Optimal interval clustering: Application to Bregman clustering and statistical mixture learning

Frank Nielsen, *Senior Member, IEEE*
Sony Computer Science Laboratories, Inc.
3-14-13 Higashi Gotanda
141-0022 Shinagawa-ku, Tokyo, Japan
Frank.Nielsen@acm.org
and Richard Nock *Non member*,
NICTA
Sydney, Australia
Richard.Nock@nicta.com.au

***Abstract*—We present a generic dynamic programming method to compute the optimal clustering of n scalar elements into k pairwise disjoint intervals. This case includes 1D Euclidean k-means, k-medoids, k-medians, k-centers, etc. We extend the method to incorporate cluster size constraints and show how to choose the appropriate k by model selection. Finally, we illustrate and refine the method on two case studies: Bregman clustering and statistical mixture learning maximizing the complete likelihood.**

***Index Terms*—Clustering, dynamic programming, k-means, Bregman divergences, statistical mixtures, exponential families.**

I. Introduction

Clustering is a fundamental and key primitive to discover structural groups of homogeneous data, called *clusters*, in data sets. The most famous clustering technique is the celebrated *k-means* [1] that seeks to minimize the sum of intra-cluster variances by prescribing beforehand the number of clusters, k. On one hand, solving the k-means problem is *NP-hard* [2] when the dimension $d > 1$ and $k > 1$ and various heuristics *locally* optimizing the k-means objective function like Lloyd's batched k-means [1] have been proposed. When $d > 1$ and $k > 1$, NP-hardness also holds for other clustering problems like k-medoids, k-medians and k-centers [3]. On the other hand, it is well-known that those center-based clustering problems are fully characterized when $k = 1$: For example, the *centroid* [1] is the solution of the 1-mean, the Fermat-Weber point [3] the solution of the geometric 1-median, the circumcenter [3] the solution of the 1-center, etc. Surprisingly, it is less known that k-means can be solved *exactly* in 1D by using *dynamic programming* [4], [5] (DP).

In this letter, we first revisit and extend the seminal dynamic programming (DP) paradigm [4] for optimally clustering n 1D elements into k pairwise disjoint intervals, the clusters. We term clustering with this property: The *1D contiguous* or *interval clustering problem*. We further show how to incorporate constraints on the minimum and the maximum cluster sizes, and perform model selection (i.e., choosing the appropriate k) from the DP table. The generic DP solver requires either $O(n^2kT_1(n))$ time using $O(nk)$ memory or $O(n^2T_1(n))$ time using $O(n^2)$ memory, where $T_1(n)$ is the time requires for solving the corresponding 1-cluster problem. Second, we consider two applications that refine the generic DP method: In the first application, we report a $O(n^2k)$-time optimal Bregman k-means relying on 1D Summed Area Tables [6] (SATs) and also consider the Bregman ℓ_r-clustering problems [8]. In the second application, we consider learning statistical mixture models from independently and identically (iid.) univariate observations by maximizing the complete likelihood: Using the one-to-one mapping between Bregman divergences and exponential families [1], we transform this problem into a series of equivalent 1D Bregman k-means clustering that can be solved optimally by DP for statistical mixtures of *singly-parametric exponential families* (like zero-centered Gaussians, Rayleigh or Poisson families). In the general case, we require that the density graphs intersect pairwise in at most a single point like the Cauchy or Laplacian location families (not belonging to the exponential families) to guarantee optimality.

This work was done while R. Nock was with Université des Antilles-Guyane - CEREGMIA, Campus de Schoelcher, 97233 Schoelcher, France.

