On learning statistical mixtures maximizing the complete likelihood

Frank Nielsen

École Polytechnique
Sony Computer Science Laboratories

Abstract. Statistical mixtures are semi-parametric models ubiquitously met in data science since they can universally model smooth densities arbitrarily closely. Finite mixtures are usually inferred from data using the celebrated Expectation-Maximization framework that locally iteratively maximizes the incomplete likelihood by assigning softly data to mixture components. In this paper, we present a novel methodology to infer mixtures by transforming the learning problem into a sequence of geometric center-based hard clustering problems that provably maximizes monotonically the complete likelihood. Our versatile method is fast and uses low memory footprint: The core inner steps can be implemented using various generalized k-means type heuristics. Thus we can leverage recent results on clustering to mixture learning. In particular, for mixtures of singly-parametric distributions including for example the Rayleigh, Weibull, or Poisson distributions, we show how to use dynamic programming to solve exactly the inner geometric clustering problems. We discuss on several extensions of the methodology.

Keywords: Statistical mixtures, maximum likelihood estimator, expectation maximization, geometric clustering, Bregman divergences, exponential families, convex conjugates
PACS: 05. Statistical physics, thermodynamics, and nonlinear dynamical systems

INTRODUCTION

Consider a finite statistical mixture with $k \in \mathbf{N}$ components of density $m(x|\Lambda,W) = \sum_{i=1}^{k} w_i p(x|\lambda_i)$ with $W \in \Delta_k$ the positive weight vector belonging to the open k-dimensional probability simplex Δ_k and $\Lambda = \{\lambda_1,...,\lambda_k\}$ the respective k parameters of the mixture components. Mixtures are universal density estimators: For example, Gaussian mixtures are defined on the support $\mathscr{X} = \mathbf{R}^d$ and find countless applications in imaging (*e.g.*, Kernel Density Estimators based on isotropic Gaussian kernels, KDEs) while Gamma mixtures are useful for modeling distances on $\mathscr{X} = \mathbf{R}^+$. Mixtures are conceptually used to probabilistically model *sub-populations* within an overall population: To illustrate this point, consider for example modeling the height of a country population: it is reasonable to assume that its distribution follows a density that is a mixture of $k = 2$ sub-populations: a Gaussian component for modeling men heights and another Gaussian component for modeling woman heights.

To sample a variate $x \in \mathscr{X}$ from a mixture $m(x|\Lambda,W)$:

- First, choose a component l according to the weight distribution $w_1,...,w_k$ (multinomial), and then
- Draw a variate x according to $p(x|\lambda_l)$.

Conversely, the most common method to infer a mixture model from a set of Independently and Identically Distributed (IID.) set of observations $x_1, ..., x_n$ (without the labels l_i called hidden/missing/latent variables) is the Expectation-Maximization [1] (1977) algorithm. The EM algorithm monotonically maximizes the *likelihood function*:

$$l(x_1, ..., x_n) = \prod_{i=1}^{n} m(x_i|\Lambda, W).$$

EM can be trapped into a local maximum and further needs a stopping criterion or loop forever, otherwise. From a technical viewpoint, handling semi-parametric mixtures is different from regular parametric models since often the mixture density exhibits the problems of identifiability and Fisher information irregularity among others, see [2].

Recently, several approaches of Theoretical Computer Science (TCS) have been proposed [3, 4] to study the *learnability complexity* of mixtures: A mixture m is said ε-*close* to a mixture $\tilde{m}$ (both with k components) when:

- $\forall i \in \{1, ..., k\}$, $|w_i - \hat{w}_{\pi(i)}| \leq \varepsilon$,
- $\forall i \in \{1, ..., k\}$, $\mathrm{KL}(p(x|\lambda_i) : p(x|\hat{\lambda}_{\pi(i)})) \leq \varepsilon$,

where $\pi(\cdot)$ denotes a permutation and $\mathrm{KL}(m : m') = \int_{x \in \mathscr{X}} m(x) \log \frac{m(x)}{m'(x)} \mathrm{d}x$ is the Kullback-Leibler information divergence (commonly called relative entropy). It has been reported that for a ε-*learnable* Gaussian mixture m that satisfies the following conditions:

- $\min_{i=1}^{k} w_i \geq \varepsilon$,
- $\mathrm{KL}(p(x|\lambda_i) : p(x|\lambda_j)) \geq \varepsilon$, $\forall i \neq j$,

there exist *polynomial-time algorithms* [3, 4] in n and $\frac{1}{\varepsilon}$ that ε-closely estimates m. Furthermore, core-set techniques [5] have been designed for dealing with massive data sets when learning mixtures.

