Batch and Online Mixture Learning: A Review with Extensions

Christophe Saint-Jean and Frank Nielsen

1 Introduction

Mixture models $f(x;\theta)$ are a powerful and flexible tool to approximate any unknown *smooth* probability density function π as a finite convex combination of parametric density functions $g_j(x;\theta_j)$:

$$\pi(x) \approx f(x;\theta) = \sum_{j=1}^{K} w_j g_j(x;\theta_j), \text{ with } w_j > 0 \text{ and } \sum_{j=1}^{K} w_j = 1, \qquad (1)$$

where $K \in \mathbb{N}$ denotes the number of components of the mixture. Fitting such a kind of semi-parametric model amounts to find a "good" candidate within a parametric family of distributions $\mathcal{F}_\theta$ defined by a set of parameters θ. Among all those distributions, the closest candidate in $\mathcal{F}_\theta$ to π will be denoted f^* (related to the approximation error). Figure 1 depicts the case of a density of a continuous random variable modeled as a mixture of three univariate normal components.

This mixture learning task received much attention in the literature since it is a core operation for both theoretical purposes, and it is widely used in many applications. Classically, one may distinguish two main approaches:

1. Maximum Likelihood Estimation (MLE), and
2. Bayesian Learning.

C. Saint-Jean (✉)
Mathématiques, Image, Applications (MIA), Université de La Rochelle,
La Rochelle, France
e-mail: christophe.saint-jean@univ-lr.fr

F. Nielsen
LIX, École Polytechnique, Palaiseau, France
e-mail: Frank.Nielsen@acm.org

F. Nielsen
Sony CSL, Tokyo, Japan

F. Nielsen et al. (eds.), *Computational Information Geometry*,
Signals and Communication Technology, DOI 10.1007/978-3-319-47058-0_11

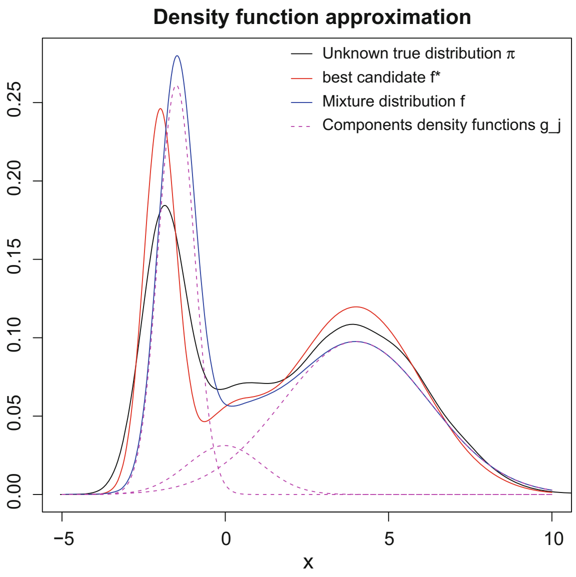


Fig. 1 Approximating an unknown distribution π (*black curve*) with a mixture distribution $f(x;\theta)$ (*blue curve*) of three normal component distributions ($K = 3$, *dashed magenta curves*) (color figure online)

While the former approach gives a *point estimate* of mixture parameters, the latter considers the *posterior distribution* of the parameters given a *prior distribution* on them. In this work, we restrict ourselves to the MLE approach since it is by far the most popular approach.

Consider a random sample $\chi = \{x_i\}_{i=1}^N$ of N independent and identically distributed (iid) observations from π. Under this assumption, the *joint probability* of set χ regarding a particular value for θ is simply $f(\chi;\theta) = \prod_i f(x_i;\theta)$. Viewing χ as a fixed set and θ as a parameter vector, the maximum likelihood estimator (MLE) $\hat{\theta}^{(N)}$ is defined as the maximizer of the likelihood, or equivalently of the average log-likelihood:

$$\bar{l}(\theta;\chi) = N^{-1}\sum_{i=1}^{N}\log f(x_i;\theta) = N^{-1}\sum_{i=1}^{N}\log\left(\sum_{j=1}^{K} w_j g_j(x_i;\theta_j)\right). \qquad (2)$$

In the remainder of this chapter, we will discuss the case when sample χ is not fully known in a whole. That is, we shall consider that the observations x_i are available one after another (e.g. in the data stream model, useful for dealing with very large data sets). Thus online methods differ from batch methods, and ideally aim to get same convergence and efficiency properties as batch ones while having a single pass over the full dataset. This topic receives increasing attention due to the recent challenges associated to massive datasets.

2 Online Learning with Gradient-Based Methods

In this section, we recall the basics of gradient-based optimization and of stochastic approximation. Most of the content below comes from the paper (Bottou 1998; Amari 1997; Bottou and Bousquet 2011).

2.1 Gradient-Based Optimization

The maximization of $\bar{l}$ takes the form of a *sum-minimization* problem (M-estimation):

$$C_N(\theta) = N^{-1} \sum_{i=1}^{N} C(x_i, \theta)$$

for the loss function $C(x, \theta) = -\log f(x; \theta)$. The empirical risk $C_N(\theta)$ evaluated on sample χ of size N is an approximation of the expected risk

$$C(\theta) = \mathbb{E}_\pi[C(x, \theta)].$$

The iterative minimization of the empirical risk with a batch gradient descent (GD) takes the following form:

At iteration t:

$$\boxed{\hat{\theta}^{(t+1)} = \hat{\theta}^{(t)} - \alpha^{(t)} R^{-1}(\hat{\theta}^{(t)}) \underbrace{N^{-1} \sum_{i=1}^{N} \nabla_\theta C(x_i, \hat{\theta}^{(t)})}_{\nabla C_N(\hat{\theta}^{(t)})}} \quad (3)$$

where $\hat{\theta}^{(t)}$ is the parameter estimates, $\alpha^{(t)}$ is the learning rate, and positive definite matrix $R \succ 0$ a rescaling matrix. When R is chosen as the identity matrix, this amounts to ordinary first-order gradient ascent. For $R = \nabla^2 C_N$ chosen as the hessian matrix of C_N, this defines the Newton–Raphson method for finding extrema. Since naïve versions of these methods involve costly operations at each iteration (computation of gradients, hessians for all observations and a matrix inversion), quasi-newton methods (e.g., L-BGFS) which approximate the inverse of hessians are generally preferred.

When the parameter space Θ is a Riemannian manifold with tensor metric G, the direction of the steepest descent at θ is given by the *natural gradient* (Amari 1997, 1998, 2016):

$$\tilde{\nabla}_\theta C_N(\theta) \doteq G^{-1}(\theta) \nabla_\theta C_N(\theta) \quad (4)$$

So, picking $R(\hat{\theta}^{(t)})$ as $G(\hat{\theta}^{(t)})$ defines the natural gradient descent method (Amari 1998).

In information geometry, a D-dimensional parametric exponential or mixture family has a dually flat structure (Amari 2016) induced by a convex potential function F with a canonical divergence a Bregman divergence (with convex generator defined modulo an affine term). The convex function induced two dual coordinate systems θ and η such that $\eta = \nabla F(\theta)$ and $\theta = \nabla F^*(\eta)$, where F^* is the Legendre convex conjugate (Amari 2016). In a dually flat space, the dual basis vectors $e^i = \partial_i = \frac{\partial}{\partial \eta_i}$ and $e_j = \partial^j = \frac{\partial}{\partial \theta^j}$ are orthogonal since $\langle e^i, e_j \rangle = \delta^i_j$ (with $\delta^i_j = 1$ iff $i = j$, and 0 otherwise). We can define a *mixed coordinate system* (Amari 2016, p. 144) ξ by choosing the first k components from the θ-coordinate system and the $D - k$ remaining coordinates from the η-coordinate system. Then then Riemannian metric G in this mixed coordinate system has a *block-diagonal* structure by construction:

$$G = \begin{bmatrix} g_{ij} & 0 \\ 0 & g^{lm} \end{bmatrix},$$

where $g_{ij} = \langle e_i, e_j \rangle$ and $g^{lm} = \langle e^l, e^m \rangle$.

It follows that when $D = 2$, the mixed coordinate systems always ensure a *diagonal* Riemannian (Fisher information) matrix (see Miura (2011) for an example of such parameter orthogonalization). Computing the inverse G^{-1} of a diagonal matrix $G = \mathrm{diag}(a_{11}, \ldots, a_{DD})$ is fast since $G^{-1} = \mathrm{diag}(a_{11}^{-1}, \ldots, a_{DD}^{-1})$, and the gradient-based optimization efficient. However,

Note that the ordinary gradient is obtained for $G = I$ (the identity matrix), and it makes sense to consider this natural gradient updating rule since $\Delta^{(t+1)} = \theta^{(t+1)} - \theta^{(t)}$ is a contravariant vector and ∇l is a covariant derivative. Therefore in the natural gradient, the factor G^{-1} converts a covariant to contravariant vector (Amari 1997).

2.2 Stochastic Gradient Descent Methods

While batch methods have good convergence properties (linear or quadratic), their costs in time and memory is prohibitive when the sample size increases. During the last decade, stochastic methods for optimization (especially those based on GD) have been proven to be very effective in the situation.

Following the seminal work of Robbins and Monro (1951), the observations $\nabla_\theta C(x_1, \theta), \nabla_\theta C(x_2, \theta), \ldots$ can be considered as "noise corrupted" ones of $\nabla_\theta C(\theta)$ for which a root θ^* is searched. Under the assumptions that learning rates $\alpha^{(t)}$ satisfy $\sum_{t \geq 0} \alpha^{(t)} = \infty$ (diverge) and $\sum_{t \geq 0} {\alpha^{(t)}}^2 < \infty$ (converge), they proved that the sequence $\hat{\theta}^{(0)}, \hat{\theta}^{(1)}, \ldots$ in Eq. 5 converges almost surely to θ^*. This method is referred in the literature as the *Stochastic Gradient Descent* (SGD):

At iteration t:

$$\boxed{\hat{\theta}^{(t+1)} = \hat{\theta}^{(t)} - \alpha^{(t)} R(\hat{\theta}^{(t)})^{-1} \nabla_\theta C(x_t, \hat{\theta}^{(t)})} \tag{5}$$

Again, if the parameter space has a non-Euclidean Riemannian structure, it is preferable to use the *stochastic natural gradient descent* (SNGD).

<u>At iteration t</u>:

$$\boxed{\hat{\theta}^{(t+1)} = \hat{\theta}^{(t)} - \alpha^{(t)} \tilde{\nabla}_\theta C(x_t, \hat{\theta}^{(t)})} \tag{6}$$

One strength of the natural gradient descent for online learning besides its invariance under reparameterization is that is it provably Fisher efficient (Amari 1997, 2016), meaning that it meets asymptotically the Cramér-Rao lower bound Amari (2016).

There exist many extensions to this algorithm. In the sequel, we report some old and new heuristics:

(Minibatch SGD) In order the reduce the variance in the parameter update, the gradient of C may be estimated from a limited sample B_t (a.k.a. mini-batch, see Sculley 2010). Since this mini-batch is created at each iteration (successive picks in the stream or through the sampling without replacement from χ), the resulting estimate is also a noisy observation of $\nabla_\theta C(\hat{\theta}^{(t)})$.

<u>At iteration t</u>:

$$\boxed{\hat{\theta}^{(t+1)} = \hat{\theta}^{(t)} - \alpha^{(t)} |B_t|^{-1} \sum_{x \in B_t} \nabla_\theta C(x, \hat{\theta}^{(t)})} \tag{7}$$

(Momentum SGD) Another strategy for regularizing the parameter update consists in doing a convex combination between the previous update and the gradient.[1]

<u>At iteration t</u>:

$$\boxed{\begin{aligned} \Delta^{(t+1)} &= \epsilon^{(t)} \Delta^{(t)} - \alpha^{(t)} \nabla_\theta C(x_t, \hat{\theta}^{(t)}) \\ \hat{\theta}^{(t+1)} &= \hat{\theta}^{(t)} + \Delta^{(t+1)} \end{aligned}} \tag{8}$$

Doing such modification enforces velocity vector Δ to accumulate directions of steepest descent. Momentum coefficient $\epsilon^{(t)}$ is an additional hyper-parameter which has to be set in $[0, 1]$. A popular setting of ϵ consists in taking it around 0.5 in the warmup phase (initial learning) then to increase it towards 0.9 simultaneously to the iterations to enforce the stability of the update.