II. 1D contiguous clustering: Interval clustering

Let $\mathbb{X}$ be a one-dimensional space totally ordered with respect to $<$ (usually, $\mathbb{X} = \mathbb{R}$), and $\mathcal{X} = \{x_1, ..., x_n\} \subset \mathbb{X}$ a set of n distinct elements. A clustering of $\mathcal{X}$ into $k \in \mathbb{N}$ clusters partitions $\mathcal{X}$ into pairwise disjoint subsets $\mathcal{C}_1 \subset \mathcal{X}, ..., \mathcal{C}_k \subset \mathcal{X}$ so that $\mathcal{X} = \biguplus_{i=1}^k \mathcal{C}_i$. Let us preliminary sort $\mathcal{X}$ in $O(n \log n)$ time, so that we assume $x_1 < ... < x_n$ in the remainder.

The output of a 1D contiguous clustering is a collection of k intervals $I_i = [x_{l_i}, x_{r_i}]$ (such that $\mathcal{C}_i = I_i \cap \mathcal{X}$) that can be encoded using $k-1$ *delimiters* l_i ($i \in \{2, ..., k\}$) since $r_i = l_{i+1} - 1$ ($i < k$ and $r_k = n$) and $l_1 = 1$:

$$\underbrace{[x_1 ... x_{l_2-1}]}_{\mathcal{C}_1} \underbrace{[x_{l_2} ... x_{l_3-1}]}_{\mathcal{C}_2} ... \underbrace{[x_{l_k} ... x_n]}_{\mathcal{C}_k} \quad (1)$$

To define an optimal clustering among the potential $\binom{n-1}{k-1}$ contiguous partitions, we ask to minimize a clustering *objective function* or *energy function*:

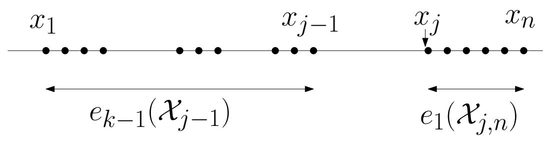

Fig. 1. The optimal 1D contiguous clustering is found by dynamic programming by observing that an optimal clustering with k clusters is necessarily found from an optimal clustering with $(k-1)$ clusters (see text and Eq. 3).

$$\min_{l_1=1<l_2<\ldots<l_k} e_k(\mathcal{X}) = \bigoplus_{i=1}^{k} e_1(\mathcal{C}_i), \qquad (2)$$

where e_1 denotes the *intra-cluster cost* and $\oplus$ is a commutative and associative operator for calculating the *inter-cluster cost*. This framework includes the k-means and the k-medians ($\bigoplus = \sum$), and the k-center [3] ($\bigoplus = \max$) criteria (and their discrete counterparts: k-medoids, etc.) among others.

A. Solving 1D contiguous clustering using DP

Recall that after sorting, we have $x_1 < \ldots < x_n$. Let $\mathcal{X}_{j,i} = \{x_j, \ldots, x_i\}$ ($j \leq i$) and $\mathcal{X}_i = \mathcal{X}_{1,i} = \{x_1, \ldots, x_i\}$. We define a $n \times k$ cost matrix $E = [e_{i,j}]$ that stores at entry (i,m) the optimal clustering cost $e_{i,m} = e_m(\mathcal{X}_i)$, where e_m is defined using Eq. 2. Similarly, we define a matrix $S = [s_{i,j}]$ of dimension $n \times k$ that stores at position (i,m) the index j of the leftmost point in the m-th cluster in $\mathcal{X}_i$. Therefore the global clustering solution shall be found at entry (n,k) with cost $e_{n,k} = e_k(\mathcal{X})$.

