**Max–Planck–Institut für biologische Kybernetik**
Max Planck Institute for Biological Cybernetics


Technical Report No. 177

# Approximation Algorithms for Bregman Clustering Co-clustering and Tensor Clustering

Suvrit Sra[1], Stefanie Jegelka[1]
Arindam Banerjee[2]

17 Oct 2008

[1] MPI für biologische Kybernetik, AGBS; [2] University of Minnesota, MN, USA.

# Approximation Algorithms for Bregman Clustering Co-clustering and Tensor Clustering

*Suvrit Sra, Stefanie Jegelka, and Arindam Banerjee*

**Abstract.** The Euclidean K-means problem is fundamental to clustering and over the years it has been intensely investigated. More recently, generalizations such as Bregman k-means [8], co-clustering [10], and tensor (multi-way) clustering [40] have also gained prominence. A well-known computational difficulty encountered by these clustering problems is the NP-Hardness of the associated optimization task, and commonly used methods guarantee at most local optimality. Consequently, approximation algorithms of varying degrees of sophistication have been developed, though largely for the basic Euclidean K-means (or $\ell_1$-norm K-median) problem. In this paper we present approximation algorithms for several Bregman clustering problems by building upon the recent paper of Arthur and Vassilvitskii [5]. Our algorithms obtain objective values within a factor $O(\log K)$ for Bregman k-means, Bregman co-clustering, Bregman tensor clustering, and weighted kernel k-means. To our knowledge, except for some special cases, approximation algorithms have not been considered for these general clustering problems. There are several important implications of our work: (i) under the same assumptions as Ackermann et al. [2] it yields a much faster algorithm (non-exponential in $K$, unlike [2]) for information-theoretic clustering, (ii) it answers several open problems posed by [4], including generalizations to Bregman co-clustering, and tensor clustering, (iii) it provides practical and easy to implement methods—in contrast to several other common approximation approaches.

---

## 1 Introduction

Partitioning data points into clusters is a fundamentally hard problem. The well-known Euclidean k-means problem that seeks to partition the input data into $K$ clusters, so that the sum of squared distances of the input points to their corresponding cluster centroids is minimized, is an NP-Hard problem [22]. Simple and frequently used procedures that rapidly obtain local minima exist since a long time [26, 32]. For example, Lloyd's algorithm [32], which is commonly referred to as the K-means algorithm is arguably the most popular approach to solving Euclidean k-means. Here, one begins with $K$ centers (usually chosen randomly) and assigns points to their closest centers. Each cluster center is then recomputed as the mean of the points assigned to it, and these two steps are repeated until the procedure converges. A similar greedy procedure also exists for the Bregman k-means problem, as shown in [8]. Despite enjoying properties such as monotonic descent in the objective function value and utter simplicity of implementation, these simplistic iterative approaches can often get stuck in poor local-optima. Therefore, heuristic local search strategies (e.g., [21]), or even guaranteed approximation algorithms have been designed for it (e.g., [31] or references therein). Heuristic strategies can be quite effective but are not accompanied by better than local optimality guarantees, while standard approximation algorithms quickly sacrifice the simplicity, and thereby the efficiency of the K-means algorithm.

Fortunately, in a recent paper Arthur and Vassilvitskii [5] presented a simple initialization scheme for Euclidean k-means along with an elegant analysis guaranteeing an $O(\log K)$ approximation to the globally optimal objective function value. The greatest advantage of their scheme is that it retains the simplicity and efficiency of the K-means algorithm, while still maintaining theoretical guarantees. This

paper is directly motivated by their work, which we greatly extend to obtain approximation algorithms for several Bregman clustering problems. We summarize our main results below.

### 1.1 Results.

We present approximation algorithms for the following Bregman divergence based clustering problems:

1. Bregman k-means [8] (§2),
2. Bregman co-clustering [10] (§5),
3. Bregman tensor clustering [9] (§5).

Additionally as an easy generalization of [5] we also obtain an approximation algorithm for weighted kernel k-means [20] (§4).

**Implications.** Our results have several important implications. Under assumptions similar to that of (Ackermann et al. [2], 2008), we obtain a much faster approximation algorithm for information-theoretic clustering as a special case of our approximation for Bregman k-means (§3.1). Ackermann et al. [2] require time exponential (or worse) in $K$, while our methods run in time linear in $K$. In fact, while preparing this paper we became aware of a very recent SODA 2009 paper of Ackermann and Blömer [1] (yet to appear in print), who provide new approximation algorithms for Bregman k-means. However, their new algorithms, while faster than those in [2], are *still* exponential (or worse) in $K$—our algorithm operates under the *same* assumptions on the Bregman divergences as made by [1], and is much faster (non-exponential in $K$).

Our results for Bregman co-clustering and Bregman tensor clustering answer two open problems posed by [4], and yield the first (to our knowledge) known approximation algorithms for these problems. Finally, using our $O(\log K)$ approximation for weighted kernel K-means, one can obtain potentially better algorithms for graph-cut objectives and certain semi-supervised clustering problems by exploiting the equivalences described by [20, 30].

### 1.2 Related work

There exist several books and a vast array of papers dealing with the problem of clustering. However, as our focus is on approximation algorithms for Bregman divergence based clustering problems, we summarize below only work dealing with approximation algorithms for clustering. Graph partitioning also forms a large class of clustering problems and algorithms. However, it lies outside the scope of this paper, apart from the connection via weighted kernel k-means as mentioned above.

#### 1.2.1 Clustering

The most directly related work is the paper [5] that has motivated our algorithms for clustering. If one fixes the number of clusters $K$, and the data dimensionality $d$, then Euclidean k-means can be solved exactly, in time $O(n^{Kd})$ [28]. We remark that using the Bregman Voronoi ideas of Nielsen et al. [35], it might be possible to generalize the work of Inaba et al. [28] to Bregman k-means.

Several other polynomial time approximation algorithms for K-means have been proposed in the literature, for example, [17, 25, 31] (also see the references therein). All of the algorithms proposed in these papers suffer from a common problem, namely exponential (or poly-exponential) dependence on $K$, rendering them impractical despite their theoretical pleasantness.

Of particular interest is the paper of Ackermann et al. [2], who extended clustering guarantees of Kumar et al. [31] to generic divergence measures (including Bregman divergences). Under the *same* assumptions on the underlying Bregman divergence as Ackermann et al. [2] we obtain a much faster and practical approximation algorithm for Bregman k-means than their methods. Their approximation factor is $(1+\epsilon)$ with a running time of $O(dn2^{(K/\epsilon)^{O(1)}})$, while our factor is $O(\log K)$ with a running time of

$O(dnK)$. In an even more recent paper that will appear in SODA 2009, Ackermann and Blömer [1] (preprint available from the authors' website) have introduced a new $O(dKn + d^2 2^{(K/\epsilon)^{\Theta(1)}} \log^{K+2} n)$ approximation algorithm for Bregman k-means that again achieves $(1+\epsilon)$ approximation, with the same assumptions as [2] on the underlying Bregman divergences. Our approximation algorithms for Bregman k-means are much more practical than theirs, both because of the lower running time as well as the implementational simplicity.

Related to Bregman k-means is the important special case of information theoretic clustering, wherein one minimizes the sum of KL divergences of input data points to their cluster centroids. In addition to the generic methods already summarized above, recent important work worth mentioning here is the paper of Chaudhuri and McGregor [14], who present a KL-divergence clustering algorithm that does not make *any* assumptions on the input data, but yields a non-constant $O(\log n)$ approximation.

Other relevant works such as [29, 34, 36] are summarized in [1, 2, 5], and we refer the reader to those papers for additional information.

#### 1.2.2 Co-clustering and Tensor clustering.

For a detailed discussion of co-clustering and several relevant references we refer the reader to [10], while for the lesser known problem of tensor clustering we refer the reader to [3, 9, 11, 23, 33, 40].

Approximation algorithms for co-clustering are much less well-studied. We are aware of only two very recent attempts (both papers are from 2008), namely, [38] and [4]–and both of the papers follow similar approaches to obtain their approximation guarantees. In this paper, we build upon [4] and obtain approximation algorithms for Bregman co-clustering as well as Bregman tensor clustering. We therefore answer *two open problems* posed by [4], namely, whether their methods for Euclidean co-clustering could be generalized to Bregman co-clustering, and more importantly, whether generalizations to tensors could be found. Our approximation results for co-clustering and tensor clustering may be viewed independently of our results for Bregman clustering, because they are based on being able to solve the 1-dimensional (standard) clustering problem with *any* guaranteed approximation method. One can, and we do, however, invoke our Bregman clustering results to obtain actual efficient algorithms.

Now we are ready to discuss details of our methods and we begin with Bregman clustering below.

## 2 Bregman Clustering

Bregman k-means (BREGM) was introduced by Banerjee et al. [8], and it can be viewed as a generalization of Euclidean k-means and information theoretic clustering (ITC) [19]. Below we derive a randomized algorithm called BREG++ for Bregman k-means and prove it to be within $O(\log K)$ of the optimal. In §3.1 we discuss the particularly interesting special case of ITC in further detail, especially because ITC has only recently (in 2008) witnessed some progress in terms of approximation algorithms [2, 14]. We also discuss implications of our BREG++ for mixture-modeling in §3.2.

### 2.1 Setup and Algorithm.

Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be the input data, and $\{w_1, \ldots, w_n\}$ corresponding non-negative weights. For a strictly convex function $f$, let $B_f$ denote a Bregman divergence defined as [13]

$$B_f(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) - f(\boldsymbol{y}) - \nabla f(\boldsymbol{y})^T(\boldsymbol{x} - \boldsymbol{y}). \tag{2.1}$$

Given $B_f$, Bregman k-means seeks a partition $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ of $\mathcal{X}$, such that the following objective is minimized:

$$J(\mathcal{C}) = \sum_{h=1}^{K} \sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i B_f(\boldsymbol{x}_i, \boldsymbol{\mu}_h), \tag{2.2}$$

where $\boldsymbol{\mu}_h$ is the weighted mean of cluster $\mathcal{C}_h$, i.e.,

$$\boldsymbol{\mu}_h = \frac{\sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i \boldsymbol{x}_i}{\sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i}. \tag{2.3}$$

The means given by (2.3) are optimal for a given clustering, as shown formally below.

**Lemma 2.1.** *Let $\mathcal{A}$ be a set of points with weighted mean $\boldsymbol{\mu}_{\mathcal{A}}$, and let $\boldsymbol{z}$ be an arbitrary point. Then,*

$$\sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{z}) = \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{\mu}_{\mathcal{A}}) + W_{\mathcal{A}} B_f(\boldsymbol{\mu}_{\mathcal{A}}, \boldsymbol{z}),$$

*where $W_{\mathcal{A}} = \sum_{x_i \in \mathcal{A}} w_i$.*

*Proof.* Follows directly from the definition (2.1) of $B_f$ and equation (2.3)—for details see [8]. □

**Algorithm.** Given $K$ initial means, the Bregman k-means (BREGM) algorithm [8] follows the outline:

1. For each $i = 1 \ldots N$, assign point $\boldsymbol{x}_i$ to its nearest (in $B_f$) mean updating cluster $\mathcal{C}_h$.
2. For each $h = 1 \ldots K$, compute $\boldsymbol{\mu}_h$ using (2.3).
3. Repeat steps 1 and 2 to convergence.

This simple k-means type approach monotonically decreases the objective function, finally stopping once the clusters stabilize. To obtain approximation guarantees, we must modify this basic algorithm a little. To that end, we generalize the careful initialization technique of [5], as shown below.

