Approximation Algorithms for Co-Clustering

Aris Anagnostopoulos
Yahoo! Research
701 First Ave.
Sunnyvale, CA 94089
aris@yahoo-inc.com

Anirban Dasgupta
Yahoo! Research
701 First Ave.
Sunnyvale, CA 94089
anirban@yahoo-inc.com

Ravi Kumar
Yahoo! Research
701 First Ave.
Sunnyvale, CA 94089
ravikuma@yahoo-inc.com

ABSTRACT

Co-clustering is the simultaneous partitioning of the rows and columns of a matrix such that the blocks induced by the row/column partitions are good clusters. Motivated by several applications in text mining, market-basket analysis, and bioinformatics, this problem has attracted severe attention in the past few years. Unfortunately, to date, most of the algorithmic work on this problem has been heuristic in nature.

In this work we obtain the first approximation algorithms for the co-clustering problem. Our algorithms are simple and obtain constant-factor approximation solutions to the optimum. We also show that co-clustering is NP-hard, thereby complementing our algorithmic result.

Categories and Subject Descriptors

F.2.0 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Algorithms

Keywords

Co-Clustering, Biclustering, Clustering, Approximation

1. INTRODUCTION

Clustering is a fundamental primitive in many data analysis applications, including information retrieval, databases, text and data mining, bioinformatics, market-basket analysis, and so on [10, 18]. The central objective in clustering is the following: given a set of points and a pairwise distance measure, partition the set into clusters such that points that are close to each other according to the distance measure occur together in a cluster and points that are far away from each other occur in different clusters. This objective sounds straightforward, but it is not easy to state a universal desiderata for clustering—Kleinberg showed in a reasonable axiomatic framework that clustering is an impossible problem to solve [19]. In general, the clustering objectives tend to be application-specific, exploiting the underlying structure in the data and imposing additional structure on the clusters themselves.

In several applications, the data itself has a lot of structure, which may be hard to capture using a traditional clustering objective. Consider the example of a Boolean matrix, whose rows correspond to keywords and the columns correspond to advertisers, and an entry is one if and only if the advertiser has placed a bid on the keyword. The goal is to cluster both the advertisers and the keywords. One way to accomplish this would be to independently cluster the advertisers and keywords using the standard notion of clustering—cluster similar advertisers and cluster similar keywords. However (even though for some criteria this might be a reasonable solution, as we argue subsequently in this work), such an endeavor might fail to elicit subtle structures that might exist in the data: perhaps, there are two disjoint sets of advertisers A_1, A_2 and keywords K_1, K_2 such that each advertiser in A_i bids on each keyword in K_j if and only if $i = j$. In an extreme case, may be there is a combinatorial decomposition of the matrix into blocks such that each block is either almost full or almost empty. To be able to discover such things, the clustering objective has to *simultaneously* intertwine information about both the advertisers and keywords that is present in the matrix. This is precisely achieved by *co-clustering* [14, 6]; other nomenclature for co-clustering include biclustering, bidimensional clustering, and subspace clustering.

In the simplest version of (k, ℓ)-co-clustering, we are given a matrix of numbers, and two integers k and ℓ. The goal is to partition the rows into k clusters and the columns into ℓ clusters such that the sum-squared deviation from the mean within each "block" induced by the row-column partition is minimized. This definition, along with different objectives, is made precise in Section 2. Co-clustering has received lots of attention in recent years, with several applications in text mining [8, 12, 29], market-basket data analysis, image, speech and video analysis, and bioinformatics [6, 7, 20]; see the recent paper by Banerjee et al. [4] and the survey by Madeira and Oliveira [22].

Even though co-clustering has been extensively studied in many application areas, very little is known about it from an algorithmic angle. Very special variants of co-clustering are known to be NP-hard [15]. A natural generalization of the k-means algorithm to co-clustering is known to converge [4].

PODS'08, June 9–12, 2008, Vancouver, BC, Canada.

Apart from these, most of the algorithmic work done on co-clustering has been heuristic in nature, with no proven guarantees of performance.