To define the *optimality equation* of dynamic programming, we observe that the optimal solution for a 1D contiguous clustering with m clusters can be defined from the solution of an optimal clustering with $(m-1)$ clusters: Indeed, consider the last cluster interval with left position index l_m, say $l_m = j$, as depicted in Figure 1. Then the clustering of the $(m-1)$ first clusters should be an optimal clustering too: namely, the optimal 1D contiguous clustering with $(m-1)$ clusters on subset $\mathcal{X}_{j-1}$. It follows the following recurrence equation:

$$e_{i,m} = \min_{m \leq j \leq i} \left\{ e_{j-1,m-1} \oplus e_1(\mathcal{X}_{j,i}) \right\}, \qquad (3)$$

with $e_{i,1} = e_1(\mathcal{X}_i)$ (note that $e_{m,m} = \bigoplus_{l=1}^{m} e_1(\{x_l\})$ for $1 \leq m \leq k$). We store the argmin of Eq. 3 in matrix S at position (i,m) (entry $s_{i,m}$). We compute the energy matrix E from left to right columns, and from bottom to top lines. This yields a $O(n^2 k T_1(n))$-time DP algorithm using $O(n \times k)$ memory, where $T_1(n)$ denotes the time required for computing $e_1(\mathcal{X})$: Indeed, each of the $n \times k$ entries of E requires $O(nT_1(n))$ time to evaluate Eq. 3.

To recover the optimal clustering, we *backtrack* the solution in $O(k)$ time from the S matrix storing the left indexes of the last cluster of the best solutions: That is, the left index l_k of the k-th cluster is stored at $s_{n,k}$: $l_k = s_{n,k}$. The cardinality of $\mathcal{C}_k$ is $n_k = |\mathcal{C}_k| = n - l_k + 1$. Then we iteratively retrieve the previous left interval indexes at entries $l_{j-1} = s_{l_j - 1, j-1}$ for $j = k-1, \ldots, j = 1$ with $n_j = |\mathcal{C}_j| = r_j - l_j + 1 = l_{j+1} - l_j$ since $r_j = l_{j+1} - 1$. Note that $l_j - 1 = n - \sum_{l=j}^{k} n_l$ denotes the remaining number of elements to cluster using $(j-1)$ clusters (thus we also have $l_j - 1 = \sum_{l=1}^{j-1} n_l$).

Note that when the clustering does not satisfy the 1D contiguous partition property, DP yields *anyway* a solution that may not be optimal. Furthermore, we may consider adding a weight $w_i > 0$ to each element $x_i \in \mathcal{X}$ (and thus assume the x_i's are all distinct).

B. Time versus memory optimization

By precomputing all the potential intra-cluster costs $e_1(\mathcal{X}_{j,i})$ in $O(n^2 T_1(n))$ time using an auxiliary matrix E_1 of size $n \times n$, we evaluate Eq. 3 as $e_{i,m} = \min_{m \leq j \leq i} \{e_{j-1,m-1} \oplus E_1[j,i]\}$, i.e. in $O(i-m) = O(n)$ time. Matrix E_1 plays the role of a *Look Up Table* (LUT), and the time complexity for the DP solver reduces to $O(n^2 k)$ once the LUT matrix E_1 has been computed.

Lemma 1: The generic 1D contiguous clustering can be solved optimally using dynamic programming in time $O(n^2 k T_1(n))$ using $O(n \times k)$ memory, or in time $O(n^2 T_1(n))$ time using $O(n^2)$ memory.
Note that $T_1 = \Omega(n)$ (in fact, usually, $T_1(n) = \Theta(n)$). In Section III, we will further improve the running time to $O(n^2 k)$ using $O(nk)$ memory when considering Bregman k-means.

C. Adding cluster size constraints

Let us add constraints on the sizes of clusters. Let n_i^- and n_i^+ denote lower and upper bound constraints on the size of the i-th cluster $n_i = |\mathcal{C}_i|$, with $\sum_{l=1}^{k} = n_i^- \leq n$ and $\sum_{l=1}^{k} = n_i^+ \geq n$. When no constraints are required, we simply add the *dummy* constraints $n_i^- = 1$ and $n_i^+ = n - k + 1$ (all clusters non-empty). In Eq. 3, j range from m to i. The m-th cluster size $n_m = |\mathcal{C}_m| = i - j + 1$ has to satisfy $n_m^- \leq n_m \leq n_m^+$. That is, $j \leq i + 1 - n_m^-$ and $j \geq i + 1 - n_m^+$. Clearly, j has also to be greater than $1 + \sum_{l=1}^{m-1} n_l^-$ (an optimal solution for the constrained optimal $(m-1)$-clustering). It follows, that the optimality equation writes as:

$$e_{i,m} = \min_{\substack{\max\{1+\sum_{l=1}^{m-1} n_l^-, i+1-n_m^+\} \leq j \\ j \leq i+1-n_m^-}} \left\{ e_{j-1,m-1} \oplus e_1(\mathcal{X}_{j,i}) \right\}, \qquad (4)$$

For example, a balanced clustering may be obtained by setting $n_i^- = \lfloor \frac{n}{\lambda k} \rfloor$ and $n_i^+ = \lceil \frac{\lambda n}{k} \rceil$ for some $\lambda \in \mathbb{N}$.

D. Choosing the appropriate k: Model selection

The task of clustering data set $\mathcal{X}$ asks also to find the appropriate number of clusters [7]: k. Clearly, the more clusters we allow and the less costly the objective function $e_k(\mathcal{X})$ is, but the more complex the clustering model to encode. Observe that function $m(k) = \frac{e_k(\mathcal{X})}{e_1(\mathcal{X})}$ is *monotonically decreasing* with k and reaches a minimum when $k = n$ (e.g., 0 for the Euclidean k-means) as depicted in Figure 2 (see IV for an explanation of the data-set). Thus we have to perform some kind of *model selection* [7] by choosing the *best model* among all potential models (with number of clusters

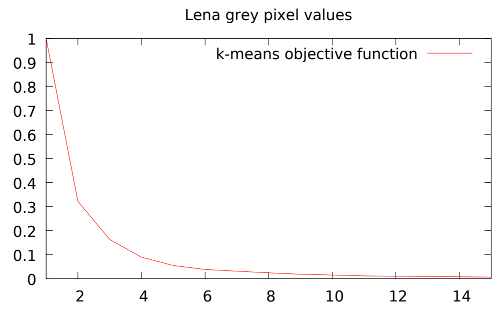

Fig. 2. Plot of function $m(k) = e_k(\mathcal{X})/e_1(\mathcal{X})$ for the optimal k-means for $k \in [1, 15]$.

ranging from 1 to n). The canonical *regularized objective clustering cost* [7] is $e'_k(\mathcal{X}) = e_k(\mathcal{X}) + f(k)$ where $f(k)$ is the cost function of choosing a model with k clusters. We can compute the best model minimizing $e'_k(\mathcal{X})$ by computing for the DP table entries for the *last matrix row* of E (indexed by n, with columns k ranging from 1 to n) the regularized cost. To compute the last row, we iteratively solve DP for $k = n, n-1, ..., 1$ and avoid redundant computations by checking whether entry $E[i,j]$ has already been computed or not. We then choose $k = \mathrm{argmin}_{g\in\{1,...,k\}} e'_g(\mathcal{X})$ by scanning the last row with column ranging from $k = 1$ to $k = n$.

E. A Voronoi condition for optimal center-based clustering

Center-based clustering methods like k-means, k-medians or k-centers store for each cluster $\mathcal{C}_j$ a *prototype* p_j, the cluster center. For discrete center-based clustering, the prototypes p_j's are required to belong to the respective $\mathcal{C}_j$'s. The ℓ_r *center-based clustering objective function* asks to minimize:

$$\sum_{i=1}^{n} w_i \min_{j=1}^{k} d^r(x_i, p_j) = \sum_{j=1}^{k} \sum_{x_l \in \mathcal{C}_j} w_l d^r(x_l, p_j), \qquad (5)$$