Better methods have been proposed when a sequence of gradients or parameters over iterations is used. This leads the following heuristics:

(Average SGD) Polyak–Ruppert averaging Polyak and Juditsky (1992) refers to a post-processing method where a second sequence $\bar{\theta}^{(0)}, \bar{\theta}^{(1)}, \ldots$ is generated by averaging estimates after t_0 iterations:

$$\bar{\theta}^{(t)} = \begin{cases} \hat{\theta}^{(t)} & t \leq t_0, \\ \frac{1}{t - t_0} \sum_{t'=t_0+1}^{t} \hat{\theta}^{(t')} & \text{otherwise} \end{cases} \tag{9}$$

[1] It is equivalent to an exponentially decaying moving average of past gradients.

In practice, recursive reformulations are always preferable since it avoids a significant memory cost.
At iteration t:

$$\boxed{\begin{aligned}\hat{\theta}^{(t+1)} &= \hat{\theta}^{(t)} - \alpha^{(t)} R(\hat{\theta}^{(t)})^{-1} \nabla_\theta C(x_t, \hat{\theta}^{(t)}) \\ \bar{\theta}^{(t+1)} &= \left((t-t_0)\,\bar{\theta}^{(t)} + \hat{\theta}^{(t+1)}\right)/(t-t_0+1) \text{ if } t > t_0\end{aligned}} \tag{10}$$

There are many policies for setting $\alpha^{(t)}$. The original proposition in Robbins and Monro (1951) is to pick $\alpha^{(t)} = t^{-1}$ which meets the requirements mentioned before. Nowadays, a classical setting is $\alpha^{(t)} = \alpha^{(0)}(1 + Ct)^{-1}$ where $\alpha^{(0)}$ and C are prescribed constants. Because the convergence of the optimization depends strongly on these constants, several authors suggest to re-evaluate them periodically using a small validation dataset (different from the training set).

(Adam) This method (named after adaptive moment estimation) is a first-order method which use estimates of first and second moments of the gradient with respect to each parameter to estimate.
At iteration t:

$$\boxed{\begin{aligned}\text{for } j = 1,2 \qquad m_j^{(t+1)} &= \beta_j\, m_j^{(t)} + (1-\beta_j)\left(\nabla_\theta C(x_t, \hat{\theta}^{(t)})\right)^{\circ j} \\ \hat{m}_j^{(t+1)} &= m_j^{(t+1)}/(1-\beta_j^t) \\ \hat{\theta}^{(t+1)} = \qquad & \hat{\theta}^{(t)} - \alpha\, \hat{m}_1^{(t+1)}/\left(\sqrt{\hat{m}_2^{(t+1)}} + \epsilon\right)\end{aligned}} \tag{11}$$

The first two steps consist in estimating moments of the gradient using exponential moving averages (the symbol $^{\circ j}$ denotes the Hadamard power) then correct their biases. The biais correction are mandatory since the m_1 and m_2 are initialized as 0's vectors. Note that the learning rate is adapted for each parameter *independently*. One of the most appealing property of this method is that the magnitudes of parameter updates are invariant to rescaling of the gradient and are controlled by hyperparameter α (the term $\hat{m}_1/\left(\sqrt{\hat{m}_2} + \epsilon\right)$ is unitless).

2.3 *From Batch Learning to Online Learning*

Observe that the previous methods (except Minibatch SGD) which do not need to remember previous observations are suitable for on-the-fly processing: the iteration number t becomes the observation number N. In such a case, since the examples are randomly drawn from the ground truth distribution π, the expected risk $C(\theta)$ is directly minimized. Note that the same methods applied to χ, a sample from π, lead to the minimization of $\mathbb{E}_{\pi_N}[C(x,\theta)]$ over an *empirical distribution* π_N with distribution:

$$\pi_N(x) = \frac{1}{N}\sum_{i=1}^{N}\delta_{x_i}(x),$$

where δ_x is the Dirac measure.

In order to prevent overfitting, the empirical risk is classically replaced by a *regularized risk* where a L_1 or L_2 penalty term is added. From the implementation perspective, a fixed dataset χ may be processed seamlessly as a data stream using function generators of modern programming languages (e.g. Python). Such kind of functions is able to yield an observation on demand by repeating infinitely χ (with shuffle). Also, let us mention that the parallelization of optimization techniques remains a very active research field leading to sophisticated hardware and software architectures.

3 Online Mixture Modelling

Before dealing with mixtures of multiple components, the simpler special case of a single component mixture is first discussed below.

3.1 Online Learning with a Single Component

Consider the case when $f = g_1$ is a (regular) exponential family (EF), that is f may be decomposed as

$$f(x;\theta) = \exp\{\langle\theta, s(x)\rangle + k(x) - F(\theta)\} \tag{12}$$

where θ, s, k, F are respectively the natural parameters, the sufficient statistics, the carrier measure, the log-partition function (see Nielsen and Garcia (2009) for further definitions). Most common distributions (but not the uniform, heavy-tailed Student t-, and Cauchy distributions) are regular exponential families: Gaussian, Dirichlet, Multinomial (including the categorical distribution), von Mises-Fisher, Wishart, Rayleigh, etc.

In case of EF, the loss function $C(x,\theta)$ takes the following expression:

$$C(x,\theta) = -\log f(x;\theta) = -\langle\theta, s(x)\rangle - k(x) + F(\theta) \tag{13}$$

The MLE $\hat{\theta}^{(N)}$ is given analytically by differentiating $C_N(\theta)$ with respect to θ:

$$\nabla F(\hat{\theta}^{(N)}) = \frac{1}{N}\sum_{i=1}^{N} s(x_i) \longrightarrow \hat{\theta}^{(N)} = (\nabla F)^{-1}\left(\frac{1}{N}\sum_{i=1}^{N} s(x_i)\right) \tag{14}$$

The functional reciprocal $(\nabla F)^{-1}$ of ∇F is generally available in an explicit formula for most (but not all) EF. It corresponds to the gradient of the dual Legendre convex conjugate (Nielsen and Garcia 2009): $(\nabla F)^{-1} = \nabla F^*$. The Fisher information matrix $I(\theta)$ of a regular exponential family is the Hessian of the log-normalizer:

$$I(\theta) = -E_\theta[\nabla^2 \log f(x;\theta)] = \nabla^2 F(\theta) \succ 0,$$

a positive-definite matrix for all $\theta \in \Theta$, where Θ denotes the natural parameter space. When switching to the online case, it is interesting to get an exact expression of the MLE by a *recursive formulation* of the average of the sufficient statistics. For that, it suffices to keep the sum of the previous sufficient statistics and update as:

$$\hat{\theta}^{(N+1)} = \quad (\nabla F)^{-1}\left(\frac{\{\sum_{i=1}^{N} s(x_i)\} + s(x_{N+1})}{N+1}\right) \tag{15}$$

$$\text{or equivalently} \quad (\nabla F)^{-1}\left(\frac{N\nabla F(\hat{\theta}^{(N)}) + s(x_{N+1})}{N+1}\right) \tag{16}$$

The recursion in Eq. 5 appears more clearly when this formula is written in the Expectation parameter space H (Nielsen and Garcia 2009). Let $\eta = \nabla F(\theta)$. The recursive computation of the exact MLE is then given by[2]:

$$\hat{\eta}^{(N+1)} = \hat{\eta}^{(N)} + \{N+1\}^{-1}(s(x_{N+1}) - \hat{\eta}^{(N)}) \text{ and } \hat{\eta}^{(0)} = 0. \tag{17}$$

It is of interest to compare this expression to the one given by the SGD update (Eq. 5) on natural parameter space Θ:

$$\hat{\theta}^{(N+1)} = \hat{\theta}^{(N)} + \alpha^{(N+1)}\left(s(x_{N+1}) - \nabla_\theta F(\hat{\theta}^{(N)})\right) \tag{18}$$

For the same optimization but in the expectation space H, recall the bijection between exponential families and Bregman divergences (Banerjee et al. 2005):

$$\log f(x;\eta) = -B_{F^*}(s(x) : \eta) + F^*(s(x)) + k(x), \tag{19}$$

where B_{F^*} is the Bregman divergence associated with F^*, the convex conjugate of F. It follows that maximizing the loss function $C(\eta) = \mathbb{E}_\pi[-\log f(x;\eta)]$ leads to the following computation:

[2] When $(\nabla F)^{-1}$ is computed with numerical approximations, this may give a different result.

$$\begin{aligned}-\nabla_\eta \log f(x;\eta) &= \nabla_\eta B_{F^*}(s(x):\eta)\\ &= -\nabla_\eta F^*(\eta) - \nabla_\eta \langle s(x)-\eta, \nabla_\eta F^*(\eta)\rangle\\ &= -H(F^*)(\eta)(s(x)-\eta) \end{aligned} \tag{20}$$

where $H(F^*)(\eta)$ is the hessian of F^* at observed point η. Thus, the minimization of $C(\eta)$ with the stochastic gradient descent on H takes the following form:

$$\hat{\eta}^{(N+1)} = \hat{\eta}^{(N)} + \alpha^{(N+1)} H(F^*)(\hat{\eta}^{(N)})(s(x_{N+1}) - \hat{\eta}^{(N)}) \tag{21}$$

Section 4 of this chapter gives an empirical comparison of Eqs. 17, 18 and 21.

Algorithm 1: Exact Online MLE for regular exponential families

```
Input: a sequence $\mathcal{S} = x_1, x_2, \ldots$ of observations
Input: Functions $s$ and $(\nabla F)^{-1}$ for some regular exponential family
Output: a sequence $\hat{\eta}^{(1)}, \hat{\eta}^{(2)}, \ldots$ of MLE where $\hat{\eta}^{(N)}$ is the exact MLE for the first $N$
        observations
1 $\hat{\eta}^{(0)} = 0$;  $N = 0$;
2 for $x_{N+1} \in \mathcal{S}$ do
3     $\hat{\eta}^{(N+1)} = \hat{\eta}^{(N)} + \{N+1\}^{-1}(s(x_{N+1}) - \hat{\eta}^{(N)})$;
4     yield $\hat{\eta}^{(N+1)}$ or yield $\hat{\theta}^{(N+1)} = (\nabla F)^{-1}(\hat{\eta}^{(N+1)})$;
5     $N = N + 1$;
```

To conclude this part, recall that for a *regular exponential family*, the natural parameter space Θ is an open convex space, and F is strictly convex and differentiable function. It follows that f is a log-concave function and that $-\log f$ is a convex function. Since we consider data stream of many different observations, we are not concerned by the problem of existence of the MLE (see Bogdan and Bogdan (2000) for a rigorous treatment of that point) in Algorithm 1.

When f does not belong to an exponential family, its mathematical properties have to be studied (especially convexity, convex relaxations, etc) and numerical methods are often required (see previous section or Shalev-Shwartz 2011).

3.2 *Batch Mixture Learning with Multiple Components*

Before carrying on with details of online mixture learning methods, let us first recall the basics of Expectation-Maximization algorithm (EM) in the next subsection.

Batch mixture learning with EM For $K > 1$, the direct maximization of $\bar{l}$ is a difficult problem since $\log f$ is the logarithm of the sum of multiple terms ($-\log f$ is no more convex). However it can be made easier if we know the component, let say

z_i, which have generated x_i. This mechanism, called data augmentation, amounts to introduce a latent (unobservable) random variable.

