbol{\mu} \sim q(\boldsymbol{\mu})$, and $c_{-i} \sim q(c_{-i})$.

$$\begin{aligned}
q^\star(c_i; \varphi_i) &\propto exp(\mathrm{E}[\log p(\mathbf{x}, \mathbf{c}, \boldsymbol{\mu})]) \\
&\propto exp(\mathrm{E}[\log p(\boldsymbol{\mu}) + \log \prod_{j=1}^{n} p(c_j) p(x_j|c_j, \boldsymbol{\mu})]) \quad (\textit{using Equation (4)}) \\
&\propto exp(\mathrm{E}[\log p(c_i) + \log p(x_i|c_i,\ \boldsymbol{\mu}); \mathbf{m},\ \mathbf{s}^2]) \quad (\textit{removed terms constant w.r.t. } c_i)
\end{aligned}$$

In the first term of the above expression $\mathrm{E}[\log p(c_i)]$ we omit the expectation since the term c_{-i} does not appear. Hence, it becomes $\log p(c_i) = -\log K$ for all i, since it is the prior of the cluster assignment that is distributed as a Categorical (2). The second term, instead, is the expected log likelihood of the Gaussian density (3). Making use of the notation $c_{ik} = \{0, 1\}$, that is the k-th

[2] The full derivation is in Equation (13) in Appendix A.

component of the indicator vector c_i (it assumes value 1 when the kth cluster is allocated to data point x_i), it is possible to write:

$$p(x_i|c_i,\ \boldsymbol{\mu}) = \prod_{k=1}^{K} p(x_i|\mu_k)^{c_{ik}} \tag{9}$$

Hence:

$$\begin{aligned}
&\mathrm{E}[\log p(x_i|c_i,\ \boldsymbol{\mu}); \mathbf{m},\ \mathbf{s}^2] = \mathrm{E}[\log \prod_{k=1}^{K} p(x_i|\mu_k)^{c_{ik}}; \mathbf{m},\ \mathbf{s}^2] \\
&= \sum_k c_{ik} \mathrm{E}[\log p(x_i|\mu_k); m_k,\ s_k^2] \\
&= \sum_k c_{ik} \mathrm{E}[\log \mathcal{N}(x_i^T \boldsymbol{\mu},\ 1)] \quad (\textit{follows from Distribution 3}) \\
&= \sum_k c_{ik} \mathrm{E}[\log(\frac{1}{1\sqrt{2\pi}}) - \frac{1}{2}(\frac{x_i - \mu_k}{1})^2; m_k,\ s_k^2] \\
&= \sum_k c_{ik} \mathrm{E}[-\frac{1}{2}(x_i - \mu_k)^2; m_k,\ s_k^2] + const \quad (\textit{removed terms constant wrt } c_i) \\
&= \sum_k c_{ik} \mathrm{E}[-\frac{1}{2}x_i^2 + \mu_k x_i - \frac{1}{2}\mu_k^2; m_k,\ s_k^2] + const \\
&= \sum_k c_{ik} (x_i \mathrm{E}[\mu_k; m_k,\ s_k^2] - \frac{1}{2}\mathrm{E}[\mu_k^2; m_k,\ s_k^2]) + const \quad (\textit{removed terms constant wrt } c_i)
\end{aligned}$$

Therefore, the variational update for the i-th cluster assignment – as a function of the variational parameters for the mixture component only – is:

$$\varphi_{ik} \propto exp\{x_i \mathrm{E}[\mu_k; m_k,\ s_k^2] - \frac{1}{2}\mathrm{E}[\mu_k^2; m_k,\ s_k^2]\} \tag{10}$$

where φ_{ik} is the probability that the i-th observation comes from the k-th cluster.

The above calculation requires $\mathrm{E}[\mu_k]$ and $\mathrm{E}[\mu_k^2]$, that are computable from the variational Gaussian on the k-th mixture component from Distribution (6). Since $\mathrm{E}[\mu_k] = m_k$ and $\mathrm{E}[\mu_k^2] = m_k^2 + s_k^2$ (using the formula $\mathrm{E}[x^2] = \mathrm{E}[x]^2 + \mathrm{Var}[x]$), Equation (10) becomes:

$$\varphi_{ik} \propto exp\{x_i m_k - \frac{1}{2}(m_k^2 + s_k^2)\} \tag{11}$$

1.2.4 CAVI update for the mixture component means (μ_k)

In this section, we derive the update for the mixture component mean μ_k.
In the equations below, the expectation is computed with respect to the terms that differ from

