Méthodes de géométrie de l'information pour les modèles de mélange

Olivier Schwander

Thèse présentée en vue de l'obtention du titre de

Docteur de l'École polytechnique

par Olivier Schwander

Information-geometric methods for mixture models

soutenue le **15 octobre 2013**

devant le jury composé de

Rapporteurs:

Ali Taylan Cemgil	Bogazici University
Alfred Hero	University of Michigan
Ali Mohammad-Djafari	CNRS - Supélec - Univ. Paris-Sud

Directeur:

Frank Nielsen	École polytechnique - Sony Computer Science Laboratories

Examinateurs:

Richard Nock	Univ. des Antilles et de la Guyane
Gabriel Peyré	CNRS - Univ. Paris-Dauphine
Michalis Vazirgiannis	École Polytechnique

Contents

List of Figures

List of Tables

Chapitre 1

Remerciements NONUMBER

J'aimerais tout d'abord remercier Frank Nielsen, mon directeur de thèse, qui a su m'encadrer au cours de ces dernières années : malgré son éloignement, j'ai eu la chance d'avoir beaucoup de discussions avec lui, face à face, trop peu souvent, mais aussi par courriers électroniques (auxquels il a toujours répondu bien plus vite que moi).

Toujours sur le plan scientifique, j'aimerais remercier tous les participants réguliers du séminaire Léon Brillouin, qui ont fait de ce séminaire à l'IRCAM un lieu d'échange et de partage de connaissances capital pour le domaine de la géométrie de l'information. Mes quelques coauteurs m'ont apporté des ouvertures insoupçonnées : Julie Bernauer, Yannick Berthoumieu, Michael Levitt, Aurélien Schutz, Adeline Sim.

De façon plus terre à terre, il me faut remercier l'équipe de Combinatoire du LIX, qui malgré ma relative indépendance m'a toujours fait bénéficier de son soutien, notamment Gilles Schaeffer et Maks Ovsjanikov. Je n'oublie pas non plus mon équipe d'adoption, MAX, pour tous les bons moment passés avec eux (mention spéciale pour les tableaux en ardoise de Cauchy) : Marc Giusti, Grégoire Lecerf, Joris van der Hoeven, François Ollivier. Les assistantes du LIX, Corinne, Évelyne, Sylvie et Valérie, et les administrateurs système, Elie, James, Jean-Marc, ont toujours été un soutien capital.

Je remercie également les lyonnais et non-lyonnais du labo, en commençant par les joueurs de cartes : Amaury, Antoine, Damien, Jean-Phillipe, Joanie, Jonas, Mikaël, Victor ; et les non-joueurs de cartes : Claire, Cécile, Florence, Guillaume, Jérémie, Mahsa, Romain. Mes coburéaux successifs dans les bureaux 2030 et 1014 sont également à louer pour la bonne ambiance qui a régné dans ces deux bureaux, par ordre chronologique : Vincent, Sylvain, David, Daouda, Grégoire, Denis, François, Bruno.

Au sein de l'X, je tiens à remercier les élèves des promotions X2007 à X2012 (6 promos !) que j'ai pu cotoyer au sein de l'orchestre (SymphoniX,

puis l'Orchestre du Plateau de Saclay, je n'oublie pas non plus les non-X qui en font partie) et de la fanfare (le Platypus Braxx Band), pas seulement pour la musique, ni pour m'avoir fait découvrir le campus, l'argot et la vie des pseudos-militaires dans leur milieu naturel, mais surtout pour les instants toujours sympathiques partagés avec eux.

Mes amis sont à remercier bien sûr, notamment Cécile et Julien, Maryline et Rash ; ainsi que mes parents et ma belle-famille.

Et bien sûr, la plus importante, Chantal.

Chapter 2

Introduction NONUMBER

2.1 Context and motivation NONUMBER

The goal of this thesis is to bring new methods for statistical mixture models which rely on the theoretical methods proposed by the framework of information geometry. The main idea behind the algorithms presented here is to introduce a geometrical point of view to statistical problems, which are not obviously geometrical at first look: the link between statistical models and geometry is allowed by the use of information geometry.

Information geometry is at the frontier between statistics and geometry: intuitively, it amounts to doing statistics and probabilities with geometric tools like distance, geodesic, centroids, etc. It thus can be seen as a translation table between statistical problems and geometric ones: parameter estimation of a distribution will be a centroid, a maximum likelihood problem will amount to finding a minimum distance, Expectation-Maximization will be a soft clustering, etc. We can also imagine novel problems like Voronoi diagrams between distributions.

The field of information geometry finds its roots in the seminal work of Radhakrishna Rao [104] in the 1940's who was the first to add a geometric structure on top of statistical objects by defining the Rao distance between distributions and to bring a structure of a Riemaniann manifold on the parameters of the distribution: from his work comes the fundamental idea that statistical objects (distributions) exist by themselves as points in the manifold of distributions and that the parameters (means and variances for Gaussians for example) of a distribution are its coordinates in some coordinate system. Introducing a notion of coordinate system raises new questions about the invariances we can expect: since distributions are points and parameter coordinates, a legitimate hope is to have geometrical tools which do not depend on the choice of the coordinate system. The independence with respect to the coordinate system is called the invariance to reparameterization: it is a fundamental property since it allows to

use the most convenient coordinate system without changing the geometric properties of the studied objects. The Rao Riemaniann metric is the only one to possess this property in a Riemaniann context but other kinds of differential manifolds have been studied in the literature. Cencov [29] studied in the 1980's these notions of invariance in the non-Riemaniann case with a categorical point of view (in the sense of the category theory) and introduced a new kind of connections, the dual affine α-connections. These connections and the α-divergences naturally associated to them have been studied independently by Amari [7] which introduced the general notion of dually flat spaces. Related divergences such as the β-divergences and the more generic (α, β) divergences are getting more and more interest nowadays [31, 39].

A particular class of families of distributions is often used in the context of information geometry: the exponential families. These families are useful from a computational point of view: with the help of their bijection with Bregman divergences [12], a lot of statistical problems can be solved in the Bregman geometric domain. A fundamental tool is that the famous Kullback-Leibler divergence between members of the same exponential family amounts to a Bregman divergence between the parameters of the two distributions. The use of exponential families and of Bregman divergences allowed to go from the purely mathematical information geometry to the growing field of computational information geometry: in addition to the mathematical developments, information geometry is now looked at with a computational point of view which takes into account the algorithmic considerations.

Bregman geometry is interesting in itself since Bregman divergences encompass a lot of usual dissimilarity measures. Problems which are common in Euclidean geometry have been studied and extended to the Bregman setup: Bregman Voronoi diagrams [91], clustering with Bregman divergences [12], data structures with Bregman balls tree [102, 95], Bregman centroids [94] (which are known in closed-form), etc. Due to the bijection with exponential families, a lot of these methods translate naturally to the statistical setup allowing to solve statistical problems with Bregman geometric tools.

A big issue in computational information geometry is the availability of closed-form formulas. The Rao distance is a neat theoretical tool but is hardly usable in practice since its evaluation requires often a numerical approximation scheme to approximate geodesics. Even for distribution for which a formula is known, like the univariate Gaussian case, useful tools are missing (the Rao centroid for the univariate Gaussian is only known when the mean of all points is 0, greatly limiting the practical utility in the context of mixture models). The Bregman divergences are more computationally friendly: the distance itself and the centroids are known in closed-form. However, the application of these closed-form formulas to particu-

lar Bregman divergences (and associated exponential families) depends on the knowledge of some other formulas which may be difficult to write (the problem is often to express the Legendre-Fenchel dual of some function or to inverse a bijective function): everything goes well for the classical Gaussian distribution but some distributions like the Gamma or Beta laws are difficult to manipulate in the framework of exponential families.

This thesis deals mainly with the problem of the estimation of an unknown density which is an important problem in many applications. Three main approaches are possible: the parametric estimation learns the parameter of a particular distribution (often with a maximum likelihood estimator or with the method of moments); the semi-parametric models use a finite mixture to model the density as a weighted sum of distributions (usually with Expectation-Maximization or a variant); the non-parametric models represent the density as a non-weighted sum of distributions, each observation leading to a component (with a kernel density estimator). The parametric estimation is often not powerful enough to model complex densities so the semi-parametric and non-parametric approaches are often used in practice.

2.2 Contributions NONUMBER

The contributions presented in this manuscript are divided in two main parts: the first part is devoted to results on simplification of kernel density estimators and the second describes methods to build mixtures of generalized Gaussian distributions and of Gamma laws using a variant of k-MLE.

A kernel density estimator (KDE) is a non-parametric statistical method which estimates an unknown density by using a non-weighted sum of kernels centered on each point of the observation set (we use here Gaussian kernels). Compared to Expectation-Maximization which usually models a density with few components, KDE builds very large models since it needs to remember each of the observations. The idea we introduce here relies on Bregman Hard Clustering, an algorithm to cluster exponential families. The clustering has already been used to simplify mixtures models but we use it here to simplify the kernel density estimator: from a very large model with one component by observation, we build a compact model which is easy to use (the memory consumption is smaller, evaluating the density is faster, and so on for drawing random values, or for evaluating approximations of the Kullback-Leibler divergence). Along with the Bregman Hard Clustering which performs the simplification using Bregman divergences and Bregman centroids, we propose another strategy which makes use of the Fisher-Rao distance. Since Fisher-Rao centroids are not known in closed-form we use Model centroids. These centroids, which are defined on constant curvature spaces, are used as an approximation of the Fisher-Rao centroids on the hyperbolic space of the parameters of the Gaussian distri-

bution. All these developments come with a `Python` library called `pyMEF` implementing the proposed algorithms.

The k-MLE algorithm has been recently introduced for mixtures of exponential families (a weighted sum of components coming from the *same* exponential family): while producing models as good as the one produced by Expectation-Maximization (EM), it is faster. Contrary to EM which uses a soft assignment procedure, k-MLE uses a hard assignment to fasten the computation. It thus maximizes the complete log-likelihood (with origins of observations known) instead of the log-likelihood maximized by EM. We introduce here a generalization of k-MLE called Extended k-MLE which allows to build mixtures where all the components do not share the same family. This new method is first applied to mixtures of generalized Gaussian distributions. Two generalized Gaussian distributions are in the same family only when their mean and shape are identical. We thus cannot use k-MLE for this type of exponential family (except for mixtures of generalized Gaussian with common mean and shape, but this is a less interesting case): with Extended k-MLE, we are able to choose the exponential family of each component by selecting the best mean and shape parameters. Extended k-MLE is also applied to mixtures of Gamma laws: although it may seem surprising at first glance since Gamma is an exponential family (for all its parameters), some necessary formulas for the direct application of k-MLE are not known in closed-form. Without these closed-form formulas, we may need costly numerical schemes to run the algorithm. Our method relies on the following remark: Gamma laws with a fixed rate parameter are still exponential families and all the needed formulas are available in closed-form. We are thus in the same case as for generalized Gaussian: we build a mixture where all the components do not belong to the same family (since we are now manipulating Gamma with fixed rate, and not classical Gamma) where the rate parameters are chosen independently for each family.

2.3 Outline NONUMBER

The content of this manuscript is organized in three parts. Part I makes a review of results on information geometry and presents all the necessary preliminaries. Part II presents results on simplification of kernel density estimators to produce mixture models with Bregman Hard Clustering and Model Hard Clustering. Part III introduces new geometrical methods to build mixtures of Gamma laws and mixtures of generalized Gaussian which are difficult or impossible to build with existing methods working on exponential families.

The most used abbreviations and notations of the document are summarized in Annex A. In Annex B, the reader will find tables summarizing the decomposition in exponential families of all the distributions used in

this thesis and in Annex C a list of the publications related to results presented here. The manuscript concludes with an index and the bibliographical references.

Part I

Information geometry, exponential families and mixture models

Chapter 3

Information geometry and exponential families

3.1 Exponential families

Exponential families form a very large class of families of probability distributions which are known since the seminal work of Andersen [10], Darmois [36], Fisher [45], Koopman [70] and Pitman [103] in the 1930's. They encompass a lot of very common distributions such as the Gaussian, Dirichlet, Rayleigh, Beta, Gamma, Weibull, Von Mises-Fisher, Pareto, Laplace, Wishart, exponential, multinomial, generalized Gaussian laws. Some notable exceptions are the Cauchy and the Student laws. Another one is the uniform law (but it can be approximated with a generalized Gaussian with a large enough shape parameter, see Section 11.4.1 in Chapter 11 for more details).

This class of distributions provides a very generic framework to study all these probability distributions. This allows to build algorithms which are not written for a particular distribution but which will be available for any exponential family. The difficulty is not anymore to derive a specifically tailored algorithm from a given distribution but to write the needed mathematical developments in order to use the generic algorithm (which raises problems of closed-form formula and of computational efficiency).

3.1.1 Definitions

Let $\mathcal{X}$ be a subset of $\mathbb{R}^n$, and x a point of this space.

Definition 3.1. An *exponential family* is a set of probability distributions whose probability density function (pdf) can be expressed as the following canonical form:

$$p_F(x;\theta) = \exp\left(\langle\theta|t(x)\rangle - F(\theta) + k(x)\right) \tag{3.1}$$

where

- θ is called the *natural parameters* of the family,
- F is called the *log-normalizer* ,
- $k(x)$ is a normalization coefficient called *carrier measure* ,
- $t(x)$ is a function over the support of the family,
- $\langle .|. \rangle$ is a scalar product.

Note 1. When we speak about probability density functions, we mean the Radon-Nikodym derivative of the probability measure which allows to use a common framework for both continuous and discrete probabilities and the integrals are made on the support of the considered distribution.

Proposition 1 (Barndorf-Nielsen, 1978)**.** *The function t is a sufficient statistic of the distribution and is even a minimal one.*

Remark 1. The carrier measure k ensures the normalization of the family to 1. It is often not needed (that is, equals to 0 for any x) which simplifies the manipulation of the family: this is the case for Gaussian distribution for example.

The natural parameters come from the *natural parameter space* defined as:

$$\Theta = \left\{ \theta \;\middle|\; \int_{\mathcal{X}} \exp\left(\langle\theta|t(x)\rangle - F(\theta) + k(x)\right)\, \mathrm{d}x < \infty \right\} \tag{3.2}$$

The family is said to be *regular* if Θ is open and the dimension of the space Θ is called the *order* of the family.

Definition 3.2. The *log-normalizer* characterizes the exponential family [25]. It is a strictly convex and differentiable function which is equal to:

$$F(\theta) = \log \int_x \exp(\langle\theta|t(x)\rangle + k(x))\, \mathrm{d}x \tag{3.3}$$

Remark 2. The usual form of the distributions which are exponential families is often not the form introduced previously: the parameters used to describe the distribution may not be the natural parameters. It means that we have (at least) two different parameterizations of an exponential family:

- the natural parameters,
- the usual parameters, known as the *source* parameters .

Example 3.1 (Gaussian distribution)**.** The usual form of the Gaussian distribution (in the univariate case) is:

$$p(x;\, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3.4}$$

with $(\mu, \sigma^2) \in \mathbb{R} \times \mathbb{R}^+$. μ and σ^2 are the source parameters of the Gaussian family. With a few work, we can rewrite the Gaussian density function in the form

$$p_F(x; \theta_1, \theta_2) = \exp\left(\langle \theta_1, \theta_2 | t(x) \rangle - F(\theta_1, \theta_2) + k(x)\right) \tag{3.5}$$

where

- $t(x) = (x, x^2)$,
- $(\theta_1, \theta_2) = \left(\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\right)$,
- $F(\theta_1, \theta_2) = -\frac{\theta_1^2}{4\theta_2} + \frac{1}{2}\log\left(-\frac{\pi}{\theta_2}\right)$,
- $k(x) = 0$,
- $\langle \theta_1, \theta_2 | t(x) \rangle = \theta_1 x + \theta_2 x^2$.

We have a one-to-one mapping between the source parameters λ and the natural parameters through the following bijection:

- $\lambda \to \theta : (\mu, \sigma^2) \mapsto \left(\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\right)$,
- $\theta \to \lambda : (\theta_1, \theta_2) \mapsto \left(-\frac{\theta_1}{2\theta_2}, -\frac{1}{2\theta_2}\right)$.

3.1.2 Dual parameterizations

We have so far two different ways to describe a particular member of an exponential family. We will now introduce a third kind of parameterization, which is based on a dual representation of the log-normalizer. Let's begin with some definitions from convex analysis from the reference book by Rockafellar [107].

Definition 3.3 (Essentially smooth function)**.** A proper convex function f is essentially smooth if:

- $C = \text{int}(\text{dom} f)$ is not empty,
- f is differentiable on C,
- $\lim_{i\to\infty} |\nabla f(x_i)| \to +\infty$ when the sequence (x_i) converges to a boundary point of C.

Proposition 2. *If a closed proper convex function f is essentially smooth, ∇f is a bijection and we can define its inverse function ∇F^{-1}.*

Definition 3.4 (Legendre-Fenchel conjugate)**.** Let C be an open subset of $\mathbb{R}^d$ and f a differentiable function $C \to \mathbb{R}$. The Legendre-Fenchel conjugate of the pair (C, f) is the pair $(D, f^\star)$ such that $D = \nabla f(C)$ and that

$$f^\star(x^\star) = \sup_{x\in C} \{\langle x|x^\star\rangle - f(x)\} \tag{3.6}$$

Notice that at this time, this conjugate has no reason to be well-defined. When it is well defined, we call it the *Legendre-Fenchel transform* .

Theorem 3. *If f is a closed proper convex function and $C = \text{int}\,\text{dom}\, f$ is not empty, then the Legendre-Fenchel conjugate $(D, f^\star)$ of the pair (C, f) is well defined with $D = \text{int}\,\text{dom}\,\nabla F$.*

The Legendre-Fenchel conjugate of a closed proper convex function is usually not a closed proper convex function: this means that even if the Legendre conjugate of a function exists, the Legendre conjugate of the Legendre conjugate has no reason to be well defined. If we want a bijective mapping, we need more hypotheses on the considered function.

Definition 3.5 (Legendre type function)**.** An essentially smooth and strictly convex function f on an open convex $C = \text{int}\,\text{dom}\, f$ is called a *Legendre type* function.

Proposition 4. *The Legendre conjugate of a Legendre type function (C, f) is a Legendre type function $(C^\star, f^\star)$ with $C^\star = \nabla f(C)$ and the Legendre conjugate of $(C^\star, f^\star)$ is (C, f). Moreover, ∇f is a bijective mapping between C and $C^\star$ and $\nabla f^\star = (\nabla f)^{-1}$.*

When talking about exponential families, a link arises naturally through the log-normalizer which is a strictly convex function on an open subset of $\mathbb{R}^n$.

We can thus deduce the existence of the Legendre-Fenchel conjugate $F^\star$ of the log-normalizer F. In the general case, we cannot apply the result from Proposition 4. To get a full duality between F and $F^\star$ through the Legendre-Fenchel transform, we need to restrict to exponential families with a Legendre type log-normalizer.

Definition 3.6 (Steep exponential family)**.** An exponential family is *steep* if its log-normalizer is essentially smooth.

Any regular family is steep but the converse is not true (the inverse Gaussian is a counter-example) [25].

Proposition 5. *When the exponential family is steep, the following properties hold:*

- $(\Theta^\star, F^\star)$ *is well defined and is a convex function of Legendre type,*
- $(\Theta^{\star\star}, F^{\star\star}) = (\Theta, F)$*,*
- ∇F *is a bijection,*
- $\nabla F^\star = (\nabla F)^{-1}$*.*

Moreover, the dual of F can be computed trough the definition of the Legendre-Fenchel transform $F^\star(\eta) = \sup_{\theta \in C} \{\langle \theta | \eta \rangle - F(\theta)\}$ by noticing that the *supremum* is reached for $\theta = (\nabla F)^{-1}(\eta)$.

This result gives two dual parameter spaces: the natural parameter space Θ and its dual $H =' \Theta^\star$ which will be called the *expectation parameter space*. The *expectation parameter* η associated to a natural parameter θ is its image by the reparameterization function ∇F. Conversely, we can transform an expectation parameter to a natural parameter using the function $(\nabla F)^{-1}$:

$$\eta(\theta) = \nabla F(\theta) \qquad \theta(\eta) = \nabla F^{-1}(\eta) \tag{3.7}$$

Remark 3. The name *expectation parameter* comes from the following equality:

$$\eta = \eta(\theta) = E_p[X] = \int x p(x) \mathrm{d}x \tag{3.8}$$

Along with the source and natural parameters, we now have a third parameterization of an exponential family. All three are bijective with one another (see Figure 3.1).

3.1.3 Statistical manifold

The first geometric point of view on distribution theory has been brought by Rao [104] in 1945 [1] by considering a family of distributions p_θ with support R^d and parameters $\theta \in \mathbb{R}^D$ (D is known as the order of the family). The following set is called the *population parameter space* :

[1] First work begins in the 1930's and were carried out with Hotelling [58]

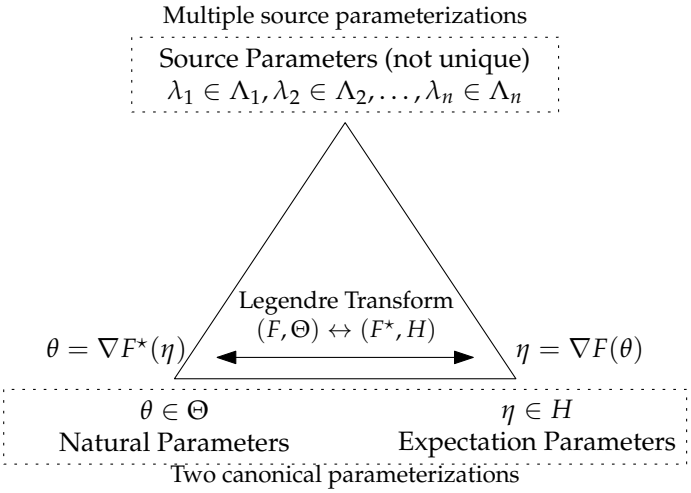


Figure 3.1: Dual parameterizations of an exponential family. The three parameter spaces, source, natural and expectation, are in bijection. The bijection between the natural parameter space and the expectation parameter space comes from the Legendre transform of the log-normalizer F.

$$\Theta = \left\{ \theta \in \mathbb{R}^D | \int_x p_\theta(x) \mathrm{d}x = 1 \right\} \tag{3.9}$$

In this setup, a distribution p_θ is seen as a point and the parameter space Θ is a coordinate system which can be used to describe the members of the family of distributions. Rao's key contribution was to add a metric structure on top of this set of distributions. Let us begin with a first definition:

Definition 3.7 (Fisher Information Matrix)**.** The *Fisher Information Matrix* is the positive definite matrix whose coefficient are:

$$I_{ij}(\theta) = E\left[\frac{\partial \log p_\theta}{\partial \theta_i} \frac{\partial \log p_\theta}{\partial \theta_i} \right] \tag{3.10}$$

$$= E\left[\frac{1}{p_\theta} \frac{\partial p_\theta}{\partial \theta_i} \frac{1}{p_\theta} \frac{\partial p_\theta}{\partial \theta_i} \right] \tag{3.11}$$

This matrix serves as a basis for defining a metric between two infinitely close points θ and $\theta + \mathrm{d}\theta$:

Definition 3.8 (Rao metric)**.** The *Rao metric* is defined as the following quadratic form:

$$\mathrm{d}s = \mathrm{d}s^2(\theta) = \sum_{i=1}^{D}\sum_{j=1}^{D} I_{ij}(\theta) d\theta_i d\theta_j \tag{3.12}$$

$$= (\nabla\theta)^T I(\theta) \nabla\theta \tag{3.13}$$

The Fisher Information Matrix thus defines a scalar product on the tangent space around θ, bringing a Riemannian structure to the *statistical manifold* .

Since the Fisher Information Matrix is invariant to a bijective transformation of the parameter space, the Rao metric is thus invariant to a bijective reparameterization, making it a tool of choice to look at intrinsic properties of the distributions without being parasited by an arbitrary choice of the coordinate system.

Chencov [29] proved that the Fisher-Rao metric is the unique invariant information metric (up to a constant scaling factor) in the discrete case. This result has been recently extended in [76] to a more general case.

From the metric, the Fisher-Rao distance (FR) is then defined as the geodesic distance between two points θ_1 and θ_2 of the statistical manifold, that is, in Riemannian geometry, the length of the shortest path joining θ_1 and θ_2.

Definition 3.9 (Fisher-Rao distance)**.** The geodesic length between two points θ_1 and θ_2 of the statistical manifold is known as the Fisher-Rao distance and is written:

$$\mathrm{FR}(\theta_1, \theta_2) = \min_{\theta(t) \text{ st } \theta(0)=\theta_1, \theta(1)=\theta_2} \int_0^1 \sqrt{ds^2} dt \tag{3.14}$$

In practice, it is often difficult to compute this geodesic: it requires either to compute a numerical approximation which amounts to solving an ordinary differential equation [7] or to find a closed-form expression of the distance (which may be difficult or impossible to obtain).

There are however some interesting particular cases where the Fisher-Rao geometry is fully known:

- for the multinomial law, the geometry amounts to the spherical geometry [72, 64],
- for the location-scale family (including the Gaussian law) we get the hyperbolic geometry [64].

Lemma 6. *For two multinomial laws p and p' with parameters $(\alpha_i)_{1\leq i\leq n}$ and $(\alpha'_i)_{1\leq i\leq n}$ (with $\sum_i \alpha_i = 1$ and $\sum_i \alpha'_i = 1$), the Fisher-Rao distance between p and p' is the spherical distance:*

$$D(p, p') = 2 \arccos\left(\sum_i \sqrt{\alpha_i \alpha'_i}\right) \tag{3.15}$$

Lemma 7. *For a probability density function p, the associated location-scale manifold*

$$\left\{\frac{1}{\sigma} p\left(\frac{x-\mu}{\sigma}\right) | (\mu, \sigma) \in \mathbb{R} \times \mathbb{R}^+\right\} \tag{3.16}$$

is a space with a constant negative curvature, that is an hyperbolic geometry.

Part II makes use of this result to compute the Fisher-Rao distance between two Gaussian distributions.

3.1.4 Invariances

As stated before, the Rao metric is invariant to reparameterizations of the parameter space. It is known since the *Erlangen program* of Felix Klein [67] that invariant transformations are a key concept of any geometry (think of the group of rigid transformations in Euclidean geometry: translations, rotations, reflections). The idea is that points exist by themselves, independently of the coordinate system used to represent them.

The family of *Csiszár f-divergences* is the only family of divergences which has the property of invariance by reparameterization. This family of divergences has been introduced independently by Csiszár [35], by Morimoto [83] and by Ali and Silvey [2] and is thus also known as Csiszár divergences, Csiszár-Morimoto divergences or Ali and Silvey divergences.

Definition 3.10. For a convex and derivable function f such that

- $f(1) = f'(1)$
- $f''(1) = 1$

the *f-divergence* is:

$$D_f(p\|q) = \int_x p(x)\, f\left(\frac{q(x)}{p(x)}\right) \mathrm{d}x \tag{3.17}$$

An interesting special case is the *Kullback-Leibler divergence* (KL) generated by the convex function $f(x) = x \log x$:

$$\mathrm{KL}\,(p\|q) = \int_x p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x \tag{3.18}$$

The Kullback-Leibler divergence is related to the Shannon entropy through the following relation:

$$\mathrm{KL}\left(p\|q\right) = H^{\times}(p;q) - H(p) \tag{3.19}$$

where $H^{\times}(p;q) = \int -p(x)\log q(x)\mathrm{d}x$ and $H(p) = H^{\times}(p;p)$. The KL divergence between p and q thus measures the information which is lost when q (coming from an estimation algorithm) is used to approximate p (the true distribution).

Moreover, at the infinitesimal scale, KL is related to the Rao distance and we have:

$$D(\theta, \theta + d\theta) = \sqrt{2\mathrm{KL}\left(\theta\|\theta + d\theta\right)} \tag{3.20}$$

3.1.5 Maximum likelihood estimator

Given a sequence of n i.i.d. observations x_1, x_2, …, x_n sampled from an exponential family $p_F(x;\theta)$, we define the following quantities:

Definition 3.11. The likelihood function is:

$$l(x_1, \ldots, x_n; \theta) = \prod_{i=1}^{n} p_F(x_i;\theta) \tag{3.21}$$

In practice, it is often easier to use the average log-likelihood defined as:

$$ll(x_1, \ldots, x_n; \theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_F(x_i;\theta) \tag{3.22}$$

The maximum likelihood estimator (MLE) is then:

$$\hat{\theta} = \arg\max_{\theta} l(x_1, \ldots, x_n; \theta) = \arg\max_{\theta} ll(x_1, \ldots, x_n; \theta) \tag{3.23}$$

The following results directly from the general expression of an exponential family:

Theorem 8. *The maximum likelihood estimator of an exponential family is:*

$$\hat{\theta} = \nabla F^{\star}\left(\frac{1}{n}\sum_{i=1}^{n} t(x_i)\right) \tag{3.24}$$

Proof. From the definition of an exponential family, we have:

$$\arg\max_{\theta} ll(x_1, \ldots, x_n; \theta) = \arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left(\langle \theta | t(x_i) \rangle - F(\theta) + k(x_i) \right) \quad (3.25)$$

$$(3.26)$$

The maximum $\hat{\theta}$ will be reached for a 0 of the derivative of the log-likelihood:

$$\frac{\partial ll}{\partial \theta} = \frac{1}{n} \sum_{i=1}^{n} \left(t(x_i) - \nabla F(\theta) \right) \quad (3.27)$$

We thus get that

$$\nabla F(\hat{\theta}) = \frac{1}{n} \sum_{i=1}^{n} t(x_i) \quad (3.28)$$

$$\hat{\theta} = \nabla F^{-1} \left(\frac{1}{n} \sum_{i=1}^{n} t(x_i) \right) \quad (3.29)$$

$$\hat{\theta} = \nabla F^{\star} \left(\frac{1}{n} \sum_{i=1}^{n} t(x_i) \right) \quad (3.30)$$

□

Using the bijection between the coordinate systems of an exponential family, we get that an estimator for the expectation parameters is simply:

$$\hat{\eta} = \frac{1}{n} \sum_{i=1}^{n} t(x_i) \quad (3.31)$$

Note 2. The maximum likelihood estimator of an exponential family admits a geometric interpretation as the barycenter of the sufficient statistics of the observed points.