Let Z_i be a categorical random variable over $1, \ldots, K$ whose parameters are $\{w_j\}_j$, that is, $Z_i \sim \mathrm{Cat}_K(\{w_j\}_j)$. Also, assuming that $X_i|Z_i = j \sim g_j(\cdot; \theta_j)$, the unconditional mixture distribution f in Eq. 1 is recovered by marginalizing their joint density p over Z_i (i.e. $f(x) = \sum_z p(x, z)$). Obviously, Z_i is a latent (unobservable) variable so that the realizations x_i of X_i (resp. (x_i, z_i) of (X_i, Z_i)) is often viewed as an incomplete (resp. complete) data observation. Alternatively, we may consider that Z_i is a random vector $[Z_{i,1}, Z_{i,2}, \ldots, Z_{i,k}]$ where $Z_{i,j} = 1$ iff. X_i arises from the j-th component of the mixture and 0 otherwise. Thus, $Z_1, \ldots, Z_N$ are unconditionally distributed according to the multinomial law $\mathcal{M}_K(1, \{w_j\}_j)$ which is an exponential family.

Similarly to Eq. 2, the average complete log-likelihood function can be written as:

$$\begin{aligned}\bar{l}_c(\theta; \chi_c) = \quad & N^{-1} \sum_{i=1}^{N} \log p(x_i, z_i; \theta) = N^{-1} \sum_{i=1}^{N} \log \prod_{j=1}^{K} \left(w_j g_j(x_i; \theta_j)\right)^{z_{i,j}} \\ = \quad & N^{-1} \sum_{i=1}^{N} \sum_{j=1}^{K} z_{i,j} \log(w_j g_j(x_i; \theta_j)) \end{aligned} \tag{22}$$

where $\chi_c = \{(x_i, z_i)\}_{i=1}^N$, is the set of complete data observations.

Here comes the EM algorithm (cf. Algorithm 2) which optimizes $\bar{l}(\theta; \chi)$ (proofs in Dempster et al. 1977; Robbins and Monro 1951; Titterington 1984; Amari 1997, 1998, 2016; Miura 2011; Cappé and Moulines 2009; Neal and Hinton 1999) by repeating two steps until convergence:

- Compute the conditional expectation of missing values

$$\begin{aligned}\mathcal{Q}(\theta; \hat{\theta}^{(t)}, \chi) &:= \mathbb{E}_{\hat{\theta}^{(t)}}[\bar{l}_c(\theta; \chi_c)|\chi] \\ &= N^{-1} \sum_{i=1}^{N} \sum_{j=1}^{K} \mathbb{E}_{\hat{\theta}^{(t)}}[Z_{i,j}|X_i = x_i] \log(w_j g_j(x_i; \theta_j))\end{aligned}$$

- Maximize $\mathcal{Q}(\theta; \hat{\theta}^{(t)}, \chi)$ over θ.

Remark that while $\hat{w}_j^{(t+1)}$ is always known in closed-form whatever the chosen g_j, $\hat{\theta}_j^{(t+1)}$ are obtained by component-wise specific optimization involving **all** observations. More generally, the improvement of $\bar{l}(\theta; \chi)$ is guaranteed whatever the increase of $\mathcal{Q}$ is in the M-Step. This leads to the *Generalized EM* algorithm (GEM) when partial maximization (i.e., not necessarily global optimization) is performed.

Batch mixture learning with EM and EF Consider now the case when all the g_j's are exponential families (EF, e.g. gaussians densities or generalized gaussians densities). The joint density $p(x, z)$ may be decomposed as follows:

Algorithm 2: EM for fitting finite mixture models

```
Input: a set $\chi = x_1, x_2, \dots, x_N$ of observations, $\hat\theta^{(0)} = \{\hat w_j^{(0)}, \hat\theta_j^{(0)}\}_j$ an initial parameter
        values where $\theta_j$ is the parameter of p.d.f. $g_j$.
Output: an estimate $\hat\theta$ of the mixture parameters
1 $t = 0$;
2 repeat
      // E-Step : This step amounts to compute:
3         $$\hat z_{i,j}^{(t)} = \mathbb{E}_{\hat\theta^{(t)}}[Z_{i,j}|X_i = x_i] = \frac{\hat w_j^{(t)} g_j(x_i; \hat\theta_j^{(t)})}{\sum_{j'} \hat w_{j'}^{(t)} g_{j'}(x_i; \hat\theta_{j'}^{(t)})} \quad (23)$$
      // M-Step: Separated maximization of $\{w_j\}_j$ and $\{\theta_j\}_j$
4         $$\hat w_j^{(t+1)} = \frac{\sum_{i=1}^N \hat z_{i,j}^{(t)}}{N}, \quad \hat\theta_j^{(t+1)} = \arg\max_{\theta_j \in \Theta_j} \sum_{i=1}^N \hat z_{i,j}^{(t)} \log\left(g_j(x_i; \theta_j)\right) \quad (24)$$
      $t = t + 1$;
5 until Convergence of $\bar l_c(\theta; \chi_c)$;
6 return $\hat\theta^{(t)}$;
```

$$\begin{aligned}
\log p(x, z; \theta) &= \sum_{j=1}^K [z = j]\{\log(w_j) + \log g_j(x; \theta_j)\} \\
&= \sum_{j=1}^K [z = j]\{\log(w_j) + \langle\theta_j, s_j(x)\rangle + k_j(x) - F_j(\theta_j)\} \\
&= \sum_{j=1}^K \left\langle \begin{pmatrix} [z = j] \\ [z = j]\, s_j(x) \end{pmatrix}, \begin{pmatrix} \log w_j - F_j(\theta_j) \\ \theta_j \end{pmatrix} \right\rangle + \sum_{j=1}^K [z = j]\, k_j(x) \\
&= \langle s(x, z), \theta_c\rangle + \sum_{j=1}^K [z = j]\, k_j(x)
\end{aligned}$$

where $[z = j]$ denotes the Iverson's bracket,

$$s(x, z) := \quad ([z = 1], [z = 1]\, s_1(x), \quad \dots, [z = K], [z = K]\, s_K(x))^T \tag{25}$$

$$\theta_c := \quad (\log w_1 - F_1(\theta_1), \theta_1, \quad \dots, \log w_K - F_K(\theta_K), \theta_K)^T \tag{26}$$

Note that notation θ_c may be considered as ambiguous but in the paper the subscript j always refers to component-specific parameters. One can then recognize the form of an exponential family. Then, it follows very simple expressions for $\bar l_c$ and $\mathcal{Q}$:

$$\bar{l}_c(\theta; \chi_c) = N^{-1}\sum_{i=1}^{N}\langle s(x_i, z_i), \theta_c\rangle + N^{-1}\sum_{i=1}^{N}\sum_{j=1}^{K} z_{i,j}\, k_j(x_i) \tag{27}$$

$$\begin{aligned}\mathcal{Q}(\theta; \hat{\theta}^{(t)}, \chi) =& N^{-1}\sum_{i=1}^{N}\langle \mathbb{E}_{\hat{\theta}^{(t)}}\left[s(X_i, Z_i)|X_i = x_i\right], \theta_c\rangle + \\ & N^{-1}\sum_{i=1}^{N}\sum_{j=1}^{K}\mathbb{E}_{\hat{\theta}^{(t)}}\left[Z_{i,j}|X_i = x_i\right]\, k_j(x_i)\end{aligned} \tag{28}$$

Since the second term is irrelevant (i.e., a constant) to the maximization $\mathcal{Q}$, the EM algorithm for such distributions can be reformulated with sufficient statistics for complete data. The E-Step amounts to compute the vector $\hat{S}^{(t)}$, the empirical average of the conditional expectation of sufficient statistics for complete data (see Eq. 30). The M-Step consists in finding the value θ_c which maximizes the inner product with $\hat{S}^{(t)}$ (see Eq. 31). If this mapping is denoted by $\theta^\dagger : H \mapsto \Theta$, the EM algorithm for the mixture of EF can be written in one recurring formula:

$$\hat{S}^{(t+1)} = N^{-1}\sum_{i=1}^{N}\mathbb{E}_{\theta^\dagger(\hat{S}^{(t)})}[s(X_i, Z_i)|X_i = x_i] \tag{29}$$

where initial values $\hat{S}^{(0)}$ is given by $\hat{\theta}^{(0)}$ and Eq. 26.

Algorithm 3: EM for fitting finite mixture models of exponential families

```
Input: a set $\chi = x_1, x_2, \ldots, x_N$ of observations, $\hat{\theta}^{(0)} = \{\hat{w}_j^{(0)}, \hat{\theta}_j^{(0)}\}_j$ an initial parameter
        values where $\theta_j$ is the parameter of exponential family $g_j$.
Output: an estimate $\hat{\theta}$ of the mixture parameters
$t = 0$;
repeat
    E-Step :   $\hat{S}^{(t)} = N^{-1}\sum_{i=1}^{N}\mathbb{E}_{\hat{\theta}^{(t)}}[s(X_i, Z_i)|X_i = x_i]$   (30)
    M-Step :   $\hat{w}_j^{(t+1)} = \hat{S}_{2j-1}^{(t)}, \quad \hat{\eta}_j^{(t+1)} = \nabla F_j(\hat{\theta}_j^{(t+1)}) = \hat{S}_{2j}^{(t)} / \hat{S}_{2j-1}^{(t)}$   (31)
    $t = t + 1$;
until Convergence;
return $\hat{\theta}^{(t)}$;
```

3.3 Online Mixture Learning with Multiple Components

The case of online mixture learning is discussed in the following. It is now more appropriate to denote $\hat{\theta}^{(N)}$ the current parameter estimate instead of $\hat{\theta}^{(t)}$.

Titterington's algorithm The first online algorithm, due to Titterington (1984), corresponds to the direct optimization of $\mathcal{Q}(\theta; \hat{\theta}^{(N)}, \chi)$ using a second-order stochastic gradient ascent:

$$\boxed{\hat{\theta}^{(N+1)} = \hat{\theta}^{(N)} + \alpha^{(N+1)} I_c^{-1}(\hat{\theta}^{(N)}) \nabla_\theta \log f(x_{N+1}; \hat{\theta}^{(N)})} \tag{32}$$

where $\{\alpha^{(N+1)}\}$ is a decreasing sequence of positive step sizes ($\alpha^{(N+1)} = N^{-1}$ in the original paper) and the hessian $\nabla^2 \mathcal{Q}$ of $\mathcal{Q}$ is approximated by the Fisher Information matrix I_c for the complete data:

$$I_c(\hat{\theta}^{(N)}) = -\mathbb{E}_{\hat{\theta}^{(N)}} \left[H(\log p(x, z; \theta)) \right],$$

where H denotes the hessian operator ∇^2.

The justification of this recursion relies on the Fisher's identity (see discussion in Dempster et al. 1977) for finite mixture models: for any value θ' for mixture parameters,

$$\begin{aligned}
\nabla_\theta \log f(x; \theta') &= f(x; \theta')^{-1} \nabla_\theta f(x; \theta') = f(x; \theta')^{-1} \sum_z \nabla_\theta p(x, z; \theta') \\
&= f(x; \theta')^{-1} \sum_z \left\{ p(x, z; \theta') \nabla_\theta \log p(x, z; \theta') \right\} \\
&= \sum_z \left\{ h(z|x; \theta') \nabla_\theta \log p(x, z; \theta') \right\} \\
&= \mathbb{E}_{\theta'} [\nabla_\theta \log p(X, Z; \theta') | X = x]
\end{aligned} \tag{33}$$

where $h(z|x; \theta)$ is the conditional density of z given x.