As its initialization, BREG++ selects cluster centers from $\mathcal{X}$ sequentially following a weighted farthest-first scheme. The first center $\boldsymbol{\mu}_1$ is chosen with probability proportional to its weight, i.e., for some $\boldsymbol{x}_i \in \mathcal{X}$

$$P(\boldsymbol{\mu}_1 = \boldsymbol{x}_i) = \frac{w_i}{\sum_{j=1}^{n} w_j}.$$

The remaining centers are chosen from $\mathcal{X}$ with a different weighting. At a given stage in the initialization, let $C$ be the set of centers already chosen. Let $D(\boldsymbol{x})$ denote the smallest Bregman divergence of a point $\boldsymbol{x}$ in $\mathcal{X}$ to an already chosen center, i.e.,

$$D(\boldsymbol{x}) = \min_{\boldsymbol{\mu} \in C} B_f(\boldsymbol{x}, \boldsymbol{\mu}). \tag{2.4}$$

Then BREG++ chooses the next center $\boldsymbol{\mu}_h$ by letting $\boldsymbol{\mu}_h = \boldsymbol{x}_i \in \mathcal{X}$, with probability

$$P(\boldsymbol{\mu}_h = \boldsymbol{x}_i) = \frac{w_i D(\boldsymbol{x}_i)}{\sum_{j=1}^{n} w_j D(\boldsymbol{x}_j)}. \tag{2.5}$$

The initialization steps (2.4) and (2.5) are repeated until we have chosen $K$ centers, which are then used to initialize the standard BREGM algorithm. Interestingly, this weighted farthest-first initialization alone is sufficient to bring BREG++ within a factor $O(\log K)$ of the optimal, as the analysis below shows.

### 2.2 Analysis.

Arthur and Vassilvitskii's [2007] analysis does not directly generalize to Bregman k-means. Some additional details must be developed as outlined in this section. We assume that the Bregman divergence being minimized has bounded curvature, i.e., $\exists\ \sigma_1, \sigma_2$ with $0 < \sigma_1 \leq \sigma_2 < \infty$, such that

$$\sigma_1 \|\boldsymbol{x} - \boldsymbol{y}\|^2 \leq B_f(\boldsymbol{x}, \boldsymbol{y}) \leq \sigma_2 \|\boldsymbol{x} - \boldsymbol{y}\|^2. \tag{2.6}$$

Note that the bounds (2.6) need not hold over the entire domain of $f$—they can be limited to convex hulls of the input data points. Specifically, we can select

$$\sigma_1 = \inf_{\substack{\boldsymbol{x}\in\mathcal{X}\\ \boldsymbol{y}\in\mathrm{conv}(\mathcal{X})}} \frac{B_f(\boldsymbol{x},\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|^2}, \quad \sigma_2 = \sup_{\substack{\boldsymbol{x}\in\mathrm{conv}(\mathcal{X})\\ \boldsymbol{y}\in\mathcal{X}}} \frac{B_f(\boldsymbol{x},\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|^2}, \tag{2.7}$$

where $\mathrm{conv}(\mathcal{X})$ denotes the convex hull of $\mathcal{X}$, i.e., the set of points that can be expressed as $\boldsymbol{y} = \sum_{\boldsymbol{x}\in\mathcal{X}} \alpha_{\boldsymbol{x}}\boldsymbol{x}$ with $\alpha_{\boldsymbol{x}} \geq 0$ and $\sum_{\boldsymbol{x}\in\mathcal{X}} \alpha_{\boldsymbol{x}} = 1$. Though these bounds might appear restrictive, they are in fact not that limiting, as our treatment of information theoretic clustering in Section 3.1 shows. In fact, it turns out the similar bounds were assumed in the very recent work of Ackermann et al. [2], and [1]. In the language of convex-optimization, these bounds are nothing but bounds on curvature (Hessian) of the convex function $B_f$ (e.g., in the context of *strong convexity* [12, §9.1.2]).

On a more intriguing note, it seems that without such assumptions on the curvature of $B_f$, one might not be able to obtain constant approximation ratios; this intuition is reinforced by the recent results of [14], who avoided making such assumptions, but ended up with an $O(\log n)$ approximation.

We now prove the approximation in three steps (following [5]). First we show that BREG++ is competitive in those clusters out of the optimal clustering $\mathcal{C}_{\text{OPT}}$ from which it happens to sample a center.

**Lemma 2.2.** *Let $\mathcal{A}$ be an arbitrary cluster in $\mathcal{C}_{OPT}$ and let $\mathcal{C}$ be the clustering with just one center that was chosen with probability proportional to the weight of points in $\mathcal{A}$. Then, if $J(\mathcal{A})$ is the contribution of points in $\mathcal{A}$ to the final objective, we have*

$$E[J(\mathcal{A})] \leq (1+\frac{\sigma_2}{\sigma_1}) J_{OPT}(\mathcal{A}).$$

*Proof.* Let $\mu_{\mathcal{A}}$ denote the weighted mean of cluster $\mathcal{A}$. Since $\mathcal{C}_{\text{OPT}}$ is optimal, it must be using $\boldsymbol{\mu}_{\mathcal{A}}$ as its center. Let $W_{\mathcal{A}} = \sum_{x_i\in\mathcal{A}} w_i$. Now invoking Lemma 2.1 we note that $E[J(\mathcal{A})]$ is given by

$$\begin{aligned}
&\sum_{\boldsymbol{\mu}_0\in\mathcal{A}} \frac{w_o}{W_{\mathcal{A}}}\Big(\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i B_f(\boldsymbol{x}_i;\boldsymbol{\mu}_0)\Big)\\
=\;& \sum_{\boldsymbol{\mu}_0\in\mathcal{A}} \frac{w_o}{W_{\mathcal{A}}} \sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i B_f(\boldsymbol{x}_i;\boldsymbol{\mu}_{\mathcal{A}}) + \sum_{\boldsymbol{\mu}_0} W_{\mathcal{A}} B_f(\boldsymbol{\mu}_{\mathcal{A}};\boldsymbol{\mu}_0)\\
=\;& J_{\text{OPT}}(\mathcal{A}) + \sum_{\boldsymbol{\mu}_0\in\mathcal{A}} W_{\mathcal{A}} \frac{B_f(\boldsymbol{\mu}_{\mathcal{A}};\boldsymbol{\mu}_0)}{B_f(\boldsymbol{\mu}_0;\boldsymbol{\mu}_{\mathcal{A}})} B_f(\boldsymbol{\mu}_0;\boldsymbol{\mu}_{\mathcal{A}})\\
\leq\;& J_{\text{OPT}}(\mathcal{A}) + \frac{\sigma_2}{\sigma_1} J_{\text{OPT}}(\mathcal{A}) = (1+\frac{\sigma_2}{\sigma_1}) J_{\text{OPT}}(\mathcal{A}),
\end{aligned}$$

where the last inequality follows from (2.7). □

The second step consists of showing how the algorithm behaves for the remaining centers that are chosen with the weighted farthest-first sampling.

**Lemma 2.3.** *Let $\mathcal{A}$ be an arbitrary cluster in $\mathcal{C}_{OPT}$, and let $\mathcal{C}$ be an arbitrary clustering. If we add a random point of $\mathcal{A}$ as a center to $\mathcal{C}$ using (2.4) and (2.5), then*

$$E[J(\mathcal{A})] \leq 4\frac{\sigma_2}{\sigma_1}\left(1+\frac{\sigma_2}{\sigma_1}\right) J_{OPT}(\mathcal{A})\,.$$

*Proof.* After choosing a center $\boldsymbol{x}_0$ from $\mathcal{A}$, any point $x_i \in \mathcal{A}$ will contribute $w_i \min(D(\boldsymbol{x}_i), B_f(\boldsymbol{x}_i,\boldsymbol{x}_0))$ to the objective. Since we sample according to (2.5), the expected value of the objective $E[J(\mathcal{A})]$ is

$$\sum_{\boldsymbol{x}_0\in\mathcal{A}} \frac{w_0 D(\boldsymbol{x}_0)}{\sum_{\boldsymbol{x}\in\mathcal{A}} w_{\boldsymbol{x}} D(\boldsymbol{x})} \sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i \min(D(\boldsymbol{x}_i), B_f(\boldsymbol{x}_i,\boldsymbol{x}_0)).$$

Let $\boldsymbol{c}_0$ and $\boldsymbol{c}_i$ be the centers closest to $\boldsymbol{x}_0$ and $\boldsymbol{x}_i$, respectively, From the triangle inequality we have

$$\|\boldsymbol{x}_0 - \boldsymbol{c}_i\| \leq \|\boldsymbol{x}_0 - \boldsymbol{x}_i\| + \|\boldsymbol{x}_i - \boldsymbol{c}_i\|.$$

Then using (2.7) we can bound the divergence

$$\begin{aligned} &B_f(\boldsymbol{x}_0, \boldsymbol{c}_i) \leq \sigma_2 \|\boldsymbol{x}_0 - \boldsymbol{c}_i\|^2 \leq \sigma_2 \big(\|\boldsymbol{x}_0 - \boldsymbol{x}_i\| + \|\boldsymbol{x}_i - \boldsymbol{c}_i\|\big)^2 \\ &\leq 2\sigma_2 \|\boldsymbol{x}_0 - \boldsymbol{x}_i\|^2 + 2\sigma_2 \|\boldsymbol{x}_i - \boldsymbol{c}_i\|^2 \\ &\leq 2\frac{\sigma_2}{\sigma_1} B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) + 2\frac{\sigma_2}{\sigma_1} B_f(\boldsymbol{x}_i, \boldsymbol{c}_i). \end{aligned}$$

Noting that $D(\boldsymbol{x}_0) = B_f(\boldsymbol{x}_0, \boldsymbol{c}_0) \leq B_f(\boldsymbol{x}_0, \boldsymbol{c}_i)$ and $D(\boldsymbol{x}_i) = B_f(\boldsymbol{x}_i, \boldsymbol{c}_i)$, we have the bound

$$D(\boldsymbol{x}_0) \leq 2\frac{\sigma_2}{\sigma_1} B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) + 2\frac{\sigma_2}{\sigma_1} D(\boldsymbol{x}_i).$$

Multiplying both sides by $w_i$ and summing over all $\boldsymbol{x}_i \in \mathcal{A}$, we have (for $W_{\mathcal{A}} = \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i$)

$$\begin{aligned} &W_{\mathcal{A}} D(\boldsymbol{x}_0) \leq 2\frac{\sigma_2}{\sigma_1} \Big( \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) + w_i D(\boldsymbol{x}_i) \Big), \text{ i.e.,} \\ &w_0 D(\boldsymbol{x}_0) \leq 2\frac{\sigma_2}{\sigma_1} \frac{w_0}{W_{\mathcal{A}}} \Big( \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) + w_i D(\boldsymbol{x}_i) \Big). \end{aligned}$$

Now letting $R(\mathcal{A}) = \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \min(D(\boldsymbol{x}_i), B_f(\boldsymbol{x}_i, \boldsymbol{x}_0))$, we see that $E[J(\mathcal{A})]$ is upper bounded by

$$\begin{aligned} & 2\frac{\sigma_2}{\sigma_1} \sum_{\boldsymbol{x}_0 \in \mathcal{A}} \frac{w_0}{W_{\mathcal{A}}} \left( \frac{\sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i (B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) + D(\boldsymbol{x}_i))}{\sum_{\boldsymbol{x} \in \mathcal{A}} w_{\boldsymbol{x}} D(\boldsymbol{x})} R(\mathcal{A}) \right) \\ \leq \quad & 2\frac{\sigma_2}{\sigma_1} \sum_{\boldsymbol{x}_0 \in \mathcal{A}} \frac{w_0}{W_{\mathcal{A}}} \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) \\ + \quad & 2\frac{\sigma_2}{\sigma_1} \sum_{\boldsymbol{x}_0 \in \mathcal{A}} \frac{w_0}{W_{\mathcal{A}}} \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) \\ = \quad & 4\frac{\sigma_2}{\sigma_1} \sum_{\boldsymbol{x}_0 \in \mathcal{A}} \frac{w_0}{W_{\mathcal{A}}} \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i B_f(\boldsymbol{x}_i, \boldsymbol{x}_0) \\ \leq \quad & 4\frac{\sigma_2}{\sigma_1} (1 + \frac{\sigma_2}{\sigma_1}) J_{\text{OPT}}(\mathcal{A}), \end{aligned}$$

where in the second line we simplified $R(\mathcal{A})$ by using $\min(D(\boldsymbol{x}_i), B_f(\boldsymbol{x}_i, \boldsymbol{x}_0)) \leq D(\boldsymbol{x}_i)$, while for the third line we used $\min(D(\boldsymbol{x}_i), B_f(\boldsymbol{x}_i, \boldsymbol{x}_0)) \leq B_f(\boldsymbol{x}_i, \boldsymbol{x}_0)$. The last inequality follows from Lemma 2.2. □

**Remark 2.3** (K-means)**.** For $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}$ we have $\sigma_1 = \sigma_2$, and Bregman k-means is reduces to Euclidean k-means, and our analysis reduces to that of [5].