Proposition 9. *For a steep family, the MLE exists and is unique if and only if $\hat{\eta}$ is in the interior of the convex closure of Θ [13].*

A sufficient condition for the existence of the MLE [21] is that the matrix

$$T = \begin{pmatrix} 1 & t_1(x_1) & \ldots & t_D(x_1) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_1(x_n) & \ldots & t_D(x_n) \end{pmatrix} \quad (3.32)$$

has rank $D+1$ (where D is the order of the exponential family). This condition means that among the n vectors $t(x_i)$ used for the estimation, we need at least $D+1$ linearly independent vectors.

When speaking about an estimator, it is also important to discuss its quality. A lower bound on the variance of any unbiased estimator has been proposed independently by Cramér [34], Rao [104] and Fréchet [46] (but we will still use the common name Cramér-Rao bound):

Theorem 10 (Cramér-Rao bound)**.** *For an unbiased estimator $\tilde{\lambda}$, the following inequality between the variance of the estimator and the Fisher Information Matrix holds:*

$$V[\tilde{\lambda}] \succeq I(\lambda)^{-1} \tag{3.33}$$

where $I(\lambda)$ is the Fisher information matrix $I_{ij}(\lambda) = E\left[\frac{\partial \log p(x;\lambda)}{\partial \lambda_i} \frac{\partial \log p(x;\lambda)}{\partial \lambda_i}\right]$ and $\succeq$ denotes the Löwner partial ordering on positive definite matrices ($A \succeq B$ iff $A - B$ is positive definite).

3.1.6 Examples of exponential families

Some examples of Exponential Families are given here; along with the resulting expression for the decomposition of each family, some steps of calculus are given in order to illustrate the process of rewriting the pdf of distribution to look like the canonical decomposition of a family. A synthetic view of all the families which appear in this document is available in Annex B.

1. Multivariate Gaussian

   The multivariate Gaussian (see Figure 3.2) is the d-dimensional extension of the classical Gaussian law. For a Gaussian with support $\mathbb{R}^d$, the order of the family is $d + \frac{d(d+1)}{2}$ (a vector of size d and a positive definite matrix of size $d \times d$). The associated pdf is:

$$f(x;\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma)}} \exp\left(\frac{(x-\mu)^T\Sigma(x-\mu)}{2}\right) \tag{3.34}$$

$$= \exp\left(\frac{(x-\mu)^T\Sigma(x-\mu)}{2} - \log\left((2\pi)^{d/2}\sqrt{\det(\Sigma)}\right)\right) \tag{3.35}$$

$$= \exp\left(\underbrace{\frac{1}{2}(x^T\Sigma x - \mu^T\Sigma x - x^T\Sigma\mu)}_{\langle(\theta_1,\theta_2)|t(x)\rangle} + \underbrace{\frac{1}{2}\mu^T\Sigma\mu - \frac{1}{2}\log\det(\Sigma) - \frac{d}{2}\log(2\pi)}_{F(\theta_1,\theta_2)}\right) \tag{3.36}$$

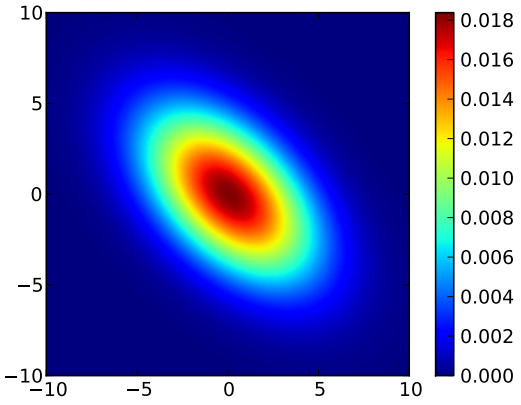


Figure 3.2: Density of a two-dimensional multivariate Gaussian law with mean $[0, 0]$ and covariance matrix $\begin{pmatrix} 10 & 5 \\ 5 & 10 \end{pmatrix}$.

We can build an exponential family decomposition:

(a) Source parameters: $(\mu, \Sigma) \in \mathbb{R}^d \times \mathbb{R}^{d \times d}$

(b) Source to Natural: $(\theta_1, \theta_2) = \left(\Sigma^{-1}\mu, \frac{1}{2}\Sigma^{-1}\right)$

(c) Natural to Source: $(\mu, \Sigma) = \left(\frac{1}{2}\theta_2^{-1}\theta_1, \frac{1}{2}\theta_2^{-1}\right)$

(d) Source to Expectation: $(\eta_1, \eta_2) = \left(\mu, -(\Sigma + \mu\mu^T)\right)$

(e) Expectation to Source: $(\mu, \Sigma) = \left(\eta_1, -(\eta_2 + \eta_1\eta_1^T)\right)$

(f) Log-normalizer: $F(\theta_1, \theta_2) = \frac{1}{4}\mathrm{tr}\left(\theta_2^{-1}\theta_1\theta_1^T\right) - \frac{1}{2}\log\det\theta_2 + \frac{d}{2}\log\pi$

(g) Dual log-normalizer: $F^\star(\eta_1, \eta_2) = \frac{1}{2}\log(1 + \eta_1^T\eta_2^{-1}\eta_1) - \frac{1}{2}\log\det(-\eta_2) - \frac{d}{2}\log(2\pi e)$

(h) Natural to Expectation:
$\nabla F(\theta_1, \theta_2) = \left(\frac{1}{2}\theta_2^{-1}\theta_1, -\frac{1}{2}\theta_2^{-1} - \frac{1}{4}\left(\theta_2^{-1}\theta_1\right)\left(\theta_2^{-2}\theta_1\right)^T\right)$

(i) Expectation to Natural:
$\nabla F^\star(\eta_1, \eta_2) = \left(-\left(\eta_2 + \eta_1\eta_1^T\right)^{-1}\eta_1, -\frac{1}{2}\left(\eta_2 + \eta_1\eta_1^T\right)\right)$

(j) Sufficient statistic: $t(x) = (x, -xx^T)$

(k) Carrier measure: $k(x) = 0$

2. Poisson distribution

Contrary to the previous one, this distribution (see Figure 3.3) uses only one parameter, with a probability mass function expressed as:

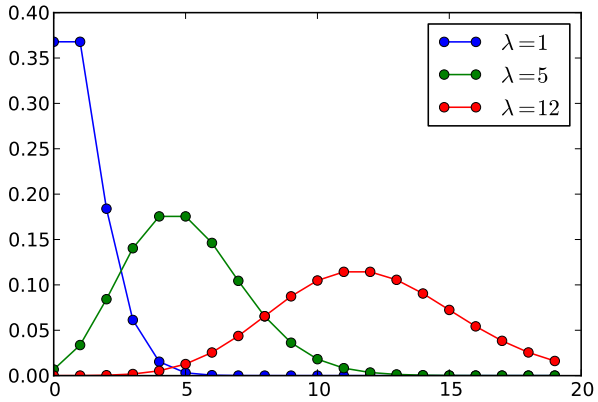


Figure 3.3: Density of a Poisson law with various values for λ (notice that Poisson is a discrete probability law and that the points are only linked to improve readability).

$$f(x;\lambda) = \frac{\lambda^x \exp(-\lambda)}{x!} \tag{3.37}$$

$$= \exp(\underbrace{x \log \lambda}_{\langle t(x)|\theta\rangle} - \underbrace{\lambda}_{F(\theta)} \underbrace{- \log x!)}_{k(x)} \tag{3.38}$$

$$\tag{3.39}$$

for $x \in \mathbb{N}^+$.

The decomposition is:

(a) Source parameters: $\lambda \in \mathbb{R}^+$
(b) Source to Natural; $\theta = \log \lambda$
(c) Natural to Source: $\lambda = \exp \theta$
(d) Source to Expectation: $\eta = \lambda$
(e) Expectation to Source: $\lambda = \eta$
(f) Log-normalizer: $F(\theta) = \exp \theta$
(g) Natural to Expectation: $\nabla F(\theta) = \exp \theta$
(h) Expectation to Natural: $\nabla F^\star(\eta) = (\nabla F)^{-1}(\eta) = \log \eta$
(i) Dual log-normalizer: $F^\star(\eta) = \eta \log \eta - \eta$
(j) Sufficient statistic: $t(x) = x$
(k) Carrier measure: $k(x) = -\log x!$

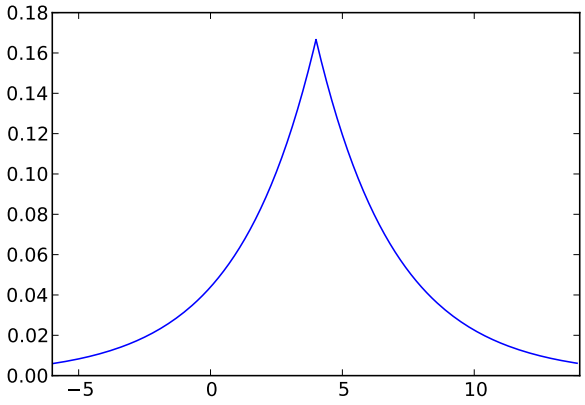


Figure 3.4: Density of a Laplace law with mean $\mu = 4$ and scale $\lambda = 3$.

3. Laplace law

   The Laplace law (see Figure 3.4) is parameterized by its mean μ and another parameter σ. However, the Laplace distribution is only an exponential family when the mean is fixed so the only source parameter will be σ for some fixed μ.

$$f_\mu(x;\lambda) = \frac{1}{2\sigma}\exp\left(\frac{-|x-\mu|}{\sigma}\right) \tag{3.40}$$

$$= \exp\left(\underbrace{-|x-\mu|\frac{1}{\sigma}}_{\langle t(x)|\theta\rangle} + \underbrace{\log\left(\frac{1}{2}\sigma\right)}_{-F(\theta)}\right) \tag{3.41}$$

   for $x \in \mathbb{R}$.

   We get the decomposition:

   (a) Source parameters: $\lambda \in \mathbb{R}^+$
   (b) Source to Natural; $\theta = -\frac{1}{\sigma}$
   (c) Natural to Source: $\lambda = -\frac{1}{\theta}$
   (d) Source to Expectation: $\eta = \sigma$
   (e) Expectation to Source: $\lambda = \eta$
   (f) Log-normalizer: $F(\theta) = \log\left(-\frac{2}{\theta}\right)$
   (g) Natural to Expectation: $\nabla F(\theta) = -\frac{1}{\theta}$
   (h) Expectation to Natural: $\nabla F^\star(\eta) = (\nabla F)^{-1}(\eta) = -\frac{1}{\eta}$

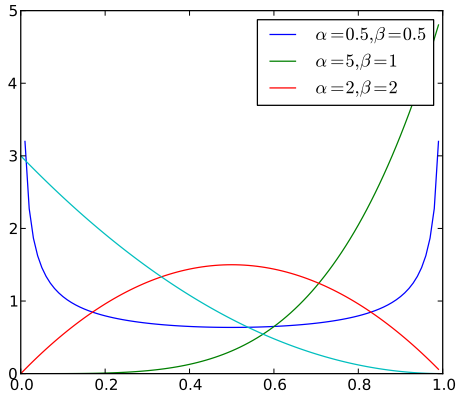


Figure 3.5: Density of a Beta law with different values for the parameters α and β.

(i) Dual log-normalizer: $F^\star(\eta) = -\log \eta$

(j) Sufficient statistic: $t(x) = |x - \mu|$

(k) Carrier measure: $k(x) = 0$

4. Beta distribution

   Even if the full decomposition is known in closed-form for the previous distributions, there are also cases where some expressions are not known. The Beta law and the Gamma law are two of them: functions $F^\star$ and $\nabla F^\star$ are not known in closed-form, which limits the practical utility of the exponential families framework for these distributions. We present here only the limitations for the Beta law, the case of the Gamma law will be presented in Chapter 11 with a workaround to avoid the computation of the unknown functions.

$$f(x; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{3.42}$$

   for $x \in \mathbb{R}$ and with $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$.

   The known elements of the decomposition are written as:

   (a) Source parameters: $(\alpha, \beta) \in \mathbb{R}^+ \times \mathbb{R}^+$

   (b) Source to Natural: $(\theta_1, \theta_2) = (\alpha - 1, \beta - 1)$

   (c) Natural to Source: $(\alpha, \beta) = (\theta_1 + 1, \theta_2 + 1)$

   (d) Log-normalizer: $F(\theta) = \log B(\theta_1 + 1, \theta_2 + 1)$

(e) Natural to Expectation: $\nabla F(\theta) = (\psi(\theta_1+1) - \psi(\theta_1+\theta_2+2), \psi(\theta_1+1) - \psi(\theta_1+\theta_2+2))$

(f) Sufficient statistic: $t(x) = (\log x, \log(1-x))$

(g) Carrier measure: $k(x) = 0$

The previous functions are the only ones which are known in closed-form: the $F^\star$ and $\nabla F^\star$ functions are not available in closed-form. As we will see in the next chapter, this lack of easy-to-use expressions is a major problem for the application of generic exponential families algorithms on the Beta law.

3.2 Bregman geometry

3.2.1 Definition and properties

Definition 3.12 (Bregman divergence)**.** Given a strictly convex function $F : \mathcal{X} \to \mathbb{R}$ defined on a closed convex subset of $\mathbb{R}^d$ and differential over the relative interior of $\mathcal{X}$, the Bregman divergence associated to the generator F is:

$$B_F(x\|y) = F(x) - F(y) - \langle x-y, \nabla F(y)\rangle \tag{3.43}$$

We deal here with differentiable generators: some other work extend the definition of the Bregman divergence to non-differentiable functions using sub-gradients [37].

Remark 4. A Bregman divergence is not symmetric in the general case.

The lack of symmetry of the Bregman divergence leads to two different notions of a Bregman ball of radius r and center c (see Figure 3.6):

- the left-sided Bregman balls: $\{x | B_F(x\|c) < r\}$
- the right-sided Bregman balls: $\{x | B_F(c\|x) < r\}$

Proposition 11 (Non-negativity)**.** *For all x, y, we have $B_F(x\|y) \geq 0$ and equality holds if and only if $x = y$.*

Proposition 12 (Convexity)**.** *A Bregman divergence is convex in its first argument. In other words, the function $x \mapsto B_F(x\|y)$ for a fixed y is convex.*

Corollary 1. *Left-sided Bregman balls are convex.*

Proposition 13 (Linearity)**.** *A Bregman divergence is linear with respect to the generator function:*

- $B_{F_1+F_2}(x\|y) = B_{F_2}(x\|y) + B_{F_2}(x\|y)$

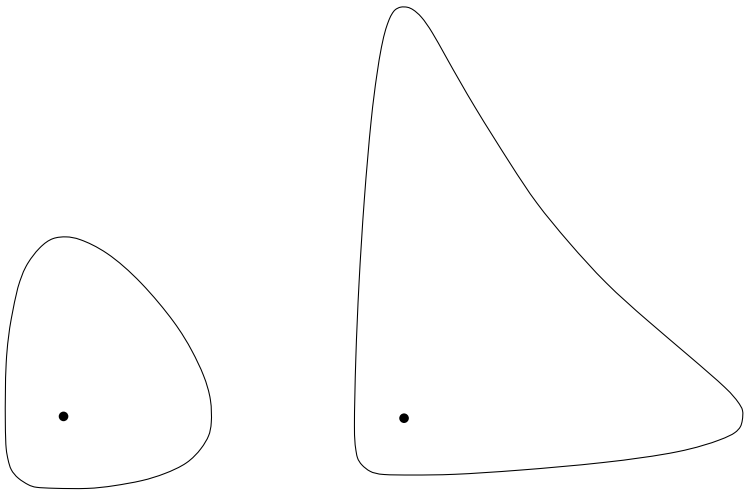


Figure 3.6: Left- and right- sided Itakura-Saito balls

- $B_{cF}(x\|y) = cB_F(x\|y)$ *for* $c \in \mathbb{R}^+$

Proposition 14 (Duality between Bregman divergences)**.** *For a Legendre type generator F, we have the following equality:*

$$B_F(x\|y) = B_{F^\star}(\nabla F(y)\|\nabla F(x)) \tag{3.44}$$

From the three-points property:

$$B_F(x\|z) = B_F(x\|y) + B_F(y\|z) - \langle (y-x)|\nabla F(y) - \nabla F(x)\rangle \tag{3.45}$$

we deduce the following theorem:

Proposition 15 (Bregman-Pythagoras theorem)**.** *For any* x, y, z *such that* $\langle (y-x)|\nabla F(x) - \nabla F(y)\rangle = 0$*, we have:*

$$B_F(x\|z) = B_F(x\|y) + B_F(y\|z) \tag{3.46}$$

Definition 3.13 (Separable divergence)**.** A divergence D on $\mathbb{R}^d$ (or a subset) is said to be separable when it is generated by a scalar divergence d on $\mathbb{R}$ (or a subset). That is, for two vectors $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$, the divergence can be written as:

$$D(x,y) = \sum_{i=1}^{d} d(x_i, y_i) \tag{3.47}$$

This is useful in practice since the divergence value can be computed by independent evaluations on each dimension of the studied vectors.

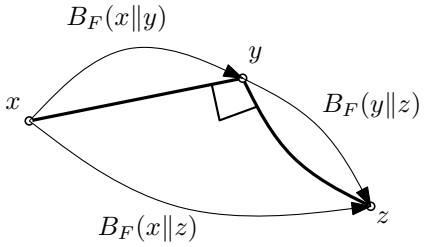


Figure 3.7: Bregman Pythagoras theorem between points x, y and z.

3.2.2 Examples

The family of Bregman divergences is parameterized by a convex function, thus providing a very large class of divergences. Among all the possible divergences, we retrieve a lot of usual divergences used in many applications.

1. Squared Euclidean distance

   The most famous distance which is a Bregman divergence is without any doubt the squared Euclidean distance which can be obtained with the generator $F(x) = x^2$.

$$
\begin{aligned}
d_2(p,q) &= \sum_i (p_i - q_i)^2 && (3.48)\\
&= \sum_i p_i^2 + q_i^2 - 2p_iq_i && (3.49)\\
&= \sum_i p_i^2 - q_i^2 - 2p_iq_i + 2q_i^2 && (3.50)\\
&= \sum_i F(p_i) - F(q_i) - \langle p_i - q_i | 2q_i \rangle && (3.51)\\
&= \sum_i F(p_i) - F(q_i) - \langle p_i - q_i | \nabla F(q_i) \rangle && (3.52)\\
&= B_F\left(p \| q\right) && (3.53)
\end{aligned}
$$

   The squared Euclidean distance is symmetrical and separable.

2. Squared Mahalanobis distance

   Not all the divergences are separable: it is not the case for the Mahalanobis distance. It is not surprising since the heart of the Mahalanobis distance is to take into account the dependencies between dimensions with the help of a covariance matrix [77]:

$$D_\Sigma(p,q) = (p-q)^T \Sigma (p-q) \quad (3.54)$$

where Σ is a positive definite matrix.

The square of the Mahalanobis distance is a Bregman divergence for the generator $F(x) = x^T \Sigma x$.

The Mahalanobis distance is not separable except for the particular case of a diagonal matrix Σ (meaning that all the dimensions are independent). It is also a symmetric divergence, the only one among the Bregman divergences [22] (with Euclidean distance, which is a special case of Mahalanobis).

3. Kullback-Leibler divergence

   The Kullback-Leibler divergence is of primary importance for statistical applications. It is a Bregman divergence for the generator $F(x) = x \log x$ (the negative of the Shannon entropy).

4. Itakura-Saito divergence

   A less common example is the Itakura-Saito divergence [51] which is generated by the Burg entropy $F(x) = -\log x$:

$$\mathrm{IS}(p, q) = \sum_i \frac{p_i}{q_i} - \log \frac{p_i}{q_i} - 1 \tag{3.55}$$

   We can see immediately that this divergence is separable and not symmetrical.

3.2.3 Bijection with the exponential families

It is known since the work of Banerjee, Merugu, Dhillon, and Ghosh [12] that there is a bijection between regular exponential families and regular Bregman divergences through the convex function: for each regular exponential family, there is a unique regular Bregman divergence, and for each regular Bregman divergence there is a unique regular exponential family.

We give here the first part of the bijection:

Lemma 16. *A regular exponential family with log-normalizer F can be rewritten in terms of a regular Bregman divergence with generator $F^\star$, the Legendre dual of F:*

$$p_F(x; \theta) = \exp\left(-B_{F^\star}\left(t(x) \| \eta\right) + F^\star\left(t(x)\right) + k(x)\right) \tag{3.56}$$

where $\eta = \nabla F(\theta)$ is the expectation parameter associated to the natural parameter θ.

Proof. The result is straightforward using the expression of p_F and the formula for the dual of F:

$$F^\star(\eta) = \langle \nabla F^\star(\eta) | \eta \rangle - F(\nabla F^\star(\eta)) \tag{3.57}$$
$$= \langle \theta | \eta \rangle - F(\theta) \tag{3.58}$$

We thus have:

$$p_F(x;\theta) = \exp\left(\langle \theta | t(x) \rangle - F(\theta) + k(x)\right) \tag{3.59}$$
$$= \exp\left(\langle \theta | t(x) \rangle - \langle \theta | \eta \rangle + F^\star(\eta) + k(x)\right) \tag{3.60}$$
$$= \exp\left(\langle \theta | t(x) - \eta \rangle + F^\star(\eta) + k(x)\right) \tag{3.61}$$
$$= \exp\left(\langle \nabla F^\star(\eta) | t(x) - \eta \rangle + F^\star(\eta) + k(x)\right) \tag{3.62}$$
$$= \exp\left(-B_{F^\star}\left(t(x) \| \eta\right) + F^\star(t(x)) + k(x)\right) \tag{3.63}$$

□

This leads to an essential result which allows to compute the Kullback-Leibler divergence between two members of the **same** exponential family with a closed-form formula.

Proposition 17. *The Kullback-Leibler divergence between two members of the same exponential family is a Bregman divergence between the parameters of these families (with swapped order):*

$$\mathrm{KL}\left(p_F(x,\theta_1) \| p_F(x,\theta_2)\right) = B_{F^\star}\left(\eta_1 \| \eta_2\right) \tag{3.64}$$
$$= B_F\left(\theta_2 \| \theta_1\right) \tag{3.65}$$

Remark 5. It is interesting to notice that the Kullback-Leibler divergence between two members of the same exponential family can be computed with the Bregman divergence generated by the log-normalizer of the family but that the Kullback-Leibler divergence is also in itself a Bregman divergence.

3.2.4 Dually flat geometry

In its pioneering work Amari [7] brought a new point of view on information geometry and extended the Riemannian vision of the Rao statistical manifold by introducing a new family of dually affine connections pairs $\left(\nabla^{(+\alpha)}, \nabla^{(-\alpha)}\right)$ known as the α-connections (intuitively, a connection describes the way to transform a tangent space into another, defining the way to move on the manifold).

In this new setup, the geodesics are not anymore the shortest paths between two points but rather the curves ensuring parallel transport of vectors between the points. The Levi-Civita connection of the Riemannian Rao geometry is obtained for $\alpha = 0$ (and a geodesic in this geometry is thus both a shortest path curve and an auto-parallel one). With $\left(\nabla^{(+1)}, \nabla^{(-1)}\right)$ we retrieve the dually flat structure of the dual affine parameters (natural and expectation) of the exponential families.

3.3 Centroids

3.3.1 Euclidean centroids

In Euclidean geometry, it is straightforward to define the centroid of a set of n points $\{p_1, \ldots, p_n\}$: it is simply the center of mass $\frac{1}{n}\sum_i p_i$ of the points. A more general case is when the set of points is weighted by a vector $(\omega_1, \ldots, \omega_n)$ (with $\sum_i \omega_i = 1$):

$$c = \sum_i \omega_i p_i \tag{3.66}$$

This Euclidean centroid can also be expressed as the solution of a minimization problem. The centroid is the point which minimizes the average squared Euclidean distances, that is:

$$c = \arg\min_{c'} \sum \omega_i \|c' - p_i\|^2 \tag{3.67}$$

Note 3 (Fermat-Weber point). If we omit the square in the previous formula, we get the Fermat-Weber point [127], that is, the generalization of the median point in any dimension.

When not dealing with Euclidean geometry, different approaches are possible: the two main approaches are the *mean by axiomatization* and the *mean by optimization*.

3.3.2 Mean by axiomatization

When we look for a definition of a centroid, the key point is to build a *representant*, a point which is able to faithfully describe a set of points. Obviously, only one point is not enough to represent a set of numerous points, so the centroid will necessarily be a compromise. One may think of various properties which may be desirable for a centroid: for example one may want the centroid not to depend on the ordering of the points, or the centroid of identical points to be the considered point, etc. We will give here

Name	Expression	f function
Arithmetic mean	$\frac{x_1 + x_2}{2}$	$x \mapsto x$
Geometric mean	$\sqrt{x_1 x_2}$	$x \mapsto \log x$
Harmonic mean	$\frac{2}{\frac{1}{x_1} + \frac{1}{x_2}}$	$x \mapsto \frac{1}{x}$

Table 3.1: Means and their expressions as f-means

the axiomatization first introduced by Kolmogorov [68] and independently by Nagumo [86] in 1930 and later refined by Aczél [1]. These axioms can be described without loss of generality for two non-negative numbers x_1 and x_2 with the centroid function M:

- **Reflexivity**: $M(x, x) = x$;
- **Symmetry**: $M(x_1, x_2) = M(x_2, x_1)$;
- **Continuity and strict monotonicity**: the function $M(.,.)$ is continuous and $M(x_1, x_2) < (x_1', x_2')$ if $x_1 < x_1'$;
- **Anonymity**: $M(M(x_1, x_2), M(x_3, x_4)) = M(M(x_1, x_3), M(x_2, x_4))$. The centroid can be computed with the help of partial centroids computed on subsets of the original points set.

It has been shown that there is only one family of functions which satisfy these axioms, the so-called family of f-means. Each member M_f of this family is parameterized by a strictly monotonous function f and can be written as:

$$M_f(x_1, x_2) = f^{-1}\left(\frac{f(x_1) + f(x_2)}{2}\right) \tag{3.68}$$

Some usual mean function can be expressed as f-means with a well-chosen f function like the arithmetic mean, the geometric mean or the harmonic mean (see Table 3.1 for the corresponding f functions).

The next step is to describe the generalization to an arbitrary number of points.

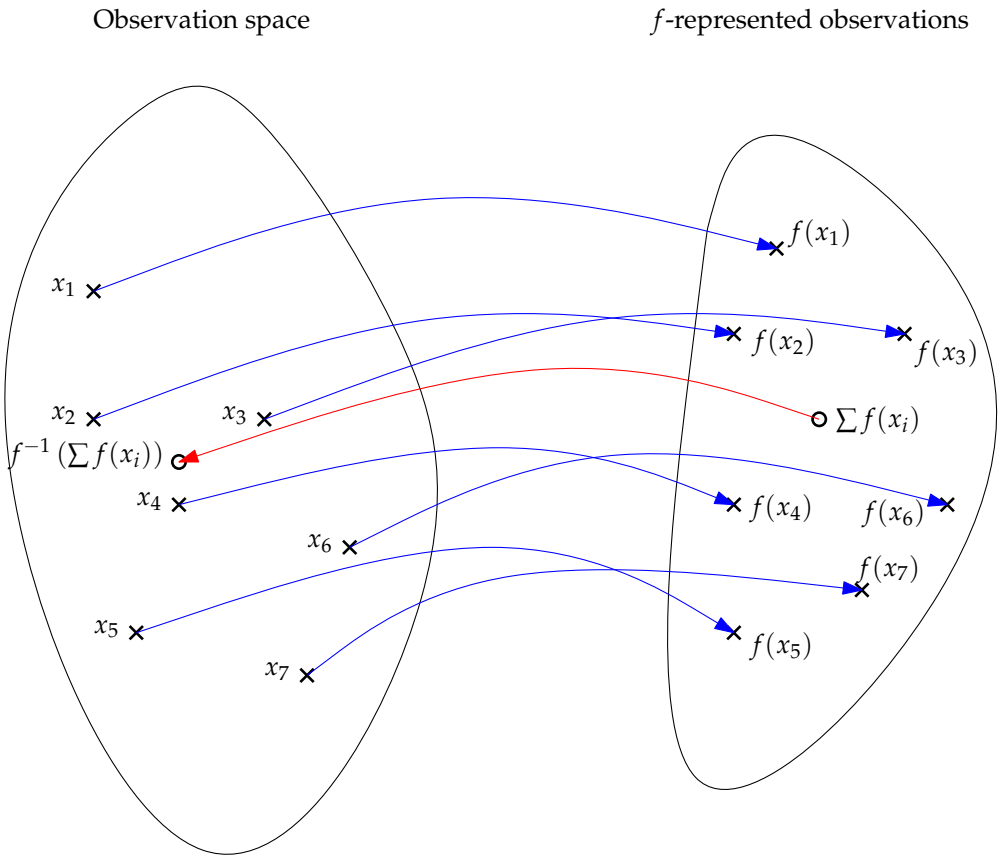


Figure 3.8: Quasi-arithmetic mean. The f-mean is an arithmetic mean in the f-represented space (right): since f is a bijection, we can come back to the original space (left).

Definition 3.14 (f-mean)**.** Given a strictly increasing function f, the f-mean of a set of n points $x_1, \ldots, x_n$ with weights $\omega_1, \ldots, \omega_n$ is:

$$M_f(x_1, \ldots, x_n; \omega_1, \ldots, \omega_n) = f^{-1}\left(\sum_{i=1}^{n} \omega_i f(x_i)\right) \tag{3.69}$$

This centroid can be seen as a *quasi-arithmetic mean* (see Figure 3.8): it is an arithmetic mean on the f-represented numbers $f(x_1), \ldots, f(x_n)$.

3.3.3 Mean by optimization

The other possible point of view is to define the mean as the solution of a minimization problem using a distance function: the mean will be the point which minimizes the average distance between the mean and the other points and thus the solution of the following problem (for some distance-

Name	Expression	f function of the f-divergence
Arithmetic mean	$\sum_i \omega_i x_i$	$x \mapsto -\log x + x$
Geometric mean	$\prod_i (x_i)^{\omega_i}$	$x \mapsto x \log x - x$
Harmonic mean	$\frac{1}{\sum_i \frac{1}{x_i}}$	$x \mapsto (1-x)^2$

Table 3.2: Means and their expressions as entropic means

like function d):

$$\min_x \sum \omega_i d(x, x_i) \tag{3.70}$$

The distance-like function is a dissimilarity measure such that:

$$d(x,y) \begin{cases} = 0 \text{ if } x = y \\ > 0 \text{ if } x \neq y \end{cases} \tag{3.71}$$

Entropic mean

For n positive numbers $x_1, \ldots, x_n$ and weights $\omega_1, \ldots \omega_n$ ($\sum_i \omega_i = 1$), Ben-Tal, Charnes, and Teboulle [16] studied the particular case of the optimization problem given by the family of f-divergences (Definition 3.10):

$$\min_x \sum \omega_i D_f(x, x_i) = \min_x \sum x f\left(\frac{x}{x_i}\right) \mathrm{d}x \tag{3.72}$$

The previous problem is convex and thus admits a unique solution $M_f(x_1, \ldots, x_n)$ which is called the *entropic mean* of the points set.