In this paper we address the problem of co-clustering from an algorithmic point of view.

Main contributions.

Our main contribution is the first constant-factor approximation algorithm for the (k, ℓ)-co-clustering problem. Our algorithm is simple and builds upon approximation algorithms for a variant of the k-median problem, which we call k-means$_p$. The algorithm works for any norm and produces a 3α-approximate solution, where α is the approximation factor for the k-means$_p$ problem; for the latter, we obtain a constant-factor approximation by extending the results of the k-median problem. We next consider the important special case of the Frobenius norm, and constant k, ℓ. For this, we obtain a $(\sqrt{2}+\epsilon)$-approximation algorithm by exploiting the geometry of the space, and results on the k-means problem.

We complement these results by considering the extreme cases of $\ell = 1$ and $\ell = n^\epsilon$, where the matrix is of size $m \times n$. We show that the $(k, 1)$-co-clustering problem can be solved exactly in time $O(mn + m^2k)$ and the (k, n^ϵ)-co-clustering problem is NP-hard, for $k \geq 2$ under the ℓ_1 norm.

Related work.

Research on clustering has a long and varied history, with work ranging from approximation algorithms to axiomatic developments of the objective functions [16, 10, 19, 18, 34, 13]. The problem of co-clustering itself has found growing applications in several practical fields, for example, simultaneously clustering words and documents in information retrieval [8], clustering genes and expression data for biological data analysis [6, 32], clustering users and products for recommendation systems [1], and so on. The exact objective function, and the corresponding definition of co-clustering varies, depending on the type of structure we want to extract from the data. The hardness of the co-clustering problem depends on the exact merit function to be used. In the simplest case, the co-clustering problem is akin to finding out a bipartite clique (or dense graph) that is known to be NP-hard even to approximate. Consequently, work on co-clustering has mostly focused on heuristics that work well in practice. Excellent references on such methods are the surveys by Madeira and Oliveira [22] and Tanay, Sharan and Shamir [30]. Dhillon et al. [4] unified a number of merit functions for the co-clustering problem under the general setting of Bregman divergences, and gave a k-means style algorithm that is guaranteed to monotonically decrease the merit function. Our objective function for the $p = 2$ case, in fact is exactly the $\|\cdot\|_F$ merit function for which their results apply.

There is little work along the lines of approximation algorithms for the co-clustering problems. The closest algorithmic work to this problem relates to finding cliques and dense bipartite subgraphs [24, 25]. These variants, are however, often hard even to approximate to within a constant factor. Hassanpour [15] shows that a version of the co-clustering problem that finds out homogeneous submatrices is hard and Feige shows that the problem of finding out the maximum biclique is hard to approximate to within $2^{(\log n)^\delta}$ [11].

Very recently, Puolamäki et al. [27] published results on the co-clustering problem for objective functions of the same form that we study. They analyze the same algorithm for two cases, the ℓ_1 norm for 0/1-valued matrices and the ℓ_2 norm for real-valued matrices. In the first case they obtain a better approximation factor than ours (2.414α as opposed to 3α, where α is the best approximation factor for one-sided clustering). On the other hand, our result is more general as it holds for any ℓ_p norm and for real-valued matrices. Their ℓ_2 result is the same as ours ($\sqrt{2}\alpha$-approximation) and their proof is similar (although presented differently).

Organization.

Sections 2 and 3 contain some background material. The problem of co-clustering is formally defined in Section 4. The algorithms for co-clustering are given in Section 5. The hardness result is shown in Section 6. Finally, Section 7 contains directions for future work.

2. SOME CO-CLUSTERING VARIANTS

In this section we mention briefly some of the variants of the objective functions that have been proposed in the co-clustering literature and are close to the ones we use in this work. Other commonly used objectives are based on information-theoretic quantities.

Let $A = \{a_{ij}\} \in \mathbb{R}^{m \times n}$ be the matrix that we want to co-cluster. A (k, ℓ)-co-clustering is a k-partitioning $\mathcal{I} = \{I_1, \ldots, I_k\}$ of the set of rows $\{1, \ldots, m\}$ and an ℓ-partitioning $\mathcal{J} = \{J_1, \ldots, J_\ell\}$ of the set of columns $\{1, \ldots, n\}$.