It follows that the gradient of $\mathcal{Q}$ at $\hat{\theta}^{(N)}$ (see Eq. 28) has a particular form especially when the model for the complete data is an exponential family:

$$\begin{aligned}
\nabla_\theta \mathcal{Q}(\hat{\theta}^{(N)}; \hat{\theta}^{(N)}, \chi) &= N^{-1} \sum_{i=1}^{N} \mathbb{E}_{\hat{\theta}^{(N)}} [\nabla_\theta \log p(X_i, Z_i; \hat{\theta}^{(N)}) | X_i = x_i] \\
&= N^{-1} \sum_{i=1}^{N} \nabla_\theta \log f(x_i; \hat{\theta}^{(N)})
\end{aligned}$$

In order to incorporate the constraint on weight components ($\sum_{j=1}^{K} w_j = 1$), the last component weight w_K is removed from the parameters to be optimized and set to

$w_K = 1 - \sum_{j=1}^{K-1} w_j$. Thus, if the θ-coordinate system is considered and parameters are ordered as $\theta = (w_1, \ldots, w_{K-1}, \theta_1, \ldots, \theta_K)$, we are able to further describe this algorithm for mixtures of exponential families (MEFs):

$$\text{For } j = 1, \ldots, K-1 \quad \frac{\partial \log f(x_{N+1}; \hat{\theta}^{(N)})}{\partial w_j} = \frac{g_j(x_{N+1}; \hat{\theta}_j^{(N)}) - g_K(x_{N+1}; \hat{\theta}_K^{(N)})}{f(x_{N+1}; \hat{\theta}^{(N)})} \tag{34}$$

$$\text{For } j = 1, \ldots, K \quad \frac{\partial \log f(x_{N+1}; \hat{\theta}^{(N)})}{\partial \theta_j} = \hat{z}_{N+1,j}^{(N)} \left(s_j(x_{N+1}) - \nabla_{\theta_j} F_j(\hat{\theta}_j^{(N)}) \right) \tag{35}$$

where $\hat{z}_{N+1,j}^{(N)} = w_j g_j(x_{N+1}; \hat{\theta}_j^{(N)}) / f(x_{N+1}; \hat{\theta}^{(N)})$.

Due to the chosen parametrization, the hessian of $\log p$ is a block diagonal matrix where the hessians $H(F_j)$ of all the F_j appear. It follows that the information matrix I_c is easier to compute:

$$I_c(\theta) = blockdiag\left(\left(diag(w_1^{-1}, \ldots, w_{K-1}^{-1}) - \frac{\mathbf{1}_{K-1}{}^t\mathbf{1}_{K-1}}{w_K} \right), w_1 H(F_1)(\theta_1), \ldots, w_K H(F_K)(\theta_K) \right) \tag{36}$$

The inverse of first block matrix is given by the Sherman-Morrison identity (see formula 160 in Petersen and Pedersen 2012). By plugging these results in Eq. 32, the update equations for a generic MEF are:

<u>For a new observation x_{N+1}</u>,

$$\boxed{\begin{aligned} \hat{w}_j^{(N+1)} &= \hat{w}_j^{(N)} + \alpha^{(N+1)} (\hat{z}_{N+1,j}^{(N)} - \hat{w}_j^{(N)}) \text{ and } \hat{w}_K^{(N+1)} = 1 - \sum_{j=1}^{K-1} \hat{w}_j^{(N+1)} && (37) \\ \hat{\theta}_j^{(N+1)} &= \hat{\theta}_j^{(N)} + \alpha^{(N+1)} \frac{\hat{z}_{N+1,j}^{(N)}}{\hat{w}_j^{(N)}} H(F_j)^{-1}(\hat{\theta}_j^{(N)}) \left(s_j(x_{N+1}) - \nabla_{\theta_j} F_j(\hat{\theta}_j^{(N)}) \right) && (38) \end{aligned}}$$

This recursive procedure does not necessarily take into account the constraints on the θ_j's (e.g. $\theta_j > 0$ for a mixture of Rayleigh distributions).

Online EM In a recent paper, Cappé and Moulines (2009) proposed to replace the E-Step by a *stochastic approximation* and leave the M-step unchanged. Here are the key ideas of their approach in the case of mixture of EFs.

When considering an infinite number of observations, the EM update given by an empirical average in Eq. 29 can be defined by the mapping $\mathcal{T} : H \mapsto H$ as follows:

$$\mathcal{T}(S) = \mathbb{E}_{\pi} \left[\mathbb{E}_{\theta^{\dagger}(S)}[s(X, Z)|X] \right] \tag{39}$$

Thus, the sequence $\hat{S}^{(0)}, \hat{S}^{(1)}, \hat{S}^{(2)}, \ldots$ converges to the sequence $\hat{S}^{(0)}, \mathcal{T}(\hat{S}^{(0)})$, $\mathcal{T}(\mathcal{T}(\hat{S}^{(0)})), \ldots$ which depends only on $\hat{S}^{(0)}$. Finding the limit of this sequence amounts to find a fixed point of $\mathcal{T}$ or equivalently to look for a root of the function $C : H \mapsto H$:

$$C(S) = \mathcal{T}(S) - S = \mathbb{E}_{\pi}\left[\mathbb{E}_{\theta^{\dagger}(S)}[s(X, Z)|X] - S\right] \tag{40}$$

According again to the framework of Robbins-Monro, one can get the solution by iterating :

$$\tilde{S}^{(N+1)} = \tilde{S}^{(N)} + \alpha^{(N+1)}\left(\mathbb{E}_{\theta^{\dagger}(\tilde{S}^{(N)})}[s(x_{N+1}, z_{N+1})|x_{N+1}] - \tilde{S}^{(N)}\right) \tag{41}$$

The initial value for parameters $\hat{\theta}^{(0)}$ is transformed $\tilde{S}^{(0)}$ by Eq. 31. Obviously, this formula is comparable to the one for $K = 1$ (see Eq. 17).

This approach guarantees that parameter constraints are automatically respected solving a known problem for Titterington's approach. The authors have proved that two algorithms are asymptotically equivalent. The link between the two approaches will be discussed later on.

4 Experiments

4.1 Online Learning for a Gaussian Distribution

The aim of this first experiment is to test several methods of optimization for the simple case of the online learning of a single univariate gaussian distribution. This experiment may appear to be unnecessary since a recursive formulation for the MLE is known from Eq. 17. Hence, many properties of previous optimization methods can be exhibited from this case. This distribution is an exponential family for which the canonical decomposition is recalled in Appendix 6.1. In particular, Eqs. 54, 58 and 66 are needed for the different update formulas 17, 18 and 21.

The experiment consists in the recursive estimation of the parameters on an univariate gaussian $\mathcal{N}(\mu = 1, \sigma^2 = 4)$ from a continuous stream of its realizations. The dataset of size 60,000 is splitted on two parts, one for training (1/3) and the other for the validation (2/3). Two criteria are used to evaluate the estimates $(\hat{\mu}^{(N)}, \hat{\sigma}^{2(N)})$: the average log-likelihood on training and testing datasets, the Kullback–Leibler (KL) divergence (see. Eq. 68) between true parameters values and their estimates.

The results of the recursive estimation with exact formula are reported on Fig. 2. As expected, since the variance of the MLE for a N-sample is $\{NI(\lambda)\}^{-1}$ (see Eq. 69), a convergence is achieved quite quickly. This method does not depend on a particular initialization and one can remark that the average log-likelihood does not necessarily increase after incorporating a new observation. This property is common to many

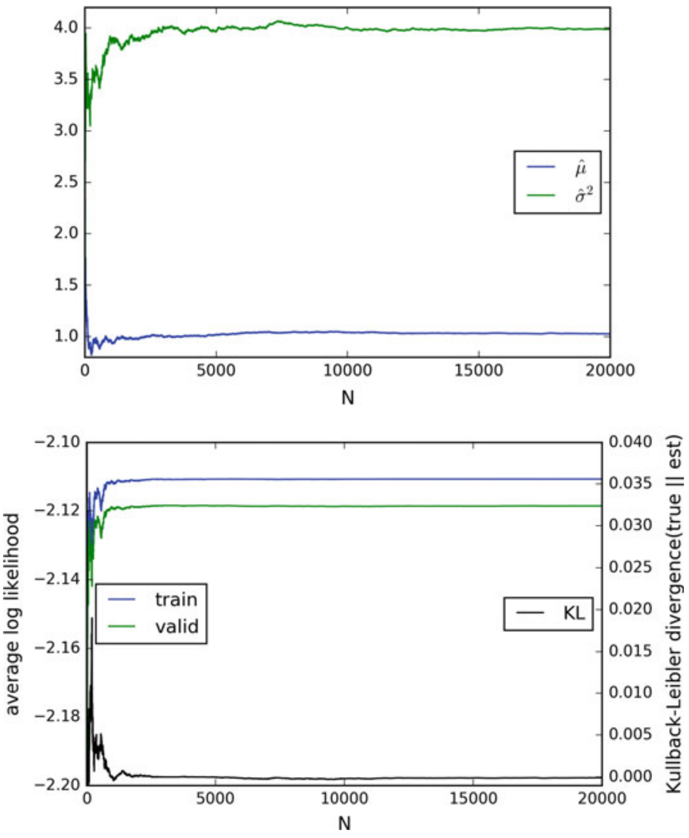


Fig. 2 Recursive estimation with exact formula: parameters estimates (*top*) - Average log-likelihood/KL divergence (*bottom*) w.r.t. the number of observations (color figure online)

recursive methods. On the right side, one may notice that green and blue curves are very similar, but the shift between them shows the training error is an optimistically biased criterion.

Figure 3 shows the estimates of μ and σ^2 through the iterations with various settings (space, fixed learning rate) but same initialization. We can immediately see that the speed convergence of SGD methods is highly dependent on the learning rate. For some good values (e.g. $\alpha = 0.0316$ for SGD on source parameters), the online method is quite competitive with recursive estimation. When, the learning rate is too low, parameter update and therefore the convergence is very slow. In contrast, when α is too large, the estimates oscillate around the global maximum. During these optimizations, the updates can lead to estimates that are outside the domain of admissible values for them. To cope with that, several strategies can be implemented: reject the update, project onto the set of admissibles values, Uzawa's method (Boyd and Vandenberghe 2004), etc.

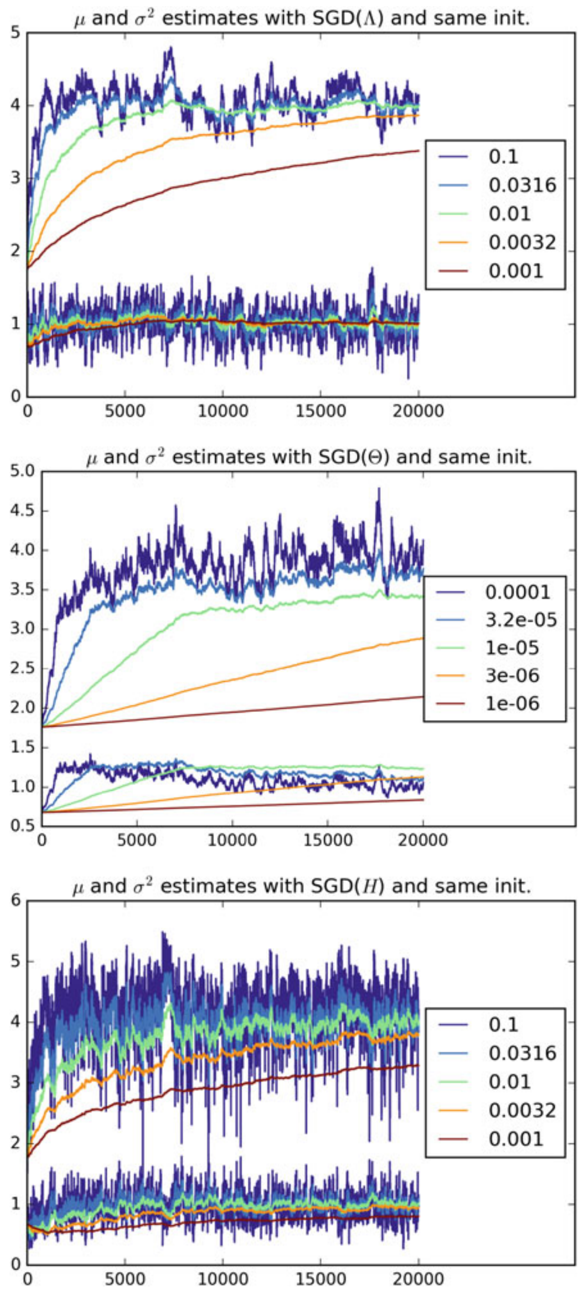


Fig. 3 Recursive estimation with SGD on source space (*left*), natural space (*middle*), expectation space (*right*) with same initialization and different $\alpha^{(N+1)}$