**Remark 2.4** (Mahalanobis)**.** For $f = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$, where $\boldsymbol{A}$ is a positive-definite matrix, the resulting Bregman divergence is a Mahalanobis distance. Here, one has $\sigma_1 = \lambda_{\min}(\boldsymbol{A})$, and $\sigma_2 = \lambda_{\max}(\boldsymbol{A})$, *independent* of the input data (the $\lambda$s denote eigenvalues of $\boldsymbol{A}$). The approximation ratio depends naturally on the condition number $\lambda_{\max}(\boldsymbol{A})/\lambda_{\min}(\boldsymbol{A})$ of $\boldsymbol{A}$.

Given Lemmas 2.2 and 2.3, the third step of our proof is simple as we can essentially invoke Lemma 3.3 of [5] to show that the total error incurred via the weighted farthest-first sampling is within a factor $O(\log K)$ of the optimal.

**Lemma 2.4.** *Let $\mathcal{C}$ be an arbitrary clustering. Choose $u > 0$ "uncovered"[1] clusters from $\mathcal{C}_{OPT}$, and let $\mathcal{X}_u$ denote the set of points in these clusters. Also let $\mathcal{X}_c = X \setminus \mathcal{X}_u$. Now suppose we add $t \leq u$ random centers to $\mathcal{C}$, chosen with the weighted farthest-first sampling. Let $\mathcal{C}'$ denote the resulting clustering, and let $J'$ denote the corresponding objective. Then, $E[J']$ is at most*

$$\Big(J(\mathcal{X}_c) + 4\frac{\sigma_2}{\sigma_1}\big(1 + \frac{\sigma_2}{\sigma_1}\big)J_{OPT}(\mathcal{X}_u)\Big) \cdot (1 + H_t) + \frac{u-t}{u} \cdot J(\mathcal{X}_u),$$

*where $H_t$ denotes the Harmonic number $1 + \frac{1}{2} + \cdots + \frac{1}{t}$.*

*Proof.* Direct from the proof of Lemma 3.3 of [5]. □

Finally, we have the main approximation theorem.

**Theorem 2.5.** *A clustering $\mathcal{C}$ obtained via* BREG++ *satisfies*

$$E[J(\mathcal{C})] \leq 4\frac{\sigma_2}{\sigma_1}\left(1 + \frac{\sigma_2}{\sigma_1}\right)(\log K + 2)J_{OPT}.$$

*Proof.* Immediate from Theorem 3.1 of [5] by a direct application of Lemma 2.4. □

## 3 Implications of BREG++

We now describe some important implications of our BREG++ method derived above.

### 3.1 Information Theoretic Clustering.

With $f(\boldsymbol{x}) = \sum_j x_j \log x_j$, the Bregman divergence $B_f$ becomes the (un-normalized) Kullback-Leibler divergence, and Bregman k-means reduces to information theoretic clustering (ITC). Even though local and greedy methods for ITC have been well studied [6, 19, 37], approximation algorithms for it have only been developed very recently [2, 14] (both papers are from 2008).

For ITC, our assumptions on bounded $\sigma_1$ and $\sigma_2$ are equivalent to those of [2] as mentioned previously. Under these assumptions we obtain an efficient k-means type $O(\log K)$ approximation algorithm, while Ackermann et al. [2] obtain an $O(1+\epsilon)$ approximation algorithm, with an impractical running time of $O(dn2^{(\frac{K}{\epsilon})^{O(1)}})$. The even more recent paper of Ackermann and Blömer [1] yields an ITC algorithm faster than that of [2], but still has an impractical running time of $O(dKn + d^2 2^{(K/\epsilon)^{\Theta(1)}} \log^{K+2} n)$.

Chaudhuri and McGregor [14] develop an approximation algorithm ITC that does not make any assumptions on the data. They first lower bound the KL-divergence using a Hellinger distance, then cluster approximately using this distance, before recovering clusters for KL. Their method is clever, but leads to a non-constant approximation ratio of $O(\log n)$ ($n$ is the number of input points). Obtaining an $O(\log K)$ algorithm for ITC without any assumptions on the data therefore remains an open problem—though we suspect that without additional assumptions, ITC might be inapproximable to better than a polylog factor.

**Details.** Observe that given the definition the Bregman divergence (2.1), using Taylor series expansion of $f$ around $\boldsymbol{x}$ we immediately have

$$B_f(\boldsymbol{x}, \boldsymbol{y}) = \tfrac{1}{2}(\boldsymbol{x} - \boldsymbol{y})^T \nabla^2 f(\xi_{\boldsymbol{x},\boldsymbol{y}})(\boldsymbol{x} - \boldsymbol{y}),$$

where $\xi_{\boldsymbol{x},\boldsymbol{y}}$ is some point between $\boldsymbol{x}$ and $\boldsymbol{y}$. Since $\nabla^2 f$ is positive definite, the constants $\sigma_1$ and $\sigma_2$ can be obtained by bounding the minimum and maximum eigenvalues of $\nabla^2 f$. For ITC, one assumes the input data to be normalized, i.e., for each $\boldsymbol{x} \in \mathcal{X}$, $\sum_j x_j = 1$. Since the Hessian for the KL-divergence is $\nabla^2 f(\boldsymbol{x}) = \text{Diag}(x_1^{-1}, \ldots, x_d^{-1})$ at a point $\boldsymbol{x} = (x_1, \ldots, x_d)$, its maximum eigenvalue is $(1 - \|1 - \boldsymbol{x}\|_\infty)^{-1}$ and the minimum eigenvalue is $\|\boldsymbol{x}\|_\infty^{-1}$.

[1] An "uncovered" cluster is one from which a center has not been chosen by the weighted farthest-first sampling procedure.

Since the interpolating point $\xi_{\boldsymbol{x},\boldsymbol{y}}$ lies in $\mathrm{conv}(\mathcal{X})$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{conv}(\mathcal{X})$, we can select

$$\sigma_1^{-1} = \max_{\boldsymbol{y} \in \mathrm{conv}(\mathcal{X})} \|\boldsymbol{y}\|_\infty = \max_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_\infty.$$

Analogously we have,

$$\sigma_2^{-1} = \min_{\boldsymbol{y} \in \mathrm{conv}(\mathcal{X})} (1 - \|1 - \boldsymbol{y}\|_\infty) = \min_{\boldsymbol{x} \in \mathcal{X}} (1 - \|1 - \boldsymbol{x}\|_\infty).$$

The reduction from $\mathrm{conv}(\mathcal{X})$ to $\mathcal{X}$ results from each $\boldsymbol{y} \in \mathrm{conv}(\mathcal{X})$ being a convex combination of the data points, whereby its coordinates are a convex combination of the data point coordinates. This convex combination is maximized by putting all weight on the maximum component. Thus, $\sigma_1 \geq 1$ and $\sigma_2$ corresponds to inverse of the minimum coordinate entry $\gamma$ in the data set, so $\sigma_2/\sigma_1 \leq \gamma^{-1}$. Ackermann et al. [2] and Ackermann and Blömer [1] will also have a similar dependence on $\gamma$.

### 3.2 Mixture Modeling on Exponential Families

The parametric mixture modeling problem entails fitting a mixture of $K$ distributions from a pre-defined family to a set of observations. Let $\boldsymbol{x}$ denote an observation, $\boldsymbol{\pi}$ a prior over the mixture components, and $\boldsymbol{\theta}_h$ the parameters corresponding to the $z^{th}$ mixture component. Then, a mixture model assumes the following generative process: (i) sample $z \sim \boldsymbol{\pi}$, and (ii) sample $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta}_z)$. Given a set of observations, the basic mixture modeling problem is that of finding the parameters $\boldsymbol{\Theta} = (\boldsymbol{\pi}, \boldsymbol{\theta}_z, \{z\}_1^K)$ such that $\log p(\boldsymbol{x}|\boldsymbol{\Theta})$ is maximized.[2] More formally, if $X$ denotes the random variable corresponding to the observations and $Z$ denotes the one corresponding to the mixture components, a direct calculation [7] shows that the problem is equivalent to maximizing

$$J_{MM}(\boldsymbol{\Theta}) = E_{Z|X} \left[\log p(X, Z|\boldsymbol{\Theta})\right] + H(Z|X) \tag{3.1}$$

over $\boldsymbol{\Theta}$, where

$$p(z|\boldsymbol{x}) = \frac{\pi_z p(\boldsymbol{x}|\boldsymbol{\theta}_z)}{p(\boldsymbol{x})},$$

and $H(\cdot)$ denotes the Shannon entropy of $Z|X$. For the purposes of analysis one can focus on the expected log-likelihood of the data, i.e., the first term in (3.1). In practice, for several real datasets, the distribution $p(z|\boldsymbol{x})$ is typically skewed in that it has a high value $\approx 1$ for some $z^*$, and low values $\approx 0$ for other $z$, so that the entropy $H(Z|X)$ is rather small. As a result, ignoring the entropy term may be reasonable in an application. A more theoretically well motivated justification can be given by considering "hard clustering" for the mixture modeling problem, where we focus on the family of posterior distributions

$$q(z|\boldsymbol{x}) = \begin{cases} 1, & \text{if} p(z|\boldsymbol{x}) > p(z'|\boldsymbol{x}), \forall z' \neq z, \\ 0, & \text{otherwise} . \end{cases}$$

For simplicity, we let $z_i^* = z$ such that $p(z|\boldsymbol{x}_i) > p(z'|\boldsymbol{x}_i), \forall z' \neq z$. If $J_Q(\boldsymbol{\Theta})$ is the corresponding objective, then following [7] we have

$$J_{MM}(\boldsymbol{\Theta}) - H(Z|X) \leq J_Q(\boldsymbol{\Theta}) \leq J_{MM}(\boldsymbol{\Theta}) .$$

Thus, $J_Q(\boldsymbol{\Theta})$ forms a tight lower bound to the original objective $J_{MM}(\boldsymbol{\Theta})$, especially when entropy of $Z|X$ is small, which is true for several real world problems.

[2] Note that the objective $\log p(\boldsymbol{x}|\boldsymbol{\Theta})$ is always non-positive since $p(\boldsymbol{x}|\boldsymbol{\Theta}) \leq 1$. All objective functions in this section share the same property.