Proposition 18. *The entropic mean is linear scale-invariant:*

$$M_f(\lambda x_1, \ldots, \lambda x_n) = \lambda M_f(x_1, \ldots, x_n) \tag{3.73}$$

Some common means can be expressed as an entropic mean, see Table 3.2 for some examples.

Bregman centroids

The centroids for the class of Bregman divergences can be studied through mean by optimization. According to the previous definition, the Bregman centroid will be the solution of the following problem:

$$\text{(MEAN)} \quad \min_x \sum \omega_i B_F\left(x \| p_i\right) \tag{3.74}$$

From the convexity of the Bregman divergence with respect to its first argument (see Proposition 12), it follows that this problem admits a unique minimizer. Moreover, it is known [12, 94] that this solution can be expressed in closed-form as a quasi-arithmetic mean:

$$c = \arg\min_x \sum_i \omega_i B_F(x, p_i) \tag{3.75}$$

$$= \nabla F^* \left(\sum_i \omega_i \nabla F(p_i) \right) \tag{3.76}$$

where $F^\star$ is the Legendre dual of the generator F.

But since Bregman divergences are not symmetrical in the general case, we can define one more notion of centroid with the following optimization problem:

$$\text{(MEAN')} \quad \min_x \sum \omega_i B_F\left(p_i \| x\right) \tag{3.77}$$

Even if this problem is not convex in the general case (a counter-example appears with $F(x) = -\log x$), the problem admits a unique solution which will be called the *right-sided Bregman centroid* [94] . Although in the case of f-divergences (which can always be made symmetrical), this new problem leads to the same point as the problem (MEAN), we get for Bregman divergence a different point. Like the *left-sided Bregman centroid* , the right-sided centroid admits a closed-form formula which is simply the center of mass of the points:

$$c^R = \arg\min_x \sum_i \omega_i B_F(p_i, x) \tag{3.78}$$

$$= \sum_i \omega_i p_i \tag{3.79}$$

Remark 6. Although these two results on left-sided and right-sided are essential since it allows efficient algorithms using Bregman centroids, the practical efficiency of the closed-form formula may be limited if an expression for $\nabla F^* = (\nabla F)^{-1}$ is not available (thus requiring numerical approximation).

The use of asymmetric divergences may not be desirable for some applications but each time we have an asymmetric divergence, it is possible to build a new symmetric divergence. For example, some work has been done on Bregman metrization [28] but the most straightforward is however the Jeffreys approach [60] known as *Jeffreys-Bregman divergences* in the case of Bregman divergences [94]:

$$S_F(p,q) = \frac{B_F\left(p\|q\right) + B_F\left(q\|p\right)}{2} \tag{3.80}$$

Another possible approach is inspired by the Jensen divergence [26] and is called *Jensen-Bregman divergence* [92]:

$$J_F(p,q) = \frac{1}{2}\left(B_F\left(p\|\frac{p+q}{2}\right) + B_F\left(\frac{p+q}{2}\|q\right)\right) \tag{3.81}$$

The Jensen-Bregman divergence is actually the *Burbea-Rao divergence* generated by F [92]:

$$\begin{aligned}
J_F(p,q) =& \frac{1}{2}\left(B_F\left(p\|\frac{p+q}{2}\right) + B_F\left(\frac{p+q}{2}\|q\right)\right) && (3.82)\\
=& \frac{1}{2}\left(F(p) - F(\frac{p+q}{2}) - \langle p - \frac{p+q}{2}, \nabla F(\frac{p+q}{2})\rangle + \right. && (3.83)\\
& \left. F(\frac{p+q}{2}) - F(q) - \langle \frac{p+q}{2} - q, \nabla F(q)\rangle\right)\\
=& \frac{1}{2}\left(F(p) - F(q) - F\left(\frac{p+q}{2}\right)\right) && (3.84)\\
=& \mathrm{BR}_F(p,q) && (3.85)
\end{aligned}$$

It is obvious that these two new families of divergences are symmetrical but the main drawback is that the centroids are not known is closed-form anymore [94]. Actually, these two symetrizations can be generalized by replacing the $\frac{1}{2}$ coefficient by an α between 0 and 1, leading to the skewed versions described in Section 3.4.2.

3.4 Information-geometric divergences

In addition to the already cited divergences, there is an impressive number of families of divergences which rely on different foundations. We give here just a quick look but an extensive review by Basseville is available in [14] (the *Encyclopedia of Distances* by Deza and Deza [40] provides an even larger view of the notion of distance).

3.4.1 Bhattacharyya coefficient

The *Bhattacharyya coefficient* [18] allows to measure the amount of overlapping between two distributions p and q:

$$C(p,q) = \int \sqrt{p(x)q(x)}\mathrm{d}x \tag{3.86}$$

This coefficient is not a divergence but it is rather a similarity measure since it falls in the interval $[0,1]$ with 1 as the value obtained for maximal similarity and 0 obtained for minimal similarity. It can be changed into a divergence with the *Bhattacharyya distance* [18] :

$$B(p,q) = -\log C(p,q) \tag{3.87}$$

But this is still not a metric. A metrized version is available through the *Hellinger metric* [56] (or Matusita metric [79]) which can be expressed in terms of the Bhattacharyya coefficient:

$$H(p,q) = \sqrt{\frac{1}{2}\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 \mathrm{d}x} \tag{3.88}$$

$$= \sqrt{\frac{1}{2}\int p(x) + q(x) - 2\sqrt{p(x)q(x)}\mathrm{d}x} \tag{3.89}$$

$$= \sqrt{1 - C(p,q)} \tag{3.90}$$

An important result is that the Bhattacharyya distance between two members of the same exponential family can be computed using a Burbea-Rao divergence between the parameters of the distributions [92]:

$$B\left(p_F\left(x,\theta_p\right), p_F\left(x,\theta_q\right)\right) = \mathrm{BR}_F\left(\theta_p,\theta_q\right) \tag{3.91}$$

3.4.2 Skew divergences

Let us recall the expression of the Jeffreys-Bregman and Jensen-Bregman divergences:

$$S_F(p,q) = \frac{B_F\left(p\|q\right) + B_F\left(q\|p\right)}{2} \tag{3.92}$$

$$J_F(p,q) = \frac{1}{2}\left(B_F\left(p\|\frac{p+q}{2}\right) + B_F\left(\frac{p+q}{2}\|q\right)\right) \tag{3.93}$$

These two expressions divergences extend naturally to two new families of divergences parameterized by a real number between 0 and 1 which are called skew Jeffreys-Bregman divergences and skew Jensen-Bregman divergences:

Definition 3.15. Given $\alpha \in [0,1]$, the *skew Jeffreys-Bregman divergence* between p and q is:

$$S_F^{(\alpha)}(p\|q) = \alpha B_F(p\|q) + (1-\alpha)B_F(q\|p) \tag{3.94}$$

Definition 3.16. Given $\alpha \in [0,1]$, the *skew Jensen-Bregman divergence* between p and q is:

$$J_F^{(\alpha)}(p\|q) = \alpha B_F(p\|\alpha p + (1-\alpha)q) + (1-\alpha)B_F(\alpha p + (1-\alpha)q\|q) \tag{3.95}$$

The equality between the Jeffreys-Bregman divergence and the Burbea-Rao divergence remains true by defining a skew version of the Burbea-Rao divergence:

Definition 3.17. Given $\alpha \in [0,1]$, the *skew Burbea-Rao divergence* between p and q is:

$$\mathrm{BR}_F^{(\alpha)}(p\|q) = \alpha F(p) + (1-\alpha)F(q) - F(\alpha p + (1-\alpha)q) \tag{3.96}$$

With this new skew divergence, we have thus the equality $J_F^{(\alpha)}(p\|q) = \mathrm{BR}_F^{(\alpha)}(p\|q)$.

3.5 Conclusion

This chapter presented an introduction course to the field of information geometry. Using this background, the next chapter will describe some algorithms on mixture model with a computational information geometry point of view.

Chapter 4

Statistical mixture models

4.1 Definition

Finite mixture models [81] are a tool of choice to model unknown probability density functions (pdf) in a lot of applications (image and video retrieval [8], classification of gene expression on microarrays [19], telecommunication network modeling [5], ultrasound image analysis [122], etc). The pdf of a mixture model can be written as a weighted sum of n parametric distributions (called the *components* of the mixture):

$$m(x) = \sum_{i=1}^{n} \omega_i f_i(x) \tag{4.1}$$

where the weights $(\omega_i)_i$ are a vector of positive reals with $\sum_i \omega_i = 1$ (to ensure normalization of the mixture distribution).

Usually, all the f_i come from the same distribution: the most common case is certainly the mixture of Gaussian in which each component is a Gaussian distribution. We have in this case:

$$m(x; \mu, \sigma, \omega) = \sum_i \omega_i \, \mathcal{N}(x; \mu_i, \sigma_i^2) \tag{4.2}$$

where $\mathcal{N}$ is the normal distribution.

Proposition 19 (Universality of Gaussian mixtures [32])**.** *Any smooth density with $\mathbb{R}$ for support can be approached within arbitrary precision with a sufficient number of components.*

The previous property justifies the wide use of the Gaussian mixtures to model arbitrary densities. However, other densities have this property of universality: for example, the Gamma mixtures are universal on $\mathbb{R}^+$.

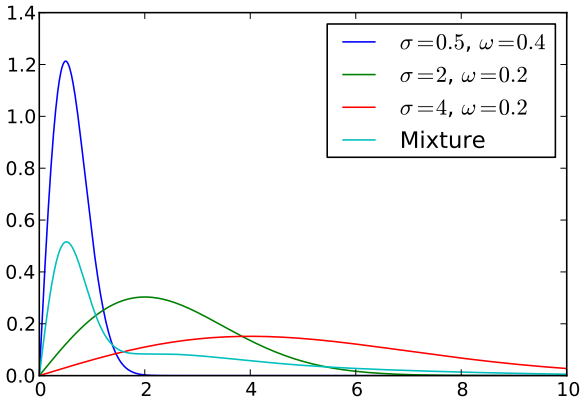


Figure 4.1: A Rayleigh mixture model.

Even if Gaussian mixtures are the most widespread kind of mixture models, a lot of other mixtures have been of use for various applications: mixtures of Laplace distributions [8], of Gamma laws [80], of Rayleigh [122], of Wishart [52], of Poisson [73], of Bernoulli [9] and many others.

We focus here on *mixtures of exponential families*, a particular case of mixture for which each component is an exponential family (the same for each component). The pdf is now expressed as:

$$m_f(x;\theta,\omega) = \sum_i \omega_i \, p_F(x;\theta_i) \tag{4.3}$$

where p_F is an exponential family with log-normalizer F as described in Chapter 3.

The use of a generic mixture such as a mixture of exponential families allows to build algorithms and methods which are generic for any exponential family, contrary to specifically tailored algorithms which have been used in many applications.

Remark 7. When learning an unknown mixture model from observed data points, it is obvious that one of the goals will be to estimate the parameter vectors θ and ω. One should not forget that the number of components is also a parameter of the distribution and is often the most challenging value to determinate.

4.2 A generative model

A mixture model is able to describe an unknown distribution but it is also a *generative model* : from the model, one can generate points following the

(a) Original image (b) Representation of the Gaussians of the mixture (c) Statistical image

Figure 4.2: Statistical image: the `RGBxy` point set is modeled with a 5d mixture of Gaussian (ellipsoids show the marginal laws on `xy` with the `RGB` mean for color) and points are sampled from this mixture. The mixture has lost all the low-frequency information from the original image. This output has been produced with the `jMEF` Java library [99] for mixtures of exponential families.

distribution of the mixture model (see Figure 4.2 for an illustration of the process). The sampling is done with a doubly stochastic process:

1. Choose a component according to the weights.
2. Draw a point from the distribution of the chosen component.

The first step, choosing the component, is made by drawing a random number from a multinomial distribution with n possible outcomes weighted by the vector ω. The second step is made by any method able to draw points from the components distribution.

Remark 8. It is thus straightforward to draw points from a mixture model provided we have an algorithm to draw points from the components. If this algorithm is not available, one can rely on generic drawing methods such as the rejection method [87] or the Metropolis-Hastings algorithm [82, 55].

4.3 Mixtures as Hidden Markov Models

When points are drawn from the algorithm described in the previous section, we know the component from which the sample comes. However, in the general case of an unknown density modeled by a mixture model, there is uncertainty about the origin of each observation: given a point it is not possible to know the component from which it comes. In this case, the origin of each sample is said to be a *hidden variable* (or a *latent variable*).

The Expectation-Maximization algorithm described in 4.4.1 has been originally proposed by Dempster, Laird, and Rubin [38] to learn parameters of a model with missing observations, another kind of hidden variable.

Mixtures are actually a special case of a more generic class of models called *Hidden Markov Models* [15]: in such a model, instead of having independence in the choice of the origin distribution, the latent variables are thus related by a Markov process.

4.4 Learning mixtures

Since mixture models are a commonly used set of tools for many practical applications, a lot of algorithms to build mixture models have been proposed. A selection of these methods is presented here, without extensivity in mind: the goal is to present some common algorithms which will be used as a basis for most of the new approaches introduced in the next parts.

For each of the following algorithms, the goal will be to learn a mixture model with n components given a set of N observations x_i.

4.4.1 Expectation-Maximization

The general case of the *Expectation-Maximization* (EM) algorithm [38] looks for the parameters of a model which maximizes the likelihood of the model when the considered kind of models depends on hidden variables. As explained in Section 4.3, this is exactly the case of a mixture model.

EM is an iterative algorithm which converges to a local maximum of the likelihood of the mixture [38]. It consists into two parts:

- the *expectation* step (E-step) estimates the expected likelihood given the current estimate of the parameters;
- the *maximization* step (M-step) updates the parameters in order to optimize the expected likelihood.

These two steps are iterated until the convergence of the likelihood has been reached.

Expectation step The posterior probability for the i-th component of the mixture given the observation x_t and the current estimate λ of the parameters is:

$$p(i|x_t, \lambda) = \frac{\omega_i p(x_t; \lambda_i)}{\sum_{j=1}^{k} \omega_j p(x_t; \lambda_j)} \tag{4.4}$$

Maximization step For the component i of the mixture, the new weights are:

$$\omega_i = \frac{1}{N} \sum_{t=1}^{N} p(i|x_t, \lambda) \tag{4.5}$$

and the update of the parameters is:

$$\lambda_i = \arg\max_{\lambda'} \sum_t p(i|x_t, \lambda) \log p(x_t; \lambda') \tag{4.6}$$

Remark 9 (Initialization). Since EM is an iterative algorithm, it heavily depends on the initialization (here, an initial guess for the parameters and the weights of the component). It is known that a bad initialization for EM can lead to an arbitrary bad estimate so it is a crucial point for real applications (a popular choice is to cluster the observations using k-means and learn a MLE on each cluster).

4.4.2 Bregman Soft Clustering

Even if the principle of EM is not limited to Gaussian mixtures, its application needs to manually compute the formula used in steps E and M. A more satisfying solution would be to have the distribution as a parameter of the algorithm: this is the idea of the Bregman Soft Clustering algorithm [12] which allows to build mixtures of exponential families (the family of the components is now simply a parameter of the algorithm).

The Bregman Soft Clustering relies heavily on the bijection between Exponential families and Bregman divergences. With this bijection, the E- and M-steps can be rewritten in a generic way:

Expectation step If we replace each appearance of an exponential family in the posterior probability by its expansion using a Bregman divergence, we get:

$$p(i|x_t, \eta) = \frac{\omega_i \exp\left(-B_{F^\star}\left(t(x_i) \| \eta_i\right)\right) \exp k(x_t)}{\sum_{j=1}^{k} \omega_j \exp\left(-B_{F^\star}\left(t(x_t) \| \eta_j\right)\right) \exp k(x_t)} \tag{4.7}$$

$$= \frac{\omega_i \exp\left(-B_{F^\star}\left(t(x_t) \| \eta_i\right)\right)}{\sum_{j=1}^{k} \omega_j \exp\left(-B_{F^\star}\left(t(x_t) \| \eta_j\right)\right)} \tag{4.8}$$

$$\tag{4.9}$$

Since $B_{F^\star}(p\|q) = F^\star(p) - F^\star(q) - \langle p - q, \nabla F^\star(q) \rangle$ we can expand the expression of the Bregman divergence in the previous equation:

$$p(i|x_t, \eta) = \frac{\omega_i \exp\left(-F^\star(t(x_t)) - F^\star(\eta_i) - \langle t(x_t) - \eta_i, \nabla F^\star(\eta_i)\rangle\right)}{\sum_{j=1}^{k} \omega_j \exp\left(-F^\star(t(x_t)) - F^\star(\eta_j) - \langle t(x_t) - \eta_j, \nabla F^\star(\eta_j)\rangle\right)} \tag{4.10}$$

$$= \frac{\omega_i \exp\left(F^\star(\eta_i) + \langle t(x_t) - \eta_i, \nabla F^\star(\eta_i)\rangle\right)}{\sum_{j=1}^{k} \omega_j \exp\left(F^\star(\eta_j) + \langle t(x_t) - \eta_j, \nabla F^\star(\eta_j)\rangle\right)} \tag{4.11}$$

Maximization step This step relies on the generic maximum likelihood estimator for exponential families [12]. The weights estimations are unchanged and the parameters which maximize the likelihood are the following:

$$\omega_i = \frac{1}{N} \sum_{t=1}^{N} p(i|x_t, \eta) \tag{4.12}$$

$$\eta_i = \sum_{t=1}^{N} \frac{p(i|x_t, \eta)}{\sum_{t=1}^{N} p(i|x_t, \eta)} t(x_t) \tag{4.13}$$

Remark 10. It justifies the naming *Bregman Soft Clustering*. The E-step and the weights estimations are equivalent to a soft assignment and the estimation of the new η_i parameters is a Bregman centroid.

The Bregman Soft Clustering algorithm is a particular instance of the EM framework applied to exponential families. It is more generic than the classical applications of EM to mixtures since any exponential family can be passed as a parameter of the algorithm. There are however some limitations: to use an EF, one needs to know some functions from the decomposition of the family (see Annex B). The functions used are:

- the sufficient statistic t,
- the dual of the log-normalizer $F^\star$,
- and its gradient $\nabla F^\star$.

The problem comes from the two functions $F^\star$ and $\nabla F^\star$: these functions may not be known in closed-form (see Section 3.1.6). For distributions where these formulas are not available in closed-form, the numerical scheme needed at each step of the algorithm may render it unusable in practice.

4.4.3 Kernel density estimators

Strictly speaking, a Kernel Density Estimator (KDE, also known as Parzen window method [100, 109]) is not a mixture model: it is a non-parametric estimator and there is no concept of a latent hidden variable involved. However, if we only look at the expression of the pdf, it can be seen as a mixture. The generic expression of an estimator built with the Parzen window method is the following:

$$m(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} K\left(\frac{x - x_i}{h}\right) \tag{4.14}$$

where K is known as the kernel function (a symmetric and normalized to 1 function) and $h \in \mathbb{R}^+$ is known as the bandwidth. In other words, building a KDE means putting a kernel centered on each of the N observations (leading to a sum with N terms).

Various kernels have been proposed in the literature (see Table 6.1 in Chapter 6 for some examples) but for convenience reasons the Gaussian kernel is often used (actually, a standard Gaussian law with mean 0 and variance 1).

Contrary to the choice of the kernel which is often not critical, the bandwidth is the most important parameter of the KDE. This parameter controls the smoothness of the density and is a compromise between a very smooth curve where all information has been lost and a sharp curve with thin peaks on each observation (see Figure 6.1 in Chapter 6 for the impact of the bandwidth on the quality of the estimation).

Due to its importance, a lot of work is devoted to the choice of this parameter but a consensus has been reached for now with two main methods: plugin selector [123, 110] and cross validation [23, 53].

4.5 Comparing mixtures

4.5.1 Kullback-Leibler divergence between mixtures

It is of interest in a lot of applications to compare mixture models. The well-known Kullback-Leibler (KL) divergence would be a tool of choice but unfortunately, there is no closed-form formula available for mixtures of Exponential families. For two mixtures m_1 with n_1 components and m_2 with n_2 components, the KL divergence becomes:

$$\mathrm{KL}\left(m_1 \| m_2\right) = \int m_1(x) \log \frac{m_1(x)}{m_2(x)} \mathrm{d}x \tag{4.15}$$

$$= \int \sum_i \omega_i^{(1)} p_F(x; \theta_i^{(1)}) \log \frac{\sum_i \omega_i^{(1)} p_F(x; \theta_i^{(1)})}{\sum_i \omega_i^{(2)} p_F(x; \theta_i^{(2)})} \mathrm{d}x \tag{4.16}$$

Except for the degenerated case where the two mixtures have exactly one component (in this case, the KL divergence between the two mixtures amounts to a KL divergence between two exponential families, that is the Bregman divergence between the parameters), there is no closed-form available to compute the divergence[1]. Due to the importance of the KL divergence in many applications, a lot of different methods have been proposed in the literature.

4.5.2 Some Kullback-Leibler approximations

It has been proposed in [57] (and also [42]) an extensive review of the existing KL approximations, we simply detail here a few of them. Each method can be evaluated using various criteria depending on the application and the goals:

- Error of the approximation
- Computation time
- Does the similarity property ($\mathrm{KL}\left(p \| p\right) = 0$) hold ?
- Does the identification property ($\mathrm{KL}\left(p \| q\right) = 0$ iff $p = q$) hold ?
- Does the positivity property ($\mathrm{KL}\left(p \| q\right) \geq 0$ for any p and q) hold ?

Monte-Carlo sampling

One of the most used method to estimate the Monte-Carlo sampling is a Monte-Carlo method which relies on the following observation:

$$\mathrm{KL}\left(p \| q\right) = \int p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x \tag{4.17}$$

$$= E_p\left[\log \frac{p}{q}\right] \tag{4.18}$$

From a set of i.i.d. points $x_i, \ldots, x_n$ sampled from the mixture p we can estimate the expectation $E_p\left[\log \frac{p}{q}\right]$ and get an approximation of the

[1] It does not mean the closed-form does not exist, the problem is still open.

Kullback-Leibler. We thus define the Monte-Carlo Kullback-Leibler divergence as:

$$\widetilde{\mathrm{KL}}_{\mathrm{MC}}(p\|q) = \frac{1}{n}\sum_i \log\frac{p(x_i)}{q(x_i)} \tag{4.19}$$

$$\underset{n\to\infty}{\longrightarrow} \mathrm{KL}\,(p\|q) \tag{4.20}$$

The big advantage of the Monte-Carlo approximation is that it is the only method which converges to the KL divergence [57]: it is thus the method of choice when a precise approximation is required. However, this property comes with a price: a lot of samples are needed in order to get a good estimation of the divergence (a common value is 10^6 [42], 10^5 still gives a significant deviation from the true value [57]).

When used with a lot of samples (there is no point in trying to speed up the method with few samples since better methods are available when speed matters), the properties of the Monte-Carlo approximation are the following:

- precise,
- slow,
- similarity holds,
- identification holds in practice (but may fail with probability 0 on unlikely examples),
- positivity does not hold.

Goldberger approximation

For two mixtures which share the same number of components, an upper bound can be derived from the log-sum inequality:

$$\sum a_i \log\frac{\sum a_i}{\sum b_i} \leq \sum a_i \log\frac{a_i}{b_i} \tag{4.21}$$

when $\sum a_i = 1$ and $\sum b_i = 1$.

A first inequality can thus be written:

$$\mathrm{KL}\left(m_1 \| m_2\right) = \int m_1 \log \frac{m_1}{m_2} \tag{4.22}$$

$$= \int \left(\sum_{i=0}^{n} \omega_i^{(1)} p_i^{(1)} \right) \log \frac{\sum_{i=0}^{n} \omega_i^{(1)} p_i^{(1)}}{\sum_{i=0}^{n} \omega_i^{(2)} f_i^{(1)})} \mathrm{d}x \tag{4.23}$$

$$\leq \int \sum \omega_i^{(1)} p_i^{(1)} \log \frac{\omega_i^{(1)} p_i^{(1)}}{\omega_i^{(2)} p_i^{(2)}} \tag{4.24}$$

$$= \int \sum \omega_i^{(1)} p_i^{(1)} \left(\log \omega_i^{(1)} + \log p_i^{(1)} - \log \omega_i^{(2)} - \log p_i^{(2)} \right) \tag{4.25}$$

$$= \sum \left(\int p_i^{(1)} \right) \omega_i^{(1)} \log \frac{\omega_i^{(1)}}{\omega_i^{(2)}} + \sum \omega_i^{(1)} \int p_i^{(1)} \log \frac{p_i^{(1)}}{p_i^{(2)}} \tag{4.26}$$

$$= \sum \omega_i^{(1)} \log \frac{\omega_i^{(1)}}{\omega_i^{(2)}} + \sum \omega_i^{(1)} \int p_i^{(1)} \log \frac{p_i^{(1)}}{p_i^{(2)}} \tag{4.27}$$

$$\mathrm{KL}\left(m \| m_2\right) \leq \mathrm{KL}\left(\omega^{(1)} \| \omega^{(2)}\right) + \sum \omega_i^{(1)} \mathrm{KL}\left(p_i^{(1)} \| p_i^{(2)}\right) \tag{4.28}$$

Note 4. The notation $\mathrm{KL}\left(\omega^{(1)} \| \omega^{(2)}\right)$ is the KL divergence between normalized arrays (or equivalently the KL divergence between multinomial laws).

The previous inequality has been used in some applications like texture classification [41]. However, a slightly better upper-bound can be obtained by noticing that the order of the components of a mixture does not matter:

$$m(x) = \sum_i \omega_i p_i(x) \tag{4.29}$$

$$= \sum_i \omega_{\sigma(i)} p_{\sigma(i)}(x) \tag{4.30}$$

for any permutation σ.

It is straightforward to see that Inequality (4.28) is true for any permutation of the components of one of the mixture: we thus have a better upper-bound by taking the permutation which minimizes the value. The Goldberger approximation is thus defined as:

$$\widetilde{\mathrm{KL}}_{\mathrm{Goldberger}}(p \| q) = \arg\min_{\sigma} \mathrm{KL}\left(\omega^{(1)} \| \sigma(\omega^{(2)})\right) + \sum \omega_i^{(1)} \mathrm{KL}\left(p_i^{(1)} \| p_{\sigma(i)}^{(2)}\right) \tag{4.31}$$

$$= \arg\min_{\sigma} \sum \omega_i^{(1)} \left(\log \frac{\omega_i^{(1)}}{\omega_i^{(2)}} + \mathrm{KL}\left(p_i^{(1)} \| p_{\sigma(i)}^{(2)}\right) \right) \tag{4.32}$$

This problem of finding the best permutation is known as the *assignment problem* (and can be seen as a discrete case of the optimal transport theory) and a solution can be found in polynomial time $O(n^3)$ [43] using the *Kuhn-Munkres algorithm* [71, 84] also known as the *Hungarian algorithm*.

Note 5 (Historical note). This algorithm has a lot of names: Kuhn introduced the algorithm in 1955 and called it the *Hungarian algorithm* since it used early work by two Hungarian mathematicians Kőnig and Egerváry. Munkres then published a study of its complexity in 1957 [84] which popularized the name Kuhn-Munkres algorithm or even simply Munkres algorithm. However it deserves to be known that some forgotten work [59] by Jacobi have been recently rediscovered in 2006 and show that he already proposed a solution during the 19th century [2].

Properties of the Goldberger approximation can be summarized as follows:

- upper bound (but not a very good one [99]),
- fast with a small number of components (unusable on a kernel density estimator),
- similarity does not hold,
- identification does not hold,
- positivity holds.

Variational approximation

The variational approximation [57] comes from the Jensen inequality and provides a closed-form approximation of the Kullback-Leibler divergence which can be expressed as:

$$\widetilde{\mathrm{KL}}_{\mathrm{Variational}}(p\|q) = \sum_{i=1}^{n_1} \omega_i^{(1)} \log \frac{\sum_{i'=1}^{n_1} \omega_{i'}^{(1)} \exp\left(-\mathrm{KL}\left(p_i^{(1)}\|p_{i'}^{(1)}\right)\right)}{\sum_{j=1}^{n_2} \omega_j^{(2)} \exp\left(-\mathrm{KL}\left(p_i^{(1)}\|p_j^{(2)}\right)\right)} \quad (4.33)$$

The variational approximation has the following properties:

- accurate,
- fast (closed-form),
- similarity holds,

[2] See `http://www.lix.polytechnique.fr/~ollivier/JACOBI/jacobiEngl.htm` for a history of this rediscovery and translation of the original Latin articles.

- identification does not hold,
- positivity does not hold.

In addition to these properties, $\widetilde{\mathrm{KL}}_{\mathrm{Variational}}$ is interesting since gradient with respect to p or q can be easily computed which can be of use for optimization problems. This approximation can be seen as an extension of the Goldberger approximation where variational coefficients are extension of the permutation used in $\widetilde{\mathrm{KL}}_{\mathrm{Goldberger}}$.

There is also an upper-bound version of the variational approximation which is not in closed-form anymore [57].

Choosing an approximation

The choice of the approximation method depends on the application and on the needed properties. If precision is the primary concern, the Monte-Carlo method is obviously the best choice, since it is the only one which converges to the true value, when the number of samples grows. This precision comes with the price of a high computational cost: when speed matters, it is probably better to use a closed-form approximation. Among these (which are not all described here), the variational approximation is the most accurate [57].

4.5.3 Closed-form distances between mixtures

Along with the work to build better approximations of the Kullback-Leibler divergence, there are also some new divergences which are available in closed-form [89] for mixtures of exponential families such as the Cauchy-Schwartz divergence [62, 63], the Jensen-Rényi divergence [126] and the total Square Loss [74].

4.6 Conclusion

We recalled in this chapter the basics on mixture model, and more particularly on mixture of exponential families. This serves as a foundation for the contributions presented in the next two parts.