LEARNING MIXTURES BY SOLVING SEQUENCES OF GEOMETRIC CLUSTERING PROBLEMS

The EM algorithm monotonically maximizes the incomplete data likelihood (or equivalently incomplete log-likelihood l_i). This is usually intractable to solve exactly in closed-form because of the log-sum terms:

$$l_i(x_1, ..., x_n) = \sum_{i=1}^{n} \log \left(\sum_{j=1}^{k} w_j p(x_i|\theta_j) \right).$$

Consider the complete likelihood by introducing the indicator variables $z_{i,j}$ with $z_{i,j} = 1$ iff. $l_i = j$ (*i.e.*, observation x_i emanated from component l_j), and $z_{i,j} = 0$ otherwise:

$$l_c(x_1,...,x_n) = \log \prod_{i=1}^{n} \prod_{j=1}^{k} \left(w_j p(x_i|\theta_j)\right)^{z_{i,j}} = \sum_{i=1}^{n} \sum_{j=1}^{k} z_{i,j} \log(w_j p(x_i|\theta_j)).$$

The k-MLE methodology: Maximizing the complete likelihood

The complete log-likelihood optimization can be rewritten as follows:

$$\begin{aligned}
\max_{W,\Lambda} l_c(W,\Lambda) &= \max_{\Lambda} \sum_{i=1}^{n} \max_{j=1}^{k} \log(w_j p(x_i|\theta_j)), && (1)\\
&\equiv \min_{W,\Lambda} \sum_{i=1}^{n} \min_{j=1}^{k} (-\log p(x_i|\theta_j) - \log w_j), && (2)\\
&= \min_{W,\Lambda} \sum_{i=1}^{n} \min_{j=1}^{k} D_j(x_i), && (3)
\end{aligned}$$

where the $c_j = (w_j, \theta_j)$'s denote the *cluster prototypes* and the $D_j(x_i) = -\log p(x_i|\theta_j) - \log w_j$ are the *potential distance-like functions*. Thus maximizing the complete likelihood amounts to a *geometric hard clustering* [6, 7] for *fixed* w_j's: $\min_\Lambda \sum_i \min_j D_j(x_i)$. Note that the distances $D_j(\cdot)$'s depend on the cluster prototypes c_j's. This viewpoint is related to the classification EM [8] (CEM, or hard EM/truncated EM) that can be used to initialize an EM.

We describe the generic $\underline{k\text{-MLE}}$ approach:

1. Initialize weight W in the open probability simplex: $W \in \Delta_k$
2. Solve $\min_\Lambda \sum_i \min_j D_j(x_i)$ (center-based clustering, weights W fixed)
3. Solve $\min_W \sum_i \min_j D_j(x_i)$ (parameters Λ fixed)
4. Test for convergence and go to step 2) otherwise.

The k-MLE method can be interpreted as a group coordinate descent optimization strategy. Consider the uniform weight $W = (\frac{1}{k},...,\frac{1}{k})$ and isotropic Gaussian components. Then step 2 amounts to solve for a *k-means clustering* problem [9]. In general, k-means is NP-hard (non-convex optimization) when $d > 1$ and $k > 1$ and solved exactly using dynamic programming [10] in $O(n^2 k)$ when $d = 1$. Various heuristics have been proposed for k-means:

- Global: Kanungo et al. [11] swap method that yields a $(9+\varepsilon)$-approximation,
- Seeding techniques: random seed (Forgy [12]), k-means++ [13], global k-means initialization [14],
- Local refinements: Lloyd's batched update [9], MacQueen's iterative update [15], Hartigan single-point swap update [16], etc.