We focus on mixture models over exponential family distributions, whose density functions can be written as

$$p(\boldsymbol{x}|\theta) = \exp(\langle \boldsymbol{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}_z)) p_0(\boldsymbol{x}) \ ,$$

where $\psi$ is a convex function of Legendre [39] type known as the cumulant, $\theta$ is the natural parameter, and $p_0(\boldsymbol{x})$ is a base measure. A particular choice of $\psi$ determines a family, such as Gaussian or Poisson, while a particular choice of $\boldsymbol{\theta}$ determines a specific distribution in the family. The expectation parameter $\boldsymbol{\mu} = E[X]$ of an exponential family distribution is uniquely tied to the natural parameter through a Legendre transform $\boldsymbol{\mu} = \nabla\psi(\boldsymbol{\theta})$, and $\boldsymbol{\theta} = \nabla\phi(\boldsymbol{\mu})$ where $\phi$ is the Legendre conjugate of $\psi$ [39].

Our results below rely on the following key connection between exponential family distributions and Bregman divergences [8]. The density function $p(\boldsymbol{x}|\boldsymbol{\theta}_z)$ of an exponential family distribution can be uniquely written as

$$p(\boldsymbol{x}|\boldsymbol{\theta}_z) = \exp(-B_\phi(\boldsymbol{x}, \boldsymbol{\mu})) f_0(\boldsymbol{x}), \tag{3.2}$$

where $\phi$ is the Legendre conjugate of $\psi$, and $\boldsymbol{\mu} = E[X] = \nabla\psi(\boldsymbol{\theta})$ is the expectation parameter.

Now we use BREG++ to optimize $J_Q(\boldsymbol{\Theta})$ based on the expectation parameter $\boldsymbol{\mu}_z$ of each component $z = 1, \ldots, K$. The only additional step is to set (for each $i$) $z_i^* = z$ if $\boldsymbol{x}_i$ is assigned to cluster $z$.

**Lemma 3.1.** *Let $\boldsymbol{\Theta}_{MM}$ denote the natural parameters corresponding to the final mean parameters after convergence, and let $\boldsymbol{\pi}_z = |C_z|/n$, where $|C_z|$ denotes the number of elements in the $z$-th cluster. Then, (recall $J_Q$ is negative)*

$$E[J_Q(\boldsymbol{\Theta}_{MM})] \geq 4\frac{\sigma_2}{\sigma_1}\left(1 + \frac{\sigma_2}{\sigma_1}\right) J_Q(\boldsymbol{\Theta}^*),$$

*where $\boldsymbol{\Theta}^*$ denotes an optimum set of parameters.*

*Proof.* By definition,

$$\begin{aligned}
\max_{\boldsymbol{\Theta}} J_Q(\boldsymbol{\Theta}) &= \max_{\boldsymbol{\Theta}} E_{Z|X\sim Q}[\log p(X, Z|\boldsymbol{\Theta})] + H_Q(Z|X) \\
&= \max_{\boldsymbol{\Theta}} \frac{1}{n}\sum_{i=1}^{n} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}_{z_i^*}) \\
&= \max_{\boldsymbol{\mu}} -\frac{1}{n}\sum_{i=1}^{n} B_\phi(\boldsymbol{x}_i, \boldsymbol{\mu}_{z_i^*}) \quad = \quad \min_{\boldsymbol{\mu}} \frac{1}{n}\sum_{i=1}^{n} B_\phi(\boldsymbol{x}_i, \boldsymbol{\mu}_{z_i^*}) \ ,
\end{aligned}$$

which is precisely the objective function for Bregman k-means. Thus, the result follows from Lemma 2.3, and a change in the direction of the inequality due to conversion of the minimization problem to a maximization problem by multiplying both sides with -1. □

## 4 Weighted Kernel K-means

In this section we present WKKM++, an $O(\log K)$ approximation algorithm for the weighted kernel k-means (WKKM) problem [20].

Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ denote the set of input data points (which may or may not be available explicitly), and let $\phi : \boldsymbol{x} \to \phi(\boldsymbol{x}) \in \mathcal{H}$, denote the feature map that takes $\boldsymbol{x}$ to its corresponding point in an RKHS $\mathcal{H}$. Further, let $w_1, w_2, \ldots, w_n$ denote non-negative weights corresponding to each input point.

WKKM seeks a clustering $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$ such that the following objective is minimized

$$J(\mathcal{C}) = \sum_{h=1}^{K} \sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i \|\phi(\boldsymbol{x}_i) - \boldsymbol{\mu}_h\|^2 \ , \tag{4.1}$$

where $\boldsymbol{\mu}_h$ is the weighted mean of cluster $\mathcal{C}_h$, i.e.,

$$\boldsymbol{\mu}_h = \frac{\sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i \phi(\boldsymbol{x}_i)}{\sum_{\boldsymbol{x}_i \in \mathcal{C}_h} w_i} \,. \tag{4.2}$$

For a given clustering, the weighted centroid $\boldsymbol{\mu}_h$ (4.2) is optimal; formally stated

**Lemma 4.1** (Optimality of Mean)**.** *Let $\mathcal{A}$ be a set of points with weighted mean $\boldsymbol{\mu}_\mathcal{A}$, and let $\phi(\boldsymbol{z})$ be an arbitrary point. Then,*

$$\begin{aligned} &\sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{z})\|^2 \\ &= \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \|\phi(\boldsymbol{x}_i) - \boldsymbol{\mu}_\mathcal{A}\|^2 + W \|\boldsymbol{\mu}_\mathcal{A} - \phi(\boldsymbol{z})\|^2, \end{aligned}$$

*where $W = \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i$.*

*Proof.* Elementary; similar to Lemma 2.2. □

The WKKM++ algorithm proceeds exactly like the BREG++ algorithm of Section 2. Specifically, WKKM++ selects $K$ initial means from amongst the data points using a particular sampling procedure. These $K$ means are then used as an initialization for the WKKM algorithm:

1. For each $i = 1..N$ assign point $\boldsymbol{x}_i$ to its nearest mean, update corresponding cluster $\mathcal{C}_h$
2. For each $h = 1..K$ update $\boldsymbol{\mu}_h$ using (4.2).
3. Repeat steps 1 and 2 until convergence.

This standard approach is guaranteed to only monotonically decrease the objective function. However, the crux of the analysis is in showing that after just the weighted farthest-first initialization based on (4.3) and (4.4), WKKM++ comes to within a factor $O(\log K)$ of the optimal. We elaborate on this below.

The first mean $\boldsymbol{\mu}_1 = \phi(\boldsymbol{x}_i)$ is chosen uniformly at random from the data points. The remaining means are chosen with a weighted farthest-first sampling procedure outlined below.

First, we define the weighting function

$$D(\boldsymbol{x}) = \min_{\boldsymbol{\mu} \in C} \|\phi(\boldsymbol{x}) - \boldsymbol{\mu}\|^2, \tag{4.3}$$

which is easily computable using dot-products only because

$$\|\phi(\boldsymbol{x}) - \boldsymbol{\mu}\|^2 = \phi(\boldsymbol{x})^T \phi(\boldsymbol{x}) + \boldsymbol{\mu}^T \boldsymbol{\mu} - 2\phi(\boldsymbol{x})^T \boldsymbol{\mu},$$

and $\boldsymbol{\mu}$ is just one of the points $\boldsymbol{x}_i$ during the initialization.

At a given stage in the algorithm, suppose we wish to select the next mean $\boldsymbol{\mu}_h$. The probability that a given point $\phi(\boldsymbol{x}_i)$ is chosen to be $\boldsymbol{\mu}_h$ is set to

$$P(\boldsymbol{\mu}_h = \phi(\boldsymbol{x}_i)) = \frac{w_i D(\boldsymbol{x}_i)^2}{\sum_{j=1}^n w_j D(\boldsymbol{x}_j)^2}. \tag{4.4}$$

The probability (4.4) is also computable using dot-products only, as it involves only the weighting function $D(\boldsymbol{x})$, which itself is so computable. We repeatedly select means using (4.3) and (4.4) until we have selected $K$ different means.

Now we proceed to show that the weighted farthest-first initialization as described above is sufficient to guarantee an $O(\log K)$ factor. First we show that WKKM++ is competitive in those clusters out of the optimal clustering $\mathcal{C}_{\text{OPT}}$ from which it happens to sample a center.

**Lemma 4.2** (First mean)**.** *Let $\mathcal{A}$ be an arbitrary cluster in $\mathcal{C}_{OPT}$, and let $\mathcal{C}$ be the clustering with just one center, which is chosen with probability proportional proportional to the weight of points in $\mathcal{A}$. Then, if $J(\mathcal{A})$ is the contribution of points in $\mathcal{A}$ to the final objective, $E[J(\mathcal{A})] \leq 2J_{OPT}(\mathcal{A})$.*

*Proof.* Let $\boldsymbol{\mu}_{\mathcal{A}}$ denote the weighted mean of cluster $\mathcal{A}$. Since $\mathcal{C}_{\text{OPT}}$ is optimal, it must be using $\boldsymbol{\mu}_{\mathcal{A}}$ as its center. Let $W_{\mathcal{A}} = \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i$. With the random initialization, assuming all points in $\mathcal{A}$ stay assigned to the cluster $\mathcal{A}$ till the end, using Lemma 4.1 we see that $E[J(\mathcal{A})]$ is given by

$$\begin{aligned}
&\sum_{\boldsymbol{x}_0 \in \mathcal{A}} \frac{w_0}{W_{\mathcal{A}}} \left( \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_0)\|^2 \right) \\
&= \sum_{\boldsymbol{x}_0 \in A} \frac{w_0}{W_{\mathcal{A}}} \left( \sum_{\boldsymbol{x}_i \in A} w_i \|\phi(\boldsymbol{x}_i) - \boldsymbol{\mu}_{\mathcal{A}}\|^2 + W_{\mathcal{A}} \|\phi(\boldsymbol{x}_0) - \boldsymbol{\mu}_{\mathcal{A}}\|^2 \right) \\
&= \sum_{\boldsymbol{x}_i \in A} w_i \|\phi(\boldsymbol{x}) - \boldsymbol{\mu}_{\mathcal{A}}\|^2 + \sum_{\boldsymbol{x}_0 \in A} w_0 \|\phi(\boldsymbol{x}_0) - \boldsymbol{\mu}_{\mathcal{A}}\|^2 \\
&= 2J_{\text{OPT}}(\mathcal{A}) \; .
\end{aligned}$$

Since the contribution of each point can only decrease in subsequent WKKM++ ++ iterations, we have $E[J(\mathcal{A})] \leq 2J_{\text{OPT}}(\mathcal{A})$. □

Next we show how WKKM++ behaves for the remaining centers that it picks.