Part II

From kernels to mixtures with simplification

This part presents results already published in the ICASSP 2012 article "Model centroids for the simplification of Kernel Density estimators" [116] and in a chapter "Learning Mixtures by Simplifying Kernel Density Estimators" [115] of the book *Matrix Information Geometry*[97]. A previous version of the software library was presented in [118]. The full bioinformatics application (in which most part is not related to statistical models and thus not relevant here) is described in details in [124].

This work has been developed in collaboration with Julie Bernauer (LIX - Amib team, École Polytechnique - Inria Saclay) who was the pillar of the bioinformatics applications presented in [124].

Chapter 5

Introduction NONUMBER

Statistical methods are nowadays commonplace in modern signal processing. There are basically two major approaches for modeling experimental data by probability distributions: we may either consider a semi-parametric modeling by a *finite* mixture model learned using the famous Expectation-Maximization (EM) procedure, or alternatively choose a non-parametric modeling using a Kernel Density Estimator (KDE).

On the one hand mixture modeling requires to fix or learn the number of components but provides a useful compact representation of data. On the other hand, KDE finely describes the underlying empirical distribution at the expense of a dense model size. In this part, we present a novel statistical modeling method that simplifies efficiently a KDE model with respect to an underlying distance between Gaussian kernels. We consider the Fisher-Rao metric and the Kullback-Leibler divergence. Since the underlying Fisher-Rao geometry of Gaussians is hyperbolic without a closed-form equation for the centroids, we rather adopt an approximation that bears the name of *hyperbolic model centroid*, and show its use in a *single-step clustering method*. We report on our experiments that the KDE simplification paradigm is a competitive approach over the classical EM, in terms of both processing time and quality.

In Chapter 6 we present the simplification algorithms: we begin with a description of the kernel density estimation method, then we introduce the Bregman Hard Clustering and Model Hard Clustering algorithms and their applications to kernel density estimators simplifications.

In Chapter 7, we describe our new software library `pyMEF` aimed at the manipulation of mixtures of exponential families. The goal of this library is to unify the various tools used to build mixtures which are usually limited to one kind of exponential family. The use of the library is further explained with a short tutorial.

In Chapter 8, we study experimentally the performance of our methods through two applications. First, we experimentally check the convergence

of Model Hard Clustering. Next, we give a simple example of the modeling of the intensity histogram of an image which shows that the proposed methods are competitive in terms of log-likelihood. Last, a real-world application in bio-informatics is presented where the models built by the proposed methods are compared to reference state-of-the-art models built using Dirichlet Process Mixtures.

Chapter 6

Simplification

6.1 Kernel density estimators

6.1.1 Definition

The *Kernel density estimation* (KDE) method has been independently proposed by Parzen [100] (hence the common name *Parzen window method*) and Rosenblatt [109]. It is a non-parametric method to estimate an unknown density from a set of observations sampled from a random variable. Strictly speaking this kind of estimator is thus not a mixture model (which is semi-parametric and which involves hidden variable). However, the pdf coming from a KDE can be expressed as a non-weighted sum of components. For a set of N observations $(x_i, \ldots, x_N)$ sampled from the density we want to estimate, the kernel density estimator is:

$$m(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} K_h(x - x_i) \tag{6.1}$$

The function K_h is called the *kernel* and the parameter h is called the *bandwidth*.

6.1.2 Kernels

Definition 6.1. A *kernel* is a non-negative function K which is

- integrable and normalized: $\int K(x)\mathrm{d}x = 1$,
- symmetric: $K(-x) = K(x)$ for all x.

The scaled kernel K_h used in Eq. (6.1) is expressed as:

$$K_h(x) = \frac{1}{h} K\left(\frac{x}{h}\right) \tag{6.2}$$

Name	Expression	Comment
Epanechnikov	$\frac{3}{4}(1-x^2)$ if $\lvert x\rvert \leq 1, 0$ otherwise	Optimal
Triangular	$(1-\lvert x\rvert)$ if $\lvert x\rvert \leq 1, 0$ otherwise	
Quartic	$\frac{15}{16}(1-x^2)^2$ if $\lvert x\rvert \leq 1, 0$ otherwise	
Gaussian	$\mathcal{N}(0,1)$	Exponential family

Table 6.1: Some common kernels.

A large range of kernels has been proposed in the literature: the *Epanechnikov kernel*

$$K(x) = \begin{cases} \frac{3}{4}(1-x^2) & \text{if } |x| \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

is the optimal kernel in the sense of the mean-squared error [44] but the other common kernels (triangular, quartic, cosine, etc, see Table 6.1) are known to be nearly as efficient in practice and the Gaussian kernel is very often used in many applications. Since we deal with exponential families, the most interesting for us is the Gaussian kernel (any EF which is symmetrical may actually be used, such as the Laplace law, but they are not widespread in applications).

6.1.3 Bandwidth

The choice of the kernel is often not critical and will not really influence the quality of the estimation. However, the choice of the other parameter, the *bandwidth*, is critical. This parameter controls the smoothness of the estimator: a too small value will produce a KDE with sharp kernels centered on each point and the values of the density outside the nearby neighborhood of the observations will not be meaningful; a too large value will produce a very smooth curve where all the meaningful information will be lost (see Figure 6.1 for an illustration of the influence of the bandwidth). The extensive study by Sheather and Jones [123] allowed to get a consensus for the best bandwidth selection method.

6.2 Simplification of kernel density estimators

Since mixtures with a low number of components have proved their capacity to model complex data (Figure 6.2), it would be useful to build a mixture avoiding the costly learning step of EM.

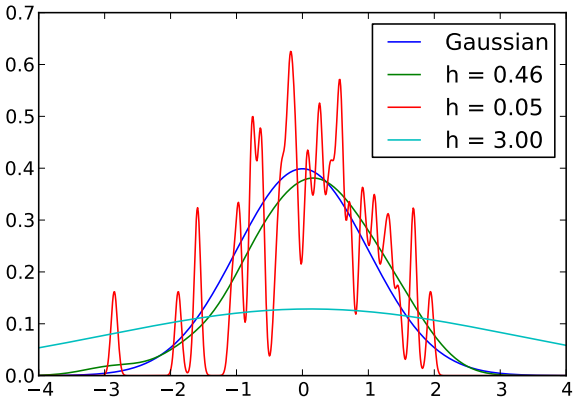


Figure 6.1: Influence of the bandwidth on the KDE. A too low value (red curve) gives meaningless peaks, a too large value (cyan curve) gives a large support density where information is diluted. The value in between (green curve) gives a good compromise.

We rely here on a clustering procedure, using a k-means like method, and we present two different variants of the simplification procedure. The first one is the Bregman Hard Clustering algorithm as introduced in [48] for the simplification of mixture models: it exploits the dually flat geometry induced by the Bregman divergence and converges monotonically to a local optimum of the cost function. The second one is a variant which makes use of the hyperbolic geometry induced by the Fisher-Rao distance (for the Gaussian law): instead of the true Fisher-Rao centroids which are computationally not tractable, we use the model centroids by Galperin [47]. Since we do not use the centroid associated to the distance in the k-means, we do not have the local convergence property, however the algorithm converges in practice.

6.2.1 Bregman Hard Clustering

The Bregman Hard Clustering algorithm is an extension of the celebrated k-means clustering algorithm to the class of Bregman divergences [12]. It has been proposed in Garcia, Nielsen, and Nock [48] to use this method for the simplification of mixtures of exponential families. Similarly to the Lloyd k-means algorithm, the goal is to minimize the following cost function, for the simplification of n components mixture to a k components mixture (with $k < n$):

$$L = \min_{\theta'_1,\dots,\theta'_k} \sum_{1<j\leq k} \sum_{i} B_F\left(\theta'_j \| \theta_i\right) \tag{6.4}$$

where F is the log-normalizer of the considered exponential family, the θ_i are the natural parameters of the source mixture and the θ'_j are the natural parameters of the target mixture.

With the bijection between exponential families and Bregman divergences, the cost function L can be written in terms of Kullback-Leibler divergences:

$$L = \min_{c_1,\ldots,c_k} \sum_{1<j\leq k} \sum_i \mathrm{KL}\left(x_i \| c_j\right) \tag{6.5}$$

where the x_i are the components of the original mixture and the c_j are the components of the target mixture. With this reformulation, the Bregman Hard Clustering is shown to be a k-means with the Kullback-Leibler divergence (instead of the usual L_2-based distance). As in the L_2 version, the k-means involves two steps: assignment and centroid updates. The centroids of the cluster are here computed using the closed-formula presented in Section 3.3.3.

Though left-, right-sided and symetrized formulations of this optimization problem can be used, it has been shown experimentally in [48] that the right-sided Bregman Hard Clustering performs better in terms of Kullback-Leibler error. This experimental result is explained theoretically by a theorem stating that the right-sided centroid is the best single-component approximation of a mixture model, in terms of Kullback-Leibler divergences. Introduced by Pelletier [101], a complete and more precise proof of this result is given in the following section.

6.2.2 Kullback-Leibler centroids as geometric projections

Pelletier proved ([101], Theorem 4.1) that the right-sided KL centroid $\bar{p}^*$ can be interpreted as the *information-theoretic projection* of the mixture model distribution $\tilde{p} \in \mathcal{P}$ onto the model exponential family sub-manifold $\mathcal{E}_F$:

$$\bar{p}^* = \arg\min_{p\in\mathcal{E}_F} \mathrm{KL}\left(\tilde{p}\|p\right) \tag{6.6}$$

Since the mixture of exponential families is not an exponential family ($\tilde{p} \notin \mathcal{E}_F$),[1] it yields a neat interpretation: the best KL approximation of a mixture of components of the same exponential family is the exponential family member defined using the right-sided KL centroid of mixture parameters.

Let θ_i^j for $j \in \{1,\ldots,d\}$ be the d coordinates in the primal coordinate system of parameter θ_i.

Let us write for short $\theta = \theta(p)$, and $\bar{\theta}^* = \theta(\bar{p}^*)$ the natural coordinates of p and $\bar{p^*}$, respectively. Similarly, denote by $\eta = \eta(p)$, $\bar{\eta} = \eta(\bar{p})$, and $\bar{\eta}^* = \eta(\bar{p}^*)$ the dual moment coordinates of p and $\bar{p^*}$, respectively.

[1] The product of exponential families is an exponential family.

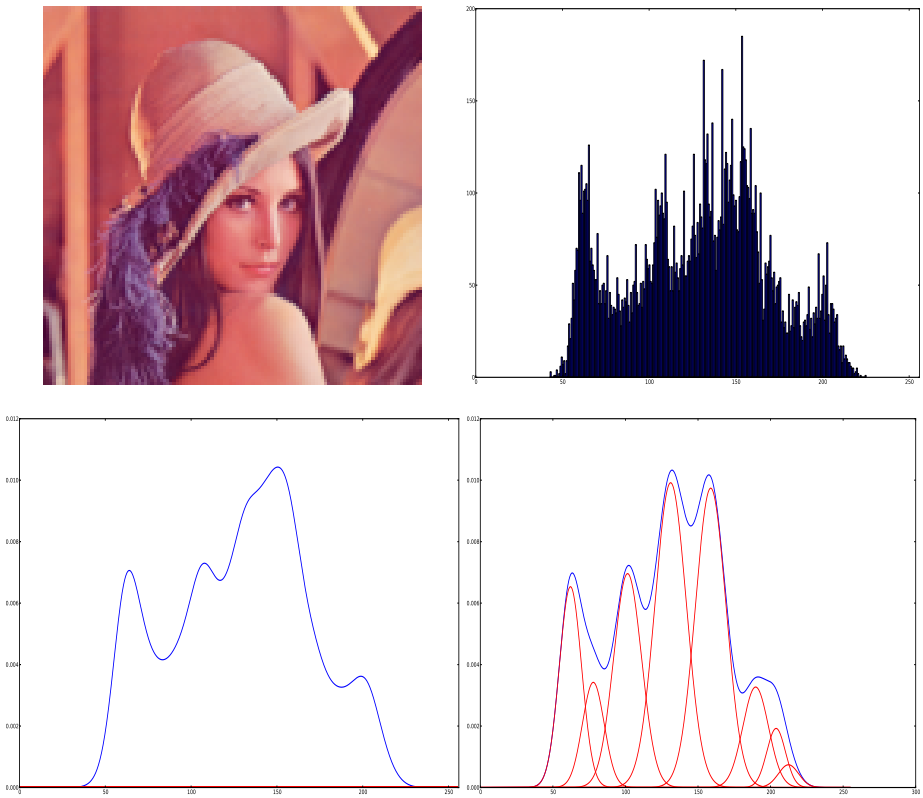


Figure 6.2: Top to bottom, left to right: original image, original intensity histogram, raw KDE (14400 components) and simplified mixture (8 components). Even with very few components compared to the mixture produced by the KDE, the simplified mixture still reproduces very well the shape of the histogram.

We have

$$\begin{aligned}
\mathrm{KL}\left(\tilde{p}\|p\right) &= \int \tilde{p}(x)\log\frac{\tilde{p}(x)}{p(x)}\mathrm{d}x && (6.7)\\
&= E_{\tilde{p}}[\log\tilde{p}] - E_{\tilde{p}}[\log p] && (6.8)\\
&= E_{\tilde{p}}[\log\tilde{p}] - E_{\tilde{p}}[\langle\theta, t(x)\rangle - F(\theta) + k(x)] && (6.9)\\
&= E_{\tilde{p}}[\log\tilde{p}] + F(\theta) - \langle\theta, E_{\tilde{p}}[t(x)]\rangle - E_{\tilde{p}}[k(x)] && (6.10)
\end{aligned}$$

since $E_{\tilde{p}}[F(\theta)] = F(\theta)\int\tilde{p}(x)\mathrm{d}x = F(\theta)$.

Using the fact that

$$\begin{aligned}
E_{\tilde{p}}[t(x)] &= E_{\sum_{i=1}^n w_i p_F(x;\theta_i)}[t(x)] && (6.11)\\
&= \sum_{i=1}^n w_i E_{p_F(x;\theta_i)}[t(x)] && (6.12)\\
&= \sum_{i=1}^n w_i\eta_i = \bar{\eta}^* && (6.13)
\end{aligned}$$

it follows that

$$\begin{aligned}
\mathrm{KL}\left(\tilde{p}\|p\right) &= E_{\tilde{p}}[\log\tilde{p}] + F(\theta) - E_{\tilde{p}}[k(x)] - \left\langle\theta, \sum_{i=1}^n w_i\eta_i\right\rangle && (6.14)\\
&= E_{\tilde{p}}[\log\tilde{p}] + F(\theta) - E_{\tilde{p}}[k(x)] - \langle\theta, \bar{\eta}^*\rangle. && (6.15)
\end{aligned}$$

Let us now add for mathematical convenience the neutralized sum $F(\bar{\theta}^*) + \langle\bar{\theta}^*, \bar{\eta}^*\rangle - F(\bar{\theta}^*) - \langle\bar{\theta}^*, \bar{\eta}^*\rangle = 0$ to the former equation.

Since

$$\mathrm{KL}\left(\bar{p}^*\|p\right) = B_F(\theta\|\bar{\theta}^*) = F(\theta) - F(\bar{\theta}^*) - \langle\theta - \bar{\theta}^*, \bar{\eta}^*\rangle, \quad (6.16)$$

and

$$\mathrm{KL}\left(\tilde{p}\|\bar{p}^*\right) = E_{\tilde{p}}[\log\tilde{p}] - E_{\tilde{p}}[k(x)] + F(\bar{\theta}^*) - \langle\bar{\theta}^*, \bar{\eta}^*\rangle, \quad (6.17)$$

We end up with the following Pythagorean sum:

$$\begin{aligned}
\mathrm{KL}\left(\tilde{p}\|p\right) &= E_{\tilde{p}}[\log\tilde{p}] + F(\theta) - E_{\tilde{p}}[k(x)] - \langle\bar{\eta}^*, \theta\rangle && (6.18)\\
&\quad + F(\bar{\theta}^*) + \langle\bar{\theta}^*, \bar{\eta}^*\rangle - F(\bar{\theta}^*) - \langle\bar{\theta}^*, \bar{\eta}^*\rangle && (6.19)\\
\mathrm{KL}\left(\tilde{p}\|p\right) &= \mathrm{KL}\left(\bar{p}^*\|p\right) + \mathrm{KL}\left(\tilde{p}\|\bar{p}^*\right) && (6.20)
\end{aligned}$$

This expression is therefore minimized for $\mathrm{KL}\left(\bar{p}^*\|p\right) = 0$ (since we have $\mathrm{KL}\left(\bar{p}^*\|p\right) \geq 0$), that is for $p = \bar{p}^*$. The closest distribution of $\mathcal{E}_F$ to $\tilde{p} \in$

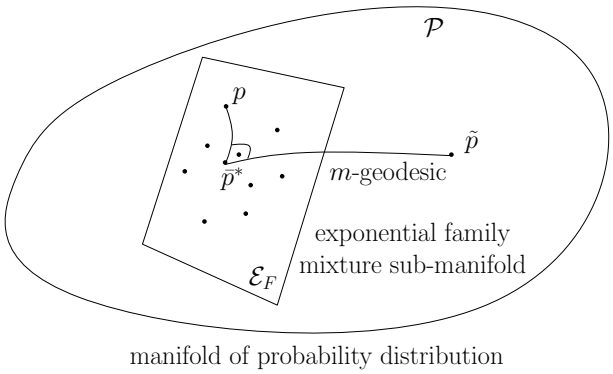


Figure 6.3: Projection operation from the mixture manifold to the model exponential family sub-manifold.

$\mathcal{P}$ is given by the dual centroid. In other words, distribution $\bar{p}^*$ is the right-sided KL projection of the mixture model onto the model sub-manifold. Geometrically speaking, it is the projection of $\tilde{p}$ via the mixture connection: the m-connection. Figure 6.3 illustrates the projection operation.

This theoretically explains why the right-sided KL centroid (ie., left-sided Bregman centroid) is preferred for simplifying mixtures [99] emanating from a kernel density estimator.

6.2.3 Model Hard Clustering

Hyperbolic distance

The statistical manifold of the parameters of exponential families can be studied through the framework of Riemaniann geometry. It has been proved by Chencov [29] and Lê [76] that the Fisher-Rao metric is the only Riemaniann metric on the statistical manifold which is invariant by reparameterization:

$$I(\theta) = \left[g_{ij}\right] = E\left[\frac{d\,\log p}{d\,\theta_i}\frac{d\,\log p}{d\,\theta_j}\right] \tag{6.21}$$

The Fisher-Rao distance (FR) between two distributions is computed using the length of the geodesic path between the two points on the statistical manifold:

$$\mathrm{FR}(p(x;\theta_1), p(x;\theta_2)) = \min_{\theta(t)} \int_0^1 \sqrt{\left(\frac{d\,\theta}{d\,t}\right)^T I\left(\theta(t)\right)\frac{d\,\theta}{d\,t}}\,dt \tag{6.22}$$

with θ such that $\theta(0) = \theta_1$ and $\theta(1) = \theta_2$.

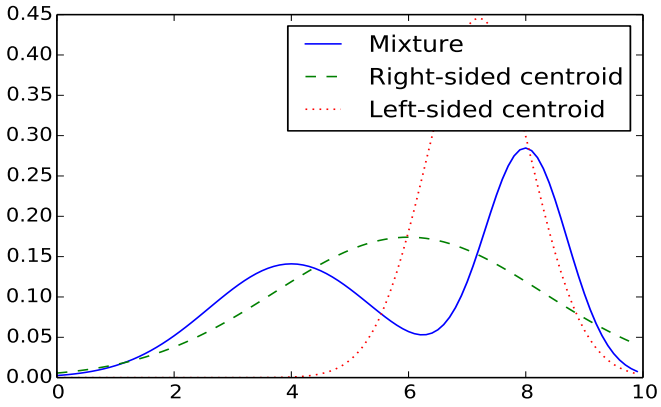


Figure 6.4: Right-sided (dashed line) and left-sided (dotted line) Kullback-Leibler centroids of a 2-components Gaussian mixture model. The left-sided centroid focuses on the highest mode of the mixture while the right-sided one tries to cover the supports of all the components. Pelletier's result says the right-sided centroid is the closest Gaussian to the mixture.

This integral is not known in the general case and is usually difficult to compute (see [106] for a numerical approximation in the case of the Gamma distribution).

However, it is known in the case of a normal distribution that the Fisher-Rao metric yields an hyperbolic geometry [65, 33].

For univariate Gaussian distributions, a closed-form formulation of the Fisher-Rao distance can be expressed, using the Poincaré hyperbolic distance in the Poincaré upper half-plane:

$$\begin{aligned}&\mathrm{FR}(f(x;\,\mu_p,\sigma_p^2),\,f(x;\,\mu_q,\sigma_q^2))\\ &\qquad=\sqrt{2}\ln\frac{\|(\frac{\mu_p}{\sqrt{2}},\sigma_p)-(\frac{\mu_q}{\sqrt{2}},\sigma_q)\|+\|(\frac{\mu_p}{\sqrt{2}},\sigma_p)-(\frac{\mu_q}{\sqrt{2}},\sigma_q)\|}{\|(\frac{\mu_p}{\sqrt{2}},\sigma_p)-(\frac{\mu_q}{\sqrt{2}},\sigma_q)\|-\|(\frac{\mu_p}{\sqrt{2}},\sigma_p)-(\frac{\mu_q}{\sqrt{2}},\sigma_q)\|}\end{aligned}\tag{6.23}$$

where $\|(u,v)\|=\sqrt{u^2+v^2}$ denotes the Euclidean norm.

Model centroid

In order to perform the k-means iterations using the Fisher-Rao distance, we need to define centroids on the hyperbolic space. Model centroids, introduced by Galperin [47] and successfully used in [108] for hyperbolic centroidal Voronoi tesselations, are a way to define centroids in the three

kinds of constant curvature spaces (namely, Euclidean, hyperbolic or spherical). For a d-dimensional curved space, it starts with finding a $(k+1)$-dimensional model in the Euclidean space. For a 2D hyperbolic space, it will be the Minkowski model , that is the upper sheet of the hyperboloid $-x^2 - y^2 + z^2 = 1$.

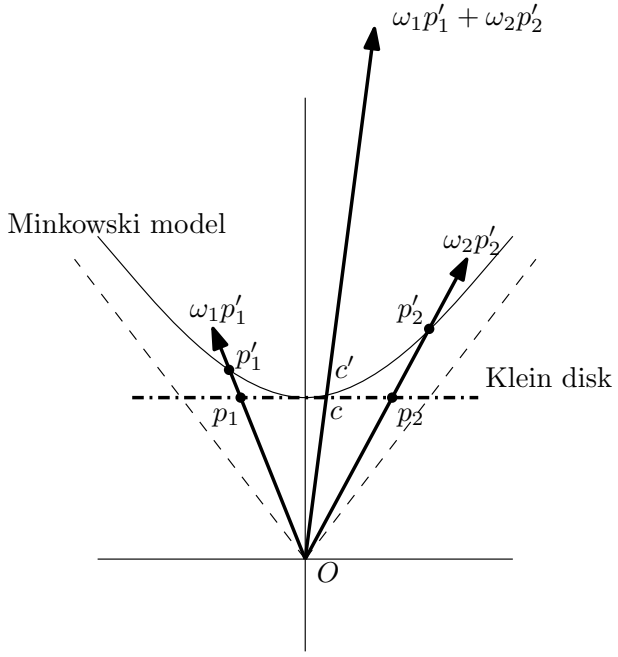


Figure 6.5: Computation of the centroid c given the system $(\omega_1, p_1), (\omega_2, p_2)$.

First, each point p (with coordinates (x_p, y_p)) lying on the Klein disk is embedded in the Minkowski model:

$$x_{p'} = \frac{x_p}{\sqrt{1-(x_p^2+y_p^2)}} \qquad y_{p'} = \frac{y_p}{\sqrt{1-(x_p^2+y_p^2)}} \qquad z_{p'} = \frac{1}{\sqrt{1-(x_p^2+y_p^2)}} \tag{6.24}$$

Next the center of mass of the points is computed

$$c'' = \sum \omega_i p'_i \tag{6.25}$$

This point needs to be normalized to lie on the Minkowski model, so we look for the intersection between the vector Oc'' and the hyperboloid:

$$c' = \frac{c''}{-x_{c''}^2 - y_{c''}^2 + z_{c''}^2} \tag{6.26}$$

From this point in the Minkowski model, we can use the reverse transform in order to get a point in the original Klein disk [93]:

$$x_c = \frac{x_{c'}}{z_{c'}} \qquad y_c = \frac{y_{c'}}{z_{c'}} \tag{6.27}$$

Although this scheme gives the centroid of points located on the Klein disk, it is not sufficient since parameters of the Gaussian distribution are in the Poincaré upper half-plane [33]. Thus we need to convert points from one model to another, using the Poincaré disk as an intermediate step. For a point (a, b) on the half-plane, let $z = a + ib$, the mapping with the Poincaré disk is:

$$z' = \frac{z - i}{z + i} \qquad z = \frac{i(z' + 1)}{1 - z'} \tag{6.28}$$

And for a point p on the Poincaré disk, the mapping with a point k on the Klein disk is:

$$k = \frac{2}{1 + \langle p|p\rangle} p \tag{6.29}$$

$$p = \frac{1 - \sqrt{1 - \langle k|k\rangle}}{\langle k|k\rangle} k \tag{6.30}$$

$$\tag{6.31}$$

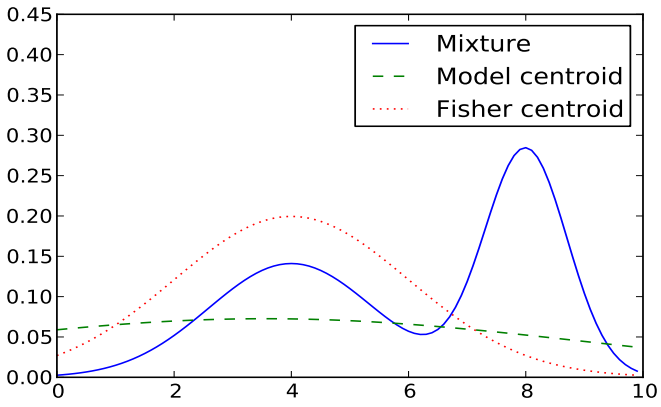


Figure 6.6: Model (dashed line) and Fisher (dotted line) Kullback-Leibler centroids of a 2-components Gaussian mixture model. The two notions of centroids give visually very different Gaussians: both centroids have roughly the same mean but the Model centroid has a large variance.

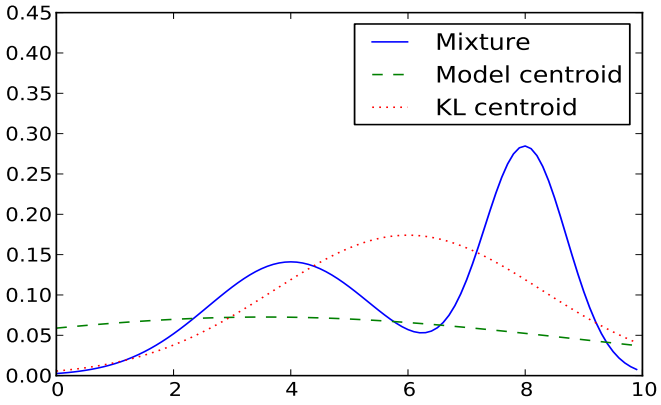


Figure 6.7: Model (dashed line) and left-sided Kullback-Leibler (dotted line) centroids of a 2-components Gaussian mixture model.

Contrary to the Bregman Hard Clustering algorithm, which use the Bregman divergence and the corresponding centroid, we do not use for the Model Hard Clustering the notion of centroid which is associated to the Fisher-Rao distance. Since the **Update** step of the k-means procedure does not minimize anymore the mean of the squared Fisher distances between points and clusters, we do not have the local convergence property. We see on Figure 6.6 that the two centroids are very different. Although there is no theoretical proof of the convergence, it appears that the algorithm works in practice and converges to a local minimum (see experiments in Section 8.1).

Quasi-arithmetic mean

The model centroid can be interpreted in a summarized version as a quasi-arithmetic mean:

$$c = q^{-1}\left(\frac{1}{\| \sum_i q\left(\mu_i, \sigma_i^2\right) \|_M} \sum_i k\left(\mu_i, \sigma_i^2\right)\right) \tag{6.32}$$

where $\|(x, y, z)\|_M = \sqrt{z^2 - (y^2 + z^2)}$ is the Minkowski norm .

The representation function q maps a couple of parameters $\left(\mu_i, \sigma_i^2\right)$ to the vector (x, y, z) according the following formula ($|u|$ denotes the modulus of a complex number):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \frac{\operatorname{Re} k}{\sqrt{1-|k|^2}} \\ \frac{\operatorname{Im} k}{\sqrt{1-|k|^2}} \\ \frac{1}{\sqrt{1-|k|^2}} \end{pmatrix} \tag{6.33}$$

where

$$\begin{cases} p = \dfrac{\mu + i\,(\sigma^2 - 1)}{\mu + i\,(\sigma^2 + 1)} \\ k = \dfrac{2}{1 + |p|^2} p \end{cases} \tag{6.34}$$

The inverse mapping q^{-1} which maps the vector (x, y, z) to the parameters (μ, σ^2), is given by the equations:

$$\begin{pmatrix} \mu \\ \sigma^2 \end{pmatrix} = \begin{pmatrix} \operatorname{Re} \frac{i\,(p+1)}{1-p} \\ \operatorname{Im} \frac{i\,(p+1)}{1-p} \end{pmatrix} \tag{6.35}$$

where

$$\begin{cases} k = \dfrac{x}{z} + i\,\dfrac{y}{z} \\ p = \dfrac{1 - \sqrt{1 - |k|^2}}{|k|^2} k \end{cases} \tag{6.36}$$

6.3 Conclusion

After an introduction about kernel density estimator, this chapter presented two variants of the simplification algorithm: the first one is based on the classical Bregman Hard Clustering whereas the second one uses the Fisher distance and the model centroids. Along with these two methods, a proof of the equality between the right-sided Kullback-Leibler centroids and geometric projection of a mixture model onto the exponential family manifold is given. The algorithms presented here are implemented in a `Python` library which is described in the next chapter.

Chapter 7

Software library

7.1 Presentation

Several tools are already available to build mixture models, either for mixtures of Gaussian distributions or for mixtures of other distributions. But these tools are usually dedicated to a particular family of distributions.

In order to provide a unified and powerful framework for the manipulation of arbitrary mixture models, we develop `pyMEF`, a `Python` library dedicated to the mixtures of exponential families.