Similar to k-means, data are assigned to their closest cluster with respect to the potential functions $D_j(x_i) = -\log p(x_i|\theta_j) - \log w_j$. Let $\mathscr{C}_1,...,\mathscr{C}_k$ denote the cluster partition. Note that if we consider a $k = 2$ mixture, we cannot classify exactly the observations

from the corresponding sub-populations because we lack the missing labels: In classification, the minimum error is called Bayes' error [17] and can be upper bounded using Chernoff information [17]. For solving the geometric clustering problems for fixed weight vectors W, we can characterize the optimal *cluster assignment* using *generalized Voronoi diagrams*.

Furthest Maximum Likelihood Voronoi diagrams

The geometric clustering problem consists in finding the prototypes (cluster centers) c_j's that minimizes the objective function: $\min_\Lambda \sum_i \min_j D_j(x_i)$. It partitions the data into k clusters and fits the MLE inside each cluster. We assign data to clusters according to the Furthest Maximum Likelihood (FML) Voronoi diagram:

$$\begin{aligned} \mathrm{Vor}_{\mathrm{FML}}(c_i = (w_i, \theta_i)) &= \{x \in \mathscr{X} : w_i p(x|\lambda_i) \geq w_j p(x|\lambda_j),\ \forall i \neq j\}, \quad (4) \\ \mathrm{Vor}(c_i) &= \{x \in \mathscr{X} : D_i(x) \leq D_j(x),\ \forall i \neq j\}. \quad (5) \end{aligned}$$

This amounts to an *additively weighted Voronoi diagram* with anchored distance $D_l(\cdot)$ at each cluster $\mathscr{C}_l$: $D_l(x) = -\log p(x|\lambda_l) - \log w_l$.

Updating the mixture component weights

In step 3 of k-MLE, we have to solve the optimization problem: $\min_W \sum_i \min_j D_j(x_i)$. This amounts to solve for:

$$\arg\min_{W \in \Delta_k} -n_j \log w_j = \arg\min_{W \in \Delta_k} -\frac{n_j}{n} \log w_j,$$

where $n_j = \#\{x_i \in \mathrm{Vor}(c_j)\} = |\mathscr{C}_j|$ denotes the cardinality of cluster $\mathscr{C}_j$. Thus, we seek for:

$$\min_{W \in \Delta_k} H^\times(N : W),$$

where $N = (\frac{n_1}{n}, ..., \frac{n_k}{n})$ is the *cluster point proportion vector* $\in \Delta_k$. Since the *cross-entropy* $H^\times(N : W)$ is minimized when $H^\times(N : W) = H(N)$, we deduce that $W = N$. In other words, at step 3, we update the component weights W of the mixture by taking the proportion of points falling into the k clusters.

Case study: Mixtures of exponential families

An exponential family mixture has component densities that write *canonically* as $p_F(x|\theta) = \exp(t(x)^\top \theta - F(\theta) + k(x))$ with:

- $t(x)$: the sufficient statistic in $\mathbf{R}^D$ where D denotes the family order,
- $k(x)$: an auxiliary carrier term with respect to the Lebesgue or counting measures,

- $F(\theta)$: the log-normalizer also called cumulant function or log-partition function.

Exponential families have log-concave densities, meaning that the potential distance functions $D_j(x)$'s are convex. Thus the geometric clustering problems are k-means type clustering problems with respect to convex "distances". Using the duality between exponential families and Bregman divergences [18], we get the potential distance functions:

$$\begin{aligned} D_{w,\theta}(x) = -\log p(x;\theta) - \log w &= F(\theta) - t(x)^\top \theta - k(x) - \log w, & (6)\\ &= B_{F^*}(t(x):\eta) + F^*(t(x)) + k(x) - \log w, & (7) \end{aligned}$$

where $F^*(\eta) = \max_\theta(\theta^\top \eta - F(\theta))$ is the Legendre-Fenchel convex conjugate. Thus the ML farthest Voronoi diagram turns out to be equivalent to an *additively-weighted Bregman Voronoi diagram* [19] (affine diagrams).