**Lemma 4.3** (Other means)**.** *Let $\mathcal{A}$ be an arbitrary cluster in $\mathcal{C}_{OPT}$, and let $\mathcal{C}$ be an arbitrary clustering. If we add a random point from $\mathcal{A}$ as a center to $\mathcal{C}$ using the the farthest-first sampling, then $E[J(\mathcal{A})] \leq 8J_{OPT}(\mathcal{A})$.*

*Proof.* After choosing a center $\phi(\boldsymbol{x}_0)$, any point $\boldsymbol{x}_i \in \mathcal{A}$ will contribute $w_i \min(D(\boldsymbol{x}_i), \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_0)\|)^2$ to the objective. Since subsequent assignments can only decrease the contribution, we have

$$E[J(\mathcal{A})] \leq \sum_{\boldsymbol{x}_0 \in A} \frac{w_0 D(\boldsymbol{x}_0)^2}{\sum_{\boldsymbol{x}' \in \mathcal{A}} w_{\boldsymbol{x}'} D(\boldsymbol{x}')^2} \times \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \min(D(\boldsymbol{x}_i), \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_0)\|)^2 \; .$$

Let $\boldsymbol{c}_0, \boldsymbol{c}_i$ be the closest centers to $\phi(x_0), \phi(x_i)$, respectively. Then, from the triangle inequality we have

$$\|\phi(\boldsymbol{x}_0) - \boldsymbol{c}_i\| \leq \|\phi(\boldsymbol{x}_0) - \phi(\boldsymbol{x}_i)\| + \|\phi(\boldsymbol{x}_i) - \boldsymbol{c}_i\| \; .$$

Further, from the Cauchy-Schwartz inequality we have

$$\|\phi(\boldsymbol{x}_0) - \boldsymbol{c}_i\|^2 \leq 2\|\phi(\boldsymbol{x}_0) - \phi(\boldsymbol{x}_i)\|^2 + 2\|\phi(\boldsymbol{x}_i) - \boldsymbol{c}_i\|^2 \; .$$

Noting that $D(\boldsymbol{x}_0) = \|\phi(\boldsymbol{x}_0) - \boldsymbol{c}_0\| \leq \|\phi(\boldsymbol{x}_0) - \boldsymbol{c}_i\|$, we hence have

$$D(\boldsymbol{x}_0)^2 \leq 2\|\phi(\boldsymbol{x}_0) - \phi(\boldsymbol{x}_i)\|^2 + 2D(\boldsymbol{x}_i)^2 \; .$$

Multiplying both sides by $w_i$ and summing over $\boldsymbol{x}_i \in \mathcal{A}$ we obtain

$$\begin{aligned}
W_{\mathcal{A}} D(\boldsymbol{x}_0)^2 &\leq 2 \left( \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_0)\|^2 + w_i D(\boldsymbol{x}_i) \right) \\
\Rightarrow w_0 D(\boldsymbol{x}_0)^2 &\leq 2 \frac{w_0}{W_{\mathcal{A}}} \left( \sum_{\boldsymbol{x}_i \in \mathcal{A}} w_i \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_0)\|^2 + w_i D(\boldsymbol{x}_i) \right) \; .
\end{aligned}$$

Hence, $E[J(\mathcal{A})]$ is upper bounded by

$$
\begin{aligned}
&2\sum_{\boldsymbol{x}_0\in\mathcal{A}}\frac{w_0}{W_{\mathcal{A}}}\left(\frac{\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|^2}{\sum_{\boldsymbol{x}'\in\mathcal{A}} w_{\boldsymbol{x}'}D(\boldsymbol{x}')^2}\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\min(D(\boldsymbol{x}_i),\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|)^2\right)\\
&\quad+2\sum_{\boldsymbol{x}_0\in\mathcal{A}}\frac{w_0}{W_{\mathcal{A}}}\left(\frac{\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_iD(\boldsymbol{x}_i)^2}{\sum_{\boldsymbol{x}'\in\mathcal{A}} w_{\boldsymbol{x}'}D(\boldsymbol{x}')^2}\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\min(D(\boldsymbol{x}_i),\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|)^2\right)\\
\le&2\sum_{\boldsymbol{x}_0\in\mathcal{A}}\frac{w_0}{W_{\mathcal{A}}}\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|^2+2\sum_{\boldsymbol{x}_0\in\mathcal{A}}\frac{w_0}{W_{\mathcal{A}}}\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|^2\\
=&4\sum_{\boldsymbol{x}_0\in\mathcal{A}}\frac{w_0}{W_{\mathcal{A}}}\sum_{\boldsymbol{x}_i\in\mathcal{A}} w_i\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|^2\\
&\le 8J_{\text{OPT}}(\mathcal{A})\,,
\end{aligned}
$$

where in the first line we used $\min(D(\boldsymbol{x}_i),\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|)^2 \le D(\boldsymbol{x}_i)^2$ in the first expression, and $\min(D(\boldsymbol{x}_i),\|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|)^2 \le \|\phi(\boldsymbol{x}_i)-\phi(\boldsymbol{x}_0)\|^2$ in the second expression, and the last line follows from Lemma 4.2. □

**Theorem 4.4** (Approximation Ratio)**.** *A clustering $\mathcal{C}$ obtained via* WKKM++ *satisfies*

$$E[J(\mathcal{C})] \le 8(\log K+2)J_{OPT}.$$

*Proof.* Exactly follows proof structure in Section 2. □

**Remark 4.1** (Implications)**.** WKKM++ has two important implications. First, by exploiting the equivalence between several graph-cut criteria and WKKM [20], one can hope to obtain better graph-cuts using WKKM++. A formal proof of this observation however remains an open problem. Second, the connection of WKKM to semi-supervised graph-clustering [30] leads to a potentially improved algorithm for semi-supervised clustering.

## 5 Approximation Algorithms for Tensor Clustering and Co-clustering

In its simplest formulation, co-clustering refers to the simultaneous partitioning of the rows and columns of the input data matrix into $K \times L$ co-clusters (sub-matrices). Co-clustering, also called bi-clustering [15, 27] has witnessed increasing interest over the years (see [10] and references therein). Anagnostopoulos et al. [4] seem to be the first to present an approximation algorithm for co-clustering based on a minimum-sum squared residue criterion of [16]. They (i.e., [4]) posed two open questions:

- Could one extend their ideas to obtain approximation algorithms for Bregman co-clustering?
- Could one design approximation algorithms for 3-way co-clustering of a tensor in $\mathbb{R}^{n_1\times n_2\times n_3}$?

Below we answer both these questions in the affirmative, leading to the first (to our knowledge) approximation algorithms for Bregman matrix and tensor co-clustering. In fact, our results hold for arbitrary $m$-way tensor co-clustering, not just the 3-way case.

We directly develop an approximation algorithm for $m$-way tensor co-clustering (hereafter *clustering*) that yields an approximation algorithm for Bregman co-clustering, which is nothing but tensor clustering for $m = 2$. Specifically, we show a competitiveness of $O(m \log K)$ for $m$-way Bregman co-clustering.

Tensors are well-studied in multilinear algebra [24], but they are not so widespread in the machine learning community. Therefore, to facilitate an easier understanding of our proofs, we briefly summarize some important tensor notation below for those unfamiliar to it.

### 5.1 Background on Tensors

Most of the material in this section is taken from the well-written paper of de Silva and Lim [18], whose notation turns out to be particularly suitable for our analysis. An order-$m$ tensor $\boldsymbol{A}$ may be viewed as an element of the vector space $\mathbb{R}^{n_1 \times \ldots \times n_m}$. A particular component of the tensor $\boldsymbol{A}$ is represented by the multiply-indexed value $a_{i_1 i_2 \ldots i_m}$, where $i_j = 1 \ldots n_j$ for $1 \leq j \leq m$.

**Multilinear matrix multiplication.** The most important operation that we do with tensors is that of multilinear matrix multiplication, which is a generalization of the familiar concept of matrix multiplication. Matrices *act* on other matrices by either left or right multiplications. For an order-3 tensor, there are three dimensions along which a matrix may act via matrix multiplication. For example, given an order-3 tensor $\boldsymbol{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, and three matrices $\boldsymbol{P} \in \mathbb{R}^{p_1 \times n_1}$, $\boldsymbol{Q} \in \mathbb{R}^{p_2 \times n_2}$, and $\boldsymbol{R} \in \mathbb{R}^{p_3 \times n_3}$, *multilinear matrix multiplication* is the operation defined by the action of these three matrices on the different dimensions of $\boldsymbol{A}$ that yields the tensor $\boldsymbol{A}' \in \mathbb{R}^{p_1 \times p_2 \times p_3}$. Formally, the entries of the tensor $\boldsymbol{A}'$ are given by

$$a'_{lmn} = \sum_{i,j,k=1}^{n_1,n_2,n_3} p_{li} q_{mj} r_{nk} a_{ijk}, \tag{5.1}$$

and this operation is written compactly as

$$\boldsymbol{A}' = (\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}) \cdot \boldsymbol{A}. \tag{5.2}$$

The notation (5.2) is particularly nice and may be viewed as the *group action* of $(\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R})$ on $\boldsymbol{A}$ (group-action refers to the situation when a group with a particular algebraic structure "acts" on another set; for the multilinear multiplication notation on can view it as the set $G = \mathbb{R}^{p_1 \times n_1} \times \mathbb{R}^{p_2 \times n_2} \times \mathbb{R}^{p_3 \times n_3}$ acting on the set $X = \mathbb{R}^{n_1 \times n_2 \times n_3}$). Addition in $G$ is defined entry-wise:

$$(\boldsymbol{P}_1, \boldsymbol{Q}_1, \boldsymbol{R}_1) + (\boldsymbol{P}_2, \boldsymbol{Q}_2, \boldsymbol{R}_2) = (\boldsymbol{P}_1 + \boldsymbol{P}_2, \boldsymbol{Q}_1 + \boldsymbol{Q}_2, \boldsymbol{R}_1 + \boldsymbol{R}_2).$$

Multilinear multiplication extends naturally to tensors of arbitrary (finite) order. If $\boldsymbol{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_m}$, and $\boldsymbol{P}_1 \in \mathbb{R}^{p_1 \times n_1}, \ldots, \boldsymbol{P}_m \in \mathbb{R}^{p_m \times n_m}$, then $\boldsymbol{A}' = (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_m) \cdot \boldsymbol{A}$ has entries

$$a'_{i_1 i_2 \ldots i_m} = \sum_{j_1, \ldots, j_m = 1}^{n_1, \ldots, n_m} p^{(1)}_{i_1 j_1} \cdots p^{(m)}_{i_m j_m} a_{j_1 \ldots j_m}, \tag{5.3}$$

where $p^{(k)}_{ij}$ denotes the $ij$-entry of matrix $\boldsymbol{P}_k$.