Given the success of the Gaussian mixture models, there are already numerous other software available to deal with it:

- some `R` packages: `MCLUST` (`http://www.stat.washington.edu/mclust/`) and `MIX` (`http://icarus.math.mcmaster.ca/peter/mix/`),
- `MIXMOD` [20] which also works on multinomial and provides bindings for `Matlab` and `Scilab`,
- `PyMIX` [49], another `Python` library which goes beyond simple mixture with Context-specific independence mixtures and dependence trees,
- `scikits.learn`, a `Python` module for machine learning (`http://scikit-learn.sf.net`),
- `jMEF` [99, 48] which is the only other library dealing with mixtures of exponential families, written in `Java`.

Although exponential families other than normal distributions have been successfully used in the literature, it was made using an implementation specific to the underlying distribution per se. The improvement of libraries such as `jMEF` and `pyMEF` is to introduce genericity: changing the exponential family means simply changing a parameter of the Bregman

Soft Clustering (equivalent to performing an EM task), and not completely rewriting the algorithm.

Moreover, the choice of the best distribution is a difficult problem in itself, and is often inspected experimentally, by looking at the shape of the histogram or by comparing a performance score (the log-likelihood or any meaningful score in the considered application) computed with mixtures of various distributions. It is worth here to use a unified framework instead of using different libraries from various sources with various interfaces.

The goal of the `pyMEF` library is to provide a consistent framework with various algorithms to build mixtures (Bregman Soft Clustering) and various Information-theoretic simplification methods (Bregman Hard Clustering, Fisher Hard Clustering) along with some widespread exponential families:

- univariate Gaussian,
- multivariate Gaussian,
- Generalized Gaussian,
- multinomial,
- Rayleigh,
- Laplace.

7.2 Extending pyMEF

The set of available exponential families can be easily extended by users. Following the principles of *Flash Cards* introduced in [99] for `jMEF` it is sufficient to implement in a `Python` class the function describing the distribution:

- the core of the family (the log-normalizer F and its gradient ∇F, the carrier measure k and the sufficient statistic t),
- the dual characterization with the Legendre dual of F ($F^\star$ and $\nabla F^\star$)
- the conversion between three parameters space (source to natural, natural to expectation, expectation to source and their reciprocal).

7.3 An example with a Gaussian Mixture Model

We present here a basic example of a `pyMEF` session. The following can be used interactively in the `Python` toplevel or be part of a larger software.

This allows both a rapid exploration of a dataset and the development of a real application with the same tools.

We begin with loading the required modules:

```
import numpy
from matplotlib import pyplot

from pyMEF.Build import BregmanSoftClustering, KDE
from pyMEF.Simplify import BregmanHardClustering
from pyMEF.Families import UnivariateGaussian
```

An example dataset (6550 samples) is loaded using standard `numpy` functions:

```
data = numpy.loadtxt("data.txt")
data = data.reshape(data.shape[0], 1)
```

An 8-component mixture model is built on this dataset using the Bregman Soft Clustering algorithm (also known as EM in the Gaussian case):

```
em = BregmanSoftClustering(data, 8, UnivariateGaussian, ())
mm_em = em.run()
```

Another mixture is built using Kernel Density Estimation (leading to a 6550-component mixture).

```
mm_kde = KDE(data, UnivariateGaussian, ())
```

This very large model is then simplified into an 8-component mixture with the Bregman Hard Clustering algorithm:

```
kmeans = BregmanHardClustering(mm_kde, 8)
mm_s = kmeans.run()
```

We finally compute the log-likelihood of the models (original and simplified).

```
print "EM:", mm_em.logLikelihood(data)
print "KDE:", mm_kde.logLikelihood(data)
print "Simplified KDE:", mm_s.logLikelihood(data)
```

For illustration purposes (see Figure 7.1), we plot the histogram of the original data and the three computed models (`pyMEF` does not provide any display functions, we rely instead on the powerful `matplotlib` [1] library).

[1] The `matplotlib` library can be downloaded on `http://matplotlib.sourceforge.net/`

Model	Log-likelihood
EM	-18486.7957123
KDE	-18985.4483699
Simplified KDE	-19015.0604457

Table 7.1: Log-likelihood of the three computed models. EM still gives the best value and the simplified KDE has nearly the same log-likelihood than the original KDE.

```
pyplot.subplot(2, 2, 1)
pyplot.hist(data, 1000)

pyplot.xlim(0, 20)
x = numpy.arange(0,20,0.1)

pyplot.subplot(2, 2, 2)
pyplot.plot(x, mm_em(x))

pyplot.subplot(2, 2, 3)
pyplot.plot(x, mm_kde(x))

pyplot.subplot(2, 2, 4)
pyplot.plot(x, mm_s(x))

pyplot.show()
```

A real application would obviously use multiple runs of the soft and hard clustering algorithms to avoid being trapped in a bad local optimum that can be reached by the two local optimization methods.

In this example, the Bregman Soft clustering gives the best result in terms of log-likelihood (Table 7.1) but the model is visually not really satisfying (there is a lot of local maxima near the first mode of the histogram, instead of just one mode). The models relying on Kernel Density Estimation give a bit worse log-likelihood but are visually more convincing. The important point is the quality of the simplified model: while having a lot less components (8 instead of 6550) the simplified model is nearly identical to the original KDE (both visually and in terms of log-likelihood).

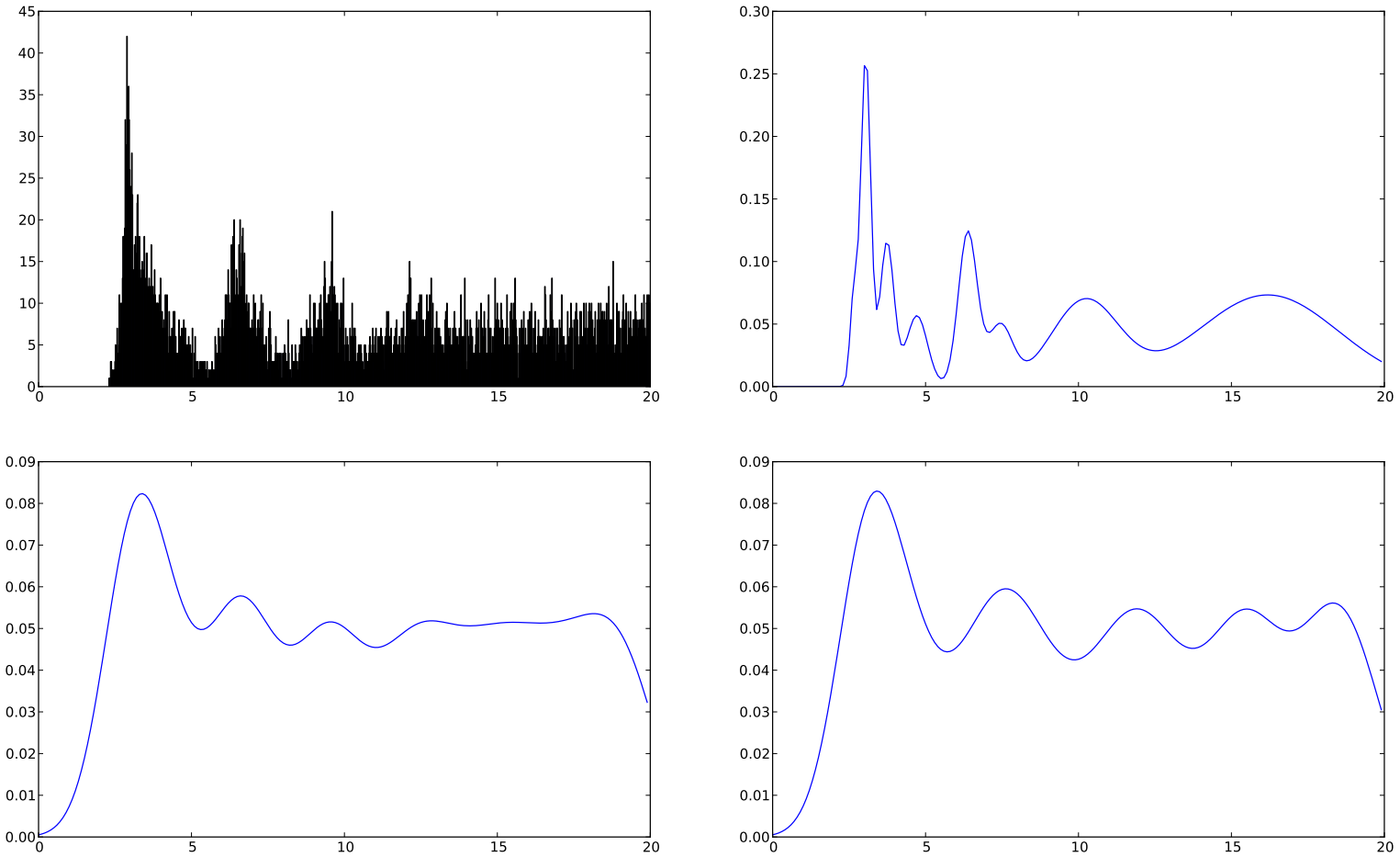


Figure 7.1: Output from the `pyMEF` demo. Top-left the histogram from the data; top-right, the model computed by EM; bottom-left the one from KDE; bottom-right the simplified KDE. Visual appearance is quite bad for EM while it is very good for both KDE and simplified KDE, even with a lot less components in the simplified version.

7.4 Examples with other exponential families

Although the Gaussian case is the more widespread and the more universal case, lots of other exponential families are useful in particular applications. We present here two examples implemented in `pyMEF` using the formula detailed in [99].

Rayleigh distribution The Rayleigh mixture models are used in the field of Intravascular UltraSound Imaging [122] for segmentation and classification tasks. We present in Figure 7.2 an example of the learning of a Rayleigh mixture model on a synthetic dataset built from a 5-components mixture of Rayleigh distributions. The graphics shown in this figure have been generated with the following script (for the sake of brevity, we omit here the loops used to select the best model among some tries). Notice how similar this code is to the previous example, showing the genericity of our library: using different exponential families for the mixtures is just a matter of changing one parameter in the program.

```
import sys, numpy

from pyMEF import MixtureModel
from pyMEF.Build import BregmanSoftClustering
from pyMEF.Simplify import BregmanHardClustering
from pyMEF.Families import Rayleigh

# Original mixture
k = 5
mm = MixtureModel(5, Rayleigh, ())
mm[0].source((1.,))
mm[1].source((10.,))
mm[2].source((3.,))
mm[3].source((5.,))
mm[4].source((7.,))

# Data sample
data = mm.rand(10000)

# Bregman Soft Clustering k=5
em5 = BregmanSoftClustering(data, 5, Rayleigh, ())
em5.run()
mm_em5 = em5.mixture()

# Bregman Soft Clustering k=32 + Simplification
em32 = BregmanSoftClustering(data, 32, Rayleigh, ())
```

```
em32.run()
mm_em32 = em.mixture()

kmeans5 = BregmanHardClustering(mm_em32, 5)
kmeans.run()
mm_simplified = kmeans.mixture()
```

Laplace distribution Although Laplace distributions are only exponential families when their means is fixed, zero-mean Laplacian mixture models are used in various applications. Figure 7.3 presents the same experiments as in Figure 7.2 and has been generated with the same script, by replacing all occurrences of the word `Rayleigh` by the word `CenteredLaplace`.

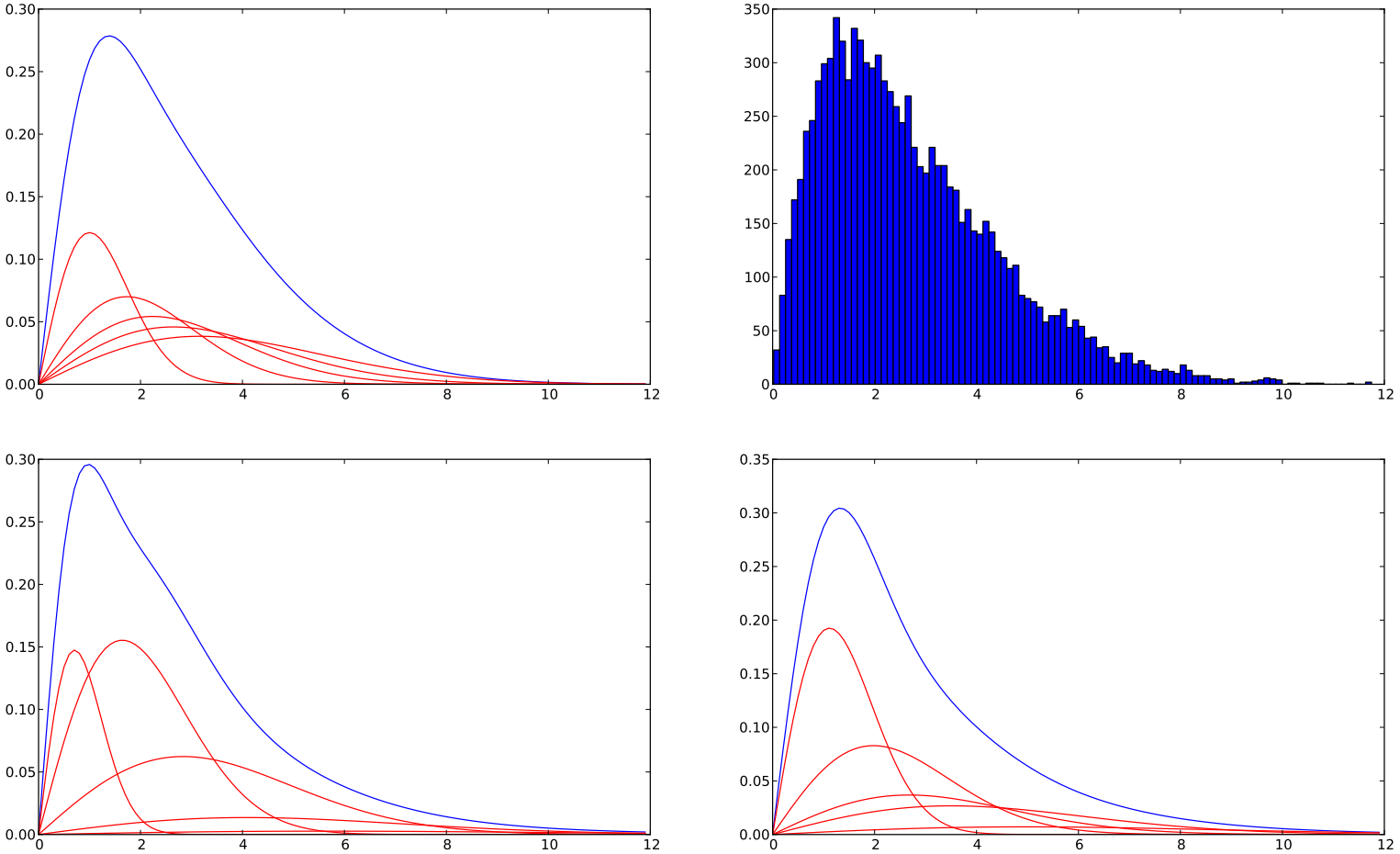


Figure 7.2: Rayleigh mixture models. The top left figure is the true mixture (synthetic data) and the top right one is the histogram of 10000 sample drawn from the true mixture. The bottom left figure is a mixture build with the Bregman Soft Clustering algorithm (with 5 components) and the bottom right one is a mixture built by first getting a 32 components mixture with Bregman Soft Clustering and then simplifying it to a 5 components mixtures with the Bregman Hard Clustering algorithm.

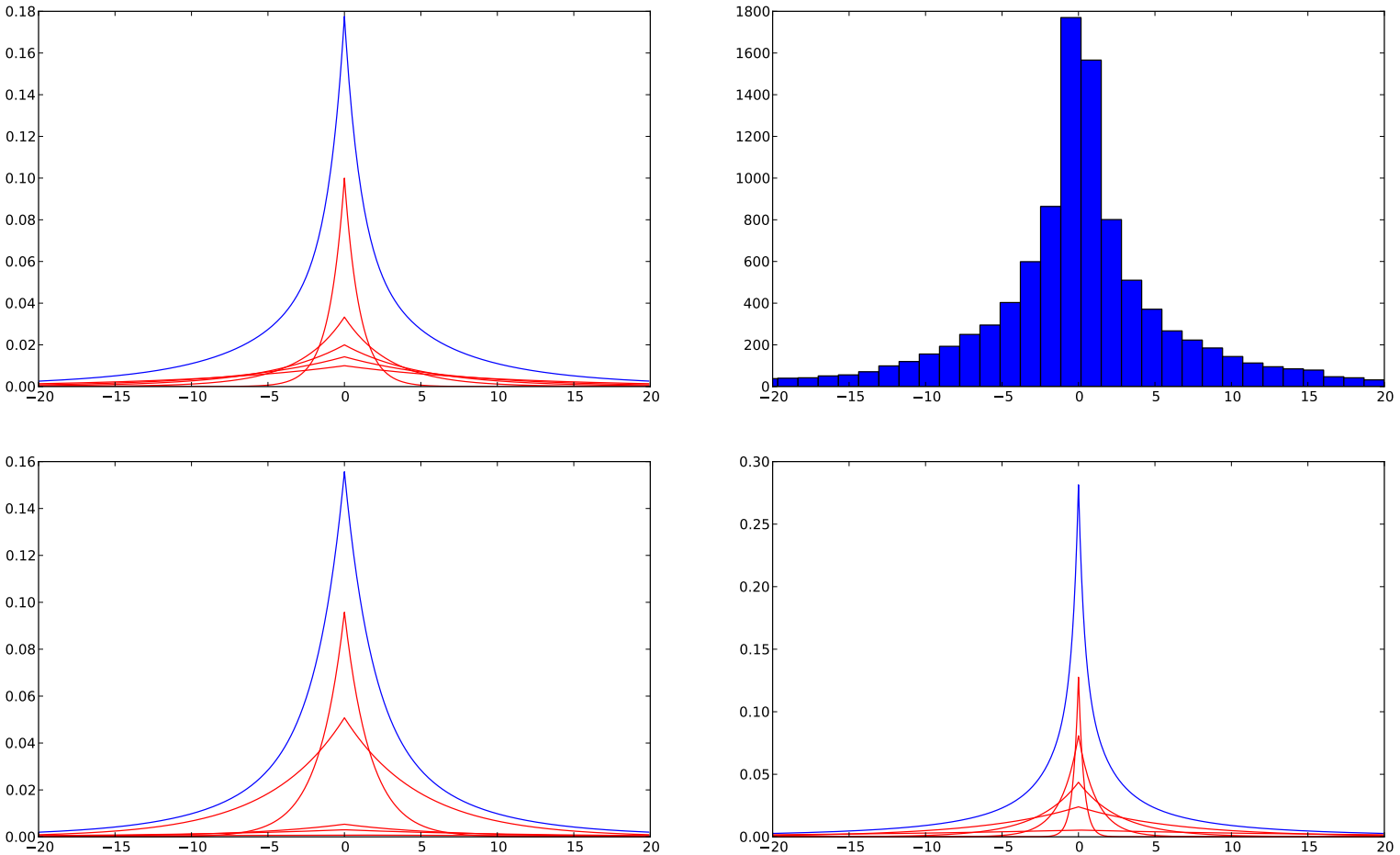


Figure 7.3: Laplace mixture models. The top left figure is the true mixture (synthetic data) and the top right one is the histogram of 10000 sample drawn from the true mixture. The bottom left figure is a mixture build with the Bregman Soft Clustering algorithm (with 5 components) and the bottom right one is a mixture built by first getting a 32 components mixture with Bregman Soft Clustering and then simplifying it to a 5 components mixtures with the Bregman Hard Clustering algorithm.

7.5 Conclusion

The `pyMEF` library presented in this chapter implements the two simplification algorithms alongside with other mixture model learning methods. This is both an easy to use and an easy to extend library which allows practical use of information geometric algorithms for mixture models which are evaluated in the next chapter.

Chapter 8

Applications and experiments

8.1 Local convergence of Model Hard Clustering

Since we do not have a theoretical proof of the convergence of Model Hard Clustering, it is import to experimentally check that the convergence happens in practice. In all the experiments presented in the following two sections, the cost function decreases and we get a local minimum. Figure 8.1 presents the decrease of the k-means cost function on one of the observations set from Section 8.3.

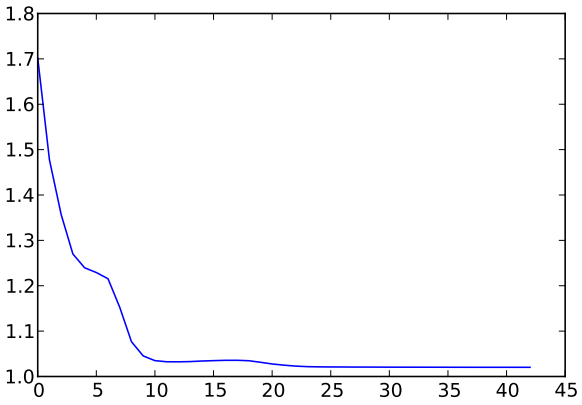


Figure 8.1: Evolution of the cost function of Model Hard Clustering wrt the number of iterations. We reach a local minimum of the cost.

8.2 Experiments on images

We study here the quality, in terms of log-likelihood, and the computation time of the proposed methods compared to a baseline Expectation-

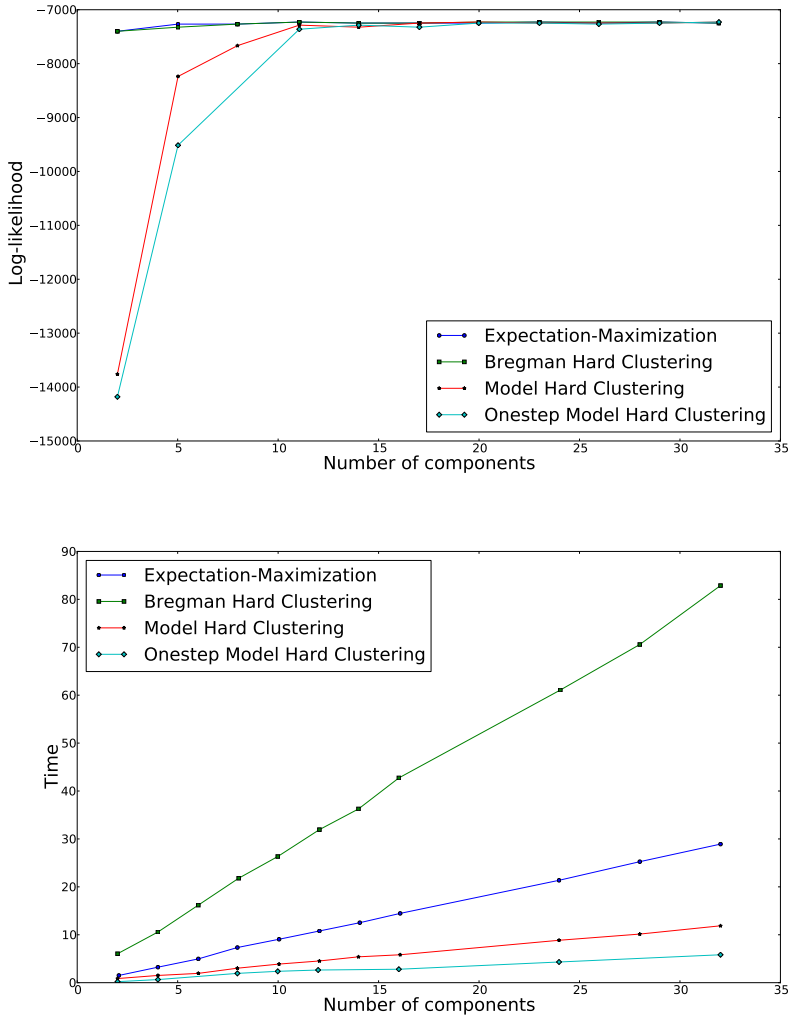


Figure 8.2: Log-likelihood of the simplified models and computation time. All the algorithms reach the same log-likelihood maximum with quite few components (but the one-step model centroid needs a few more components than all the others). Model centroid based clusterings are the fastest methods, Kullback-Leibler clustering is even slower than EM due to the computational cost of the KL distance and centroids.

Maximization algorithm. The source distribution is the intensity histogram of the famous `Lena` image (see Figure 6.2). As explained in Section 6.2.1, for the Kullback-Leibler divergence, we report only results for right-sided centroids since it performs better (as indicated by the theory) than the two other flavors and has the same computation cost. The third and fourth methods are the Model centroid, both with a full k-means and with only one iteration.

The top part of Figure 8.2 shows the evolution of the log-likelihood as a function of the number of components k. First, we see that all the algorithms perform nearly the same and converge very quickly to a maximum value (the KL curve is merged with the EM one).

Kullback-Leibler divergence and Fisher-Rao metric perform similarly but they are rather different from a theoretical standpoint: KL assumes an underlying flat geometry while Fisher-Rao is related to the curved hyperbolic geometry of Gaussian distributions. However at infinitesimal scale (or on dense compact clusters) they behave the same.

The bottom part of Figure 8.2 describes the running time (in seconds) as a function of k. Despite the fact that the quality of mixtures is nearly identical, the costs are very different. Kullback-Leibler divergence is very slow (even in closed-form, the formulas are quite complex to calculate). While achieving the same log-likelihood, model centroid is the fastest method, significantly faster than EM.

While being slower to converge when k increases, the one step model clustering performs still well and is roughly two times faster than a complete k-means.

8.3 Prediction of 3D structures of RNA molecules

RNA molecules play an important role in many biological processes. The understanding of the functions of these molecules depends on the study of their 3D structure. A common approach is to use knowledge-based potential built from inter-atomic distance coming from experimentally determining structures. Recent work use mixture models [17] to model the distribution of the inter-atomic distances.

In the original work by Bernauer, Huang, Sim, and Levitt [17] the authors use Dirichlet Process Mixtures [105] to build the mixture models. This gives high quality mixtures, both in terms of log-likelihood and in the context of the application, but with a high computational cost which is not affordable for building thousands of mixtures.

We study here the effectiveness of our proposed simplification mixtures compared to reference high quality mixtures built with Dirichlet Process Mixtures. We evaluate the quality of our simplified models by computing mixture in an absolute way, with the log-likelihood, and in a relative way,

Method	Log-likelihood
DPM	-18420.6999452
KDE	-18985.4483699
KDE + Bregman Hard Clustering	-18998.3203038
KDE + Model Hard Clustering	-18974.0717664
KDE + One step Model Hard Clustering	-19322.2443988

Table 8.1: Log-likelihood of the model built by the state-of-the-art Dirichlet Process Mixture, by Kernel Density Estimation, and by our new simplified models. DPM is better but the proposed simplification methods perform as well as the KDE.

KL	DPM	KDE	BHC	MHC	One step MHC
DPM	0.0	0.051	0.060	0.043	0.066
KDE	0.090	0.0	0.018	0.002	0.016

Table 8.2: Kullback-Leibler divergence matrix for models built by Dirichlet Process Mixture (DPM), by Kernel Density Estimation (KDE), by the Bregman Hard Clustering (BHC), by the Model Hard Clustering (MHC) and by the one-step Model Hard Clustering. We limit the lines of the table to only DPM and KDE since by the nature of Kullback-Leibler, the left term of the divergence is supposed to be the "true" distribution and the right term the estimated distribution (left term comes from the lines and right term from the columns).

with the Kullback-Leibler divergence between a mixture built with Dirichlet and a simplified mixture.

Only the relevant statistical part of this application is described: the computational chemistry process of constructing potential from the mixture models and the bio-informatic analysis which follows is out of scope here and the readers will find all the details in [124].

Both DPM and KDE produce high quality models (see Table 8.1): for the first with high computational cost, for the second with a high number of components. Moreover, these two models are very close for the Kullback-Leibler divergence: this means that one may choose between the two algorithms depending on the most critical point, time or size, in their application.

Simplified models get nearly identical log-likelihood values. Only the one-step Model Hard Clustering leads to a significant loss in likelihood.

Simplified models using Bregman and Model Hard Clustering are both close to the reference DPM model and to the original KDE (Table 8.2). Moreover, the Model Hard Clustering outperforms the Bregman Hard Clus-

tering in the two cases. As expected, the one-step Model Hard Clustering is the furthest: it will depend on the application to know if the decrease in computation time is worth the loss in quality.

Chapter 9

Conclusion NONUMBER

We presented a novel modeling paradigm which is both *fast* and *accurate*. From the Kernel Density Estimates which are precise but difficult to use due to their size, we are able to build new models which achieve the same approximation quality while being faster to compute and compact. We introduce a new mixture simplification method, the Model Hard Clustering, which relies on the Fisher-Rao metric to perform the simplification. Since closed-form formulas are not known in the general case we exploit the underlying hyperbolic geometry, allowing to use the Poincaré hyperbolic distance and the Model centroids, which are a notion of centroids in constant curvature spaces.

Models simplified by the Bregman Hard Clustering and by Model Hard Clustering have both a quality comparable to models built by Expectation-Maximization or by Kernel Density Estimation. But the Model Hard Clustering does not only give very high quality models, it is also faster than the usual Expectation-Maximization. The quality of the models simplified by the Model Hard Clustering justifies the use of the Model centroids as a substitute for the Fisher-Rao centroids.

Both Model and Bregman Hard Clustering are also competitive with state-of-the-art approaches in a bio-informatics application for the modeling of the 3D structure of a RNA molecule, giving models which are very close, in terms of Kullback-Leibler divergence, to reference models built with Dirichlet Process Mixtures.

Part III

Extended k-MLE: mixtures of Gamma and Generalized Gaussian distributions

This part presents results already published in the ICPR 2012 article "k-MLE for mixtures of generalized Gaussians" [121] and in the SIMBAD 2013 article "Fast Learning of Gamma Mixture Models with k-MLE" [114].

The work about generalized Gaussian distributions in [121] has been carried out with Aurélien Schutz and Yannick Berthoumieu (Laboratoire IMS, Université de Bordeaux).

Chapter 10

Introduction NONUMBER

After decades of successful use, statistical mixture models are still an active research field: while the Expectation-Maximization method can be used for any kind of mixture (of Gaussians, of Laplace, of Gamma, etc), other works want to improve some particular points. The k-Maximum Likelihood Estimator algorithm (k-MLE) recently introduced by Nielsen [96, 90] aims at improving the speed of learning a mixture of exponential families. k-MLE exploits a different paradigm than the classical EM: instead of working as a soft clustering algorithm, it performs a hard clustering of the observations. The quantity which is optimized is not anymore the expected log-likelihood but the complete log-likelihood and this optimization problem can be solved with the help of the Bregman Hard Clustering algorithm [12]. Although it maximizes a slightly different quantity, it appears that k-MLE also produces mixtures with a high log-likelihood while being significantly faster. k-MLE works in the framework of exponential families which may be limitating in some conditions: this is the case when we do not want the same exponential family for each component (as for the generalized Gaussian mixtures) or when the necessary closed-form formulas (for efficient computation) are not known (as for the Gamma mixtures).