$q(\mu_k)$, i.e. $\mu_{-k} \sim q(\mu_{-k})$ and $c \sim q(c)$.

$$
\begin{aligned}
q^\star(u_k) &\propto exp(\mathrm{E}[\log p(\mathbf{x}, \mathbf{c}, \boldsymbol{\mu})]) \\
&\propto exp(\mathrm{E}[\log \prod_{z=1}^{K} p(\mu_z) + \log \prod_{i=1}^{n} p(c_i) p(x_i|c_i, \boldsymbol{\mu})]) \quad \textit{(joint as Equation (4))} \\
&\propto \exp(\log p(\mu_k) + \sum_{i=1}^{n} \mathrm{E}[\log p(x_i|\mathrm{c}_i,\ \boldsymbol{\mu}); \varphi_i,\ m_{-k},\ s^2_{-k}]) \quad \textit{(removed terms constant wrt } \mu_k\textit{)}
\end{aligned}
$$

Applying the log to both sides of the above equation, we get:

$$
\begin{aligned}
\log(q_k) &= \log p(\mu_k) + \sum_i \mathrm{E}[\log p(x_i|\mathrm{c}_i,\ \boldsymbol{\mu}); \varphi_i,\ m_{-k},\ s^2_{-k}] + const \\
&= \log p(\mu_k) + \sum_i \mathrm{E}[c_{ik} \log p(x_i|\mu_k); \varphi_i] + const \quad \textit{(removed terms constant wrt } \mu_k \textit{ + use of Notation (9))} \\
&= -\mu_k^2/2\sigma^2 + \sum_i \mathrm{E}[c_{ik}; \varphi_i] \log p(x_i|\mu_k) + const \\
&= -\mu_k^2/2\sigma^2 + \sum_i \varphi_{ik}(-(x_i - \mu_k)^2/2) + const \\
&= -\mu_k^2/2\sigma^2 + \sum_i \varphi_{ik} x_i \mu_k - \varphi_{ik}\mu_k^2/2 + const \\
&= (\sum_i \varphi_{ik} x_i)\mu_k - (1/2\sigma^2 + 1/2 \sum_i \varphi_{ik})\mu_k^2 + const
\end{aligned}
$$

In practice it means that the CAVI optimal variational density $q(\mu_k)$ is a member of the exponential family, (Gaussian) with sufficient statistics $\{\mu_k,\ \mu_k^2\}$ and natural parameters $\{\sum_{i=1}^{n} \varphi_{ik} x_i,\ -\frac{1}{2}\sigma^2 - \frac{1}{2}\sum_{i=1}^{n} \varphi_{ik}\}$. So, the expressions of the updates of the variational mean and the variance are:

$$
m_k = \frac{\sum_i \varphi_{ik} x_i}{1/\sigma^2 + \sum_i \varphi_{ik}} \ ;\ s_k^2 = \frac{1}{1/\sigma^2 + \sum_i \varphi_{ik}} \tag{12}
$$

1.3 CAVI algorithm for GMM

Finally, we can state the full coordinate-ascent variational algorithm for the Bayesian mixture model under analysis. It consists of iteratively optimizing the categorical parameters φ_i with variational updates as derived in Equation (10), and the Gaussian parameters m_k, s_k^2 as in Equation (12), that approximate the posterior cluster assignment of the i-th data point, and the posterior of the k-th mixture components, respectively.
The ELBO as in Equation (8) is computed to check the convergence of the algorithm.

Algorithm 1: CAVI for a Gaussian Mixture Model

Input: Data $x_{1:n}$, number of components K, prior variance σ^2
Initialize variational parameters $\mathbf{m} = m_{1:k}$, $\mathbf{s}^2 = s^2_{1:k}$, $\boldsymbol{\varphi} = \varphi_{1:n}$
while *the ELBO has not converged* **do**
  for $i \leftarrow 1$ **to** n **do**
    Set $\varphi_{ik} \propto exp\{x_i \mathrm{E}[\mu_k; m_k,\ s_k^2] - \frac{1}{2}\mathrm{E}[\mu_k^2; m_k,\ s_k^2]\}$
  end
  for $k \leftarrow 1$ **to** K **do**
    Set $m \leftarrow \frac{\sum_i \varphi_{ik} x_i}{1/\sigma^2 + \sum_i \varphi_{ik}}$
    Set $s_k^2 \leftarrow \frac{1}{1/\sigma^2 + \sum_i \varphi_{ik}}$
  end
  Compute ELBO($\mathbf{m}$, $\mathbf{s}^2$, $\boldsymbol{\varphi}$)
end