where $d(\cdot,\cdot)$ is a *dissimilarity measure function* (*not* necessarily a distance). We do not take the $\frac{1}{r}$ power of the sum since it changes the value of e_1 but not the argmin (prototype). Note that in 1D, ℓ_s-norm distance is always $d_s(p,q) = |p-q|$, independent of $s \geq 1$. Thus the intra-cluster cost $e_1(\mathcal{C}_j)$ of a ℓ_r center-based clustering has to solve the following minimization problem: $e_1(\mathcal{C}_j) = \min_{p_j} \sum_{x_l\in\mathcal{C}_j} w_l d^r(x_l, p_j)$ and retrieve the j-th cluster prototype by $p_j = \mathrm{argmin}_{p_j} \sum_{x_l\in\mathcal{C}_j} w_l d^r(x_l, p_j)$.

In order for DP to return the optimal clustering, we need to assume that we have the 1D contiguous clustering property. For Euclidean k-means, this was proved in [9]. In general, consider the *Voronoi cell* of prototype p_j of $\mathcal{C}_j$:

$$V(p_j) = \{x \in \mathbb{X} \ : \ d^r(x, p_j) \leq d^r(x, p_l)\ \forall l \in \{1, ..., k\}\}. \qquad (6)$$

Since x^r is a monotonically increasing function on $\mathbb{R}^+$, it is equivalent to $V'(p_j) = \{x \in \mathbb{X} \ : \ d(x : p_j) < d(x : p_l)\}$. A sufficient condition is to prove that for *all* potential choices of the k cluster prototypes $\mathcal{P} = \{p_1, ..., p_k\}$ the induced 1D dissimilarity Voronoi diagram is made of *connected Voronoi cells*. A 2-clustering displays the Voronoi bisector. We now consider two case studies to illustrate and refine the DP method.

III. Optimal 1D Bregman clustering

The ℓ_r-norm Bregman center [8] is defined for $d(p,q) = B_F(p : q)$, where $B_F(p : q)$ is a univariate Bregman divergence [1]:

$$B_F(p : q) = F(p) - F(q) + (p-q)F'(q), \qquad (7)$$

induced by a strictly convex and differentiable function F. When $F(x) = x^2$, we recover the squared Euclidean distance. Bregman divergences are *not* metric [10], since they violate the triangular inequality and are *asymmetric* except when $F(x) = \lambda x^2$ for $\lambda > 0$.

For Bregman k-means, the *Bregman information* [1] of a cluster generalizes the notion of cluster variance. It is the *intra-cluster sum of Bregman divergences* (Bregman k-means, for $r = 1$):

$$e_1(\mathcal{C}_j) = \min_{p_j} \sum_{x_l \in \mathcal{C}_j} w_l B_F(x_l : p_j). \qquad (8)$$

The cluster prototype [1] is $p_j = \frac{1}{\sum_{x_l\in\mathcal{C}_j} w_l} \sum_{x_l\in\mathcal{C}_j} w_l x_l$ and the Bregman information is [11]: $e_1(\mathcal{C}_j) = \left(\sum_{x_l\in\mathcal{C}_j} w_l\right)(p_j F'(p_j) - F(p_j)) + \left(\sum_{x_l\in\mathcal{C}_j} w_l F(x_l)\right) - F'(p_j)\left(\sum_{x\in\mathcal{C}_j} w_l x\right)$. Observe that the Bregman information relies on three sums $\sum_{x_l\in\mathcal{C}_j} w_l$, $\sum_{x\in\mathcal{C}_j} w_l x$ and $\sum_{x_l\in\mathcal{C}_j} w_l F(x_l)$ that can be preprocessed using *Summed Area Tables* [6] (SATs) since $\mathcal{C}_j$ is a contiguous cluster. That is, by computing all the *cumulative sums* $S_1(j) = \sum_{l=1}^{j} w_l$, $S_2(j) = \sum_{l=1}^{j} w_l x_l$, and $S_3(j) = \sum_{l=1}^{j} w_l F(x_l)$ in $O(n)$ time at preprocessing stage, we can evaluate the Bregman information $e_1(\mathcal{X}_{j,i})$ in constant time $O(1)$. For example, $\sum_{l=j}^{i} w_l F(x_l) = S_3(i) - S_3(j-1)$ with the convention that $S_3(0) = 0$.