Cho et al. [7] define for every element a_{ij} that belongs to the (I, J)-co-cluster its *residue* as

$$h_{ij} = a_{ij} - a_{IJ}, \tag{1}$$

or

$$h_{ij} = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}, \tag{2}$$

where $a_{IJ} = \frac{1}{|I| \cdot |J|} \sum_{i \in I, j \in J} a_{ij}$ is the average of all the entries in the co-cluster, $a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}$ is the mean of all the entries in row i whose columns belong into J, and $a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$ is the mean of all the entries in column j whose rows belong into I.

Having defined the residues, the goal is to minimize some norm of the residue matrix $H = (h_{ij})$. The norm most commonly used in the literature is the Frobenius norm, $\|\cdot\|_F$, defined as the square root of the sum of the squares of the elements:

$$\|H\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} h_{ij}^2}.$$

One can attempt to minimize some other norm; for example, Yang et al. [33] minimize the norm

$$|||H|||_1 = \sum_{i=1}^{m} \sum_{j=1}^{n} |h_{ij}|.$$

More generally, one can define the norm

$$|||H|||_p = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |h_{ij}|^p \right)^{1/p}. \tag{3}$$

Note that the Frobenius norm is a special case, where $p = 2$.

In this work we study the general case of norms of the form of Equation (3), for $p \geq 1$, using the residual definition of Equation (1). We leave the application of our techniques to other objectives as future work.

3. ONE-SIDED CLUSTERING

In the standard clustering problem, we are given n points in a metric space, possibly $\mathbb{R}^d$, and an objective function that measures the quality of any given clustering of the points. Various such objective functions have been extensively used in practice, and have been analyzed in the theoretical computer science literature (k-center, k-median, k-means, etc.). As an aid to our co-clustering algorithm, we are particularly interested in the following setting of the problem, which we call k-means$_p$. Given a set of vectors $a_1, a_2, \ldots a_n$, the distance metric $\|\cdot\|_p$, and an integer k, we first define the cost of a partitioning $\mathcal{I} = \{I_1, \ldots, I_k\}$ of $\{1, \ldots, n\}$ as follows. For each cluster I, the center of the cluster I is defined to be the vector μ_I such that

$$\mu_I = \arg\min_{x \in \mathbb{R}^d} \sum_{a_j \in I} \|a_j - x\|_p^p .$$

The cost $c(\mathcal{I})$ of the clustering $\mathcal{I}$ is then defined to be the sum of distances of each point to the corresponding cluster center, raised to the power of $1/p$:

$$c(\mathcal{I}) = \left(\sum_{I \in \mathcal{I}} \sum_{a_j \in I} \|a_j - \mu_I\|_p^p \right)^{1/p} .$$

This differs from the k-median problem, where the cost of the clustering is given by

$$\sum_{I \in \mathcal{I}} \sum_{a_j \in I} \|a_j - \mu_I\|_p .$$

In the case of $p = 1$, k-means$_p$ is the k-median problem, while for $p = 2$, it is the k-means problem. We have not seen any other versions of this problem in the literature.

In matrix notation, the points in the space define a matrix $A = [a_1, \ldots, a_d]^T$. We will represent each clustering $\mathcal{I} = \{I_1, \ldots, I_k\}$ of n points in $\mathbb{R}^d$ by a clustering index matrix $R \in \mathbb{R}^{n \times k}$. Each column of matrix R will essentially be the index vector of the corresponding cluster, $R_{iI} = 1$ if a_i belongs to cluster I, and 0 otherwise (see Figure 1). Similarly, the matrix $M \in \mathbb{R}^{k \times d}$ is defined to be the set of centers of the clusters, that is, $M = [\mu_1, \ldots, \mu_k]^T$. Thus, the aim is to find out the clustering index matrix R that minimizes