**Example 5.1** (Matrix Multiplication)**.** *Let $\boldsymbol{A} \in \mathbb{R}^{n_1 \times n_2}$, $\boldsymbol{P} \in \mathbb{R}^{p \times n_1}$, and $\boldsymbol{Q} \in \mathbb{R}^{q \times n_2}$ be given. The matrix product $\boldsymbol{P}\boldsymbol{A}\boldsymbol{Q}^T$ can be written as the multilinear multiplication $(\boldsymbol{P}, \boldsymbol{Q}) \cdot \boldsymbol{A}$.*

**Example 5.2** (Basic Properties)**.** *The following properties of multilinear multiplication are easily verified (and generalized to tensors of arbitrary order):*

1. ***Linearity:*** *Let $\alpha, \beta \in \mathbb{R}$, and $\boldsymbol{A}$ and $\boldsymbol{B}$ be tensors with same dimensions, then*

$$(\boldsymbol{P}, \boldsymbol{Q}) \cdot (\alpha \boldsymbol{A} + \beta \boldsymbol{B}) = \alpha (\boldsymbol{P}, \boldsymbol{Q}) \cdot \boldsymbol{A} + \beta (\boldsymbol{P}, \boldsymbol{Q}) \cdot \boldsymbol{B}$$

2. ***Product rule:*** *For matrices $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, $\boldsymbol{Q}_1$, $\boldsymbol{Q}_2$ of appropriate dimensions, and a tensor $\boldsymbol{A}$ we have*

$$(\boldsymbol{P}_1, \boldsymbol{P}_2) \cdot ((\boldsymbol{Q}_1, \boldsymbol{Q}_2) \cdot \boldsymbol{A}) = (\boldsymbol{P}_1 \boldsymbol{Q}_1, \boldsymbol{P}_2 \boldsymbol{Q}_2) \cdot \boldsymbol{A}$$

3. ***Multilinearity:*** *Let $\alpha, \beta \in \mathbb{R}$, and $\boldsymbol{P}$, $\boldsymbol{Q}$, and $\boldsymbol{R}$ be matrices of appropriate dimensions. Then, for a tensor $\boldsymbol{A}$ the following holds*

$$(\boldsymbol{P}, \alpha \boldsymbol{Q} + \beta \boldsymbol{R}) \cdot \boldsymbol{A} = \alpha (\boldsymbol{P}, \boldsymbol{Q}) \cdot \boldsymbol{A} + \beta (\boldsymbol{P}, \boldsymbol{R}) \cdot \boldsymbol{A}$$

**Inner Product:** The Frobenius norm induces an inner-product that can be defined as

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle = \sum\nolimits_{i_1,\ldots,i_m} a_{i_1\ldots i_m} b_{i_1\ldots i_m}, \tag{5.4}$$

so that $\|\boldsymbol{A}\|_{\mathrm{F}}^2 = \langle \boldsymbol{A}, \boldsymbol{A} \rangle$ holds as usual. The following property of this inner product is easily verified

$$\left\langle (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_m) \cdot \boldsymbol{A}, (\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m) \cdot \boldsymbol{B} \right\rangle = \left\langle \boldsymbol{A}, (\boldsymbol{P}_1^T \boldsymbol{Q}_1, \ldots, \boldsymbol{P}_m^T \boldsymbol{Q}_m) \cdot \boldsymbol{B} \right\rangle. \tag{5.5}$$

*Proof:* Using definition (5.3) along with (5.4) we have

$$\begin{aligned}
\left\langle (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_m) \cdot \boldsymbol{A}, (\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m) \cdot \boldsymbol{B} \right\rangle &= \sum_{i_1,\ldots,i_m} \sum_{\substack{j_1,\ldots,j_m \\ k_1,\ldots,k_m}} p^{(1)}_{i_1 j_1} q^{(1)}_{i_1 k_1} \cdots p^{(m)}_{i_m j_m} q^{(m)}_{i_m k_m} a_{j_1\ldots j_m} b_{k_1\ldots k_m}, \\
&= \sum_{\substack{j_1,\ldots,j_m \\ k_1,\ldots,k_m}} \Big(\sum_{i_1} p^{(1)}_{i_1 j_1} q^{(1)}_{i_1 k_1}\Big) \cdots \Big(\sum_{i_m} p^{(m)}_{i_m j_m} q^{(m)}_{i_m k_m}\Big) a_{j_1\ldots j_m} b_{k_1\ldots k_m} \\
&= \sum_{\substack{j_1,\ldots,j_m \\ k_1,\ldots,k_m}} (\boldsymbol{P}_1^T \boldsymbol{Q}_1)_{j_1 k_1} \cdots (\boldsymbol{P}_m^T \boldsymbol{Q}_m)_{j_m k_m} a_{j_1\ldots j_m} b_{k_1\ldots k_m} = \sum_{j_1\ldots j_m} a_{j_1\ldots j_m} b'_{j_1\ldots j_m} = \langle \boldsymbol{A}, \boldsymbol{B}' \rangle,
\end{aligned}$$

where $\boldsymbol{B}' = (\boldsymbol{P}_1^T \boldsymbol{Q}_1, \ldots, \boldsymbol{P}_m^T \boldsymbol{Q}_m) \cdot \boldsymbol{B}$.

### 5.2 Tensor clustering

Given the background above, we are now ready to formally state the Bregman tensor clustering problem.

Let $\boldsymbol{A} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$ be an order-$m$ tensor. Tensor clustering refers to a partitioning of $\boldsymbol{A}$ into sub-tensors or simply clusters, so that the entries of each cluster are as coherent as possible. The goal of (one simple version) of Bregman tensor clustering is to partition $\boldsymbol{A}$ into sub-tensors so that the sum of the Bregman divergences of individual elements in the sub-tensor to their corresponding cluster representatives is minimized. The cluster representatives turn out to be simply the means of the associated sub-tensors because we are minimizing Bregman divergences.

A cluster (sub-tensor) is indexed by subsets of indices along each dimension. Let $I_j \subseteq \{1, \ldots, n_j\}$ denote such an index subset for dimension $j$. Then the cluster representative corresponding to a sub-tensor is simply its mean, i.e.

$$M_{I_1\ldots I_m} = \frac{1}{|I_1| \cdots |I_m|} \sum_{i_1 \in I_1, \ldots, i_m \in I_m} a_{i_1\ldots i_m}. \tag{5.6}$$

Assuming that each dimension $j$ is partitioned into $k_j$ clusters, we can collect all the different representatives (each of which can be written in the form (5.6)) into a *means tensor* $\boldsymbol{M} \in \mathbb{R}^{k_1 \times \ldots \times k_m}$. Thus, we have a total of $\prod_j k_j$ tensor clusters. Let $\overline{\boldsymbol{C}}_j \in \{0,1\}^{n_j \times k_j}$ denote the cluster indicator matrix for dimension $j$. In such a matrix, entry $i$ of column $k$ is one if and only if $i$ is in the $k$th index set for tensor dimension $j$.

Given this notation, we can now formally state the Bregman tensor clustering problem:

$$\underset{\overline{\boldsymbol{C}}_1,\ldots,\overline{\boldsymbol{C}}_m}{\text{minimize}} \quad B_f(\boldsymbol{A}, (\overline{\boldsymbol{C}}_1, \ldots, \overline{\boldsymbol{C}}_m) \cdot \boldsymbol{M}), \qquad \text{s.t. } \overline{\boldsymbol{C}}_j \in \{0,1\}^{n_j \times k_j}. \tag{5.7}$$

Problem (5.7) can be rewritten in a more useful form. To that end, let $\boldsymbol{C}_j$ be the normalized cluster indicator matrix obtained from $\overline{\boldsymbol{C}}_j$ by normalizing the columns to have unit-norm (i.e., $\boldsymbol{C}_j^T \boldsymbol{C}_j = \boldsymbol{I}_{k_j}$). Then (5.7) may be rewritten as

$$\underset{\boldsymbol{C}_1,\ldots,\boldsymbol{C}_m}{\text{minimize}} \quad J(\mathcal{C}) = B_f(\boldsymbol{A}, (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_m) \cdot \boldsymbol{A}), \qquad \text{where } \boldsymbol{P}_j = \boldsymbol{C}_j \boldsymbol{C}_j^T, \tag{5.8}$$

In the sequel, we will refer to a clustering by its parametrizations via both indicator and projection matrices. A little remark on the side: since $\boldsymbol{C}_j^T \boldsymbol{C}_j = \boldsymbol{I}_{k_j}$, one can relax the "hard"-clustering constraints on $\boldsymbol{C}_j$ to just orthogonality constraints. Indeed, such relaxations form the basis of spectral-relaxations for Euclidean K-means as well as co-clustering [16]. However, we will not use such relaxations in this paper.

**Summary of the Algorithm:** Broadly speaking, our tensor clustering approximation algorithm is based on clustering along subsets of dimensions using a guaranteed approximation algorithm, and then combining the resulting clusterings to obtain tensor clusters. A particular example of such a scheme would be to cluster along single dimensions using a method such as BREG++. clustering. Note that clustering along a single dimension in a tensor is a generalization of clustering the one-dimensional sub-tensors, i.e. vectors, in a matrix. In a tensor, we form groups of $m-1$-way tensors, for instance, grouping matrices in a 3-way tensor. Thanks to the separability of our Bregman divergences, BREG++directly extends to sub-tensor objects. Taking the 3-way example with sub-matrices $\boldsymbol{A}, \boldsymbol{B}$, recall that $B_f(\boldsymbol{A}, \boldsymbol{B}) = B_f(\mathrm{vec}(\boldsymbol{A}), \mathrm{vec}(\boldsymbol{B}))$. So we simply treat the sub-tensors as vectors.

Our analysis below establishes that given a clustering algorithm that clusters along $t$ of the dimensions at a time with an approximation factor of $\alpha_t$, we can achieve an objective within $O(\lceil m/t \rceil \frac{\sigma_2}{\sigma_1} \alpha_t)$ of the optimal; the scaling factors $\sigma_1$ and $\sigma_2$ are defined as

$$\sigma_1 = \inf_{\substack{x \in \{a_{ij}\} \\ y \in \mathrm{conv}(\{a_{ij}\})}} \frac{B_f(x,y)}{(x-y)^2}, \qquad \sigma_2 = \sup_{\substack{x \in \{a_{ij}\} \\ y \in \mathrm{conv}(\{a_{ij}\})}} \frac{B_f(x,y)}{(x-y)^2}. \tag{5.9}$$

**Note:** For simplicity of exposition we assume that we cluster an order-$m$ tensor along $t$ dimensions at a time and to eventually combine the resulting $m/t$ sub-clusterings. Our analysis can be generalized (at the expense of laborious algebra) to the case where we cluster along *partitions* of varying sizes, say $\{t_1, \ldots, t_r\}$, where $t_1 + \cdots + t_r = m$.

### 5.2.1 Analysis

In this section we prove our tensor clustering approximation theorem, which yields as corollaries efficient approximation algorithms based on BREG++ for both Bregman co-clustering, and Bregman tensor clustering.

**Theorem 5.3** (Approximation guarantee)**.** *Let $\boldsymbol{A}$ be the input order-$m$ tensor, and let $\mathcal{C}_j$ denote the clustering of $\boldsymbol{A}$ along the $j$th subset of $t$ dimensions ($1 \leq j \leq m/t$), as obtained by a multiway clustering algorithm with guarantee[3] $\alpha_t$. Let $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_{m/t})$ denote the induced tensor clustering. Then[4]*

$$J(\mathcal{C}) \leq 2^{\log(m/t)} \frac{\sigma_2}{\sigma_1} \alpha_t J_{OPT}(m)$$

**Corollary 5.4** (Approximation with BREG++)**.** *Let $\boldsymbol{A}$ be the input order-$m$ tensor, and let $\mathcal{C}_j$ denote the clustering of $\boldsymbol{A}$ along dimension $j$ ($1 \leq j \leq m$), as obtained via* BREG++*. Let $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_m)$ denote the induced tensor clustering. Then,*

$$E[J(\mathcal{C})] \leq 4m \frac{\sigma_2^2}{\sigma_1^2} \left(1 + \frac{\sigma_2}{\sigma_1}\right) (\log K^* + 2) J_{OPT}(m),$$

*where $K^* = \max_{1 \leq j \leq m} k_j$ is the maximum number of clusters across all dimensions.*

[3] By "guarantee $\alpha$", we mean that the algorithm yields a solution that is guaranteed to have an objective value within a factor of $O(\alpha_t)$ of the optimum.

[4] Here and in the sequel, the argument $m$ to $J_{\mathrm{OPT}}$ denotes the best $m$-way clustering to avoid confusions about dimensions.

To establish Theorem 5.3, we will first bound the quality of a combination of dimension-wise clusterings for the Frobenius norm, with the help of the Pythagorean Property (Lemma 5.5). It is clear that compressing along only a subset of dimensions achieves lower divergence than clustering along all dimensions. Generalizing an idea of [4], we upper bound the full combined clustering in terms of the (approximately) optimal clustering along a subset of dimensions (Prop. 5.6). Finally, we extend this upper bound to general Bregman divergences and relate it to the optimal tensor clustering.