The k-MLE method for mixtures of exponential families, k-MLEEF, is therefore rewritten as follows:

1. Initialize weight $W \in \Delta_k$
2. Solve additive Bregman k-means: $\min_\Lambda \sum_i \min_j D_j(x)$ with $D_j(x) = B_{F^*}(t(x) : \eta_j) - \log w_j$
3. Update weight vector W as cluster point proportion
4. Test for convergence and go to step 2) otherwise

Step 2 is solved using an *extended version of Bregman k-means* (convergence proofs for Lloyd's batched heuristic is reported in [20] and for Hartigan's single swap heuristic in [25]). Given a ML farthest Voronoi partition, we compute the MLEs $\hat{\theta}_j$'s inside each cluster as follows:

$$\hat{\theta}_j = \arg\max_{\theta\in\Theta} \prod_{x_i \in \mathrm{Vor}(c_j)} p_F(x_i;\theta).$$

The MLE is found by solving the moment equation:

$$\nabla F(\hat{\theta}_j) = \eta(\hat{\theta}_j) = \frac{1}{n_j} \sum_{x_i \in \mathrm{Vor}(c_j)} t(x_i) = \overline{t} = \hat{\eta}.$$

The MLE for exponential families is *consistent*, *efficient* with *asymptotic normal distribution*:

$$\hat{\theta}_j \sim \mathrm{Nor}\left(\theta_j, \frac{1}{n_j} I^{-1}(\theta_j)\right),$$

where the Fisher information matrix is:

$$I(\theta_j) = \mathrm{var}[t(X)] = \nabla^2 F(\theta_j) = (\nabla^2 F^*(\eta_j))^{-1}.$$

The MLE may be biased (*e.g.*, normal distributions) and is guaranteed to exist and be unique [21, 22] when:

$$T(x_1,...,x_n) = \begin{bmatrix} 1 & t_1(x_1) & ... & t_D(x_1) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_1(x_n) & ... & t_D(x_n) \end{bmatrix} \quad (8)$$

of dimension $n \times (D+1)$ has *rank* $D+1$ [21]. For example, there are problems for undefined MLEs of multivariate normals (MVNs) with $n < d$ observations (unbounded likelihood is ∞). The maximal likelihood is $l(x_1,...,x_n) = F^*(\hat{\eta}) + \sum_{i=1}^{n} k(x_i)$, where $\hat{\eta} = \nabla F(\hat{\theta})$.

The generalized k-MLE method

Weibull distributions or generalized Gaussians are *parametric families of exponential families* [23]: They are *not* exponential families when considering all free parameters but can be interpreted as parametric families $F(\gamma)$ of exponential families when considering some fixed parameters γ. Reducing the number of free parameters of high-order exponential families is also useful to obtain one free parameter with convex conjugate F^* approximated *efficiently* by line search (*e.g.*, Gamma distributions [24] or generalized Gaussians [23]). (Indeed, fixing some of their parameters yields *nested families* of exponential families [24].) To extend k-MLE to those kind of distributions, we further attach to each cluster prototype c_j the family F_j of distributions (*i.e.*, $c_j = (w_j, \theta_j, F_j)$) and we set $D_{w_j,\theta_j,F_j}(x) = -\log p_{F_j}(x;\theta_j) - \log w_j$. The standard k-MLE considers all families identical: $F_j = F$. We describe the $\underline{k\text{-GMLE}}$ methodology:

1. Initialize weight $W \in \Delta_k$ and family type $(F_1,...,F_k)$ for each cluster
2. Solve $\min_\Lambda \sum_i \min_j D_j(x_i)$ (center-based clustering for W fixed) with potential functions: $D_j(x_i) = -\log p_{F_j}(x_i|\theta_j) - \log w_j$
3. Solve family types maximizing the MLE in each cluster $\mathscr{C}_j$ by choosing the parametric family of distributions $F_j = F(\gamma_j)$ that yields the best likelihood: $\min_{F_1=F(\gamma_1),...,F_k=F(\gamma_k)\in F(\gamma)} \sum_i \min_j D_{w_j,\theta_j,F_j}(x_i)$.
4. Update W as the cluster point proportion
5. Test for convergence and go to step 2) otherwise.