The first contribution introduced here is the following: generalized Gaussian distributions (GG) are a powerful generalization of the Gaussian distribution which contain many distributions from heavy-tailed laws to the limit case of the uniform distribution. Although a generalized Gaussian is an exponential family when the parameter which controls the shape of density is fixed, two generalized Gaussian with two different shape parameters are not in the same family which renders the k-MLE algorithm unsuitable. To unleash the full power of GG, we introduce here an extension of k-MLE which allows to learn mixtures of generalized Gaussian with various shape parameters.

The second contribution is to provide an efficient way to build Gamma mixture models. In fact, a Gamma law is an exponential family (for all the

parameters, contrary to the generalized Gaussian) so the exponential families tools like k-MLE are theoretically usable but this is not true in practice. Some necessary functions are not known in closed-form, rendering the original k-MLE too slow (with numerical approximation to replace the closed-form formula) to be useful. We rely here on a computational trick: a Gamma law with a fixed rate parameter is still an exponential family (of order 1) for which all the formulas are known in closed-form. Using this remark, we can now apply the same extended k-MLE as the one used for generalized Gaussian: the goal is to learn a mixture of fixed-rate Gamma laws where the rates are different among the components.

In Chapter 11, we first present the necessary background on exponential families and Bregman divergences and introduce the original k-MLE algorithm. The extension allowing to learn mixtures where all the components are not part of the same exponential family is then described.

In Chapter 12, we describe `libmef`, the software library used in the experiments. This library is written in the `C` language and is complementary to the `pyMEF` Python library described in the previous part since it focuses on speed rather than on the ease of use in a dynamic programming language. In the long term, the two libraries should merge in order to provide a fast `C` backend with a neat `Python` API.

In Chapter 13, we present some experimental results: k-MLE for generalized Gaussian is compared to an EM for generalized Gaussian; k-MLE for Gamma is a compared to an EM for Gamma laws. In both cases, the comparison is made in terms of log-likelihood and computation time.

Chapter 11

Extended k-MLE

11.1 Motivation and prior work

k-MLE is a rather new algorithm for mixtures of exponential families but it finds its roots on previous work which studied the meaning of the soft clustering procedure. It is very close to the Hard EM algorithm introduced by Banerjee, Dhillon, Ghosh, and Sra [11] for Von Mises-Fisher distributions which suggests to replace the soft assignment step of EM with a simpler hard assignment; it is also similar to Classification EM [27]. As a compromise between hard and soft assignment, it has also been proposed in the literature [88] to sparsify the posterior probability matrix by filtering out some values. Transforming the soft assignment into a hard one in EM has also been suggested for Hidden Markov Model with an algorithm called *Viterbi Training* [61, 69].

The main contribution of k-MLE, providing a generic algorithm for exponential families, is not enough for the two distributions we consider here: although theoretically possible, the method is computationally unsuitable for Gaussian mixtures and two generalized Gaussian with different shape parameters are simply not in the same family, which imposes to use the same value for all the components with the original k-MLE.

The generalized Gaussian distribution is historically strongly linked with applications related to texture classification or retrieval: from seminal work by Mallat [78] on wavelets, this family of distributions has been used for modeling wavelet decomposition of textures [41] and is still actively used [112]. An Expectation-Maximization algorithm for generalized Gaussian has been introduced by Allili [3, 4]: we will use this method as a reference in the chapter about experimental evaluation.

Gamma mixture models are a very interesting kind of mixture models since like the Gaussian mixtures, they have the property of universality: contrary to the Gaussian mixtures which are universal on full $\mathbb{R}$, Gamma mixtures are universal for densities with a positive support. This makes

mixtures of Gamma laws a natural and more statistically meaningful kind of mixtures to model densities from positive observations such as the distances used in Part II. Some algorithms are available in the literature for these mixtures, with various applications in mind: telecommunication network modeling [5], medical service analysis [125] or in bio-informatics to model molecular sequence evolution [80]. We will use here in experiments the EM variant for Gamma laws by Almhana, Liu, Choulakian, and McGorman [5] as a reference.

11.2 Learning mixtures of exponential families with k-MLE

Assume we have a set $\mathcal{X} = \{x_1, \ldots, x_n\}$ of n observations which have been sampled from a finite mixture model with k components. The joint probability distribution of these samples with the missing components z_i (indicating from which component each observation x_i comes from) is:

$$p(x_1, z_1, \ldots, x_n, z_n) = \prod_i p(z_i|\omega) p(x_i|z_i, \theta) \tag{11.1}$$

Since the variables z_i are not observed in practice, we marginalize them and we get:

$$p(x_1, \ldots, x_n|\omega, \theta) = \prod_i \sum_j p(z_i = j|\omega) p(x_i|z_i = j, \theta) \tag{11.2}$$

The straightforward way to optimize this distribution would be to test the k^n labels but this is not tractable in practice. Instead, Expectation-Maximization optimizes the following quantity, the average log-likelihood:

$$ll(x_1, \ldots, x_n) = \frac{1}{n} \log p(x_1, \ldots, x_n) \tag{11.3}$$

$$= \frac{1}{n} \sum_i \log \sum_j p(z_i = j|\omega) p(x_i|z_i = j, \theta) \tag{11.4}$$

Contrary to this approach, the k-Maximum Likelihood Estimator maximizes the average complete log-likelihood:

$$ll'(x_1, z_1, \ldots, x_n, z_n) = \frac{1}{n} \log p(x_1, z_1, \ldots, x_n, z_n) \tag{11.5}$$

$$= \frac{1}{n} \sum_i \log \prod_j \left(\left(\omega_j p_F(x_i, \theta_j) \right)^{\delta(z_i)} \right) \tag{11.6}$$

$$= \frac{1}{n} \sum_i \sum_j \delta_j(z_i) \left(\log p_F(x_i, \theta_j) + \log \omega_j \right), \tag{11.7}$$

where $\delta_j(z_i) = 1$ if and only if z_i emanates from the j-th component.

Since p_F is an exponential family, we have:

$$\log p_F(x_i, \theta_j) = -B_{F^*} \left(t(x) \| \eta_j \right) + \underbrace{F^\star(t(x)) + k(x)}_{\text{does not depend on } \theta} \tag{11.8}$$

The terms which do not depend on θ are of no interest for the maximization problem and can be removed: we can then rewrite Eq. (11.7) to get the equivalent problem:

$$\arg\min \sum_i \sum_j \delta(z_i) \left(B_{F^*} \left(t(x) \| \eta_j \right) - \log \omega_j \right) \tag{11.9}$$

As stated in [96] this problem can be solved for a fixed set of weights ω_i using the Bregman k-means algorithm with the Bregman divergence B_{F^*} (actually, any heuristic for k-means is convenient such as the Hartigan procedure [54], used for Wishart mixtures with k-MLE by Saint-Jean and Nielsen [111]).

The weights can now be optimized by taking $\omega_i = \frac{|\mathcal{C}_i|}{n}$ (where $|C_i|$ is the number of observations put in the cluster $\mathcal{C}_i$ by the solution of the previous clustering problem). This step amounts to maximizing the cross-entropy of the mixture [96].

The full algorithm can be summarized as follows (see Fig. 11.2 for a block diagram):

1. **Initialization** (choose seeds θ_i randomly or by using k-MLE ++[96]);
2. **Assignment** $z_i = \arg\max_j \log(\omega_j p_F(x_i|\theta_j))$;
3. **Update** of the η parameters $\eta_i = \frac{1}{n_j} \sum_{x \in \mathcal{C}_j} t(x)$;
   Goto step 2 until local convergence;
4. **Update** of the parameters ω_j;
   Goto step 2 until local convergence of the complete likelihood.

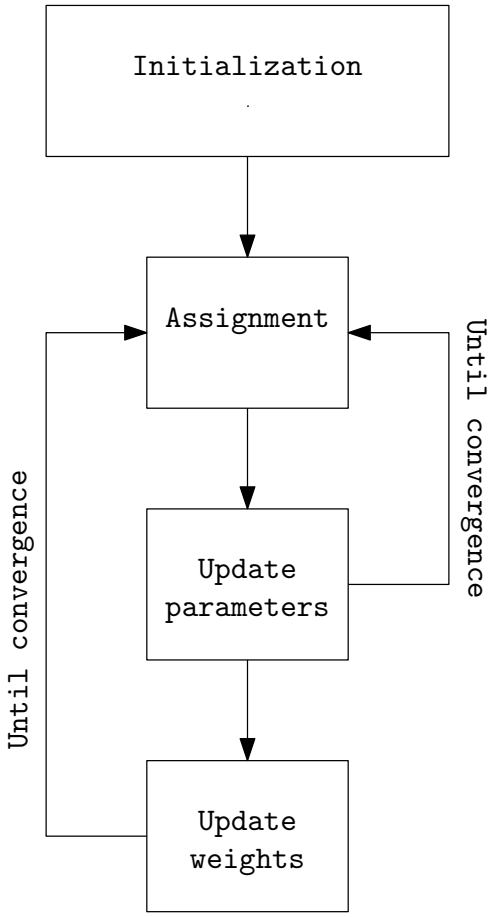


Figure 11.1: Block diagram for the *k*-MLE algorithm.

Using the Lloyd k-means heuristic may lead to empty clusters, especially with a large number of clusters and high-dimensional data [111]. Another problem is degenerate clusters: when there are not enough observations (less than the order of the family) in the cluster, the MLE does not exist (see Section 3.1.5 in Chapter 3 more details).

The strategy used here is to remove empty or to small clusters when they appear: this allows to select a number of components larger than necessary since the number will decrease during the iterations. Another strategy is to use the Hartigan clustering heuristic [54] which can be tuned to avoid empty clusters [111]. In the Agglomerative Bregman Clustering algorihm [37], a smoothing procedure to deal with degenerate clusters has also been proposed.

11.3 Extension to the non-fixed family case

11.3.1 Choosing the exponential family

The original k-MLE algorithm builds mixture models where all the components belong to the same exponential family. The extended k-MLE aims at building mixtures where the family is not fixed *a priori* but chosen at each step (see Figure 11.2): the convergence to a local maximum of the complete log-likelihood is preserved when each component is part of a different family and this maximization can still be done with a Bregman Hard Clustering-like algorithm. It is not anymore the original Bregman Hard Clustering but a generalized version in which the cost function is now:

$$\sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j} \left(B_{F_j^*} \left(t(x_i) \| \eta_j \right) - {F_j}^*(t(x_i)) - k_j(x_i) - \log \omega_j \right) \tag{11.10}$$

where $\mathcal{C}_j$ is the set of the indices of the observations sampled from the j-th component.

This new extended k-MLE algorithm can be summarized as follows (see Figure 11.2):

1. **Initialization** (random or using k-MLE ++[96]);
2. **Assignment** $z_i = \arg\max_j \log(\omega_j p_{F_j}(x_i|\theta_j))$;
3. **Update** of the η parameters $\eta_i = \frac{1}{n_j} \sum_{x \in \mathcal{C}_j} \log(x)$;
   Goto step 2 until stability (local convergence of the k-means);
4. **Update** of the weights ω;
   Choice of the exponential family of each components;
   Goto step 2 until local convergence of the complete likelihood.

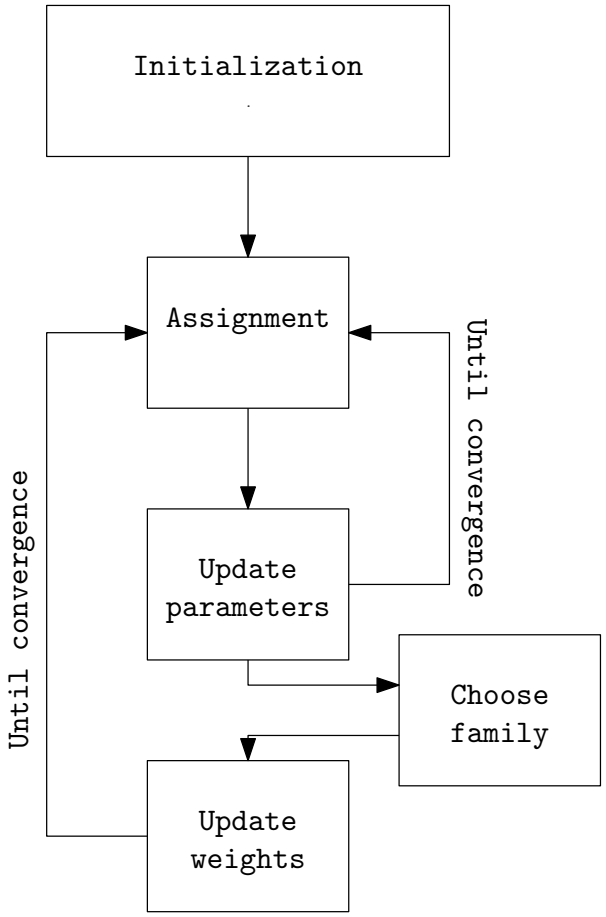


Figure 11.2: Block diagram for Extended *k*-MLE.

11.3.2 New cost function

As the one proposed for generalized Gaussian, this algorithm converges to a local maximum of the complete log-likelihood. We want to minimize the same cost function as the original k-MLE algorithm, the complete average log-likelihood of the mixture, with the slight difference that the log-normalizer is not shared among components but now depends on the component j; and is thus now written F_j instead of F:

$$ll'(x_1, z_1, ..., x_n, z_n|w, \theta) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\delta_j(z_i)(\log p_{F_j}(x_i|\theta_j) + \log\omega_j) \tag{11.11}$$

$$\begin{aligned} =\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\delta_j(z_i)\Big(&- B_{F_j^*}\left(t(x_i)\|\eta_j\right) \\ &+ F_j^*(t(x_i)) + k_j(x_i) + \log\omega_j\Big) \end{aligned} \tag{11.12}$$

Maximizing the complete average log-likelihood ll' is equivalent to minimizing the cost function $K = -ll'$:

$$K = -ll' = \frac{1}{n}\sum_{j=1}^{k}\sum_{i\in\mathcal{C}_j} U_j\left(x_i, \eta_j\right) \tag{11.13}$$

where

$$U_j(x_i, \eta_j) = -\Big(\log p_{F_j}(x_i|\theta_j) + \log\omega_j\Big) \tag{11.14}$$

$$= B_{F_j^*}\left(t(x_i)\|\eta_j\right) - F_j^*(t(x_i)) - k_j(x_i) - \log\omega_j \tag{11.15}$$

is the cost for the observation i to have been sampled from the component j. Notice this cost depends on j since each component has a different generator F_j and a different auxiliary carrier measure k_j.

This minimization problem can be solved with the Lloyd k-means algorithm [75] using the cost function U (which is not a distance nor a divergence and can even be negative): the proof of this statement is given in the next section.

After the execution of the Lloyd algorithm (or any other heuristic), the log-likelihood has been optimized for fixed ω_j and a fixed family p_{F_j}. The final step is to update the weights using the proportion of samples in each cluster and to choose the exponential family (for generalized Gaussian and Gamma distributions this amounts to estimating the fixed parameters but we may imagine even more generic mixtures where the family of each component is chosen at each step among a discrete dictionary for exponential families).

11.3.3 Convergence to a local maximum

We now prove that the Lloyd method allows to find a local minimum by decreasing monotonically the cost function. The Lloyd method iterates over two main steps: *assignment* and *centroid updates*. The sketch of the proof is very similar to the usual proof, but we emphasize the fact the costs U_j are not distances nor divergences and may even be negative.

Let us denote by $\mathcal{C}_i^{(t)}$ the content of the cluster i at the t-th iteration and by $\eta_i^{(t)}$ the center of the cluster i at the t-th iteration. The cost function at the iteration t is:

$$K^{(t)} = \frac{1}{n} \sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j^{(t)}} U_j \left(x_i, \eta_j^{(t)} \right) \tag{11.16}$$

The assignment step allocates each point x_i to the center η_j which minimizes the cost $U_j(x_i, \eta_j)$, so we have:

$$K^{(t)} \leq \frac{1}{n} \sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j^{(t+1)}} U_j \left(x_i, \eta_j^{(t)} \right) \tag{11.17}$$

For each cluster, the centroid update step must solve the optimization problem $\eta_j^{(t+1)} = \arg\min_{\eta_j^*} \sum_{i \in \mathcal{C}_j^{(t+1)}} U_j \left(x_i, \eta_j^* \right)$ where the cost function is:

$$\sum_{i \in \mathcal{C}_j^{(t+1)}} B_{F_j^*} \left(t(x_i) \| \eta_j \right) - {F_j}^*(t(x_i)) - k_j(x_i) - \log \omega_j$$

By removing the constant terms $-{F_j}^*(t(x_i)) - k_j(x_i) - \log \omega_j$ which do not depend on the centroids $\eta_i^{(t)}$ the original problem becomes equivalent to the following problem: $\eta_j^{(t+1)} = \arg\min_{\eta^*} \sum_{i \in \mathcal{C}_j^{(t+1)}} B_{F_j^*} \left(t(x_i) \| \eta_j \right)$ which is the problem of the computation of a right-sided Bregman centroid [94]: this centroid is known in closed-form [12] $\eta_j^{(t+1)} = \sum_{i \in \mathcal{C}_j^{(t+1)}} t(x_i)$. Since the centroid update minimizes the average cost to each centroid, we now have:

$$\frac{1}{n} \sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j^{(t+1)}} U_j \left(x_i, \eta_j^{(t+1)} \right) \leq \frac{1}{n} \sum_{j=1}^{k} \sum_{i \in \mathcal{C}_j^{(t+1)}} U_j \left(x_i, \eta_j^{(t)} \right) \tag{11.18}$$

Combining the equations (11.17) and (11.18) we get a global inequality which characterizes the monotonic decrease of the cost function.

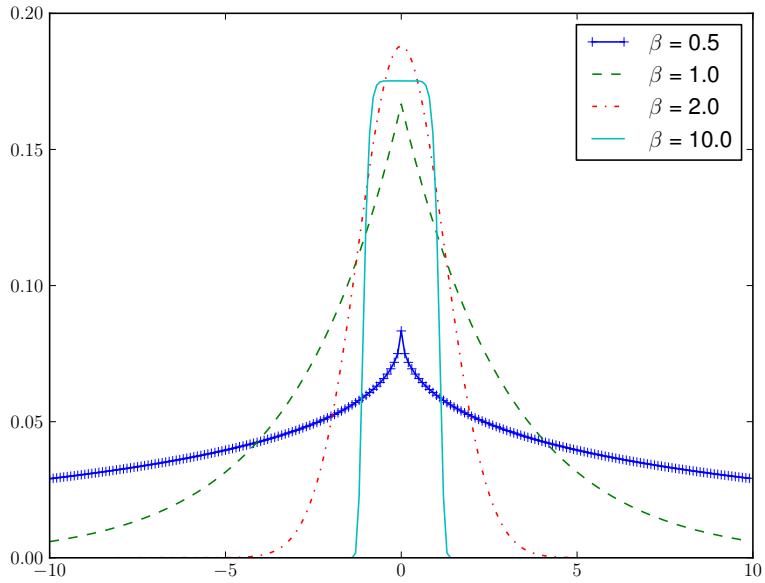


Figure 11.3: Probability density function for various β (with a normal distribution $\beta = 2.0$, a Laplace $\beta = 1.0$ and a nearly uniform distribution $\beta = 10.0$).

11.4 Mixtures of generalized Gaussians with various shape parameters

11.4.1 Generalized Gaussian distributions

The generalized Gaussian (GG) [78, 85] replaces the square in the usual Gaussian distribution by a parameter β. This family thus contains the normal law ($\beta = 2$), the Laplace law ($\beta = 1$) and even the uniform law as a limit case ($\beta \to +\infty$):

$$f(x; \mu, \alpha, \beta) = \frac{\beta}{2\alpha\Gamma(1/\beta)} \exp\left(-\frac{|x-\mu|^\beta}{\alpha^\beta}\right) \tag{11.19}$$

with $\alpha > 0$ (the scale parameter) and $\beta > 0$ (the shape parameter).

Generalized Gaussian distributions have been successfully used in problems of texture classification [41, 30] and mixtures of these distributions have been used for image and video segmentation [4]. We focus here on one dimensional distributions but a multivariate generalized Gaussian can be written as a product of one dimensional laws.

Generalized Gaussian are exponential families for fixed μ and β with the parameters (refer to Table B.6 in Annex B for full details):

- $t(x) = -|x - \mu|^{\beta}$,
- $\theta = \alpha^{-\beta}$,
- $F(\theta) = \beta \log \theta - \log \beta + \log \left(2\Gamma\left(\frac{1}{\beta}\right)\right)$,
- $k(x) = 0$.

11.4.2 Maximum likelihood estimator for a fixed shape

There is no maximum likelihood estimator known in closed-form but a numerical scheme to estimate the parameters α and β has been proposed in [41] . Using results from the framework of exponential families, we can estimate the expectation parameters $\eta = \nabla F^*(\theta) = -\frac{1}{\theta} = \sum_{i=1}^{N} t(x_i)$ which is equivalent to the proposed estimator for α.

The shape parameter β can be estimated as the solution of the equation:

$$1 + \frac{\psi(1/\beta)}{\beta} - \frac{\sum_{i=1}^{N} |x_i - \mu|^{\beta} \log |x_i - \mu|}{\sum_{i=1}^{N} |x_i - \mu|^{\beta}} + \frac{\frac{\beta}{N} \sum_{i=1}^{N} |x_i - \mu|^{\beta}}{\beta} = 0 \quad (11.20)$$

This can be solved using the Newton-Raphson method initialized by a dichotomic search between $\beta = 0$ and $\beta = 20$ (for a high enough β the generalized Gaussian law is very close to a uniform law). In some applications, it may be worth limiting the estimation to this dichotomic search in order to reduce the computation time.

11.5 Efficient learning of mixtures of Gamma distributions

11.5.1 Gamma distribution

The general case of the Gamma distribution is

$$p(x; \alpha, \beta) = \frac{\beta^{\alpha} x^{\alpha - 1} \exp(-\beta x)}{\Gamma(\alpha)} \quad (11.21)$$

with $\alpha, \beta > 0$: α is called the *shape* parameter and β is called the *rate* parameter (or *inverse scale* parameter).

This distribution is an exponential family with the following parameterization:

Natural parameters $(\theta_1, \theta_2) = (-\beta, \alpha - 1)$

Sufficient statistics $t(x) = (x, \log x)$

Log normalizer $F(\theta_1, \theta_2) = (-(\theta_2 + 1)\log(-\theta_1) + \log\Gamma(\theta_2 + 1))$

Gradient log normalizer $\nabla F(\theta_1, \theta_2) = \left(\frac{\theta_2+1}{-\theta_1}, -\log(-\theta_1) + \psi(\theta_2 + 1)\right)$

Dual log normalizer $F^\star(\eta_1, \eta_2) =$

$$\left\langle (\nabla F)^{-1}(\eta_1, \eta_2), (\eta_1, \eta_2) \right\rangle - F\left((\nabla F)^{-1}(\eta_1, \eta_2)\right)$$

Although the log-normalizer F is known in closed-form, it is not the case for its dual $F^\star$. It thus requires numerical approximation, which is computationally costly.

11.5.2 Restriction of the exponential family

The previous section was devoted to the *full* exponential family of the Gamma distribution, we now focus on a *restricted* exponential family, that is, the Gamma distribution with fixed parameter β:

$$p_\beta(x; \alpha) = \frac{\beta^\alpha x^{\alpha-1} \exp(-\beta x)}{\Gamma(\alpha)} \tag{11.22}$$

with $\alpha, \beta > 0$.

Such a restriction can be seen as a particular case of *restricted exponential family* [66, 6]: we consider only a subset of the possible parameters by imposing constraints on the space of parameters.

This is still an exponential family with the following parameterization (full details in Table B.8):

Natural parameters $\theta = \alpha - 1$

Log normalizer $F(\theta) = -(\theta + 1)\log(\beta) + \log\Gamma(\theta + 1))$

Gradient log normalizer $\nabla F(\theta) = -\log(\beta) + \psi(\theta + 1)$

Dual log normalizer $F^\star(\eta) = \langle \nabla F^\star(\eta), \eta \rangle - F\left(\nabla F^\star(\eta)\right)$

Gradient of the dual log normalizer $\nabla F^\star(\eta) = (\nabla F)^{-1}(\eta)$

The ∇F can be inverted in closed-form with respect to the inverse digamma function ψ^{-1}, giving:

$$(\nabla F)^{-1}(\eta) = \psi^{-1}(\eta + \log\beta) - 1 = \nabla F^\star(\eta) \tag{11.23}$$

We can now compute the $F^\star$ function by directly applying the Legendre transform to the log-normalizer F:

$$\begin{aligned} F^\star(\eta) &= \langle \nabla F^\star(\eta), \eta \rangle - F\left(\nabla F^\star(\eta)\right) && (11.24)\\ &= \eta\left(\psi^{-1}(\eta+\log\beta) - 1\right) + \psi^{-1}(\eta+\log\beta)\log\beta \\ &\quad - \log\Gamma\left(\psi^{-1}(\eta+\log\beta)\right) && (11.25) \end{aligned}$$

The function ψ^{-1} can be efficiently computed with a dichotomic search (see Algorithm 1).

Algorithm 1 Computation of the inverse digamma function ψ^{-1}

Require: $y = \psi(x)$, a precision ϵ
Ensure: $\tilde{x} \approx x$
  $L \leftarrow 1$
  $\tilde{x} \leftarrow \exp(y)$
  while $L > \epsilon$ **do**
    if $y - \psi(\tilde{x}) > 0$ **then**
      $\tilde{x} \leftarrow \tilde{x} + L$
    else
      $\tilde{x} \leftarrow \tilde{x} - L$
    end if
  end while
  return $\tilde{x}$

11.5.3 Maximum likelihood estimator

Results from exponential families give an estimator for the expectation parameters of the fixed rate family:

$$\hat{\eta} = \frac{1}{n}\sum t(x_i) = -\log\hat{\alpha} + \psi(\beta) \tag{11.26}$$

By derivation of the likelihood function, we get an estimator for the rate parameter β:

$$\hat{\beta} = \frac{n\hat{\alpha}}{\sum x_i} \tag{11.27}$$

11.6 Conclusion

After a presentation of the original *k*-MLE algorithm, the *extended k-MLE* algorithm is introduced: this generalization allows to deal with mixture in which the exponential family is not the same for all the components. Such an extension allows to build efficiently mixtures of generalized Gaussian and of Gamma laws. In the next chapter, we present `libmef` which is the `C` library implementing this algorithm.

Chapter 12

Software library

12.1 Presentation

The `pyMEF` library used in Chapter 7 was a first step to have a generic library to deal with mixtures of exponential families but it is not adapted to all use cases: it inherits from the `Python` drawback of being rather slow (even with the use of the couple `numpy` and `scipy`), it is unsuitable for bindings to foreign languages such as `Scilab` or `Matlab`; moreover it had been designed for mixtures of exponential families in the strict sense, where all the components share the same exponential family. For all these reasons, we decided to build a new library written in the *C* language (with the help of the `GNU GSL` library [50] for scientific computing): it is faster, allowing to compete with state-of-art methods on the field of computation time; it is easier to write bindings to others languages, easing the spread of our methods; it is designed to bear mixture with different families inside, allowing to build mixtures of generalized Gaussian distributions.

12.2 Learning mixtures

12.2.1 Manually creating a mixture

In the following example (see Figure 12.1 for the complete listing), a mixture of Gaussian with 3 components is created from scratch: the mixture is then displayed (in a human friendly format), the densities of the mixture and all its individual components are printed.

The first part is to create an empty container that will receive the components:

```
mixture mix = mixture_create(3);
```

We then create the family which will be shared between all the components and put it inside each component:

```
#include <stdio.h>
#include <mef.h>

int main(int argc, char **argv) {
  mixture mix = mixture_create(3);

  Gaussian_family ef = Gaussian();
  mix->ef[0] = (family)ef;
  mix->ef[1] = (family)ef;
  mix->ef[2] = (family)ef;

  mix->params[0] = (param)Gaussian_create_source(ef, \
                                                 0., 1.);
  mix->params[1] = (param)Gaussian_create_source(ef, \
                                                -10., 3.);
  mix->params[2] = (param)Gaussian_create_source(ef, \
                                                 5., 5.);

  mix->weights[0] = 0.2;
  mix->weights[1] = 0.5;
  mix->weights[2] = 0.3;

  mixture_fprint(stderr, mix);

  unsigned int n = 1000;
  double x_min = -20., x_max = 20., dx = (x_max-x_min)/n;
  double x = x_min;
  for(int i=0; i < n; i++) {
    printf("%f %f %f %f %f\n", x, mixture_pdf(mix, x),
         mix->weights[0] * ef->pdf(ef, x, mix->params[0]),
         mix->weights[1] * ef->pdf(ef, x, mix->params[1]),
         mix->weights[2] * ef->pdf(ef, x, mix->params[2])
         );
    x += dx;
  }
}
```

Figure 12.1: Building a mixture by hand with `libmef`

```
Gaussian_family ef = Gaussian();
mix->ef[0] = (family)ef;
mix->ef[1] = (family)ef;
mix->ef[2] = (family)ef;
```

What we create with the constructor `Gaussian` is a pointer to a C `struct` of type `Gaussian_family` but in order to use it with the generic mixture structure we need to "forget" the fact that it is a Gaussian with the cast `(family)ef`. (The `libmef` library relies heavily on this kind of pseudo subtyping using cast and `struct`.)

The next step is to create the parameters and the weights used for the components: we choose here to create source parameters but constructors are also available for natural and expectation parameters.

```
mix->params[0] = (param)Gaussian_create_source(ef,  0., 1.);
mix->params[1] = (param)Gaussian_create_source(ef,-10., 3.);
mix->params[2] = (param)Gaussian_create_source(ef,  5., 5.);

mix->weights[0] = 0.2;
mix->weights[1] = 0.5;
mix->weights[2] = 0.3;
```

As for the family, we forget the precise family to build generic parameters with the cast `(param)`: this ensures that the parameters can only be used through the generic functions implemented in the family definition.