In the analysis below we assume without loss of generality that $m = 2^h t$ for an integer $h$ (otherwise, pad in empty dimensions). We assume that we have access to an algorithm that can cluster along a subset of $t$ dimensions, while achieving an objective function within a factor $\alpha_t$ of the optimal (for those $t$ dimensions), i.e., $\alpha_t J_{\text{OPT}}(t)$. For example, when $t = 1$ we can use BREG++ (or in theory, even the approximation algorithms of [1]).

**Lemma 5.5** (Pythagorean Property)**.** *Let $\mathscr{P} = (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_t)$, $\mathscr{Q} = (\boldsymbol{P}_{t+1}, \ldots, \boldsymbol{P}_m)$, and $\mathscr{P}^\perp = (\boldsymbol{I} - \boldsymbol{P}_1, \ldots, \boldsymbol{I} - \boldsymbol{P}_t)$ be combinations of projection matrices $\boldsymbol{P}_j$. Then*

$$\|(\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A} + (\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B}\|^2 = \|(\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A}\|^2 + \|(\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B}\|^2, \tag{5.10}$$

*where $\mathscr{R}$ is some arbitrary combination of $m - t$ projection matrices.*

*Proof.* Using the inner-product (5.4) we can rewrite (5.10) as

$$\|(\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A} + (\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B}\|^2 = \|(\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A}\|^2 + \|(\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B}\|^2 + 2\left\langle (\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A}, (\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B} \right\rangle.$$

With (5.5) the latter term simplifies to

$$\left\langle (\mathscr{P}, \mathscr{Q}) \cdot \boldsymbol{A}, (\mathscr{P}^\perp, \mathscr{R}) \cdot \boldsymbol{B} \right\rangle = \left\langle \boldsymbol{A}, (\mathscr{P}^T \mathscr{P}^\perp, \mathscr{Q}^T \mathscr{R}) \cdot \boldsymbol{B} \right\rangle = 0,$$

thus yielding the claim (5.10). □

**Some more notation.** Before diving into the proofs, we outline some more useful notation. Since we can only cluster along $t$ dimensions at a time, we recursively half the initial set of $m$ dimensions until, after $\log(m/t) + 1$ recursions, the sets have length $t$. Let $l$ denote the level of recursion, starting at $l = \log(m/t) = h$ down to $l = 0$, where the sets have length $t$. At level $l$, the sets will have length $2^l t$. Each clustering along a subset of $2^l t$ dimensions is represented by the corresponding $2^l t$ projection matrices. We denote their combination by $\mathscr{P}_i^l$. At level $l$, $i$ ranges from 1 to $2^{h-l}$.

For illustration, consider an order-8 tensor, and $t = 2$. Then $h = \log(m/t) = 2$, so we will need 3 levels. For simplicity, we always partition the set of dimensions in the middle, i.e. $\{1, \ldots, 8\}$ into $\{1, \ldots, 4\}$ and $\{5, \ldots, 8\}$ and so on, ending with $\{\{1,2\}, \{3,4\}, \{5,6\}, \{7,8\}$. The projection matrix for dimension $i$ is $\boldsymbol{P}_i$. The full tensor clustering is $(\boldsymbol{P}_1, \ldots, \boldsymbol{P}_8\}$. So here we get

$$\begin{array}{ccccc} & & \mathscr{P}_1^2 = (\boldsymbol{P}_1, \boldsymbol{P}_2, \boldsymbol{P}_3, \boldsymbol{P}_4, \boldsymbol{P}_5, \boldsymbol{P}_6, \boldsymbol{P}_7, \boldsymbol{P}_8) & & \\ & \mathscr{P}_1^1 = (\boldsymbol{P}_1, \boldsymbol{P}_2, \boldsymbol{P}_3, \boldsymbol{P}_4), & & \mathscr{P}_2^1 = (\boldsymbol{P}_5, \boldsymbol{P}_6, \boldsymbol{P}_7, \boldsymbol{P}_8) & \\ \mathscr{P}_1^0 = (\boldsymbol{P}_1, \boldsymbol{P}_2), & \mathscr{P}_2^0 = (\boldsymbol{P}_3, \boldsymbol{P}_4), & \mathscr{P}_3^0 = (\boldsymbol{P}_5, \boldsymbol{P}_6), & \mathscr{P}_4^0 = (\boldsymbol{P}_7, \boldsymbol{P}_8) & \end{array}$$

To represent a clustering of the tensor along only a subset of dimensions, we pad the corresponding $\mathscr{P}_i^l$ with $m - 2^l t$ identity matrices for the non-clustered dimensions. We refer to this padded collection as $\mathscr{Q}_i^l$. In the example above, e.g. $\mathscr{Q}_1^0 = (\boldsymbol{P}_1, \boldsymbol{P}_2, \boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I})$, $\mathscr{Q}_2^1 = (\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I}, \boldsymbol{I}, \mathscr{P}_2^1)$, and $\mathscr{Q}_1^2 = \mathscr{P}_1^2$. With recursive partitions of the dimensions, $\mathscr{Q}_i^l$ subsumes $\mathscr{Q}_j^0$ for $2^l(i-1) < j \leq 2^l i$: $\mathscr{Q}_i^l = \sum_{j=2^l(i-1)}^{2^l i} \mathscr{Q}_j^0$. The algorithm for the subsets of dimensions will yield the $\mathscr{Q}_i^0$ and $\mathscr{P}_i^0$. The remaining clusterings are simply combinations of those level-0 clusterings. Finally, we refer to the collection of $m - 2^l t$ identity matrices (for simplicity, we assume that they have the correct dimensionalities) as $\mathscr{I}^l$, so, for instance, $\mathscr{Q}_1^l = (\mathscr{P}_1^l, \mathscr{I}^l)$.

Note that the order of the dimensions is arbitrary, as long as the index sets remain the same and we reorder the dimensions of all tensors and matrices correspondingly. Hence, we always shift the identity matrices to the back for "ease" of notation. Furnished with this notation, we can now turn towards the details of the proofs. We start with the relation of the combined clustering to a subclustering with the Frobenius norm objective function.

**Proposition 5.6.** *Let $\boldsymbol{A}$ be an order-$m$ tensor and $m \geq 2^l t$. The objective function for any $2^l t$-way clustering $\mathscr{P}_1^l = (\mathscr{P}_1^0, \ldots, \mathscr{P}_{2^l}^0)$ can be bounded via the subclusterings along only one set of dimensions of size $t$:*

$$\|\boldsymbol{A} - \mathscr{Q}_1^l \cdot \boldsymbol{A}\|^2 = \|\boldsymbol{A} - (\mathscr{P}_1^l, \mathscr{I}_1^l) \cdot \boldsymbol{A}\|^2 \leq \max_{1 \leq j \leq 2^l} 2^l \|\boldsymbol{A} - \mathscr{Q}_j^0 \cdot \boldsymbol{A}\|^2. \tag{5.11}$$

Not that the Proposition actually holds for any set of $2^l$ sub-clusterings by permuting dimensions accordingly:

$$\|\boldsymbol{A} - \mathscr{Q}_i^l \cdot \boldsymbol{A}\|^2 \leq \max_{2^l(i-1) < j \leq 2^l i} 2^l \|\boldsymbol{A} - \mathscr{Q}_j^0 \cdot \boldsymbol{A}\|^2 \tag{5.12}$$

We will prove the proposition for $i = 1$ for ease of notation. If $m = 2^h t$ then the factor is $2^h = m/t$. (If $m/t$ is not a power of 2, then we get the factor with $h = \lceil \log(m/t) \rceil$).

*Proof.* We prove the proposition by induction on $l$.

*Base case:* Let $l = 0$. Then $\mathscr{Q}_j^0 = \mathscr{P}_1^0 = \mathscr{P}_1^l$ and the claim holds trivially.

*Induction step:* Assume the claim holds for $l \geq 0$. Let us look at a clustering $\mathscr{P}_1^{l+1} = (\mathscr{P}_1^l, \mathscr{P}_2^l)$. Then $\boldsymbol{A}$ decomposes as

$$\boldsymbol{A} = \Big( (\mathscr{P}_1^l, \mathscr{P}_2^l, \mathscr{I}^{l+1}) + ((\mathscr{P}_1^l)^\perp, \mathscr{P}_2^l, \mathscr{I}^{l+1}) + (\mathscr{P}_1^l, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) + ((\mathscr{P}_1^l)^\perp, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \Big) \cdot \boldsymbol{A},$$

whereby the Pythagorean Property 5.5 yields

$$\|\boldsymbol{A} - (\mathscr{P}_1^{l+1}, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 = \|\boldsymbol{A} - (\mathscr{P}_1^l, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 \tag{5.13}$$

$$= \|((\mathscr{P}_1^l)^\perp, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 + \|(\mathscr{P}_1^l, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 + \|((\mathscr{P}_1^l)^\perp, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2, \tag{5.14}$$

Assuming without loss of generality that

$$\|((\mathscr{P}_1^l)^\perp, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 \geq \|\mathscr{P}_1^l, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|,$$

we have from (5.13) and (5.14) the following inequalities

$$\begin{aligned}
\|\boldsymbol{A} - (\mathscr{P}_1^l, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 & \\
&\leq 2(\|((\mathscr{P}_1^l)^\perp, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 + \|((\mathscr{P}_1^l)^\perp, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2) \\
&= 2\|((\mathscr{P}_1^l)^\perp, \mathscr{P}_2^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A} + ((\mathscr{P}_1^l)^\perp, (\mathscr{P}_2^l)^\perp, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 \\
&= 2\|((\mathscr{P}_1^l)^\perp, \mathscr{I}^l, \mathscr{I}^{l+1}) \cdot \boldsymbol{A}\|^2 \\
&= 2\|\boldsymbol{A} - \mathscr{Q}_1^l \cdot \boldsymbol{A}\|^2 \\
&\leq 2 \max_{1 \leq j \leq 2} \|\boldsymbol{A} - \mathscr{Q}_j^l \cdot \boldsymbol{A}\|^2 \\
&\leq 2 \cdot 2^l \max_{1 \leq j \leq 2^{l+1}} \|\boldsymbol{A} - \mathscr{Q}_j^0 \cdot \boldsymbol{A}\|^2,
\end{aligned}$$

where the last step follows from the induction hypothesis and (5.12), and the two norm terms are combined using the Pythagorean Property. □

The proof of Theorem 5.3 generalizes Proposition 5.6 to arbitrary Bregman divergences, and then relates the objective for the sub-clustering to the objective for the full $m$-way clustering.

*Proof.* (Theorem 5.3.) Let $m = 2^h t$. Via the approximation algorithm with guarantee $\alpha_t$, we cluster the subsets of size $t$ to obtain $\mathscr{P}_i^0$. Let $\check{\mathscr{Q}}_i^0$ be the optimal subclustering of dimension set $i$, i.e. the result that $\mathscr{Q}_i^0$ would be if $\alpha_t = 1$. With $\sigma_1$ and $\sigma_2$ from (5.9), we can bound the combination $\mathscr{P}_1^h$ of the sub-clusterings $\mathscr{P}_i^0$:

$$\begin{aligned} B_f(\boldsymbol{A}, \mathscr{P}_1^h \cdot \boldsymbol{A}) &= \sum_{i_1,\dots,i_m} B_f(a_{i_1,\dots,i_m}, \mu_{I_1(i_1),\dots,I_m(i_m)}) \\ &\le \sigma_2 \sum_{i_1,\dots,i_m} (a_{i_1,\dots,i_m} - \mu_{I_1(i_1),\dots,I_m(i_m)})^2 = \sigma_2 \|\boldsymbol{A} - \mathscr{P}_1^h \cdot \boldsymbol{A}\|^2 \\ &\le 2^h \sigma_2 \max_j \|\boldsymbol{A} - \mathscr{Q}_j^0 \cdot \boldsymbol{A}\|^2 \qquad (5.15) \\ &\le 2^h \sigma_2 \sigma_1^{-1} \max_j B_f(\boldsymbol{A}, \mathscr{Q}_j^0 \cdot \boldsymbol{A}) \le 2^h \sigma_2 \sigma_1^{-1} \alpha_t \max_j B_f(\boldsymbol{A}, \check{\mathscr{Q}}_j^0 \cdot \boldsymbol{A}). \qquad (5.16) \end{aligned}$$

Inequality (5.15) follows from Proposition 5.6, and (5.16) from the guarantee of the algorithm we used to get the separate sub-clustering $\mathscr{Q}_j^0$.