Theorem 1 *The k-GMLE algorithm learns a mixture from a set of n IID. observations by solving a sequence of geometric hard clustering problems: The k-GMLE algorithm guarantees the monotonous convergence of the complete likelihood into a (possibly local) optimum.*

In [25], we build upon recent results on k-means to propose a k-MLE algorithm that learns automatically the number k of mixture components, and present several probabilistically guaranteed initializations for k-MLE (Step 1). The k-MLE algorithm is fast and uses only linear memory: This contrasts with EM that requires to store $O(nk)$ soft weights, the soft membership weights $z_{i,j} \in (0,1)$. Furthermore, cluster assignment in k-MLE can be accelerated over the naïve brute force search by using tree search structures like the vantage point trees [26] or the ball trees [27].

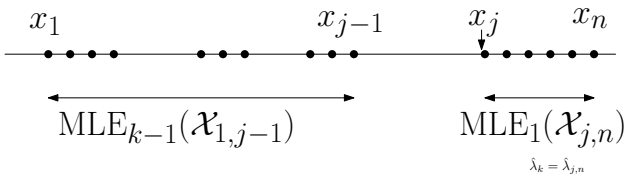

FIGURE 1. Learning a mixture of singly-parametric distributions using dynamic programming.

k-MLE for learning univariate mixtures of singly-parametric distributions

Cauchy, Rayleigh or Poisson families of distributions are univariate indexed by a single parameter. For exponential families (say, Rayleigh or Poisson, but not Cauchy), the geometric clustering problem amounts to a dual 1D weighted Bregman clustering [18] on 1D scalars $y_i = t(x_i)$ (where t denotes the sufficient statistic). The farthest ML Voronoi diagram has *connected cells*, meaning that an optimal clustering has necessarily the structure of *non-overlapping intervals*. In 1D, k-means (with additive weights) can be solved exactly using *dynamic programming* in $O(n^2k)$ time [10].

Consider the mixture weight vector W given, the k-MLE cost is: $\sum_{j=1}^{k} l_c(\mathscr{C}_j)$ where $\mathscr{C}_j$ are point clusters. The optimality equation of dynamic programming is illustrated in Figure 1:

$$\mathrm{MLE}_k(x_1,...,x_n) = \max_{j=2}^{n} \left(\mathrm{MLE}_{k-1}(\mathscr{X}_{1,j-1}) + \mathrm{MLE}_1(\mathscr{X}_{j,n})\right),$$

where $\mathscr{X}_{l,r} = \{x_l, x_{l+1}, ..., x_{r-1}, x_r\}$.

We build the dynamic programming table from $l = 1$ to $l = k$ columns, and from the $m = 1$ to $m = n$ rows. We then retrieve the clusters $\mathscr{C}_j$'s from the table by backtracking on the $\arg\max_j$. See [10] for implementation details of 1D k-MLE.

Theorem 2 *Learning a finite mixture of singly-parametric distributions with prescribed component weights can be done optimally with respect to the complete likelihood using dynamic programming provided that the Maximum Likelihood Voronoi diagram of distributions has connected cells.*

CONCLUSION AND DISCUSSION

We described a generic methodology, dubbed k-MLE (and its extension k-GMLE), to learn finite statistical mixtures by solving iteratively sequences of geometric hard clustering problems [7]. k-MLE optimizes the complete likelihood while Expectation-Maximization locally optimizes the incomplete likelihood. In particular, for exponential families, k-MLE geometric problems are solved by dual additively-weighted Bregman hard clustering problems. It is therefore different from the soft Bregman clustering proposed in [18] that was shown to be the EM algorithm in disguise. We showed how to extend the basic k-MLE method to handle independently for each cluster the family of distributions that can be used for the mixture component. For singly-parametric

family, we presented a simple dynamic programming method for solving the sequence of geometric interval clustering problems. Experimental results are reported in [23, 24, 25, 10]. We end up with the following open problem: Find the best $(1+\varepsilon)$-approximation algorithm for learning mixtures maximizing the complete or incomplete likelihood.