1.4 Variational Inference in the Exponential family

In the case of mixture of univariate Gaussians above, we have seen that the variational updates are available in closed-form and can be easily computed. This is a desirable condition, and in this section we introduce a class of model, that includes the aforementioned example, for which these properties hold.
We are referring to the exponential-family-conditional models, as known as **conditionally conjugate models**. As the name is suggesting each complete conditional is in the exponential family, where z_j is the sufficient statistic, $h(\cdot)$ is a base measure, and $a(\cdot)$ is the log normalizer:

$$p(z_j|\mathrm{z}_{-j},\ \mathrm{x}) = h(z_j)\exp\{\eta_j(\mathrm{z}_{-j},\ \mathrm{x})^{\mathrm{T}} z_j - a(\eta_j(\mathrm{z}_{-j},\ \mathrm{x}))\}$$

From the below calculations in which we apply Algorithm 1, we get that the optimal variational factor is in the same exponential family as its full conditional $p(z_j|\mathrm{z}_{-j},\ \mathrm{x})$ with natural parameter equal to $\mathrm{E}_{q-j}[\eta_j(\mathrm{z}_{-j},\ \mathrm{x})]^{\mathrm{T}}$:

$$\begin{aligned} q_j^*(z_j) &\propto \exp\{\mathrm{E}_{q-j}[\log p(z_j|\mathrm{z}_{-j},\ \mathrm{x})]\} \\ &= \exp\{\log h(z_j) + \mathrm{E}_{q-j}[\eta_j(\mathrm{z}_{-j},\ \mathrm{x})]^{\mathrm{T}} z_j - \mathrm{E}_{q-j}[a(\eta_j(\mathrm{z}_{-j},\ \mathrm{x}))]\} \\ &\propto h(z_j)\exp\{\mathrm{E}_{q-j}[\eta_j(\mathrm{z}_{-j},\ \mathrm{x})]^{\mathrm{T}} z_j\} \quad (\textit{removed terms constant w.r.t. } q_j(z_j)) \end{aligned}$$

This is a really convenient result since it implies that, for conditionally conjugate models, not only the updates are available in closed-form, but that they also have a simpler known form. Indeed, we can replace the general CAVI update by setting the natural parameter of $q_j(z_j)$ equal to $\mathrm{E}_{q-j}[\eta_j(\mathrm{z}_{-j},\ \mathrm{x})]^{\mathrm{T}}$.

This result, as said, simplifies the CAVI derivation for the class of conditionally conjugate models, which includes widely used models such as Bayesian mixtures of exponential families with conjugate priors, Hierarchical Hidden Markov Models, mixed membership models of exponential families and Bayesian linear regression.

Appendix A Code

A.1 Application: Bayesian mixture of Gaussians

In this section we continue with the example of mixture of univariate Gaussian as described, and we implement the coordinate ascent algorithm (Algorithm 2) with Python.

A.1.1 ELBO explicitation

The variational updates for the individual parameters can be found in Equation (11), whereas for the cluster parameters in Equation (12). In Equation (8), we report the form of the ELBO as in Blei et al. (2017), that is monitored to check the progress of the algorithm. Below, we make the calculation of the ELBO explicit in order to being able to implement it in the code.

$$
\begin{aligned}
ELBO(m,\ s^2,\ \varphi) \propto & \sum_{k=1}^{K} \mathrm{E}[\log(\frac{1}{2\pi\sigma^2}\exp(-\frac{u_k^2}{2\sigma^2}))] + \sum_{i=1}^{n} \mathrm{E}[-\log(K) - \sum_{k=1}^{K} c_{ik}\log(\frac{1}{\sqrt{2\pi}}\exp(\frac{-(x_i-u_k)^2}{2}))] \\
& - \sum_{i=1}^{n}\log\prod_{k=1}^{K}\varphi_{ik} - \sum_{k=1}^{K}\mathrm{E}[\log(\frac{1}{\sqrt{2\pi}s_k}\exp(-\frac{(u_k-m_k)^2}{2s_k^2}))] \\
& \propto \sum_{k=1}^{K}\frac{-s_k^2-m_k^2}{2\sigma^2} + \sum_{i=1}^{n}\sum_{k=1}^{K}\varphi_{ik}(\frac{2x_i m_k - x_i^2 - m_k^2 - s_k^2}{2}) - \sum_{i=1}^{n}\sum_{k=1}^{K}\log\varphi_{ik} + \sum_{k=1}^{K}\frac{\log s_k^2}{2}
\end{aligned}
\tag{13}
$$