The Voronoi cells of prototypes are defined by $V'(p_j) = \{x \in \mathbb{X} \ : \ B_F(x : p_j) < B_F(x : p_l)\}$. Since *Bregman Voronoi diagrams* have connected cells [10], it follows that the 1D hard ℓ_r Bregman clustering satisfies the contiguous interval property, and therefore DP yields the optimal solution. A similar argument directly hold for the Bregman k-center that is also the limit case of ℓ_r Bregman clustering when $p \to \infty$.

Lemma 2: The 1D ℓ_r Bregman clustering and Bregman k-center can be solved exactly using dynamic programming in $O(n^2 k T_1(n))$ time using $O(n \times k)$ memory, where $T_1(n)$ denotes the time to solve the case $k = 1$ for n elements. The optimal Bregman k-means can be solved in $O(n^2 k)$ time.

IV. Mixture learning by hard clustering

Statistical mixtures are semi-parametric probability models often met in practice. Consider a finite *statistical mixture* M with $k \in \mathbb{N}$ components. The probability measure m of M with respect to a dominating measure ν (usually the Lebesgue or counting measure) can be written as:

$$m(x; \Omega) = \sum_{i=1}^{k} \alpha_i p(x; \Theta_i), x \in \mathbb{X}, \qquad (9)$$

with $\alpha = (\alpha_1, ..., \alpha_k) \in \Delta_{k-1}$ a normalized positive weight vector belonging to the $(k-1)$-dimensional *probability simplex*, $\Theta = (\Theta_1, ..., \Theta_k)$, $\Omega = (\alpha, \Theta)$ and $\mathbb{X}$ the support of the distribution. Let $D = \dim(\Theta_i) \in \mathbb{N}$ denote the number of scalar parameters indexing the probability family $\mathcal{F} = \{p(x; \Theta) : \Theta \in \mathbf{\Theta}\}$, called the *order*. Mixture m is defined by a vector $\Omega \in \mathbf{\Omega} \subseteq \mathbb{R}^g$ with $g = k(D+1)-1$, and $\mathbf{\Theta}$ is called the *parameter space*. Mixtures are inferred from data usually using the Expectation-Maximization algorithm [1]. Since EM locally maximizes the *incomplete likelihood* [1] and is often trapped into a local maximum, we need some proper mixture parameter initialization or several guided restarts to hopefully reach the optimal solution. On the other hand, maximizing the *complete log-likelihood* l_c for a iid. observation data-set $\mathcal{X}$ amounts to maximize [12]:

$$l_c(\mathcal{X}; L, \Omega) = \sum_{i=1}^{n} \log(\alpha_{l_i} p(x_i; \theta_{l_i})), \tag{10}$$

where $L = \{l_i\}_i$ denotes the hidden labels of the x_i's. Thus maximizing the complete likelihood is equivalent to minimizing the following objective function:

$$\max l_c \equiv \min_{\theta_1,...,\theta_k} \sum_{i=1}^{n} \min_{j=1}^{k} (-\log p(x_i; \theta_j) - \log \alpha_j). \tag{11}$$

This is a hard clustering problem for the dissimilarity function $d(x, (\alpha, \theta)) = -\log p(x; \theta) - \log \alpha$ (given fixed α). As proved in [12], the cluster weights α_j's are then updated as the cluster proportion of observations, and the algorithm reiterates by solving Eq. 11. Initially, we choose $\alpha = \frac{1}{k}(1, ..., 1)$.