The mixture is then printed in a human friendly format with the function `mixture_fprint`:

```
mixture_fprint(stderr, mix);
```

which produces the following output:

```
0.200000 Gaussian(mu: 0.000000, sigma2: 1.000000)
0.500000 Gaussian(mu: -10.000000, sigma2: 3.000000)
0.300000 Gaussian(mu: 5.000000, sigma2: 5.000000)
```

The last step in this example is to output the values of the pdf. We see here the use of the `mixture_pdf` function and of the `pdf` pseudo-methods of each component (each `struct` describing a family contains pointers to the functions which implements the canonical decomposition):

```
for(int i=0; i < n; i++) {
  printf("%i %f %f %f %f %f\n", i, x,
         mixture_pdf(mix, x),
         mix->weights[0] * ef->pdf(ef, x, mix->params[0]),
```

```
          mix->weights[1] * ef->pdf(ef, x, mix->params[1]),
          mix->weights[2] * ef->pdf(ef, x, mix->params[2])
          );
  x += dx;
}
```

12.2.2 Learning a mixture

The main interest of a library like `libmef` is obviously to learn a mixture from a set of observations. The example of Figure 12.2 learns a Gaussian mixture model using the Bregman Soft Clustering algorithm from a set of random data (generated using the `GSL` built-in random generator).

After some initialization and the random generation part, we begin the exponential families related part by creating an instance of the Gaussian distribution which is then used to create the EM problem (the `Gaussian_family` object is casted to the generic `family` type since Bregman Soft Clustering is not supposed to know the precise family used):

```
Gaussian_family ef = Gaussian();
em em1 = em_create((family)ef, k, data, k*n, d);
```

The next step is to find an initialization of the iterative method. The choice here is to use the best solution among 10 runs of k-means on the observations and to use a MLE on each cluster:

```
double best_error = DBL_MAX;
kmeans km, km0;
for (unsigned int i=0; i<10; i++) {
  km0 = kmeans_create(k, data, k*n, d);
  kmeans_initialize_random(km0, rng);
  kmeans_run(km0);

  if (km0->old_error < best_error) {
    km = km0;
    best_error = km0->old_error;
  }
  else {
    kmeans_destroy(km0);
  }
}
em_initialize_from_clustering(em1, km->weights, km->affectation);
```

Given this initial solution, the EM is launched, displayed, and finally destroyed:

```
mixture mix = em_run(em1);
mixture_fprint(stdout, mix);
em_destroy(em1);
```

Contrary to Bregman Soft Clustering, some algorithms are intrinsically linked to a specific family. In the example of Figure 12.3 which uses EM for Gamma laws, the main difference is that there is no need to create the family and to give it in argument to the problem creation function, since it is already handled by this creation function. We thus simply have:

```
emgamma em = emgamma_create(k, data, k*n, d);
```

12.3 Extending `libmef`

The core of `libmef` is not only the algorithms on exponential families, it is also the implementations of the different families. Since it is obviously more time consuming to write a pure `C` code instead of a `Python` module using `scipy` and `numpy`, lots of efforts have been made to ease the production of the necessary code. We illustrate the development of a family through some extracts of the `GammaFixedRate` family.

The first point to help potential developers is the mechanism which allows to describe at runtime the data structures used in the implementation: with the following description lots of useful operations are available without writing supplementary code (memory management, pretty printing, arithmetic operations, dot product, etc).

```
unsigned int U(arguments_len) = 1;
struct dynamic_type U(arguments_descr)[1] = {
  {"rate"  , offsetof(struct family, rate), DOUBLE},
};

unsigned int U(source_len) = 1;
struct dynamic_type U(source_descr)[1] = {
  {"shape", offsetof(struct source, shape), DOUBLE},
};

unsigned int U(natural_len) = 1;
struct dynamic_type U(natural_descr)[1] = {
  {"theta1", offsetof(struct natural, theta1), DOUBLE},
};

unsigned int U(expectation_len) = 1;
struct dynamic_type U(expectation_descr)[1] = {
```

```
#include <mef.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>

int main(int argc, char **argv) {
  int k = 3, n = 5000, d = 1,
  double *data = malloc(k * n * sizeof(double));
  const gsl_rng_type *T; gsl_rng_env_setup();
  T = gsl_rng_default; gsl_rng *rng = gsl_rng_alloc(T);
  for (unsigned int i=0; i<n; i++) {
    data[i] = gsl_ran_gaussian(rng, 1.0) + 1;
    data[2 * i] = gsl_ran_gaussian(rng, 1.0) + 0;
    data[3 * i] = gsl_ran_gaussian(rng, 1.0) + -10;
  }

  Gaussian_family ef = Gaussian();
  em em1 = em_create((family)ef, k, data, k*n, d);

  double best_error = DBL_MAX;
  kmeans km, km0;
  for (unsigned int i=0; i<10; i++) {
    km0 = kmeans_create(k, data, k*n, d);
    kmeans_initialize_random(km0, rng);
    kmeans_run(km0);

    if (km0->old_error < best_error) {
      km = km0;
      best_error = km0->old_error;
    }
    else {
      kmeans_destroy(km0);
    }
  }
  em_initialize_from_clustering(em1, km->weights, \
                                km->affectation);
  mixture mix = em_run(em1);
  mixture_fprint(stdout, mix);
  em_destroy(em1);
```

Figure 12.2: Learning a mixture using Bregman Soft Clustering

```
#include <mef.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>

int main(int argc, char **argv) {
  int k = 3, n = 5000, d = 1;
  double *data = malloc(k * n * sizeof(double));
  /* random generation of observations in data as
     in the previous example */

  emgamma em = emgamma_create(k, data, k*n, d);
  double best_error = DBL_MAX;
  kmeans km, km0;
  for (unsigned int i=0; i<10; i++) {
    km0 = kmeans_create(k, data, k*n, d);
    kmeans_initialize_random(km0, rng);
    kmeans_run(km0);

    if (km0->old_error < best_error) {
      km = km0;
      best_error = km0->old_error;
    }
    else {
      kmeans_destroy(km0);
    }
  }

  emgamma_initialize_from_clustering(em, km->weights,\
                                     km->affectation);
  emgamma_run(em);
  mixture_fprint(stdout, em->mixture);

  kmeans_destroy(km);
  emgamma_destroy(em);
}
```

Figure 12.3: Learning a mixture with the EM-Gamma algorithm

```
  {"eta1", offsetof(struct expectation, eta1), DOUBLE},
};
}
```

The common functions such as the pdf have to be written by hand: we rely heavily on the `GNU Scientific Library` for the special functions such as Γ or ψ. Some functions must accept any type of parameters in argument: these functions expect a generic `param` object which is converted to the desired kind of parameter (for example, with `(source)ef->as_source(ef, p)` in the function `pdf`).

```
double U(pdf)(family ef, double x, param p) {
  CHECK_EF(ef, p);
  source lambda = (source)ef->as_source(ef, p);
  double alpha  = lambda->shape;
  double beta   = ef->rate;

  return pow(beta, alpha) * pow(x, alpha-1) * exp(-beta * x) \
         / gsl_sf_gamma(alpha);
}
```

Other functions expect a specific kind of argument (for example, the type conversion functions, or the F, ∇F functions): in this case, we simply use a macro to check the type of the parameters.

```
void U(lambda2theta)(family ef, param lambda0, param theta0) {
  debug("lambda2theta\n");
  CHECK_EF(ef, lambda0);
  CHECK_EF(ef, theta0);
  CHECK_TYPE(lambda0, SOURCE);
  CHECK_TYPE(theta0, NATURAL);
  source lambda = (source)lambda0;
  natural theta = (natural)theta0;

  double alpha  = lambda->shape;

  theta->theta1 = alpha - 1;
}

...

double U(F)(family ef, param theta0) {
  CHECK_EF(ef, theta0);
  CHECK_TYPE(theta0, NATURAL);
```

```
  double beta  = ef->rate;
  double theta = ((natural)theta0)->theta1;

  return -(theta + 1) * log(beta) + log(gsl_sf_gamma(theta + 1));
}

void U(gradF)(family ef, param theta0, param eta0) {
  CHECK_EF(ef, theta0);
  CHECK_EF(ef, eta0);
  CHECK_TYPE(theta0, NATURAL);
  CHECK_TYPE(eta0,   EXPECTATION);

  double beta     = ef->rate;
  expectation eta = (expectation)eta0;
  double    theta = ((natural)theta0)->theta1;

  eta->eta1 = -log(beta) + gsl_sf_psi(theta + 1);
}
```

The random generator is simply a wrapper around the corresponding `GSL` function when it is available (but some families like the generalized Gaussian may need more work):

```
double U(rand)(family ef, param param, gsl_rng *rng) {
  source lambda = (source)ef->as_source(ef, param);

  return gsl_ran_gamma(rng, lambda->shape, 1/ef->rate);
}
```

All the process of linking the runtime description of data structure to dynamic pretty printer and other dynamic functions is handled automatically with the help of some `C` preprocessor macros. All this architecture allows the developer to focus on scientific computing rather than programming details.

12.4 Conclusion

The `libmef` library about mixtures of exponential families has been introduced: this `C` library implements the original and extended k-MLE algorithm but also other mixture algorithms to serve as a reference. These implementations are used in the next chapter to evaluate experimentally the proposed methods.

Chapter 13

Experiments

13.1 Introduction

The goal of these experiments is to evaluate the efficiency of the proposed methods using two criteria: we look both at the quality of the mixtures produced (using the log-likelihood) and at the computation time. The new k-MLE methods are not evaluated for a particular application: we focus here on the pure mixture modeling efficiency and we do not want to perturb the experiments with the other steps of a complete application. Moreover we expect these experiments to show the interest of k-MLE methods for *any* application using mixture models: k-MLE can be seen as a drop-in replacement in any process which uses mixture models.

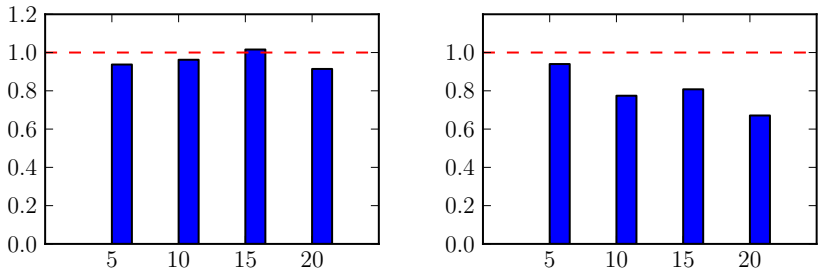
13.2 Mixtures of generalized Gaussian

The k-MLE variant for mixtures of generalized Gaussians is evaluated against the EM procedure proposed by Allili [3]. Since closed-form formulas for the M-step are not known, Allili uses a Newton-Raphson approximation scheme to estimate the β parameter of each component.

The results presented here are computed on textures from the Brodatz textures dataset [24]. The task is to model the low frequency part of the wavelet decomposition of each texture: this has originally been done with a single generalized Gaussian [41] but it has been shown that mixture models give better results [4].

We see on Figure 13.1 experimental results: the log-likelihood and the computation time are shown with respect to the number of components of the mixture. We plot here a relative value which is the ratio between the value from k-MLE and the value from EM (the higher the better for log-likelihood, the lower the better for time).

Figure 13.1(a) shows that we obtain results always very close to the reference EM algorithm. It is interesting to notice that even if k-MLE was not



(a) Log-likelihood (b) Computation time

Figure 13.1: Ratio between k-MLE et EM for mixtures of generalized Gaussian (EM is our reference and has always the score of 1). We get similar qualities, but k-MLE is always faster.

designed to maximize the log-likelihood (it is rather maximizing the *complete* log-likelihood) it still achieves very good performances on this point.

Figure 13.1(b) displays the computation time: we see that k-MLE is always faster than EM by 5 to 35 percents.

13.3 Mixtures of Gamma

The reference used in the experiments on Gamma mixtures is the EM variant introduced by Almhana, Liu, Choulakian, and McGorman with a specific M-step used for the α and β parameters. Given the current estimate for the parameters ω, α and β, the new values can be computed in closed-form with:

$$\omega_i^{(k+1)} = \frac{1}{n} \sum_{t=1}^{n} p(i|x_t, \theta^{(k)}) \tag{13.1}$$

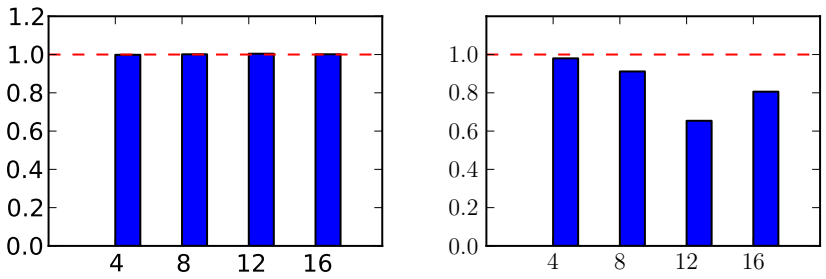
$$\beta_i^{(k+1)} = \frac{\alpha_i^{(k)} \sum_{t=1}^{n} p(i|x_t, \theta^{(k)})}{\sum_{t=1}^{n} x_t p(i|x_t, \theta^{(k)})} \tag{13.2}$$

$$\alpha_i^{(k+1)} = \alpha_i^{(k)} + \frac{1}{k} G \tag{13.3}$$

where

$$G = \frac{1}{n} \sum_{t=1}^{n} \left(\log x_t + \log \beta_i^{(k)} - \psi(\alpha_i^{(k)}) \right) p(i|x_t, \theta^{(k)}) \tag{13.4}$$

The experiments here are made on the same bio-informatics dataset as in Part II: since this dataset is made of distance measures (which are pos-



(a) Log-likelihood (b) Computation time

Figure 13.2: Ratio between k-MLE et EM for mixtures of Gamma laws (EM is our reference and has always the score of 1). k-MLE behaves the same as in the previous experiments: the qualities are similar, but k-MLE is always faster.

itive), it is more meaningful to use a mixture of distributions defined on a the positive support.

Figure 13.2 is made in the same way as Figure 13.1: we plot the ratio between the value from k-MLE and the value from EM (the higher the better for log-likelihood, the lower the better for time).

k-MLE for Gamma behaves the same as k-MLE for generalized Gaussian: we see on Figure 13.2(a) that the quality is the same as for EM and the computation time is up to 35 percents lower.

Chapter 14

Conclusion NONUMBER

We presented a powerful extension of the k-MLE algorithm for exponential families: our extension allows to build mixture models where all the components do not share the same exponential family. With this new work, we are able to efficiently build mixtures of generalized Gaussian and mixtures of Gamma laws. For both kinds of mixtures, we get a big improvement in speed without losing quality: even if k-MLE maximizes the complete log-likelihood instead of the log-likelihood, we still get high quality mixtures in terms of log-likelihood. Since our method is faster but as good as EM, we may also imagine the extension of k-MLE to be used as a drop-in replacement of EM in all applications which use mixture models.

Our extension also offers interesting perspectives: although it may seem at first glance that our extension is a specialization of k-MLE either targeted to generalized Gaussians or to Gamma laws, it is in reality a generalization of the original algorithm. k-MLE is actually a special case of extended k-MLE where the choice of the family is limited to only one possible family.

In the two applications of extended k-MLE described here, the families were parameterized by a real number: we may imagine a more generic setup where the choice is made among a discrete dictionary of available families.

Chapter 15

General conclusion and perspectives NONUMBER

15.1 Concluding remarks NONUMBER

In this thesis, we proposed new information-geometric methods for mixture model learning. This was not the first computational glance at information geometry and mixture models but contrary to many of the previous work, we did not looked for generalization at all cost: the aim was not to build meta-algorithms where the exponential family or the divergence was a new parameter of the algorithm (like Bregman Soft Clustering and k-MLE for mixtures of exponential families, or all the other recent developments on Bregman divergences). In fact, we accepted to specialize the proposed methods to some families. In the work about simplification of kernel density estimators, we are dealing only with exponential families which can also be considered as kernels and we focus on Gaussian mixtures since it is one of the most widespread kind of kernels. The use of the Fisher-Rao distance and model centroids specializes even more our method since we are thus strictly limited to Gaussian distributions since the Fisher-Rao distance is only known in closed-form for Gaussians and the model centroid needs the hyperbolic geometry of the parameters of the Gaussian (or any other location-scale family). Even it may be theoretically possible to use the simplification without closed-form, the practical use of the method would be impossible, specially when working on kernel density estimators which have a really big number of components. In the k-MLE part, we deal with generalized Gaussian distributions and Gamma laws. Although extended k-MLE can be seen as a generalization of k-MLE able to work on mixtures of multiple exponential families and offers interesting perspectives for this still unexplored kind of mixtures (using the framework of information geometry), it was not the primary goal. The goal was instead to solve problems which do not enter in the strict case of mixtures of exponential fam-

ilies (either theoretically, like for generalized Gaussian, or practically, like for Gamma laws). This kind of specialization can be seen as a limitation of our tools compared to other existing generic algorithms but we prefer to see it as a chance. The space of exponential families or Bregman divergences is parameterized by the infinite space of convex functions and is thus probably too wide to be explored in a smart and efficient way with today knowledge. This is why most applications of generic algorithms use generally well known families, such as Gaussian, Gamma, Wishart or Von Mises-Fisher, or well known divergences, such as squared Euclidean, Kullback-Leibler, Itakura-Saito or squared Mahalanobis. What we suggest to do with the contributions here is to take the best of the world of specialized algorithms and of generic algorithms: mixture simplification and extended k-MLE use a generic architecture and make a specialized adaptation to solve a specific problem.

15.2 Future work NONUMBER

Although information geometry made a long walk since its early beginning in the 1930's, computational information geometry is still at its beginning: we have a lot of powerful theoretical tools but they are difficult to use in a computational point of view. First, some closed-form formulas may be missing. The most striking example is the Fisher-Rao distance: although it is the canonical divergence of information geometry, it is nearly impossible to use in practice except in rare cases. Another example is the Legendre dual of the log-normalizer of an exponential family (and its gradient) without which it is nearly impossible to apply a generic algorithm. Second, we have the problem of the choice of the family or of the divergence. The catalog of divergences, distances, dissimilarity measures, metrics available in the machine learning and related topics literature is nearly as vast as the literature itself: information geometry provides a unified framework to operate on a subclass of theses divergences but does not give any clue to select a divergence among all the possibilities. This is probably the most challenging and motivating problem for the long term: developing theoretical foundation helping this choice, or at least some rules of thumbs. A track which is undoubtedly worth to explore is distance learning: using a cross-validation setup, we may learn the parameters of the divergences (beginning with a single real number for α-divergences, but we may imagine for Bregman divergences to build a basis of common convex functions and to learn weights using the property of linearity with respect to the generator of these divergences). The same kind of setup could be also used of the family itself.

In shorter term, the full potential of Extended k-MLE is still to be explored: with this algorithm we are able to learn mixtures of multiple ex-

ponential families but this has been used only for very limited cases when the components are actually members of the same family of distributions. Instead of choosing a parameter on the real line, the algorithm may pick an exponential family from a dictionary of families by choosing the most likely one. Using a dictionary here avoids the pitfall described previously by artificially limiting the search space for the family.

We may also use a completely different approach to model complex densities: instead of using a mixture of unimodal exponential families, it may be possible to use a single multimodal family. The challenge in this case would be to learn the log-normalizer of this family, along with the other necessary parameters, probably using a likelihood criterion.

Appendix A

Notations and abbreviations

A.1 Exponential families NONUMBER

EF	Exponential family.
$\mathcal{X}$, x	Support of the exponential family, a point from this subset.
d, D	Dimension of the support, order of the family.
$p_F(x, \lambda)$, $p_F(x, \theta)$, $p_F(x, \eta)$	Exponential family with log-normalizer F and source (resp. natural and expectation) parameter λ (resp. θ, η).
λ, θ, η, Λ, Θ, H	Source, natural and expectation parameters of an exponential family and source, natural and expectation parameters spaces.
$l(x_1, \ldots, x_n; \theta)$, $ll(x_1, \ldots, x_n; \theta)$	Likelihood function and average log-likelihood of a distribution with parameters θ.
$\mathcal{N}(x; \mu, \sigma^2)$	Gaussian law with mean μ and variance σ^2.
MLE	Maximum likelihood estimator.

A.2 Divergences and centroids NONUMBER

In the following, the $\|$ notation emphasize the non-symmetry of the divergences.

$B_F(\cdot\|\|\cdot)$	Bregman divergence with generator F.
$\mathrm{FR}(\cdot,\cdot)$	Fisher-Rao distance.
c, c^L, c^R	Respectively centroid associated with a divergence, and left-sided and right-sided centroids when the divergence is not symmetrical.
$\mathrm{KL}(\cdot\|\|\cdot)$	Kullback-Leibler divergence.

A.3 Mixture models NONUMBER

$m(x), m(x;\theta,\omega)$	Mixture model, mixture model with vector of parameters θ and vector of weights ω.
EM	Expectation-Maximization
KDE	Kernel Density Estimator
n, N	Number of components of a mixture model, of a kernel density estimator

A.4 Norms NONUMBER

$\|\cdot\|$	Modulus of a complex number.
$\|\|\cdot\|\|$	Euclidean norm
$\|\|\cdot\|\|_M$	Minkowski norm

Appendix B

Exponential families flashcards

PDF	$f(x;\, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
$\Lambda \to \Theta$	$(\theta_1, \theta_2) = \left(\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\right)$
$\Theta \to \Lambda$	$(\mu, \sigma^2) = \left(-\frac{\theta_1}{2\theta_2}, -\frac{1}{2\theta_2}\right)$
$\Lambda \to H$	$(\eta_1, \eta_2) = \left(\mu, \sigma^2 + \mu^2\right)$
$H \to \Lambda$	$(\mu, \sigma^2) = (\eta_1, \eta_2 - \eta_1^2)$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta_1, \theta_2) = -\frac{\theta_1^2}{4\theta_2} + \frac{1}{2}\log\left(-\frac{\pi}{\theta_2}\right)$
Gradient log normalizer	$\nabla F(\theta) = \left(-\frac{-\theta_1}{2\theta_2}, -\frac{1}{2\theta_2} + \frac{\theta_1^2}{4\theta_2^2}\right)$
Dual log normalizer	$F^\star(\eta) = -\frac{1}{2}\log\left(\eta_1^2 - \eta_2\right)$
Gradient dual log normalizer	$\nabla F^\star(\eta) = \left(-\frac{\eta_1}{\eta_1^2 - \eta 2}, \frac{1}{2(\eta_1^2 - \eta 2)}\right)$
Sufficient statistic	$t(x) = (x, x^2)$
Carrier measure	$k(x) = 0$

Table B.1: Canonical decomposition of the Gaussian distribution

PDF	$f(x;\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma)}} \exp\left(\frac{(x-\mu)^T\Sigma(x-\mu)}{2}\right)$
$\Lambda \to \Theta$	$(\theta_1,\theta_2) = \left(\Sigma^{-1}\mu, \frac{1}{2}\Sigma^{-1}\right)$
$\Theta \to \Lambda$	$(\mu,\Sigma) = \left(\frac{1}{2}\theta_2^{-1}\theta_1, \frac{1}{2}\theta_2^{-1}\right)$
$\Lambda \to H$	$(\eta_1,\eta_2) = \left(\mu, -(\Sigma+\mu\mu^T)\right)$
$H \to \Lambda$	$(\mu,\Sigma) = \left(\eta_1, -(\eta_2+\eta_1\eta_1^T)\right)$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta_1,\theta_2) = \frac{1}{4}\mathrm{tr}\left(\theta_2^{-1}\theta_1\theta_1^T\right) - \frac{1}{2}\log\det\theta_2 + \frac{d}{2}\log\pi$
Gradient log normalizer	$\nabla F(\theta_1,\theta_2) = \left(\frac{1}{2}\theta_2^{-1}\theta_1, -\frac{1}{2}\theta_2^{-1} - \frac{1}{4}\left(\theta_2^{-1}\theta_1\right)\left(\theta_2^{-2}\theta_1\right)^T\right)$
Dual log normalizer	$F^\star(\eta_1,\eta_2) = \frac{1}{2}\log(1 + \eta_1^T\eta_2^{-1}\eta_1) - \frac{1}{2}\log\det(-\eta_2) - \frac{d}{2}\log(2\pi e)$
Gradient dual log normalizer	$\nabla F^\star(\eta_1,\eta_2) = \left(-\left(\eta_2+\eta_1\eta_1^T\right)^{-1}\eta_1, -\frac{1}{2}\left(\eta_2+\eta_1\eta_1^T\right)\right)$
Sufficient statistic	$t(x) = (x, -xx^T)$
Carrier measure	$k(x) = 0$

Table B.2: Canonical decomposition of the multivariate Gaussian distribution

PDF	$f(x;\lambda) = \frac{\lambda^x \exp(-\lambda)}{x!}$
$\Lambda \to \Theta$	$\theta = \log \lambda$
$\Theta \to \Lambda$	$\lambda = \exp \theta$
$\Lambda \to H$	$\eta = \lambda$
$H \to \Lambda$	$\lambda = \eta$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta) = \exp \theta$
Gradient log normalizer	$\nabla F^\star(\eta) = (\nabla F)^{-1}(\eta) = \log \eta$
Dual log normalizer	$F^\star(\eta) = \eta \log \eta - \eta$
Gradient dual log normalizer	$\nabla F^\star(\eta) = \log \eta$
Sufficient statistic	$t(x) = x$
Carrier measure	$k(x) = -\log x!$

Table B.3: Canonical decomposition of the Poisson law

PDF	$f_\mu(x;\lambda) = \frac{1}{2\sigma}\exp\left(\frac{-\lvert x-\mu\rvert}{\sigma}\right)$
$\Lambda \to \Theta$	$\theta = -\frac{1}{\sigma}$
$\Theta \to \Lambda$	$\lambda = -\frac{1}{\theta}$
$\Lambda \to H$	$\eta = \sigma$
$H \to \Lambda$	$\lambda = \eta$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta) = \log\left(-\frac{2}{\theta}\right)$
Gradient log normalizer	$\nabla F(\theta) = -\frac{1}{\theta}$
Dual log normalizer	$F^\star(\eta) = -\log\eta$
Gradient dual log normalizer	$\nabla F^\star(\eta) = -\frac{1}{\eta}$
Sufficient statistic	$t(x) = \lvert x-\mu\rvert$
Carrier measure	$k(x) = 0$

Table B.4: Canonical decomposition of the Laplace law

PDF	$f(x;\alpha,\beta) = \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}$
$\Lambda \rightarrow \Theta$	$(\theta_1, \theta_2) = (\alpha - 1, \beta - 1)$
$\Theta \rightarrow \Lambda$	$(\alpha, \beta) = (\theta_1 + 1, \theta_2 + 1)$
$\Lambda \rightarrow H$	**unknown in closed-form**
$H \rightarrow \Lambda$	**unknown in closed-form**
$\Theta \rightarrow H$	$\eta = \nabla F(\theta)$
$H \rightarrow \Theta$	$\theta = \nabla F^{\star}(\eta)$
Log normalizer	$F(\theta) = \log B(\theta_1 + 1, \theta_2 + 1)$
Gradient log normalizer	$\nabla F(\theta) = (\psi(\theta_1 + 1) - \psi(\theta_1 + \theta_2 + 2), \psi(\theta_1 + 1) - \psi(\theta_1 + \theta_2 + 2))$
Dual log normalizer	**unknown in closed-form**
Gradient dual log normalizer	**unknown in closed-form**
Sufficient statistic	$t(x) = (\log x, \log(1-x))$
Carrier measure	$k(x) = 0$

Table B.5: Canonical decomposition of the Beta law

PDF	$f_{\mu,\beta}(x;\alpha) = \frac{\beta}{2\alpha\Gamma(1/\beta)} \exp\left(-\frac{\lvert x-\mu\rvert^\beta}{\alpha^\beta}\right)$
$\Lambda \to \Theta$	$\theta = \alpha^{-\beta}$
$\Theta \to \Lambda$	$\alpha = \theta^{-1/\beta}$
$\Lambda \to H$	$\eta = -\frac{\alpha^\beta}{\beta}$
$H \to \Lambda$	$\alpha = (-\beta\eta)^{1/\beta}$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta) = -\frac{\log\theta}{\beta} - \log\beta + \log\left(2\Gamma\left(\frac{1}{\beta}\right)\right)$
Gradient log normalizer	$\nabla F(\theta) = -\frac{1}{\beta\theta}$
Dual log normalizer	$F^\star(\eta) = -\frac{1}{\beta} - \frac{1}{\beta}\log(-\beta\eta) + \log\beta - \log\left(2\Gamma\left(\frac{1}{\beta}\right)\right)$
Gradient dual log normalizer	$\nabla F^\star(\eta) = -\frac{1}{\beta\eta}$
Sufficient statistic	$t(x) = -\lvert x-\mu\rvert^\beta$
Carrier measure	$k(x) = 0$

Table B.6: Canonical decomposition of generalized Gaussian distribution

PDF	$f(x;\alpha,\beta) = \frac{\beta^\alpha x^{\alpha-1}\exp(-\beta x)}{\Gamma(\alpha)}$
$\Lambda \to \Theta$	$(\theta_1,\theta_2) = (-\beta, \alpha - 1)$
$\Theta \to \Lambda$	$(\alpha,\beta) = (\theta_2 + 1, -\theta_1)$
$\Lambda \to H$	**unknown in closed-form**
$H \to \Lambda$	**unknown in closed-form**
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta_1,\theta_2) = (-(\theta_2 + 1)\log(-\theta_1) + \log\Gamma(\theta_2+1))$
Gradient log normalizer	$\nabla F(\theta_1,\theta_2) = \left(\frac{\theta_2+1}{-\theta_1}, -\log(-\theta_1) + \psi(\theta_2+1)\right)$
Dual log normalizer	**unknown in closed-form**
Gradient dual log normalizer	**unknown in closed-form**
Sufficient statistic	$t(x) = (x, \log x)$
Carrier measure	$k(x) = 0$

Table B.7: Canonical decomposition of the Gamma law

PDF	$f_\beta(x;\alpha) = \frac{\beta^\alpha x^{\alpha-1}\exp(-\beta x)}{\Gamma(\alpha)}$
$\Lambda \to \Theta$	$\theta = \alpha - 1$
$\Theta \to \Lambda$	$\alpha = \theta + 1$
$\Lambda \to H$	$\eta = -\log\beta + \Psi(\alpha)$
$H \to \Lambda$	$\alpha = \psi^{-1}(\eta + \log\beta)$
$\Theta \to H$	$\eta = \nabla F(\theta)$
$H \to \Theta$	$\theta = \nabla F^\star(\eta)$
Log normalizer	$F(\theta) = -(\theta+1)\log\beta + \log\Gamma(\theta+1)$
Gradient log normalizer	$\nabla F(\theta) = -\log\beta + \psi(\theta+1)$
Dual log normalizer	$F^\star(\eta) = \eta(\psi^{-1}(\eta + \log\beta) - 1) + \psi^{-1}(\eta + \log\beta)\log\beta + \log\Gamma(\psi^{-1}(\eta + \log\beta))$
Gradient dual log normalizer	$\nabla F^\star(\eta) = \psi^{-1}(\eta + \log\beta) - 1$
Sufficient statistic	$t(x) = \log x$
Carrier measure	$k(x) = -\beta x$

Table B.8: Canonical decomposition of the Gamma law with fixed rate

Appendix C

Publications

C.1 Book chapter NONUMBER

- Olivier Schwander and Frank Nielsen. "Learning Mixtures by Simplifying Kernel Density Estimators." In: (Jan. 2013). Ed. by Frank Nielsen and Rajendra Bhatia, pp. 403–426

C.2 Journal articles NONUMBER

- Adelene Y. L. Sim, Olivier Schwander, Michael Levitt, and Julie Bernauer. "Evaluating Mixture Models for Building RNA Knowledge-Based Potentials." In: *Journal of Bioinformatics and Computational Biology* 10.02 (Apr. 2012)

C.3 International conferences NONUMBER

- Olivier Schwander and Frank Nielsen. "Fast Learning of Gamma Mixture Models with k-MLE." In: *Similarity-Based Pattern Recognition*. Ed. by Edwin Hancock and Marcello Pelillo. Lecture Notes in Computer Science 7953. Springer Berlin Heidelberg, Jan. 2013, pp. 235–249

- Olivier Schwander, Aurélien Schutz, Frank Nielsen, and Yannick Berthoumieu. "k-MLE for mixtures of generalized Gaussians." In: *2012 21st International Conference on Pattern Recognition (ICPR)*. 2012, pp. 2825–2828

- Olivier Schwander and Frank Nielsen. "Model centroids for the simplification of Kernel Density estimators." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012, pp. 737–740

- Olivier Schwander and Frank Nielsen. "pyMEF: A framework for exponential families in Python." In: *2011 IEEE Statistical Signal Processing Workshop (SSP)*. 2011, pp. 669–672
- Olivier Schwander and Frank Nielsen. "Non-flat clustering with alpha-divergences." In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2011, pp. 2100–2103
- Frank Nielsen, Sylvain Boltz, and Olivier Schwander. "Bhattacharyya Clustering with Applications to Mixture Simplifications." In: *2010 20th International Conference on Pattern Recognition (ICPR)*. 2010, pp. 1437–1440
- Olivier Schwander and Frank Nielsen. "Reranking with Contextual Dissimilarity Measures from Representational Bregman k-Means." In: *VISAPP (1)*. Ed. by Paul Richard and José Braz. INSTICC Press, 2010, pp. 118–123

C.4 National conferences NONUMBER

- Olivier Schwander and Frank Nielsen. "Simplification de modèles de mélange issus d'estimateur par noyau." In: *Gretsi 2011*. 2011
- Olivier Schwander and Frank Nielsen. "Apprentissage rapide de modèles de mélanges avec *k*-MLE et ses extensions." In: *Gretsi 2013*. 2013

Index

Bibliography

[1] J. Aczél. "On mean values." EN. In: *Bulletin (New Series) of the American Mathematical Society* 54.4 (Apr. 1948), pp. 392–400 (cit. on p. 42).