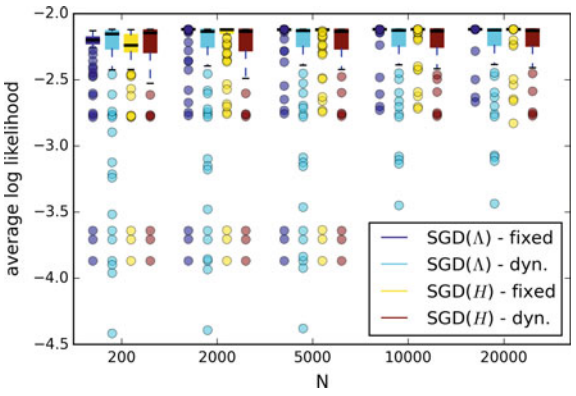


Fig. 4 Average log-likelihood over 100 runs with SGD on source space and on expectation space ($\alpha^{(N+1)} = 0.0316$ or $\alpha^{(N+1)} = n^{-0.85}$)

Remark that the SGD on H seems to be less stable. Figure 4 shows the average log-likelihood over 100 runs at different steps of the SGD on Λ and on H for a fixed learning rate or when it decreases after each iterations ($\alpha^{(N+1)} = n^{-0.85}$). The two algorithms seem to have more or less a similar behavior. Note that adapting the learning rate yields very good estimates in few iterations (black stroke indicates the median value) which are competitive over the exact MLE (cf. Fig. 2). However, this strategy seems to be too aggressive when the $\theta^{(0)}$ is far from the global optimum.

4.2 Online Learning for a Mixture of Gaussian Distributions

In this part, we focus on the online learning for a mixture of gaussian distributions. Firstly, consider the expression of Titterington's recursive EM for this case. By plugging several formulas of the appendix in Eq. 38, the following update equations are obtained:

<u>For a new observation x_{N+1},</u>

Estimate $\hat{z}^{(N)}_{N+1,j}$ and $\hat{w}^{(N+1)}_j$ using Eq. 38.

$$\boxed{\begin{aligned}\hat{\theta}^{(N+1)}_j = \hat{\theta}^{(N)}_j + \alpha^{(N+1)} \frac{2\hat{z}^{(N)}_{N+1,j}}{\hat{w}^{(N+1)}_j} & \begin{pmatrix} \hat{\theta}^{2(N)}_{1_j} + \hat{\theta}^{(N)}_{2_j} & \hat{\theta}^{(N)}_{1_j}\hat{\theta}^{(N)}_{2_j} \\ \hat{\theta}^{(N)}_{1_j}\hat{\theta}^{(N)}_{2_j} & \hat{\theta}^{2(N)}_{2_j} \end{pmatrix} \\ & \begin{pmatrix} x_{N+1} - \frac{\hat{\theta}^{(N)}_{1_j}}{2\hat{\theta}^{(N)}_{2_j}} \\ -x^2_{N+1} + \frac{\hat{\theta}^{2(N)}_{1_j}}{4\hat{\theta}^{2(N)}_{2_j}} + \frac{1}{2\hat{\theta}^{(N)}_{2_j}} \end{pmatrix}\end{aligned}} \quad (42)$$

This latter expression appears to be quite complicated. If this algorithm is applied on $\lambda = (\mu, \sigma^2)$-coordinates, the matrix I_c is almost diagonal:

$$I_c(\lambda) = blockdiag\left(\left(diag(w_1^{-1}, \ldots, w_{K-1}^{-1}) - \frac{\mathbf{1}_{K-1}{}^t\mathbf{1}_{K-1}}{w_K}\right),\right.$$
$$\left. w_1 I(\lambda_1), \ldots, w_K I(\lambda_K)\right) \tag{43}$$

where I represents in this case the Fisher information matrix on λ for the gaussian distribution (see Eq. 69). With this parametrization, the score vector given by Eq. 35 is partially composed by the following expressions:

$$\text{For } j = 1, \ldots, K \quad \frac{\partial \log f(x_{N+1}; \hat{\lambda}^{(N)})}{\partial \mu_j} = \hat{z}_{N+1,j}^{(N)} \frac{x_{N+1} - \hat{\mu}_j^{(N)}}{\hat{\sigma}_j^{2(N)}} \tag{44}$$
$$\frac{\partial \log f(x_{N+1}; \hat{\lambda}^{(N)})}{\partial \sigma_j^2} = \hat{z}_{N+1,j}^{(N)} \frac{(x_{N+1} - \hat{\mu}_j^{(N)})^2 - \hat{\sigma}_j^{2(N)}}{\hat{\sigma}_j^{4(N)}} \tag{45}$$

After few simplifications, the update equations in this coordinates system are:

$$\boxed{\hat{\mu}_j^{(N+1)} = \hat{\mu}_j^{(N)} + \alpha^{(N+1)} \frac{\hat{z}_{N+1,j}^{(N)}}{\hat{w}_j} \left(x_{N+1} - \hat{\mu}_j^{(N)}\right)} \tag{46}$$
$$\boxed{\hat{\sigma}_j^{2(N+1)} = \hat{\sigma}_j^{2(N)} + \alpha^{(N+1)} \frac{\hat{z}_{N+1,j}^{(N)}}{\hat{w}_j} \left((x_{N+1} - \hat{\mu}_j^{(N)})^2 - \hat{\sigma}_j^{2(N)}\right)} \tag{47}$$

Note the estimation of weight components remains unchanged.

In order to compare Titterington's algorithm with online EM, consider its formulation in the η-coordinates system. Recall that for a regular exponential family g_j:

$$\nabla_{\eta_j} \log g_j(x; \eta_j) = H(F_j^*)(\eta_j)(s_j(x) - \eta_j) \tag{48}$$

Moreover, since the matrix $I_c(\eta)$ is

$$I_c(\eta) = blockdiag\left(\left(diag(w_1^{-1}, \ldots, w_{K-1}^{-1}) - \frac{\mathbf{1}_{K-1}{}^t\mathbf{1}_{K-1}}{w_K}\right),\right.$$
$$\left. w_1 H(F_1^*)(\eta_1), \ldots, w_K H(F_K^*)(\eta_K)\right), \tag{49}$$

the recursion no longer requires to invert a matrix:

$$\boxed{\hat{\eta}_j^{(N+1)} = \hat{\eta}_j^{(N)} + \alpha^{(N+1)} \frac{\hat{z}_{N+1,j}^{(N)}}{\hat{w}_j^{(N)}} \left(s_j(x_{N+1}) - \hat{\eta}_j^{(N)}\right)} \tag{50}$$

Unfortunately, for all the above methods, the constraints on parameters ($\sigma_j^2 > 0$) have to be checked beforehand in order to accept the parameters update. Looking at equations Eqs. 31 and 41, we can conclude that the online EM differs in the way components parameters are updated:

$$\hat{\eta}_j^{(N+1)} = \frac{\hat{w}_j^{(N)}}{\hat{w}_j^{(N+1)}}\hat{\eta}_j^{(N)} + \alpha^{(N+1)}\left(\frac{\hat{z}_{N+1,j}^{(N)}}{\hat{w}_j^{(N+1)}} s_j(x_{N+1}) - \frac{\hat{w}_j^{(N)}}{\hat{w}_j^{(N+1)}}\hat{\eta}_j^{(N)}\right) \tag{51}$$

For more details about the convergence of these algorithms, the interested reader is referred to Cappé and Moulines (2009) for further information.

To illustrate these algorithms, two experiments on synthetic datasets are provided (see Fig. 5). Their respective parameters are:
Dataset 1 : $(w_1 = 0.5, \mu_1 = 0, \sigma_1^2 = 1), (w_2 = 0.5, \mu_2 = 4, \sigma_2^2 = 4)$
Dataset 2 : $(w_1 = 0.25, \mu_1 = 0.25, \sigma_1^2 = 0.15), (w_2 = 0.65, \mu_2 = -1, \sigma_2^2 = 0.4)$
$(w_3 = 0.1, \mu_3 = -0.5, \sigma_3^2 = 0.6)$
All the methods were initialized with same parameters values coming from the k-means algorithm.

The policy for the learning rate is also identical: $\alpha^{(N)} = \left(\frac{1}{N_0+N}\right)^{0.7}$ where N_0 is the number of observations used for k-means. The criteria used to evaluated the results are the average log-likelihood and the Kullback–Liebler divergence (KL). Since there is no closed form to evaluate this divergence, we rely on numerical integration which is reasonably fast and accurate in 1d. Figure 6 reports the results of all estimators on the two datasets. Additionally, Figs. 7 and 8 illustrates the best estimates KL resulting components.

As expected, since the dataset 1 contains two relatively separated components, the estimation converges very quickly for all methods except for the Recursive EM on θ-coordinates. For this experiment, we observe that most of the first updates are rejected due to the constraints on parameters ($\theta_2 > 0 \equiv \sigma^2 > 0$). Later in the recursion, the learning rate has decreased and the updates do not violate the constraints. Undoubtedly, and as also observed in first experiment, the choice of the learning rate policy should be different on natural parameter space. Also, one can observe that some methods are trapped in different local minima even if their initialization are

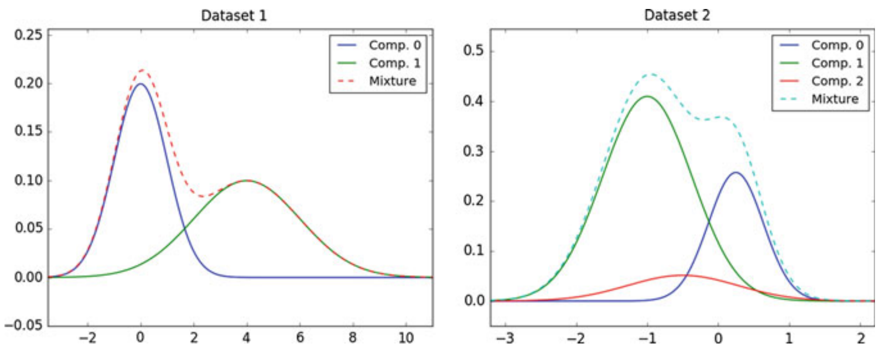
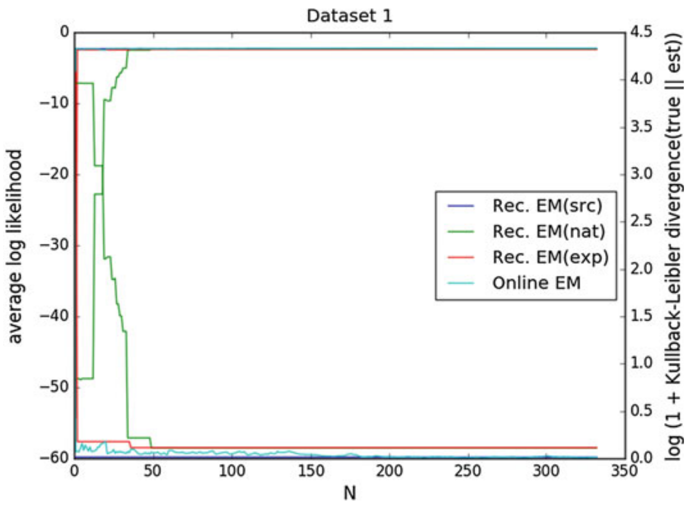