Let the *additively-weighted minus log-likelihood Voronoi cell* be defined by $V(p_j) = \{x \in \mathbb{X} : -\log p(x; \theta_j) - \log \alpha_j \leq -\log p(x; \theta_l) - \log \alpha_l\}$. In order for DP to return the optimal solution, we need to assert the contiguity property. Using the one-to-one mapping between exponential families [13], [14] and Bregman divergences [1], it turns out that the optimization problem of Eq. 11 yields an equivalent additively-weighted Bregman k-means problem (and additively-weighted Bregman Voronoi cells are connected [10]). Thus when the order of the exponential family is $D = 1$, we have the contiguity property and DP returns the optimal solution. This works also for curved exponential families with one free parameter like the family of Gaussian distributions $\mathcal{F} = \{N(\mu, \mu^2) : \mu \in \mathbb{R}\}$. In general, the contiguity property holds when density graphs in $\mathcal{F}$ are pairwise intersecting at exactly one point of the support $\mathbb{X}$. For example, some (unimodal) *location families* with density $\mathcal{F} = \{f(x; \mu) = \frac{1}{\sigma} f_0(\frac{x-\mu}{\sigma}), \mu \in \mathbb{R}\}$ for a prescribed value of $\sigma > 0$ and a standard density $f_0(x)$ (e.g., isotropic gaussian densities $N(\mu_1, \sigma)$ and $N(\mu_2, \sigma)$ intersect at $x = \frac{\mu_1 + \mu_2}{2}$). This includes location Cauchy distributions and location Laplacian distributions (both not belonging to the exponential families [13]) among others. Note that 1-order exponential families may have pairwise densities intersecting in more than one point (like the family $\mathcal{F} = \{N(0, \sigma), \sigma \in \mathbb{R}^+\}$) but after reparameterization by their sufficient statistic [13] $y_i = t(x_i)$, data-set $\mathcal{Y} = \{y_i\}_i$ satisfies the contiguous property.

Consider fitting a Gaussian Mixture Model (GMM) on the intensity histogram of the renown `lena` color image. For each

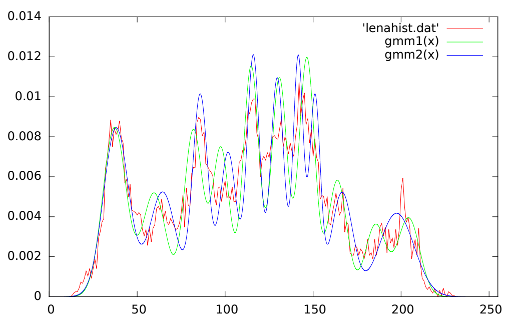

Fig. 3. 1D GMMs with $k = 10$ components maximizing the complete data likelihood of the intensity histogram of `lena` image: gmm_1 retrieved from an optimal Euclidean k-means, and gmm_2 allowing different standard deviations. The average complete data log-likelihood of gmm_1 is -3.075 and that of gmm_2 is -3.039 (better than the one for gmm_1).

pixel, we compute its grey value and add a small perturbation noise to ensure that we get distinct x_i's (alternatively, without adding noise, we set the weight w_i of x_i as the proportion of pixels having grey value x_i). We then compute the optimal Euclidean 1D k-means for $k = 10$ (it corresponds to fitting a 1D GMM gmm_1 with Gaussian components having identical[1] standard deviation), and calculate the 1D GMM gmm_2 allowing different standard deviations. In that case, we do *not* have the contiguous clustering property (densities pairwise intersect in two points) and DP may *not* yield the optimal clustering (give prescribed weights). However, in this case, we experimentally obtained a better GMM. The results are illustrated in Figure 3. For model selection in mixtures, to choose the optimal k, we use the *Akaike Information Criterion* [15] (AIC): $\mathrm{AIC}(x_1, ..., x_n) = -2l(x_1, ..., x_n) + 2k + \frac{2k(k+1)}{n-k-1}$. Other criteria like the Bayesian Information Criterion (BIC), Minimum Description Length (MDL), etc can also be used.