[2] S. M. Ali and S. D. Silvey. "A General Class of Coefficients of Divergence of One Distribution from Another." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 28.1 (Jan. 1966), pp. 131–142 (cit. on p. 28).

[3] M. S. Allili. "Wavelet-Based Texture Retrieval Using a Mixture of Generalized Gaussian Distributions." In: *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE. 2010, pp. 3143–3146 (cit. on pp. 103, 127).

[4] M. S. Allili, D. Ziou, N. Bouguila, and S. Boutemedjet. "Image and Video Segmentation by Combining Unsupervised Generalized Gaussian Mixture Modeling and Feature Selection." In: *Circuits and Systems for Video Technology, IEEE Transactions on* 20.10 (2010), pp. 1373–1377 (cit. on pp. 103, 111, 127).

[5] J. Almhana, Z. Liu, V. Choulakian, and R. McGorman. "A Recursive Algorithm for Gamma Mixture Models." In: *IEEE International Conference on Communications, 2006. ICC '06*. Vol. 1. June 2006, pp. 197–202 (cit. on pp. 49, 104, 128).

[6] Shun-Ichi Amari. "Differential Geometry of Curved Exponential Families-Curvatures and Information Loss." EN. In: *The Annals of Statistics* 10.2 (June 1982), pp. 357–385 (cit. on p. 113).

[7] Shun-Ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*. en. American Mathematical Soc., Apr. 2000 (cit. on pp. 14, 27, 40).

[8] T. Amin, M. Zeytinoglu, and Ling Guan. "Application of Laplacian Mixture Model to Image and Video Retrieval." In: *IEEE Transactions on Multimedia* 9.7 (2007), pp. 1416–1429 (cit. on pp. 49, 50).

[9] Yali Amit and Alain Trouvé. "Generative Models for Labeling Multi-object Configurations in Images." In: *Toward Category-Level Object Recognition*. Ed. by Jean Ponce, Martial Hebert, Cordelia Schmid, and Andrew Zisserman. Lecture Notes in Computer Science 4170. Springer Berlin Heidelberg, Jan. 2006, pp. 362–381 (cit. on p. 50).

[10] Erling Bernhard Andersen. "Sufficiency and Exponential Families for Discrete Sample Spaces." In: *Journal of the American Statistical Association* 65.331 (Sept. 1970) (cit. on p. 21).

[11] Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. "Clustering on the unit hypersphere using von mises-fisher distributions." In: *Journal of Machine Learning Research* 6 (2005), p. 2005 (cit. on p. 103).

[12] Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. "Clustering with Bregman Divergences." In: *J. Mach. Learn. Res.* 6 (Dec. 2005), 1705–1749 (cit. on pp. 14, 39, 45, 53, 54, 69, 101, 110).

[13] O. Barndorff Nielsen. *Information and exponential families : in statistical theory*. Wiley series in probability and mathematical statistics. Chichester [etc.]: Wiley, 1978 (cit. on p. 30).

[14] M. Basseville. "Divergence measures for statistical data processing-An annotated bibliography." In: *Signal Processing* (2012) (cit. on p. 46).

[15] Leonard E. Baum and Ted Petrie. "Statistical Inference for Probabilistic Functions of Finite State Markov Chains." In: *The Annals of Mathematical Statistics* 37.6 (Dec. 1966), pp. 1554–1563 (cit. on p. 52).

[16] Aharon Ben-Tal, Abraham Charnes, and Marc Teboulle. "Entropic means." In: *Journal of Mathematical Analysis and Applications* 139.2 (May 1989), pp. 537–551 (cit. on p. 44).

[17] Julie Bernauer, Xuhui Huang, Adelene Y. L. Sim, and Michael Levitt. "Fully differentiable coarse-grained and all-atom knowledge-based potentials for RNA structure evaluation." en. In: *RNA* 17.6 (June 2011), pp. 1066–1075 (cit. on p. 91).

[18] Anil Bhattacharyya. "On a measure of divergence between two statistical populations defined by their probability distributions." In: *Bull. Calcutta Math. Soc* 35.99-109 (1943), p. 4 (cit. on p. 47).

[19] Debjani Bhowmick, A. C. Davison, Darlene R. Goldstein, and Yann Ruffieux. "A Laplace mixture model for identification of differential expression in microarray experiments." en. In: *Biostatistics* 7.4 (Oct. 2006), pp. 630–641 (cit. on p. 49).

[20] C. Biernacki, G. Celeux, G. Govaert, and F. Langrognet. "Model-based cluster and discriminant analysis with the MIXMOD software." In: *Computational Statistics & Data Analysis* 51.2 (2006), pp. 587–600 (cit. on p. 79).

[21] Krzysztof Bogdan and Małgorzata Bogdan. "On Existence of Maximum Likelihood Estimators in Exponential Families." In: *Statistics* 34.2 (2000), pp. 137–149 (cit. on p. 30).

[22] Jean-Daniel Boissonnat, Frank Nielsen, and Richard Nock. *Bregman Voronoi Diagrams: Properties, Algorithms and Applications*. Anglais. Rapport de recherche RR-6154. INRIA, 2007, p. 48 (cit. on p. 39).

[23] Adrian W. Bowman. "An alternative method of cross-validation for the smoothing of density estimates." en. In: *Biometrika* 71.2 (Jan. 1984), pp. 353–360 (cit. on p. 55).

[24] Phil Brodatz. *Textures: a photographic album for artists and designers*. Vol. 66. Dover New York, 1966 (cit. on p. 127).

[25] Lawrence D. Brown. *Fundamentals of Statistical Exponential Families: With Applications in Statistical Decision Theory*. en. IMS, 1986 (cit. on pp. 22, 25).

[26] J. Burbea and C. Rao. "On the convexity of some divergence measures based on entropy functions." In: *IEEE Transactions on Information Theory* 28.3 (1982), pp. 489–495 (cit. on p. 46).

[27] Gilles Celeux and Gérard Govaert. "A Classification EM algorithm for clustering and two stochastic versions." In: *Comput. Stat. Data Anal.* 14.3 (Oct. 1992), 315–332 (cit. on p. 103).

[28] P. Chen, Y. Chen, and M. Rao. "Metrics defined by Bregman Divergences." EN. In: *Communications in Mathematical Sciences* 6.4 (Dec. 2008), pp. 915–926 (cit. on p. 46).

[29] Nikolai Nikolaevich Chencov. *Statistical Decision Rules and Optimal Inference*. en. American Mathematical Soc., 1982 (cit. on pp. 14, 27, 73).

[30] S.-K. Choy and C.-S. Tong. "Supervised Texture Classification Using Characteristic Generalized Gaussian Density." In: *Journal of Mathematical Imaging and Vision* 29 (1 2007), pp. 35–47 (cit. on p. 111).

[31] Andrzej Cichocki, Sergio Cruces, and Shun-ichi Amari. "Generalized Alpha-Beta Divergences and Their Application to Robust Nonnegative Matrix Factorization." In: *Entropy* 13.12 (Jan. 2011), pp. 134–170 (cit. on p. 14).

[32] Loren Cobb. "The Multimodal Exponential Families of Statistical Catastrophe Theory." en. In: *Statistical Distributions in Scientific Work*. Ed. by Charles Taillie, Ganapati P. Patil, and Bruno A. Baldessari. NATO Advanced Study Institutes Series 79. Springer Netherlands, Jan. 1981, pp. 67–90 (cit. on p. 49).

[33] S.I.R. Costa, S.A. Santos, and J.E. Strapasson. "Fisher information matrix and hyperbolic geometry." In: *2005 IEEE Information Theory Workshop*, p. 3 (cit. on pp. 74, 76).

[34] Harald Cramér. *Mathematical methods of statistics.* en. Princeton University Press, 1946 (cit. on p. 31).

[35] I. Csiszár. "Information-type measures of difference of probability distributions and indirect observations." In: *Studia Sci. Math. Hungar.* 2 (1967), pp. 299–318 (cit. on p. 28).

[36] G. Darmois. "Sur les lois de probabilites à estimation exhaustive." In: *C.R. Acad. Sci. Paris* 200 (1935), 1265–1266 (cit. on p. 21).

[37] Sanjoy Dasgupta and Matus J Telgarsky. "Agglomerative Bregman Clustering." In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. 2012, pp. 1527–1534 (cit. on pp. 36, 107).

[38] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (Jan. 1977), pp. 1–38 (cit. on p. 52).

[39] Arnaud Dessein. "Méthodes Computationnelles en Géométrie de l'Information et Applications Temps Réel au Traitement du Signal Audio." PhD thesis. Université Pierre et Marie Curie - Paris VI, Dec. 2012 (cit. on p. 14).

[40] Michel Deza and Elena Deza. *Encyclopedia of Distances*. en. Springer, May 2009 (cit. on p. 46).

[41] M. N. Do and M. Vetterli. "Wavelet-based texture retrieval using generalized Gaussian density and Kullback-Leibler distance." In: *Image Processing, IEEE Transactions on* 11.2 (2002), pp. 146–158 (cit. on pp. 58, 103, 111, 112, 127).

[42] J.-L. Durrieu, J.-P. Thiran, and F. Kelly. "Lower and upper bounds for approximation of the Kullback-Leibler divergence between Gaussian Mixture Models." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2012, pp. 4833–4836 (cit. on pp. 56, 57).

[43] Jack Edmonds and Richard M. Karp. "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems." In: *J. ACM* 19.2 (Apr. 1972), pp. 248–264 (cit. on p. 59).

[44] V. A. Epanechnikov. "Non-Parametric Estimation of a Multivariate Probability Density." In: *Theory of Probability & Its Applications* 14.1 (Jan. 1969), pp. 153–158 (cit. on p. 68).

[45] R. A. Fisher. "Two New Properties of Mathematical Likelihood." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 144.852 (Mar. 1934), pp. 285–307 (cit. on p. 21).

[46] Maurice Fréchet. "Sur l'extension de certaines évaluations statistiques au cas de petits echantillons." In: *Revue de l'Institut International de Statistique / Review of the International Statistical Institute* 11.3/4 (1943), p. 182 (cit. on p. 31).

[47] G. A. Galperin. "A concept of the mass center of a system of material points in the constant curvature spaces." In: *Communications in Mathematical Physics* 154.1 (1993), pp. 63–84 (cit. on pp. 69, 74).

[48] V. Garcia, F. Nielsen, and R. Nock. "Levels of details for gaussian mixture models." In: *Computer Vision–ACCV 2009* (2010), pp. 514–525 (cit. on pp. 69, 70, 79).

[49] B. Georgi, I. G. Costa, and A. Schliep. "PyMix- The Python mixture package- a tool for clustering of heterogeneous biological data." In: *BMC bioinformatics* 11.1 (2010), p. 9 (cit. on p. 79).

[50] Brian Gough. *GNU Scientific Library Reference Manual - Third Edition*. 3rd. Network Theory Ltd., 2009 (cit. on p. 117).

[51] R.M. Gray, A. Buzo, Jr. Gray A., and Y. Matsuyama. "Distortion measures for speech processing." In: *IEEE Transactions on Acoustics, Speech and Signal Processing* 28.4 (1980), pp. 367–376 (cit. on p. 39).

[52] L. R. Haff, P. T. Kim, J.-Y. Koo, and D. St P. Richards. "Minimax estimation for mixtures of Wishart distributions." EN. In: *The Annals of Statistics* 39.6 (Dec. 2011), pp. 3417–3440 (cit. on p. 50).

[53] Peter Hall, J. S. Marron, and Byeong U. Park. "Smoothed cross-validation." en. In: *Probability Theory and Related Fields* 92.1 (Mar. 1992), pp. 1–20 (cit. on p. 55).

[54] J. A. Hartigan and M. A. Wong. "Algorithm AS 136: A K-Means Clustering Algorithm." In: *Applied Statistics* 28.1 (1979), p. 100 (cit. on pp. 105, 107).

[55] W. K. Hastings. "Monte Carlo sampling methods using Markov chains and their applications." en. In: *Biometrika* 57.1 (Jan. 1970), pp. 97–109 (cit. on p. 51).

[56] E. Hellinger. "Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen." German. In: (1909) (cit. on p. 47).

[57] J.R. Hershey and P.A. Olsen. "Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models." In: *IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007*. Vol. 4. Apr. 2007, pp. IV–317–IV–320 (cit. on pp. 56, 57, 59, 60).

[58] H Hotelling. "Spaces of statistical parameters." In: *Bull. Amer. Math. Soc.* 36 (1930), p. 191 (cit. on p. 25).

[59] CGJ Jacobi. "Theoria novi multiplicatoris systemati aequationum differentialium vulgarium applicandi." In: *Journal für die reine und angewandte Mathematik* 27 (1844), pp. 199–268 (cit. on p. 59).

[60] Harold Jeffreys. "An Invariant Form for the Prior Probability in Estimation Problems." en. In: *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 186.1007 (Sept. 1946), pp. 453–461 (cit. on p. 46).

[61] F. Jelinek. "Continuous speech recognition by statistical methods." In: *Proceedings of the IEEE* 64.4 (1976), pp. 532–556 (cit. on p. 103).

[62] Robert Jenssen, Jose C. Principe, Deniz Erdogmus, and Torbjørn Eltoft. "The Cauchy-Schwarz divergence and Parzen windowing: Connections to graph theory and Mercer kernels." In: *Journal of the Franklin Institute* 343.6 (Sept. 2006), pp. 614–629 (cit. on p. 60).

[63] K. Kampa, E. Hasanbelliu, and J.C. Principe. "Closed-form Cauchy-Schwarz PDF divergence for mixture of Gaussians." In: *The 2011 International Joint Conference on Neural Networks (IJCNN)*. 2011, pp. 2578–2585 (cit. on p. 60).

[64] Robert E Kass. "The geometry of asymptotic inference." In: *Statistical Science* (1989), pp. 188–219 (cit. on p. 27).

[65] Robert E. Kass and Paul W. Vos. *Geometrical Foundations of Asymptotic Inference*. en. John Wiley & Sons, Sept. 2011 (cit. on p. 74).

[66] Robert W. Keener. "Curved Exponential Families." en. In: *Theoretical Statistics*. Springer Texts in Statistics. Springer New York, Jan. 2010, pp. 85–99 (cit. on p. 113).

[67] Felix Klein. "Vergleichende Betrachtungen über neuere geometrische Forschungen." de. In: *Mathematische Annalen* 43.1 (Mar. 1893), pp. 63–100 (cit. on p. 28).

[68] Andrey Kolmogorov. "Sur la notion de la moyenne." In: *Atti Accad. Naz. Lincei* 12 (1930), 388—391 (cit. on p. 42).

[69] Alexey Koloydenko, Meelis Käärik, and Jüri Lember. "On Adjusted Viterbi Training." en. In: *Acta Applicandae Mathematicae* 96.1-3 (May 2007), pp. 309–326 (cit. on p. 103).

[70] B. O. Koopman. “On Distributions Admitting a Sufficient Statistic.” In: *Transactions of the American Mathematical Society* 39.3 (May 1936), p. 399 (cit. on p. 21).

[71] H. W. Kuhn. “The Hungarian method for the assignment problem.” en. In: *Naval Research Logistics Quarterly* 2.1-2 (1955), 83–97 (cit. on p. 59).

[72] John Lafferty and Guy Lebanon. “Diffusion Kernels on Statistical Manifolds.” In: 6 (Dec. 2005), pp. 129–163 (cit. on p. 27).

[73] Jia Li and Hongyuan Zha. “Two-way Poisson mixture models for simultaneous document classification and word clustering.” In: *Computational Statistics & Data Analysis* 50.1 (Jan. 2006), pp. 163–180 (cit. on p. 50).

[74] Meizhu Liu, B.C. Vemuri, S.-I. Amari, and F. Nielsen. “Shape Retrieval Using Hierarchical Total Bregman Soft Clustering.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.12 (2012), pp. 2407–2419 (cit. on p. 60).

[75] S. P. Lloyd. “Least squares quantization in PCM.” In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137 (cit. on p. 109).

[76] Hông Vân Lê. *The uniqueness of the Fisher metric as information metric*. arXiv e-print 1306.1465. June 2013 (cit. on pp. 27, 73).

[77] PC Mahalanobis. “On the generalised distance in statistics.” In: vol. 2. Apr. 1936, pp. 49–55 (cit. on p. 38).

[78] S.G. Mallat. “A theory for multiresolution signal decomposition: the wavelet representation.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.7 (1989), pp. 674–693 (cit. on pp. 103, 111).

[79] Kameo Matusita. “Decision Rules, Based on the Distance, for Problems of Fit, Two Samples, and Estimation.” In: *The Annals of Mathematical Statistics* 26.4 (Dec. 1955), pp. 631–640 (cit. on p. 47).

[80] I. Mayrose, N. Friedman, and T. Pupko. “A Gamma mixture model better accounts for among site rate heterogeneity.” In: *Bioinformatics* 21.Suppl 2 (2005) (cit. on pp. 50, 104).

[81] Geoffrey McLachlan and David Peel. *Finite Mixture Models*. en. John Wiley & Sons, Apr. 2004 (cit. on p. 49).

[82] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. “Equation of State Calculations by Fast Computing Machines.” In: *The Journal of Chemical Physics* 21.6 (June 1953), pp. 1087–1092 (cit. on p. 51).

[83] Tetsuzo Morimoto. “Markov Processes and the H-Theorem.” In: *Journal of the Physical Society of Japan* 18 (Mar. 1963), p. 328 (cit. on p. 28).

[84] James Munkres. "Algorithms for the Assignment and Transportation Problems." In: *Journal of the Society for Industrial and Applied Mathematics* 5.1 (Mar. 1957), pp. 32–38 (cit. on p. 59).

[85] Saralees Nadarajah. "A generalized normal distribution." In: *Journal of Applied Statistics* 32.7 (2005), pp. 685–694 (cit. on p. 111).

[86] Mitio Nagumo. "Uber eine klasse der mittelwerte." In: *Japanese Journal of Mathematics* 7 (1930), pp. 71–79 (cit. on p. 42).

[87] Radford M. Neal. "Slice sampling." English. In: *Ann. Stat.* 31.3 (2003), pp. 705–767 (cit. on p. 51).

[88] Radford M. Neal and Geoffrey E. Hinton. "A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants." en. In: *Learning in Graphical Models*. Ed. by Michael I. Jordan. NATO ASI Series 89. Springer Netherlands, Jan. 1998, pp. 355–368 (cit. on p. 103).

[89] F. Nielsen. "Closed-form information-theoretic divergences for statistical mixtures." In: *2012 21st International Conference on Pattern Recognition (ICPR)*. 2012, pp. 1723–1726 (cit. on p. 60).

[90] F. Nielsen. "*k*-MLE: A fast algorithm for learning statistical mixture models." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2012, pp. 869–872 (cit. on p. 101).

[91] F. Nielsen, J. D. Boissonnat, and R. Nock. "On Bregman Voronoi diagrams." In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2007, p. 755 (cit. on p. 14).

[92] F. Nielsen and S. Boltz. "The Burbea-Rao and Bhattacharyya Centroids." In: *IEEE Transactions on Information Theory* 57.8 (2011), pp. 5455–5466 (cit. on pp. 46, 47).

[93] F. Nielsen and R. Nock. "Hyperbolic Voronoi diagrams made easy." In: *Arxiv preprint arXiv:0903.3287* (2009) (cit. on p. 76).

[94] F. Nielsen and R. Nock. "Sided and symmetrized Bregman centroids." In: *IEEE Transactions on Information Theory* 55.6 (2009), pp. 2882–2904 (cit. on pp. 14, 45, 46, 110).

[95] F. Nielsen, P. Piro, and M. Barlaud. "Bregman vantage point trees for efficient nearest Neighbor Queries." In: *IEEE International Conference on Multimedia and Expo, 2009. ICME 2009*. 2009, pp. 878–881 (cit. on p. 14).

[96] Frank Nielsen. "*k*-MLE: A fast algorithm for learning statistical mixture models." In: *CoRR* (2012) (cit. on pp. 101, 105, 107).

[97] Frank Nielsen and Rajendra Bhatia. *Matrix Information Geometry*. en. Springer, Aug. 2012 (cit. on p. 63).

[98] Frank Nielsen, Sylvain Boltz, and Olivier Schwander. "Bhattacharyya Clustering with Applications to Mixture Simplifications." In: *2010 20th International Conference on Pattern Recognition (ICPR)*. 2010, pp. 1437–1440 (cit. on p. 150).

[99] Frank Nielsen and Vincent Garcia. "Statistical exponential families: A digest with flash cards." In: *CoRR* 09114863 (2009) (cit. on pp. 51, 59, 73, 79, 80, 84).

[100] Emanuel Parzen. "On Estimation of a Probability Density Function and Mode." In: *The Annals of Mathematical Statistics* 33.3 (Sept. 1962), pp. 1065–1076 (cit. on pp. 55, 67).

[101] B. Pelletier. "Informative barycentres in statistics." In: *Annals of the Institute of Statistical Mathematics* 57.4 (2005), pp. 767–780 (cit. on p. 70).

[102] Paolo Piro. "Learning prototype-based classification rules in a boosting framework: application to real-world and medical image categorization." PhD thesis. Université de Nice Sophia-Antipolis, Jan. 2010 (cit. on p. 14).

[103] E. J. G. Pitman. "Sufficient statistics and intrinsic accuracy." In: *Mathematical Proceedings of the Cambridge Philosophical Society* 32.04 (1936), pp. 567–579 (cit. on p. 21).

[104] C. Radhakrishna Rao. "Information and accuracy attainable in the estimation of statistical parameters." In: *Bulletin of the Calcutta Mathematical Society* 37.3 (1945), pp. 81–91 (cit. on pp. 13, 25, 31).

[105] C. E. Rasmussen. "The infinite Gaussian mixture model." In: *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, Dec. 2000, pp. 554–560 (cit. on p. 91).

[106] F. Reverter and JM Oller. "Computing the Rao distance for Gamma distributions." In: *Journal of computational and applied mathematics* 157.1 (2003), pp. 155–167 (cit. on p. 74).

[107] R. Tyrrell Rockafellar. *Convex Analysis*. en. Princeton University Press, 1970 (cit. on p. 23).

[108] Guodong Rong, Miao Jin, and Xiaohu Guo. "Hyperbolic centroidal Voronoi tessellation." In: *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling*. SPM '10. New York, NY, USA: ACM, 2010, 117–126 (cit. on p. 74).

[109] Murray Rosenblatt. "Remarks on Some Nonparametric Estimates of a Density Function." EN. In: *The Annals of Mathematical Statistics* 27.3 (Sept. 1956), pp. 832–837 (cit. on pp. 55, 67).

[110] Mats Rudemo. "Empirical Choice of Histograms and Kernel Density Estimators." In: *Scandinavian Journal of Statistics* 9.2 (Jan. 1982), pp. 65–78 (cit. on p. 55).

[111] Christophe Saint-Jean and Frank Nielsen. "A new implementation of k-MLE for mixture modeling of Wishart distributions." In: *Geometry Science of Information (GSI'2013)*. Mar. 2013 (cit. on pp. 105, 107).

[112] Aurélien Schutz, Lionel Bombrun, and Yannick Berthoumieu. "*k*-centroids-based supervised classification of texture images: handling the intra-class diversity." Anglais. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vancouver, Canada, 2013, pp. 1498–1502 (cit. on p. 103).

[113] Olivier Schwander and Frank Nielsen. "Apprentissage rapide de modèles de mélanges avec *k*-MLE et ses extensions." In: *Gretsi 2013*. 2013 (cit. on p. 150).

[114] Olivier Schwander and Frank Nielsen. "Fast Learning of Gamma Mixture Models with k-MLE." In: *Similarity-Based Pattern Recognition*. Ed. by Edwin Hancock and Marcello Pelillo. Lecture Notes in Computer Science 7953. Springer Berlin Heidelberg, Jan. 2013, pp. 235–249 (cit. on pp. 99, 149).

[115] Olivier Schwander and Frank Nielsen. "Learning Mixtures by Simplifying Kernel Density Estimators." In: (Jan. 2013). Ed. by Frank Nielsen and Rajendra Bhatia, pp. 403–426 (cit. on pp. 63, 149).

[116] Olivier Schwander and Frank Nielsen. "Model centroids for the simplification of Kernel Density estimators." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012, pp. 737–740 (cit. on pp. 63, 149).

[117] Olivier Schwander and Frank Nielsen. "Non-flat clustering with alpha-divergences." In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2011, pp. 2100–2103 (cit. on p. 150).

[118] Olivier Schwander and Frank Nielsen. "pyMEF: A framework for exponential families in Python." In: *2011 IEEE Statistical Signal Processing Workshop (SSP)*. 2011, pp. 669–672 (cit. on pp. 63, 150).

[119] Olivier Schwander and Frank Nielsen. "Reranking with Contextual Dissimilarity Measures from Representational Bregman k-Means." In: *VISAPP (1)*. Ed. by Paul Richard and José Braz. INSTICC Press, 2010, pp. 118–123 (cit. on p. 150).

[120] Olivier Schwander and Frank Nielsen. "Simplification de modèles de mélange issus d'estimateur par noyau." In: *Gretsi 2011*. 2011 (cit. on p. 150).

[121] Olivier Schwander, Aurélien Schutz, Frank Nielsen, and Yannick Berthoumieu. "k-MLE for mixtures of generalized Gaussians." In: *2012 21st International Conference on Pattern Recognition (ICPR)*. 2012, pp. 2825–2828 (cit. on pp. 99, 149).

[122] J.C. Seabra, F. Ciompi, O. Pujol, J. Mauri, P. Radeva, and J. Sanches. "Rayleigh Mixture Model for Plaque Characterization in Intravascular Ultrasound." In: *IEEE Transactions on Biomedical Engineering* 58.5 (2011), pp. 1314–1324 (cit. on pp. 49, 50, 84).

[123] S. J. Sheather and M. C. Jones. "A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 53.3 (Jan. 1991), pp. 683–690 (cit. on pp. 55, 68).

[124] Adelene Y. L. Sim, Olivier Schwander, Michael Levitt, and Julie Bernauer. "Evaluating Mixture Models for Building RNA Knowledge-Based Potentials." In: *Journal of Bioinformatics and Computational Biology* 10.02 (Apr. 2012) (cit. on pp. 63, 92, 149).

[125] Sergio Venturini, Francesca Dominici, and Giovanni Parmigiani. "Gamma shape mixtures for heavy-tailed distributions." EN. In: *The Annals of Applied Statistics* 2.2 (June 2008). Zentralblatt MATH identifier: 05591297; Mathematical Reviews number (MathSciNet): MR2524355, pp. 756–776 (cit. on p. 104).

[126] Fei Wang, Tanveer Syeda-Mahmood, Baba C. Vemuri, David Beymer, and Anand Rangarajan. "Closed-Form Jensen-Renyi Divergence for Mixture of Gaussians and Applications to Group-Wise Shape Registration." In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009*. Ed. by Guang-Zhong Yang, David Hawkes, Daniel Rueckert, Alison Noble, and Chris Taylor. Lecture Notes in Computer Science 5761. Springer Berlin Heidelberg, Jan. 2009, pp. 648–655 (cit. on p. 60).

[127] Alfred Weber. "Über den Standort der Industrien. Erster Teil: Reine Theorie des Standorts." In: *Mohr, Tübingen* (1909) (cit. on p. 41).