Fig. 5 Synthetic datasets



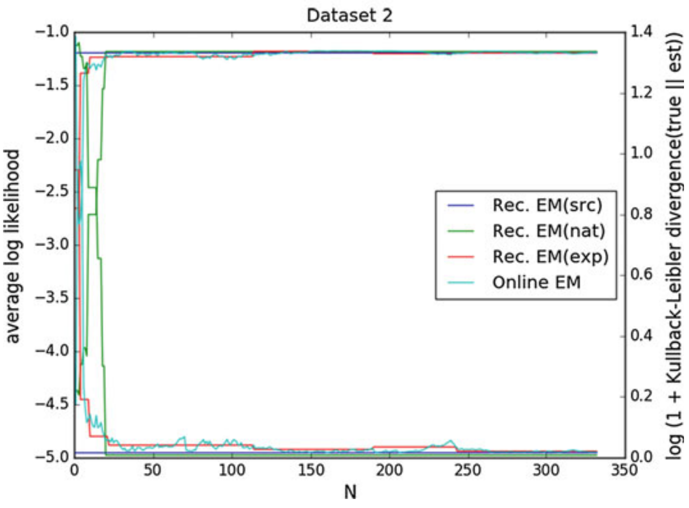


Fig. 6 Average log-likelihood and Kullback–Leibler for all estimators

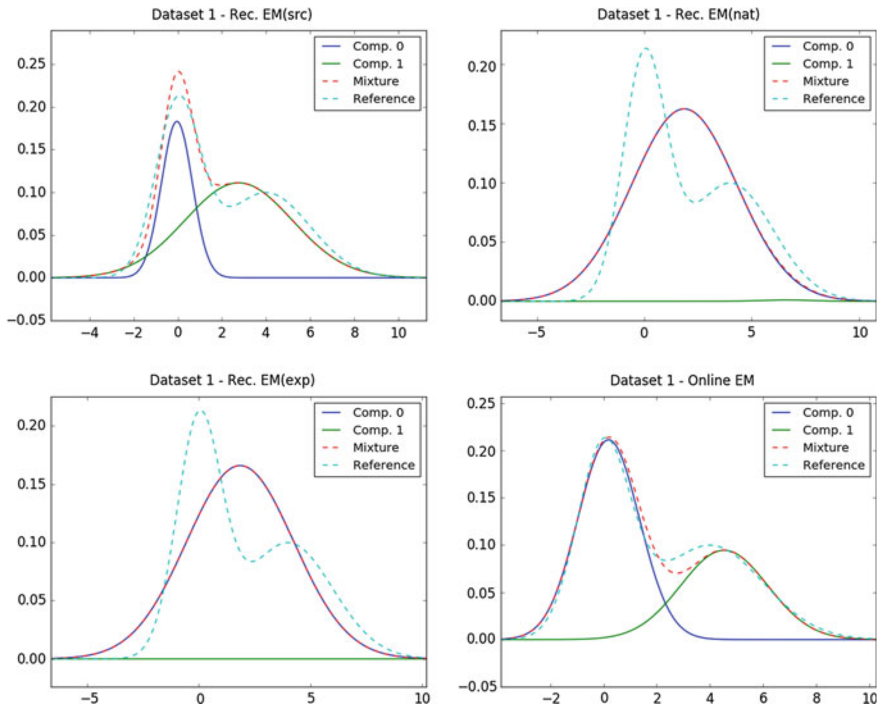


Fig. 7 Dataset 1: Best estimates w.r.t. to KL divergence

the same (see Fig. 7). For dataset 2, we remark the constraints prevents some updates for Recursive EM on natural and expectation parameters. Despite this, the mixture estimate is very good (see Recursive EM(natural) on Fig. 8).

As a conclusion of these experiments, Recursive EM on η-coordinates and online EM do not require the computation and the inversion of matrix. This is a very appealing property especially when the components have a more complicated parametric distribution (e.g. Wishart distributions Saint-Jean and Nielsen 2014). But in practice, this provides only easier to implement methods and does not guarantee better estimates. Since online EM makes a stochastic approximation of the E-Step of EM, the constraints on parameters are automatically guaranteed by the maximization step which is particularly efficient for exponential families.

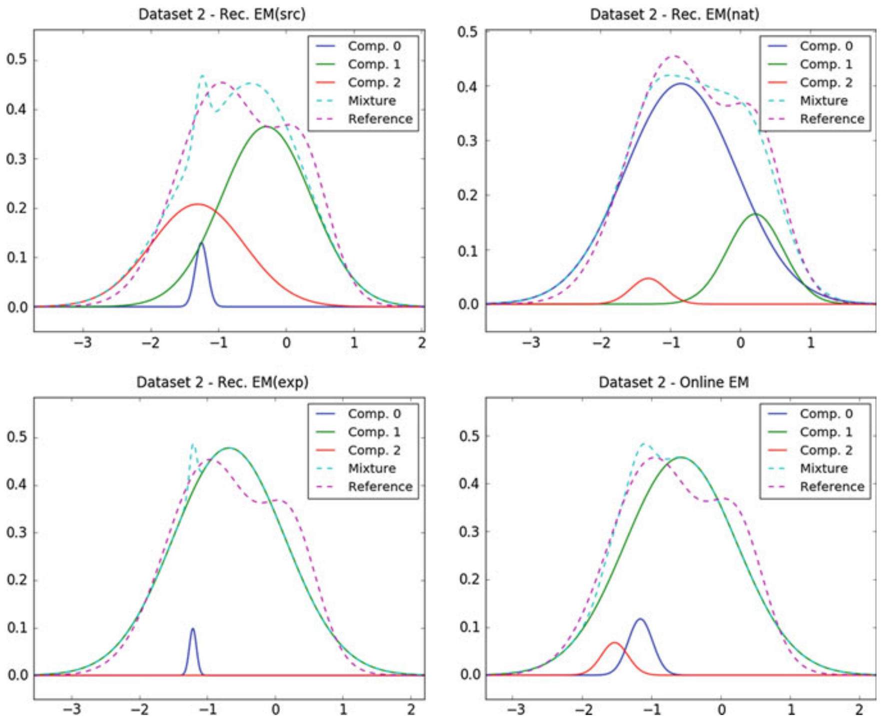


Fig. 8 Dataset 2: Best estimates w.r.t. to KL divergence

5 Conclusion

This paper addresses the problem of online learning of finite statistical mixtures with a special focus on distribution components belonging to the exponential families. Many details to compare Recursive EM and online EM from the practical point of view are given. The presented methods are fast since they require only one pass over the data stream. However, there is still room for improvement, especially for the Recursive EM method which is roughly a classical second-order stochastic gradient ascent. More recent optimization methods are described in the paper and leads to overcome the difficulty to choose an adequate policy for the learning rate. We might have also mentioned the incremental EM by Neal and Hinton (1999) which shares many properties with the online EM (partial E-Step). Further speed increase may be achieved by using distributed computing on a cluster of machines by aggregating partial sums of sufficient statistics (see Liu and Ihler 2014) since the statistical estimation is a decomposable problem.

Appendices

Univariate Gaussian Distribution as an Exponential Family

Canonical Decomposition and *F*

$$
\begin{aligned}
f(x;\mu,\sigma^2) &= \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}\\
&= \exp\left\{-\frac{1}{2\sigma^2}(x^2-2x\mu+\mu^2) - \frac{1}{2}\log\left(2\pi\sigma^2\right)\right\}\\
&= \exp\left\{\langle\frac{1}{2\sigma^2}, -x^2\rangle + \langle\frac{\mu}{\sigma^2}, x\rangle - \frac{\mu^2}{2\sigma^2} - \frac{1}{2}\log\left(2\pi\sigma^2\right)\right\}
\end{aligned}
$$

In the sequel, the vector of source parameters is denoted $\lambda = (\mu, \sigma^2)$. One may recognize the canonical form of an exponential family

$$f(x;\theta) = \exp\{<\theta, s(x)> + k(x) - F(\theta)\}$$

by setting $\theta = (\theta_1, \theta_2)$ with

$$\theta_1 = \frac{\mu}{\sigma^2} \Longleftrightarrow \mu = \frac{\theta_1}{2\theta_2} \tag{52}$$

$$\theta_2 = \frac{1}{2\sigma^2} \Longleftrightarrow \sigma^2 = \frac{1}{2\theta_2} \tag{53}$$

$$s(x) = (x, -x^2) \tag{54}$$

$$k(x) = 0 \tag{55}$$

$$
\begin{aligned}
f(x;\theta_1,\theta_2) &= \exp\left\{\langle\theta_2, -x^2\rangle + \langle\theta_1, x\rangle - \frac{1}{2}\frac{(\theta_1/2\theta_2)^2}{1/2\theta_2} - \frac{1}{2}\log(2\pi/2\theta_2)\right\}\\
&= \exp\left\{\langle\theta_2, -x^2\rangle + \langle\theta_1, x\rangle - \frac{\theta_1^2}{4\theta_2} - \frac{1}{2}\log(\pi) + \frac{1}{2}\log\theta_2\right\}
\end{aligned}
$$

with the log normalizer F as

$$F(\theta_1,\theta_2) = \frac{\theta_1^2}{4\theta_2} + \frac{1}{2}\log(\pi) - \frac{1}{2}\log\theta_2 \tag{56}$$

Gradient of the Log-Normalizer

The gradient of the log-normalizer is given by:

$$\frac{\partial F}{\partial \theta_1}(\theta_1, \theta_2) = \frac{\theta_1}{2\theta_2} \tag{57}$$

$$\frac{\partial F}{\partial \theta_2}(\theta_1, \theta_2) = -\frac{\theta_1^2}{4\theta_2^2} - \frac{1}{2\theta_2} \tag{58}$$

In order to get the dual coordinate system $\eta = (\eta_1, \eta_2)$, the following set of equations has to be inverted:

$$\eta_1 = \frac{\theta_1}{2\theta_2} \tag{59}$$

$$\eta_2 = -\frac{\theta_1^2}{4\theta_2^2} - \frac{1}{2\theta_2} \tag{60}$$

By plugging the first equation into the second one, it follows:

$$\eta_2 = -\eta_1^2 - \frac{1}{2\theta_2} \Longleftrightarrow \theta_2 = -\frac{1}{2(\eta_1^2 + \eta_2)} = \frac{\partial F^*}{\partial \eta_2}(\eta_1, \eta_2) \tag{61}$$

$$\theta_1 = 2\theta_2\eta_1 = -\frac{\eta_1}{(\eta_1^2 + \eta_2)} = \frac{\partial F^*}{\partial \eta_1}(\eta_1, \eta_2) \tag{62}$$

Formulas are even simpler regarding the source parameters since we know that

$$\eta_1 = \mathbb{E}[X] = \mu \Longleftrightarrow \quad \mu = \eta_1 \tag{63}$$

$$\eta_2 = \mathbb{E}[-X^2] = -\left\{\mu^2 + \sigma^2\right\} \Longleftrightarrow \quad \sigma^2 = -\left\{\eta_1^2 + \eta_2\right\} \tag{64}$$

In order to compute F^*, we simply have to reuse our previous results in

$$F^*(H) = \langle (\nabla F)^{-1}(H), H \rangle - F((\nabla F)^{-1}(H))$$

and obtain the following expression

$$\begin{aligned} F^*(\eta_1, \eta_2) = & \left\langle -\frac{\eta_1}{(\eta_1^2 + \eta_2)}, \eta_1 \right\rangle + \left\langle -\frac{1}{2(\eta_1^2 + \eta_2)}, \eta_2 \right\rangle \\ & - \left\{ \frac{\left(-\frac{\eta_1}{(\eta_1^2+\eta_2)}\right)^2}{4\left(-\frac{1}{2(\eta_1^2+\eta_2)}\right)} + \frac{1}{2}\log(\pi) - \frac{1}{2}\log\left(-\frac{1}{2(\eta_1^2 + \eta_2)}\right) \right\} \end{aligned}$$

$$\begin{aligned}
&= -\frac{\eta_1^2}{(\eta_1^2+\eta_2)} - \frac{\eta_2}{2(\eta_1^2+\eta_2)} + \frac{\frac{\eta_1^2}{(\eta_1^2+\eta_2)^2}}{\frac{2}{(\eta_1^2+\eta_2)}}\\
&- \frac{1}{2}\log(\pi) + \frac{1}{2}\log((-2(\eta_1^2+\eta_2))^{-1})\\
&= -\frac{\eta_1^2}{2(\eta_1^2+\eta_2)} - \frac{\eta_2}{2(\eta_1^2+\eta_2)} - \frac{1}{2}\log(\pi) - \frac{1}{2}\log(-2(\eta_1^2+\eta_2))\\
&= -\frac{1}{2} - \frac{1}{2}\log(\pi) - \frac{1}{2}\log(-2(\eta_1^2+\eta_2))\\
&= -\frac{1}{2}\log(e\pi) - \frac{1}{2}\log(-2(\eta_1^2+\eta_2))
\end{aligned}$$