REFERENCES

1. A. P. Dempster, N. M. Laird, and D. B. Rubin, *Journal of the Royal Statistical Society (Series B)* **39**, 1–38 (1977).
2. J. Chen, *The Annals of Statistics* pp. 221–233 (1995).
3. A. Moitra, and G. Valiant, "Settling the Polynomial Learnability of Mixtures of Gaussians," in *51st IEEE Annual Symposium on Foundations of Computer Science*, 2010, pp. 93–102.
4. M. Hardt, and E. Price, "Sharp bounds for learning a mixture of two Gaussians", in *CoRR* **abs/1404.4997** (2014).
5. D. Feldman, M. Faulkner, and A. Krause, "Scalable Training of Mixture Models via Coresets," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 2142–2150.
6. M. Teboulle, *Journal of Machine Learning Research* **8**, 65–102 (2007).
7. D. Feldman, M. Monemizadeh, and C. Sohler, "A PTAS for *k*-means clustering based on weak coresets," in *Proceedings of the Symposium on Computational geometry*, 2007, pp. 11–18.
8. G. Celeux, and G. Govaert, *Comput. Stat. Data Anal.* **14**, 315–332 (1992), ISSN 0167-9473.
9. S. P. Lloyd, Least squares quantization in PCM, Tech. rep., Bell Laboratories (1957).
10. F. Nielsen, and R. Nock, *IEEE Signal Processing Letters* **21**, 1289–1292 (2014), ISSN 1070-9908.
11. T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, *Computational Geometry: Theory & Applications* **28**, 89–112 (2004).
12. E. W. Forgy, *Biometrics* (1965).
13. A. Bhattacharya, R. Jaiswal, and N. Ailon, "A Tight Lower Bound Instance for *k*-means++ in Constant Dimension," in *Theory and Applications of Models of Computation*, 2014, *LNCS* 8402, pp. 7–22, ISBN 978-3-319-06088-0.
14. J. Xie, S. Jiang, W. Xie, and X. Gao, *Journal of computers* **6** (2011).
15. J. B. MacQueen, "Some methods of classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
16. J. A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, New York, NY, USA, 1975, ISBN 047135645X.
17. F. Nielsen, *Pattern Recognition Letters* **42**, 25 – 34 (2014), ISSN 0167-8655.
18. A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, *Journal of Machine Learning Research* **6**, 1705–1749 (2005), ISSN 1532-4435.
19. J.-D. Boissonnat, F. Nielsen, and R. Nock, *Discrete Comput. Geom.* **44**, 281–307 (2010), ISSN 0179-5376.
20. F. Nielsen, *CoRR* **abs/1203.5181** (2012).
21. K. Bogdan, and M. Bogdan, *Statistics* **34**, 137–149 (2000), ISSN 0233-1888.
22. W. Miao, and M. Hahn, *Scandinavian Journal of Statistics* **24**, 371–386 (1997), ISSN 1467-9469.
23. O. Schwander, A. J. Schutz, F. Nielsen, and Y. Berthoumieu, "*k*-MLE for mixtures of generalized Gaussians," in *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2012, pp. 2825–2828.
24. O. Schwander, and F. Nielsen, "Fast Learning of Gamma Mixture Models with *k*-MLE," in *Similarity-Based Pattern Recognition (SIMBAD)*, 2013, pp. 235–249.
25. C. Saint-Jean, and F. Nielsen, "Hartigan's Method for *k*-MLE: Mixture Modeling with Wishart Distributions and Its Application to Motion Retrieval," in *Geometric Theory of Information*, Springer, 2014, pp. 301–330.
26. F. Nielsen, P. Piro, and M. Barlaud, "Bregman vantage point trees for efficient nearest Neighbor Queries," in *Proceedings of International Conference on Multimedia and Expo (ICME)*, 2009, pp. 878–881.
27. P. Piro, F. Nielsen, and M. Barlaud, "Tailored Bregman ball trees for effective nearest neighbors," in *European Workshop on Computational Geometry (EuroCG)*, 2009.