Let us look at the relation between on optimal clustering $\check{\mathscr{Q}}^0$ of an arbitrary subset of $t$ dimensions and the optimal full tensor clustering $\check{\check{\mathscr{Q}}}^0$ of all $2^h t$ dimensions. Let $\overline{\check{\boldsymbol{C}}}_j$ be the cluster indicator matrices of the clustered dimensions in $\check{\mathscr{Q}}^0$, and $\check{\boldsymbol{C}}_j$ their normalized versions such that $\check{\mathscr{Q}}^0 = (\check{\boldsymbol{C}}_1, \dots, \check{\boldsymbol{C}}_t, \mathscr{I}^0)(\check{\boldsymbol{C}}_1, \dots, \check{\boldsymbol{C}}_t, \mathscr{I}^0)^\top$[5]. By definition, $\check{\mathscr{Q}}^0$ solves

$$\underset{\overline{\boldsymbol{C}}_1,\dots,\overline{\boldsymbol{C}}_t,\boldsymbol{M}}{\text{minimize}} \quad B_f(\boldsymbol{A}, (\overline{\boldsymbol{C}}_1, \dots, \overline{\boldsymbol{C}}_t, \mathscr{I}^0) \cdot \boldsymbol{M}), \qquad \text{s.t. } \overline{\boldsymbol{C}}_j \in \{0,1\}^{n_j \times k_j},$$

with $(\overline{\check{\boldsymbol{C}}}_1, \dots, \overline{\check{\boldsymbol{C}}}_t, \mathscr{I}^0) \cdot \check{\boldsymbol{M}} = \check{\mathscr{Q}}^0 \cdot \boldsymbol{A}$. In that respect, $\check{\mathscr{Q}}^0$ even beats the sub-clustering $(\check{\check{\boldsymbol{C}}}_1, \dots, \check{\check{\boldsymbol{C}}}_t)$ taken from the optimal full $m$-way clustering $\check{\check{\mathscr{Q}}}_1^h = \check{\check{\mathscr{P}}}_1^h$, i.e.

$$\begin{aligned} &B_f\Big(\boldsymbol{A}, (\check{\boldsymbol{C}}_1, \dots, \check{\boldsymbol{C}}_t, \mathscr{I}^0) \cdot ((\check{\boldsymbol{C}}_1, \dots, \check{\boldsymbol{C}}_t, \mathscr{I}^0)^\top \cdot \boldsymbol{A})\Big) \le \min_{\boldsymbol{B}} B_f(\boldsymbol{A}, (\check{\check{\boldsymbol{C}}}_1, \dots, \check{\check{\boldsymbol{C}}}_t, \mathscr{I}^0) \cdot \boldsymbol{B}) \\ &\le B_f\left(\boldsymbol{A}, (\check{\check{\boldsymbol{C}}}_1, \dots, \check{\check{\boldsymbol{C}}}_t, \mathscr{I}^0)(I, \dots, I, \check{\check{\boldsymbol{C}}}_{t+1}, \dots, \check{\check{\boldsymbol{C}}}_m) \cdot \check{\check{\boldsymbol{M}}}\right) = B_f\left(\boldsymbol{A}, \check{\check{\mathscr{P}}}_1^h \cdot \boldsymbol{A}\right), \end{aligned}$$

where $\check{\check{\boldsymbol{M}}}$ is the tensor of means for the optimal $m$-way clustering, $(\check{\check{\boldsymbol{C}}}_1, \dots, \check{\check{\boldsymbol{C}}}_m) \cdot \check{\check{\boldsymbol{M}}} = \check{\check{\mathscr{P}}}_1^h \cdot \boldsymbol{A}$. Combining this bound with (5.16) yields the final bound for the combined clustering $\mathscr{P}_1^h$,

$$B_f(\boldsymbol{A}, \mathscr{P}_1^h \cdot \boldsymbol{A}) \le 2^h \sigma_2 \sigma_1^{-1} \alpha_t B_f(\boldsymbol{A}, \check{\check{\mathscr{P}}}_1^h \cdot \boldsymbol{A}) = 2^h \sigma_2 \sigma_1^{-1} \alpha_t J_{\text{OPT}}(m),$$

and completes the proof of the theorem. □

Special cases of Theorem 5.3 are Euclidean $m$-way co-clustering and Bregman co-clustering.

**Corollary 5.7** ($m$-way Euclidean tensor clustering)**.** *For $f = \frac{1}{2}x^2$, we have $\sigma_1 = \sigma_2 = 1$. Thus, using* KMEANS++ *as the base algorithm for obtaining the combined clustering $\mathscr{P}_1^h$ leads to the approximation guarantee:*

$$E[J(\mathscr{P}_1^h)] \le 8m(\log K^* + 2) J_{OPT}(m).$$

**Corollary 5.8** (Co-clustering)**.** *Let $\mathcal{C}_R, \mathcal{C}_C$ be the clusterings of the rows and columns of $\boldsymbol{A}$ obtained via* BREG++*, and $(\mathcal{C}_R, \mathcal{C}_C)$ the induced co-clustering. Then the expected objective value $J$ is bounded as*

$$E[J(C_R, \mathcal{C}_C)] \le 8\frac{\sigma_2^2}{\sigma_1^2}\left(1 + \frac{\sigma_2}{\sigma_1}\right)(\log K^* + 2) J_{OPT}(2),$$

*where $\sigma_1$ and $\sigma_2$ are as defined in Equation 5.9.*

[5] Without loss of generality, we again assume that the clustered dimensions are the first $t$. Otherwise, permute the dimensions

## References


[1] M. R. Ackermann and Johannes Blömer. Coresets and Approximate Clustering for Bregman Divergences. In *Proc. 20th ACM-SIAM Symposium on Discrete Algorithms (SODA '09)*, 2009. To appear.

[2] M. R. Ackermann, J. Blomer, and C. Sohler. Clustering for metric and non-metric distance measures. In *ACM-SIAM SODA*, April 2008.

[3] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie. Beyond pairwise clustering. In *IEEE CVPR*, 2005.

[4] A. Anagnostopoulos, A. Dasgupta, and R. Kumar. Approximation algorithms for co-clustering. In *PODS*, 2008.

[5] D. Arthur and S. Vassilvitskii. `k-means++`: The Advantages of Careful Seeding. In *SODA*, pages 1027–1035, 2007.

[6] L. D. Baker and A. McCallum. Distributional clustering for text classification. In *Proc. SIGIR*, 1998.

[7] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the Unit Hypersphere using von Mises-Fisher Distributions. *JMLR*, 6:1345–1382, Sep 2005.

[8] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with Bregman Divergences. *JMLR*, 6(6):1705–1749, October 2005.

[9] A. Banerjee, S. Basu, and S. Merugu. Multi-way Clustering on Relation Graphs. In *SIAM Data Mining*, 2007.

[10] A. Banerjee, I. S. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A Generalized Maximum Entropy Approach to Bregman Co-clustering and Matrix Approximation. *JMLR*, 8:1919–1986, 2007.

[11] R. Bekkerman, R. El-Yaniv, and A. McCallum. Multi-way distributional clustering via pairwise interactions. In *ICML*, 2005.

[12] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[13] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

[14] K. Chaudhuri and A. McGregor. Finding metric structure in information theoretic clustering. In *Conf. on Learning Theory, COLT*, July 2008.

[15] Y. Cheng and G. Church. Biclustering of expression data. In *Proceedings ISMB*, pages 93–103. AAAI Press, 2000.

[16] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum Sum Squared Residue based Co-clustering of Gene Expression data. In *Proc. 4th SIAM International Conference on Data Mining (SDM)*, pages 114–125, Florida, 2004. SIAM.

[17] W. Fernandez de la Vega, M. Karpinski, C. Kenyon, and Y. Rabani. Approximation schemes for clustering problems. In *STOC'03*, 2003.

[18] V. de Silva and L.-H. Lim. Tensor Rank and the Ill-Posedness of the Best Low-Rank Approximation Problem. *SIAM J. on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

[19] I. S. Dhillon, S. Mallela, and R. Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *JMLR*, 3:1265–1287, March 2003.

[20] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted Graph Cuts without Eigenvectors A Multilevel Approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(11):1944–1957, 2007.

[21] Inderjit S. Dhillon, Yuqiang Guan, and J. Kogan. Iterative Clustering of High Dimensional Text Data Augmented by Local Search. *IEEE International Conference on Data Mining*, page 131, 2002.

[22] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine Learning*, 56:9–33, 2004.

[23] V. M. Govindu. A tensor decomposition for geometric grouping and segmentation. In *IEEE CVPR*, 2005.

[24] W. H. Greub. *Multilinear Algebra*. Springer, 1967.

[25] S. Har-Peled and S. Mazumdar. On coresets for $k$-means and $k$-median clustering. In *STOC'04*, 2004.

[26] J. A. Hartigan. *Clustering Algorithms*. Wiley, 1975.

[27] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, March 1972.

[28] M. Inaba, N. Katoh, and H. Imai. Applications of weighted voronoi diagrmas and randomization to variance based $k$-clustering. In *SCG'94: Proc. 10th Symp. on Comp. Geo.*, 1994.

[29] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Comput. Geom. Theory Appl.*, 28(2-3):89–112, 2004.

[30] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: a kernel approach. *Machine Learning*, 2008. To appear.

[31] A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time $(1 + \epsilon)$-approximation algorithms for k-means clustering in any dimensions. In *IEEE Symp. on Foundations of Comp. Sci.*, 2004.

[32] S. P. LLoyd. Least squares quantization in PCM. *IEEE Tran. on Inf. Theory*, 28(2):129–136, 1982.

[33] B. Long, X. Wu, and Z. Zhang. Unsupervised learning on k-partite graphs. In *SIGKDD*, 2006.

[34] R. R. Mettu and C. G. Plaxton. Optimal Time Bounds for Approximate Clustering. *Mach. Learn.*, 56(1-3):35–60, 2004.

[35] F. Nielsen, J.-D. Boissonnat, and R. Nock. On Bregman Voronoi Diagrams. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 746–755, 2007.

[36] R. Ostrovsky, Y. Rabani, L. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *FOCS*, 2006.

[37] F. C. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Meeting of the Assoc. for Comp. Linguistics*, 1993.

[38] K. Puolamäki, S. Hanhijärvi, and G. C. Garriga. An approximation ratio for biclustering. *Inf. Process. Lett.*, 108(2):45–49, 2008.

[39] R. T. Rockafellar. *Convex Analysis*. Princeton Univ. Press, 1970.

[40] A. Shashua, R. Zass, and T. Hazan. Multi-way Clustering Using Super-Symmetric Non-negative Tensor Factorization. *LNCS*, 3954:595–608, 2006.