The hessians $H(F)$, $H(F^*)$ of respectively F and F^* are

$$H(F)(\theta_1,\theta_2) = \begin{pmatrix} \frac{1}{2\theta_2} & -\frac{\theta_1}{2\theta_2^2} \\ -\frac{\theta_1}{2\theta_2^2} & \frac{\theta_1^2+\theta_2}{2\theta_2^3} \end{pmatrix} \tag{65}$$

$$H(F^*)(\eta_1,\eta_2) = \begin{pmatrix} \frac{\eta_1^2-\eta_2}{(\eta_1^2+\eta_2)^2} & \frac{\eta_1}{(\eta_1^2+\eta_2)^2} \\ \frac{\eta_1}{(\eta_1^2+\eta_2)^2} & \frac{1}{2(\eta_1^2+\eta_2)^2} \end{pmatrix} \tag{66}$$

Since the univariate normal distribution is an exponential family, the Kullback–Leibler divergence is a Bregman divergence for F^* on expectation parameters:

$$\begin{aligned}
KL(\mathcal{N}(\mu_p,\sigma_p^2)||\mathcal{N}(\mu_q,\sigma_q^2)) &= B_{F^*}(\eta_p : \eta_q)\\
&= F^*(\eta_p) - F^*(\eta_q) - \langle \eta_p - \eta_q, \nabla F^*(\eta_q)\rangle
\end{aligned}$$

After calculations, it follows:

$$B_F^*(\eta_p : \eta_q) = \frac{1}{2}\left(\log\left(\frac{\eta_{1_q}^2+\eta_{2_q}}{\eta_{1_p}^2+\eta_{2_p}}\right) + \frac{2(\eta_{1_p}-\eta_{1_q})\eta_{1_q}}{(\eta_{1_q}^2+\eta_{2_q})} + \frac{\eta_{2_p}-\eta_{2_q}}{(\eta_{1_q}^2+\eta_{2_q})}\right) \tag{67}$$

A simple rewrite of it with the source parameters leads to the known closed form:

$$\begin{aligned}
\frac{1}{2}\left(\log\left(\frac{\eta_{1_q}^2+\eta_{2_q}}{\eta_{1_p}^2+\eta_{2_p}}\right) + \frac{2(\eta_{1_p}-\eta_{1_q})\eta_{1_q}}{(\eta_{1_q}^2+\eta_{2_q})} + \frac{\eta_{2_p}-\eta_{2_q}}{(\eta_{1_q}^2+\eta_{2_q})}\right) &=\\
\frac{1}{2}\left(\log\left(\frac{\eta_{1_q}^2+\eta_{2_q}}{\eta_{1_p}^2+\eta_{2_p}}\right) + \frac{(\eta_{1_p}^2+\eta_{2_p}) - (\eta_{1_p}-\eta_{1_q})^2 - (\eta_{1_q}^2+\eta_{2_q})}{(\eta_{1_q}^2+\eta_{2_q})}\right) &=\\
\frac{1}{2}\left(\log\left(\frac{\sigma_q^2}{\sigma_p^2}\right) + \frac{\sigma_p^2}{\sigma_q^2} + \frac{(\mu_p-\mu_q)^2}{\sigma_q^2} - 1\right)
\end{aligned} \tag{68}$$

The Fisher information matrix $I(\lambda)$ is obtained by computing the expectation of the product of Fisher score and its transposition:

$$\begin{aligned} I(\lambda) &\overset{def}{=} \mathbb{E}\left[\nabla_\lambda \log f(x;\lambda).\nabla_\lambda \log f(x;\lambda)^T\right] \\ &= \mathbb{E}\left[\begin{pmatrix} \frac{x-\mu}{\sigma^2} \\ \frac{(x-\mu)^2-\sigma^2}{2\sigma^4} \end{pmatrix} \cdot \begin{pmatrix} \frac{x-\mu}{\sigma^2} & \frac{(x-\mu)^2-\sigma^2}{2\sigma^4} \end{pmatrix}\right] \\ &= \begin{pmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{2\sigma^4} \end{pmatrix}. \end{aligned} \tag{69}$$

By change in coordinates or direct computation, the Fisher information matrix is also:

$$I(\theta) = H(F)(\theta) = \begin{pmatrix} \frac{1}{2\theta_2} & -\frac{\theta_1}{2\theta_2^2} \\ -\frac{\theta_1}{2\theta_2^2} & \frac{\theta_1^2+\theta_2}{2\theta_2^3} \end{pmatrix} \text{ and } I(\eta) = \frac{1}{(\eta_1^2+\eta_2)^2}\begin{pmatrix} (\eta_1^2-\eta_2) & \eta_1 \\ \eta_1 & \frac{1}{2} \end{pmatrix} \tag{70}$$

Multivariate Gaussian Distribution as an Exponential Family

Canonical Decomposition and *F*

$$\begin{aligned} f(x;\mu,\Sigma) &= \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left\{-\frac{{}^t(x-\mu)\Sigma^{-1}(x-\mu)}{2}\right\} \\ &= \exp\left\{-\frac{{}^t x\Sigma^{-1}x - {}^t\mu\Sigma^{-1}x - {}^t x\Sigma^{-1}\mu + {}^t\mu\Sigma^{-1}\mu}{2} - \log\left((2\pi)^{d/2}|\Sigma|^{1/2}\right)\right\} \\ &= \exp\left\{-\frac{tr({}^t x\Sigma^{-1}x) - \langle {}^t\Sigma^{-1}\mu, x\rangle - \langle x, \Sigma^{-1}\mu\rangle + \langle {}^t\Sigma^{-1}\mu, \Sigma\Sigma^{-1}\mu\rangle}{2} - \log\left(\pi^{d/2}|2\Sigma|^{1/2}\right)\right\} \end{aligned}$$

Due to the cyclic property of the trace and to the symmetry of Σ^{-1}, it follows:

$$\begin{aligned} f(x;\mu,\Sigma) &= \exp\left\{tr\left({}^t\left(\frac{1}{2}\Sigma^{-1}\right)(-x^t x)\right) + \langle\Sigma^{-1}\mu, x\rangle - \frac{1}{2}\langle\Sigma^{-1}\mu, \Sigma\Sigma^{-1}\mu\rangle - \frac{d}{2}\log(\pi) - \frac{1}{2}\log|2\Sigma|\right\} \\ &= \exp\left\{\langle\frac{1}{2}\Sigma^{-1}, -x^t x\rangle_F + \langle\Sigma^{-1}\mu, x\rangle - \frac{1}{4}{}^t(\Sigma^{-1}\mu)2\Sigma(\Sigma^{-1}\mu) - \frac{d}{2}\log(\pi) - \frac{1}{2}\log|2\Sigma|\right\} \end{aligned}$$

where $\langle\cdot,\cdot\rangle_F$ is the Frobenius scalar product. One may recognize the canonical form of an exponential family

$$f(x;\Theta) = \exp\{< \Theta, t(x) > + k(x) - F(\Theta)\}$$

by setting:

$$\Theta = (\theta_1, \theta_2)$$

$$\theta_1 = \Sigma^{-1}\mu \iff \mu = \frac{1}{2}\theta_2^{-1}\theta_1 \tag{71}$$

$$\theta_2 = \frac{1}{2}\Sigma^{-1} \iff \Sigma = \frac{1}{2}\theta_2^{-1} \tag{72}$$

$$t(x) = (x, -x^t x) \tag{73}$$

$$k(x) = 0 \tag{74}$$

$$f(x;\theta_1,\theta_2) = \exp\left\{\langle\theta_2, -x^t x\rangle_F + \langle\theta_1, x\rangle - \frac{1}{4}{}^t\theta_1\theta_2^{-1}\theta_1 - \frac{d}{2}\log(\pi) + \frac{1}{2}\log|\theta_2|\right\} \tag{75}$$

with the log normalizer F:

$$F(\theta_1,\theta_2) = \frac{1}{4}{}^t\theta_1\theta_2^{-1}\theta_1 + \frac{d}{2}\log(\pi) - \frac{1}{2}\log|\theta_2| \tag{76}$$

Gradient of the Log-Normalizer

By applying the following formulas from the matrix cookbook (Petersen and Pedersen 2012)

identity 57

$$\frac{\partial \log|X|}{\partial X} = ({}^tX)^{-1} = {}^t(X^{-1})$$

identity 61

$$\frac{\partial {}^t a X^{-1} b}{\partial X} = -{}^tX^{-1}a{}^tbX^{-1}$$

identity 81

$$\frac{\partial {}^t x B x}{\partial x} = (B + {}^tB)x$$

the gradient of the log-normalizer is given by:

$$\frac{\partial F}{\partial \theta_1}(\theta_1,\theta_2) = \frac{1}{4}(\theta_2^{-1} + {}^t\theta_2^{-1})\theta_1 = \frac{1}{2}\theta_2^{-1}\theta_1 \tag{77}$$

$$\frac{\partial F}{\partial \theta_2}(\theta_1,\theta_2) = -\frac{1}{4}{}^t\theta_2^{-1}\theta_1{}^t\theta_1\theta_2^{-1} - \frac{1}{2}{}^t\theta_2^{-1} = -\left(\frac{1}{2}\theta_2^{-1}\theta_1\right)^t\left(\frac{1}{2}\theta_2^{-1}\theta_1\right) - \frac{1}{2}\theta_2^{-1} \tag{78}$$

In order to emphasize the coherence of these formulas, recall that the gradient of the log-normalizer corresponds the expectation of the sufficient statistics:

$$\mathbb{E}[x] = \mu \qquad \equiv \frac{1}{2}\theta_2^{-1}\theta_1 \tag{79}$$

$$\mathbb{E}[-x^t x] = -\mathbb{E}[x^t x] = -\mu^t \mu - \Sigma \quad \equiv -\left(\frac{1}{2}\theta_2^{-1}\theta_1\right)^t \left(\frac{1}{2}\theta_2^{-1}\theta_1\right) - \frac{1}{2}\theta_2^{-1} \tag{80}$$

Last equation comes from the expansion of $\mathbb{E}[(x-\mu)^t(x-\mu)]$.