A.1.2 Code in Python

```
#package needed
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

def generate_mixture(n_sample,cluster_mean):

    """
    Sample from a mixture of univariate Gaussians with given mean,
    and unit variance. An equal number of observations is sampled from
    each of the k-th mixture.

    Parameters
    ----------
    n_sample: int
        number of samples for each mixture
    cluster_mean: list
        mixture means
```

```
    Returns
    -------
    sample: np.array of dim (n,1)

    """

    sample = []

    for k in range(len(cluster_mean)):
        sample.extend(list(np.random.normal(cluster_mean[k], 1, n_sample)))

    sample = np.array(sample)[:,np.newaxis]
    return sample

class CaviGaussianMixture:

    def __init__(self,
                 input_data,K,sigma_sq=2):

        """
        Parameters
        ----------
        input_data: np.array of dim (n,1)
        K: int
            known number of clusters
        sigma_sq: int
            hyperparameter prior variance of
            mean
        """

        self.x = input_data
        self.n = len(input_data)
        self.K = K
        self.sigma_sq = sigma_sq

        #initialization of variational parameters
        self.varphi = np.random.random(size=(self.n,self.K)) #shape: (n,K)
        self.m = np.random.randint(min(self.x),max(self.x),K) #shape: (K,1)
        self.s_sq = np.random.random(K) #shape: (K,1)

        self.ELBO = None
        self.ELBO_list = []

    def updates_variational_parameters(self):
```

```
        #update varphi
        update_varphi = np.exp(np.outer(self.x, self.m) -(self.m**2 + self.s_sq)/2)
        #normalize varphi to be in [0,1]
        self.varphi = update_varphi/np.sum(update_varphi, axis=1)[:,np.newaxis]

        #update m
        self.m = np.sum(self.varphi * self.x,axis=0) / (1/self.sigma_sq + \
        + np.sum(self.varphi,axis=0))

        #update s_sq
        self.s_sq = 1 / ((1/self.sigma_sq) + np.sum(self.varphi,axis=0))

    def track_ELBO(self):

        #ELBO computation
        self.ELBO = -np.sum((self.s_sq + self.m**2), axis=0)/(2*self.sigma_sq) + \
        np.sum((self.varphi)*(np.outer(self.x, self.m) - (self.x**2 + self.m**2 + \
        + self.s_sq)/2)) - np.sum(np.log(self.varphi)) + np.sum(np.log(self.s_sq)/2)

        return self.ELBO

    def CAVI(self, n_iters = 1000):
        """
        Update to the variational paramaters,
        and computation of the ELBO. When it does not
        vary much from the previous iteration (1e-3),
        the algorithm has converged.

        Parameters
        ----------
        n_iters: int
           maximum number of iteration to perform

        """

        #ELBO in the initialization
        ELBO = self.track_ELBO()
        self.ELBO_list.append(ELBO)

        for i in range(n_iters):

            self.updates_variational_parameters()

            ELBO = self.track_ELBO()
            self.ELBO_list.append(ELBO)
```

```
            if np.abs(self.ELBO_list[-1] - self.ELBO_list[-2])<= 1e-3:
                print(f"convergence at iteration: {i}")
                print(f"approx of cluster means: {np.sort(self.m)} ")
                break
        return ELBO
```

A.2 Visualization

```
#demonstration
data = generate_mixture(1000,[2,4,8,13,17])

caviclass = CaviGaussianMixture(data,5)

caviclass.CAVI()
#output:
#convergence at iteration: 64
#approx of cluster means: [ 2.04204083  3.98115717  7.95572191 13.01265879
↪  16.98532241]
```