V. Conclusion

We first described a clustering algorithm based on dynamic programming (whose seminal idea was briefly outlined in Bellman's 2-page paper [4] in 1973) that computes the generic optimal 1D contiguous clustering either in $O(n^2 k T_1(n))$-time using $O(nk)$ memory, or in $O(n^2 T_1(n))$ time using $O(n^2)$ memory, where $T_1(n)$ denotes the time required for solving the case $k = 1$ on n scalar elements. We then extended the method to incorporate cluster size constraints and show how to perform model selection from the DP table. This algorithm solves optimally and generically 1D k-means, k-median and k-center among others. Second, we reported two tailored center-based clustering applications of the optimal 1D contiguous clustering: (1) Bregman k-means and k-centers clustering, and (2) learning statistical mixtures maximizing the complete likelihood provided that (a) their densities belong to a 1-order exponential family or (b) their density graphs pairwise intersect in one point. For Bregman k-means, we showed how to use Summed Area Tables (SATs) to further speed the DP solver in $O(n^2 k)$-time using $O(nk)$ memory.

[1]Once we get the optimal Euclidean cluster decomposition, we fit in each cluster its maximum likelihood estimator (MLE) mean and standard deviation from the cluster data, and set α as the relative proportion of points.

References

[1] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with Bregman divergences," *Journal of Machine Learning Research*, vol. 6, pp. 1705–1749, 2005.

[2] S. Dasgupta, "The hardness of k-means clustering," technical report CS-2008-0916, University of California, San Diego, USA.

[3] N. Megiddo and K. J. Supowit, "On the complexity of some common geometric location problems," *SIAM Journal on Computing*, vol. 13, no. 1, pp. 182–196, 1984.

[4] R. Bellman, "A note on cluster analysis and dynamic programming," *Mathematical Biosciences*, vol. 18, no. 3-4, pp. 311 – 312, 1973.

[5] H. Wang and M. Song, "Ckmeans.1d.dp: Optimal k-means clustering in one dimension by dynamic programming," *R Journal*, vol. 3, no. 2, 2011.

[6] F. C. Crow, "Summed-area tables for texture mapping," in *Proc. Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, New York, NY, USA: ACM, 1984, pp. 207–212.

[7] D. Pelleg and A. Moore, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proc. International Conf. on Machine Learning*. Morgan Kaufmann, 2000, pp. 727–734.

[8] M. Liu, B. C. Vemuri, S. i. Amari, and F. Nielsen, "Shape retrieval using hierarchical total Bregman soft clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2407–2419, 2012.

[9] W. D. Fisher, "On grouping for maximum homogeneity," *Journal of the American Statistical Association*, vol. 53, no. 284, pp. 789–798, 1958.

[10] J.-D. Boissonnat, F. Nielsen, and R. Nock, "Bregman Voronoi diagrams," *Discrete Computational Geometry*, vol. 44, no. 2, pp. 281–307, Sep. 2010.

[11] F. Nielsen and R. Nock, "Sided and symmetrized Bregman centroids," *IEEE Transactions on Information Theory*, vol. 55, no. 6, pp. 2882–2904, 2009.

[12] F. Nielsen, "k-MLE: A fast algorithm for learning statistical mixture models," *CoRR*, vol. abs/1203.5181, 2012.

[13] L. D. Brown, *Fundamentals of statistical exponential families: with applications in statistical decision theory*. Hayworth, CA, USA: Institute of Mathematical Statistics, 1986.

[14] F. Nielsen and V. Garcia, "Statistical exponential families: A digest with flash cards," 2009, arXiv.org:0911.4863.

[15] J. Cavanaugh, "Unifying the derivations for the Akaike and corrected Akaike information criteria," *Statistics & Probability Letters*, vol. 33, no. 2, pp. 201–208, Apr. 1997.