Convex Conjugate G of F and Its Gradient

In order to get the dual coordinate system $H = (\eta_1, \eta_2)$, the following set of equations has to be inverted:

$$\eta_1 = \frac{1}{2}\theta_2^{-1}\theta_1 \tag{81}$$

$$\eta_2 = -\left(\frac{1}{2}\theta_2^{-1}\theta_1\right)^t \left(\frac{1}{2}\theta_2^{-1}\theta_1\right) - \frac{1}{2}\theta_2^{-1} \tag{82}$$

By plugging the first equation into the second one, it follows

$$\eta_2 = -{\eta_1}^t\eta_1 - \frac{1}{2}\theta_2^{-1} \iff \theta_2 = \frac{1}{2}(-{\eta_1}^t\eta_1 - \eta_2)^{-1} = \frac{\partial G}{\partial \eta_2}(\eta_1, \eta_2) \tag{83}$$

and

$$\theta_1 = 2\theta_2\eta_1 = (-{\eta_1}^t\eta_1 - \eta_2)^{-1}\eta_1 = \frac{\partial G}{\partial \eta_1}(\eta_1, \eta_2) \tag{84}$$

Formulas are even simpler regarding the source parameters since we know from Eqs. 79 and 80 that

$$\eta_1 = \mu \iff \qquad \mu = \eta_1 \tag{85}$$

$$\eta_2 = -\mu^t\mu - \Sigma \iff \qquad \Sigma = -{\eta_1}^t\eta_1 - \eta_2 \tag{86}$$

In order to compute $G := F^*$, we simply have to reuse our previous results in

$$G(H) = \langle (\nabla F)^{-1}(H), H\rangle - F((\nabla F)^{-1}(H))$$

and obtain the following expression

$$\begin{aligned} G(\eta_1, \eta_2) = {} & \langle (-{\eta_1}^t\eta_1 - \eta_2)^{-1}\eta_1, \eta_1\rangle + \langle \frac{1}{2}(-{\eta_1}^t\eta_1 - \eta_2)^{-1}, \eta_2\rangle_F \\ & - \frac{1}{4}{}^t((-{\eta_1}^t\eta_1 - \eta_2)^{-1}\eta_1)2(-{\eta_1}^t\eta_1 - \eta_2)(-{\eta_1}^t\eta_1 - \eta_2)^{-1}\eta_1 \end{aligned}$$

$$
\begin{aligned}
&- \frac{d}{2}\log(\pi) + \frac{1}{2}\log|\frac{1}{2}(-\eta_1{}^t\eta_1 - \eta_2)^{-1}| \\
&= {}^t\eta_1(-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_1 + \frac{1}{2}tr({}^t(-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_2) \\
&- \frac{1}{2}{}^t\eta_1{}^t(-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_1 \\
&- \frac{d}{2}\log(\pi) + \frac{1}{2}\log|(2(-\eta_1{}^t\eta_1 - \eta_2))^{-1}| \\
&= \frac{1}{2}{}^t\eta_1(-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_1 + \frac{1}{2}tr((-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_2) \\
&- \frac{d}{2}\log(\pi) - \frac{1}{2}\log|2(-\eta_1{}^t\eta_1 - \eta_2)| \\
&= \frac{1}{2}\left(tr((-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_1{}^t\eta_1) + tr((-\eta_1{}^t\eta_1 - \eta_2)^{-1}\eta_2)\right) \\
&- \frac{d}{2}\log(\pi) - \frac{1}{2}\log|2(-\eta_1{}^t\eta_1 - \eta_2)| \\
&= -\frac{1}{2}tr((-\eta_1{}^t\eta_1 - \eta_2)^{-1}(-\eta_1{}^t\eta_1 - \eta_2)) - \frac{d}{2}\log(\pi) \\
&- \frac{1}{2}\log|2(-\eta_1{}^t\eta_1 - \eta_2)| \\
&= -\frac{1}{2}tr(I_d) - \frac{d}{2}\log(\pi) - \frac{1}{2}\log|2(-\eta_1{}^t\eta_1 - \eta_2)| \\
&= -\frac{d}{2}\log(e\pi) - \frac{1}{2}\log|2(-\eta_1{}^t\eta_1 - \eta_2)|
\end{aligned}
$$

Let us rewrite this expression with source parameters:

$$G(\mu, \Sigma) = -\frac{d}{2}\log(e\pi) - \frac{1}{2}\log|2\Sigma| \tag{87}$$

Kullback–Leibler Divergence

First recall that the Kullback–Leibler divergence between two p.d.f. p and q is

$$KL(p||q) = \int p(x)\log\frac{p(x)}{q(x)}dx$$

For two multivariate normal distributions, it is known in closed form

$$KL(\mathcal{N}(\mu_p, \Sigma_p)||\mathcal{N}(\mu_q, \Sigma_q)) = \frac{1}{2}\left(\log\left(\frac{|\Sigma_q|}{|\Sigma_p|}\right) + tr(\Sigma_q^{-1}\Sigma_p) + {}^t(\mu_q - \mu_p)\Sigma_q^{-1}(\mu_q - \mu_p) - d\right) \tag{88}$$

Since the multivariate normal distribution is an E.F., the same result must be obtained using the bregman divergence for G on expectation parameters H_p and H_q:

$$KL(\mathcal{N}(\mu_p, \Sigma_p)||\mathcal{N}(\mu_q, \Sigma_q)) = B_G(H_p||H_q) = G(H_p) - G(H_q) - \langle H_p - H_q, \nabla G(H_q)\rangle$$

$$\begin{aligned}
G(H_p) - G(H_q) &= -\frac{d}{2}\log(e\pi) - \frac{1}{2}\log|-2({\eta_1}_p{}^t{\eta_1}_p + {\eta_2}_p)| \\
&+ \frac{d}{2}\log(e\pi) + \frac{1}{2}\log|-2({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)| \\
&= \frac{1}{2}\log\frac{|-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)|}{|-({\eta_1}_p{}^t{\eta_1}_p + {\eta_2}_p)|} \\
-\langle H_p - H_q, \nabla G(H_q)\rangle &= -\langle {\eta_1}_p - {\eta_1}_q, -({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}{\eta_1}_q\rangle \\
&- tr\left({}^t({\eta_2}_p - {\eta_2}_q)\left(-\frac{1}{2}({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}\right)\right) \\
&= {}^t{\eta_1}_p({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}{\eta_1}_q - {}^t{\eta_1}_q({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}{\eta_1}_q \\
&- \frac{1}{2}tr({}^t{\eta_2}_p(-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1})) + \frac{1}{2}tr({}^t{\eta_2}_q(-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}))
\end{aligned}$$

In order to go further, we can express these two formulas using μ and $\Sigma^{-1} = (-\eta_1{}^t\eta_1 - \eta_2)^{-1} = -(\eta_1{}^t\eta_1 + \eta_2)^{-1}$ (cf. Eq. 86):

$$\frac{1}{2}\log\frac{|-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)|}{|-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)|} = \frac{1}{2}\log\frac{|\Sigma_q|}{|\Sigma_p|}$$

$$\begin{aligned}
{}^t{\eta_1}_p({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}{\eta_1}_q &= -{}^t\mu_p\Sigma_q^{-1}\mu_q \\
-{}^t{\eta_1}_q({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1}{\eta_1}_q &= {}^t\mu_q\Sigma_q^{-1}\mu_q
\end{aligned}$$

$$\begin{aligned}
-\frac{1}{2}tr({}^t{\eta_2}_p(-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1})) &= \frac{1}{2}tr((\mu_p{}^t\mu_p + \Sigma_p)\Sigma_q^{-1}) \\
&= \frac{1}{2}tr(\mu_p{}^t\mu_p\Sigma_q^{-1}) + \frac{1}{2}tr(\Sigma_p\Sigma_q^{-1}) \\
&= \frac{1}{2}{}^t\mu_p\Sigma_q^{-1}\mu_p + \frac{1}{2}tr(\Sigma_q^{-1}\Sigma_p) \\
+\frac{1}{2}tr({}^t{\eta_2}_q(-({\eta_1}_q{}^t{\eta_1}_q + {\eta_2}_q)^{-1})) &= -\frac{1}{2}tr((\mu_q{}^t\mu_q + \Sigma_q)\Sigma_q^{-1}) \\
&= -\frac{1}{2}tr(\mu_q{}^t\mu_q\Sigma_q^{-1}) - \frac{1}{2}tr(\Sigma_q\Sigma_q^{-1}) \\
&= -\frac{1}{2}{}^t\mu_q\Sigma_q^{-1}\mu_q - \frac{1}{2}d
\end{aligned}$$

By summing up of these terms, the standard formula for KL divergence is recovered:

$$\begin{aligned} KL(\mathcal{N}(\mu_p, \Sigma_p)||\mathcal{N}(\mu_q, \Sigma_q)) = \frac{1}{2}\log\frac{|\Sigma_q|}{|\Sigma_p|} - {}^t\mu_p\Sigma_q^{-1}\mu_q + {}^t\mu_q\Sigma_q^{-1}\mu_q + \\ \frac{1}{2}{}^t\mu_p\Sigma_q^{-1}\mu_p + \frac{1}{2}tr(\Sigma_q^{-1}\Sigma_p) - \frac{1}{2}{}^t\mu_q\Sigma_q^{-1}\mu_q - \frac{1}{2}d \\ = \frac{1}{2}\Bigg(\log\frac{|\Sigma_q|}{|\Sigma_p|} + tr(\Sigma_q^{-1}\Sigma_p) - d - \\ \left\{2{}^t\mu_p\Sigma_q^{-1}\mu_q - 2{}^t\mu_q\Sigma_q^{-1}\mu_q -{}^t\mu_p\Sigma_q^{-1}\mu_p + {}^t\mu_q\Sigma_q^{-1}\mu_q\right\}\Bigg) \\ = \frac{1}{2}\left(\log\frac{|\Sigma_q|}{|\Sigma_p|} + tr(\Sigma_q^{-1}\Sigma_p) - {}^t(\mu_p - \mu_q)\Sigma_q^{-1}(\mu_p - \mu_q) - d\right) \end{aligned}$$

References

Amari, S. (1997). Neural learning in structured parameter spaces — Natural Riemannian gradient. *Neural Information Processing Society (NIPS)*, *9*, 127–133.

Amari, S. (1998). Natural gradient works efficiently in learning. *Neural Computation*, *10*(2), 251–276.

Amari, S. (2016). *Information geometry and its applications*. Applied Mathematical Sciences. Japan: Springer.

Banerjee, A., Merugu, S., Dhillon, I. S., & Ghosh, J. (2005). Clustering with Bregman divergences. *Journal of Machine Learning Research*, *6*, 1705–1749.

Bogdan, K., & Bogdan, M. (2000). On existence of maximum likelihood estimators in exponential families. *Statistics*, *34*(2), 137–149.

Bottou, L. (1998). Online algorithms and stochastic approximations. In S. David (Ed.), *Online learning and neural networks*. Cambridge: Cambridge University Press.

Bottou, L., & Bousquet, O. (2011). In S. Sra, S. Nowozin, & S. J. Wright (Eds.), *The tradeoffs of large scale learning* (pp. 351–368). Cambridge: MIT Press.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge: Cambridge University Press.

Cappé, O., & Moulines, E. (2009). On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society. Series B (Methodological)*, *71*(3), 593–613.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*, 1–38.

Liu, Q., & Ihler, A. T. (2014). Distributed estimation, information loss and exponential families. *Advances in Neural Information Processing Systems*, *27*, 1098–1106.

Miura, K. (2011). An introduction to maximum likelihood estimation in information geometry. *Interdisciplinary Information Sciences*, *17*(3), 155–174.

Neal, R. M., & Hinton, G. E. (1999). A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan (Ed.), *Learning in graphical models* (pp. 355–368). Cambridge: MIT Press.

Nielsen, F., & Garcia, V. (2009). Statistical exponential families: A digest with flash cards. arXiv:0911.4863.

Petersen, K. B., & Pedersen, M. S. (2012). The matrix cookbook. http://www2.imm.dtu.dk/pubdb/p.php?3274.

Polyak, B. T., & Juditsky, A. B. (1992). Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, *30*(4), 838–855.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, *22*(3), 400–407.

Saint-Jean, C., & Nielsen, F. (2014). Hartigan's method for k-MLE: Mixture modeling with Wishart distributions and its application to motion retrieval. *Geometric theory of information* (pp. 301–330). New York: Springer.

Sculley, D. (2010). Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web* (pp. 1177–1178).

Shalev-Shwartz, S. (2011). Online learning and online convex optimization. *Foundations and Trends Machine Learning*, *4*(2), 107–194.

Titterington, D. M. (1984). Recursive parameter estimation using incomplete data. *Journal of the Royal Statistical Society. Series B (Methodological)*, *46*(2), 257–267.