The code above is a demonstration of our toy-implementation. The dataset is composed by 5000 synthetic sample from a mixture of 5 univariate Gaussian distributions with cluster mean equal to $\boldsymbol{\mu} = [2, 4, 8, 13, 17]$. The progresses in the ELBO are monitored, and the convergence is assessed when its change across subsequent iteration is below the threshold of $1e-3$. From Table 1, it is possible to notice that the CAVI algorithm guarantee to provide values for ELBO(q) that cannot decrease as the updates are carried out. Additionally, the curvature of the ELBO is not constant during the maximization, since it is not a convex function in term of the variational parameters. This can be also seen in Table 2 where different initialization lead to different ELBO trajectories, and slightly different variational parameters. Indeed, CAVI guarantees convergence only to a local optimum, which makes it sensitive to initialization. Both from Table 1 and 2 it is possible to see that before reaching the stopping point, the ELBO presents lower plateau than the optimal one, in which the algorithm is stucked in local mode.
Finally Table 3 presents the resulted mean-field approximation.

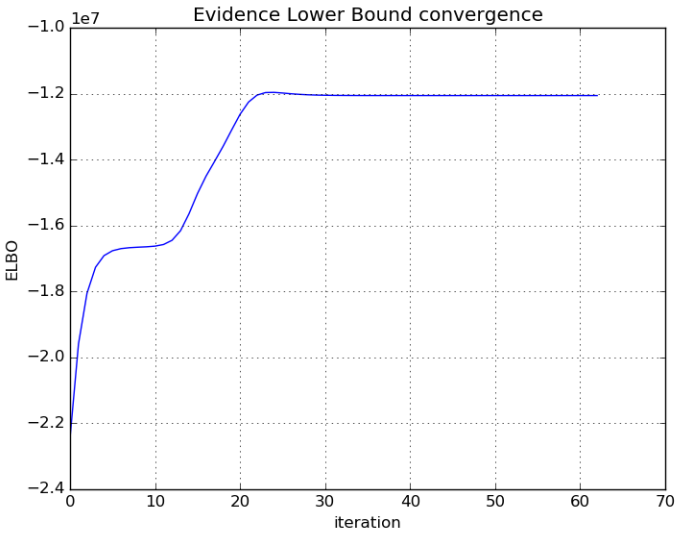

Table 1

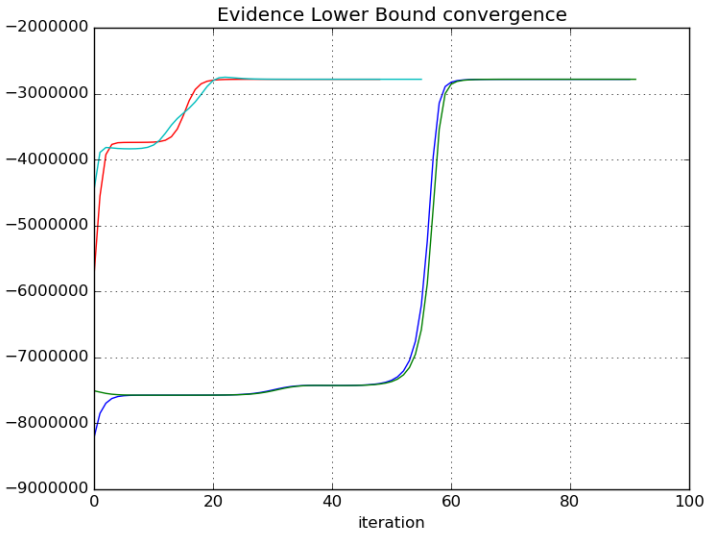

Table 2

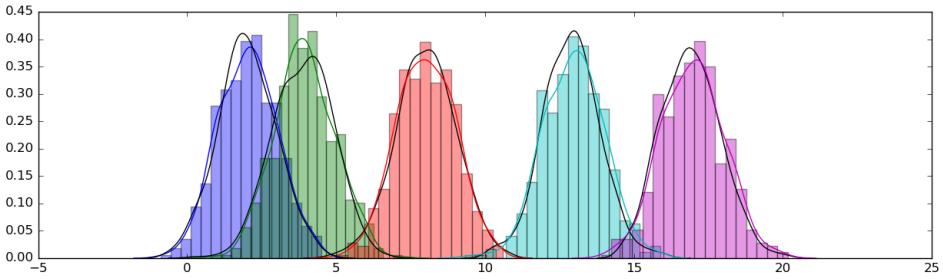

Table 3

References

Blei, David M., Kuculkebir, Alp, and McAucliffe, Jon D. (2017). "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518, pp. 859–877.