$$\|\|A - RM\|\|_p ,$$

where M is defined as the matrix in $\mathbb{R}^{k \times d}$ that minimizes

$$M = \arg\min_{X \in \mathbb{R}^{k \times \ell}} \|\|A - RX\|\|_p .$$

Let m_I be the size of the row-cluster I, and $A_I \in \mathbb{R}^{m_I \times d}$ the corresponding submatrix of A. Also let $A_{i\star}$ be the ith row vector of A. We can write

$$\begin{aligned} \|\|A - RM\|\|_p^p &= \sum_{I \in \mathcal{I}} \|\|A_I - R_I M\|\|_p^p \\ &= \sum_{I \in \mathcal{I}} \sum_{i \in I} \|A_{i\star} - \mu_I\|_p^p . \end{aligned}$$

The two norms that are of particular interest to us are $p = 1$ and $p = 2$. For the $p = 2$ case, the center μ_I for each cluster is nothing but the average of all the points A_i in that cluster. For the case $p = 1$, the center μ_I is the median of all the points $A_i \in I$. The $p = 2$ case, commonly known as *k-means clustering problem*, has a $(1 + \epsilon)$-approximation algorithm.

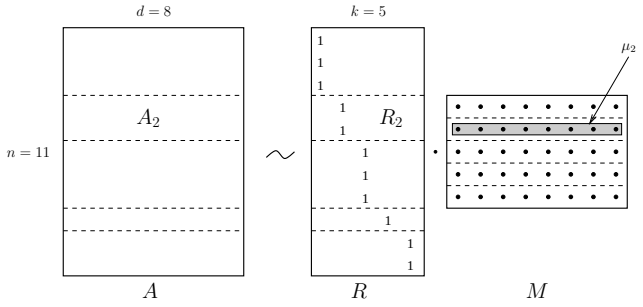

Figure 1: An example of a row-clustering, where we have rows and columns that appear in the same cluster next to each other. We have $A_I \sim R_I \cdot M \sim \mu_I$. For example, $A_2 \sim R_2 \cdot M \sim \mu_2$.

THEOREM 1 ([21]). *For any $\epsilon > 0$ there is an algorithm that achieves a $(1+\epsilon)$-factor approximation for the k-means objective, if k is a constant.*

The same holds true in the case of $p = 1$, for constant values of k.

THEOREM 2 ([3]). *For any $\epsilon > 0$ there is an algorithm that achieves a $(1+\epsilon)$-factor approximation for the k-median problem, if k is a constant.*

The general case where $p \geq 1$ and k is not necessarily constant has not been addressed before. In Theorem 3 we show that there exists a constant approximation algorithm for the problem.

THEOREM 3. *For any $k > 1$, there is an algorithm that achieves a 24-approximation to the k-means$_p$ problem for ℓ_p^p with $p \geq 1$.*

PROOF SKETCH. The problem is similar to the k-median problem, which has been studied extensively. However the results do not apply directly in the k-means$_p$ problem since the ℓ_p^p norm does not induce a metric as it does not satisfy the triangle inequality. Nevertheless, it nearly satisfies it (it follows from Hölder's inequality) and this allows (at the expense of some constant factors) many of the results that hold true for the k-median problem to hold true for the k-means$_p$ problem as well (as long as the triangle inequality is only applied a constant number of times).

The theorem can be proven, for example, by the process presented in [31, Chapters 24, 25], which has also appeared in [17] (the case of $p = 2$ is Exercise 25.6 in [31]). The details will appear in the full version of this work. □

While the value of the constant 24 holds in general, it is not necessarily the best possible, especially for particular values of p. For example, for $p = 1$ we can obtain a value of $3 + \epsilon$, for any $\epsilon > 0$ if $k = \omega(1)$ [2] (if $k = O(1)$ then Theorem 2 applies). For $p = 2$ we have a $\sqrt{108}$-approximation [17].

4. CO-CLUSTERING

In the co-clustering problem, the data is given in the form of a matrix A in $\mathbb{R}^{m \times n}$. We denote a row of A as $A_{i\star}$ and a column of A as $A_{\star j}$. The aim in co-clustering is to simultaneously cluster the rows and columns of A, so as to optimize

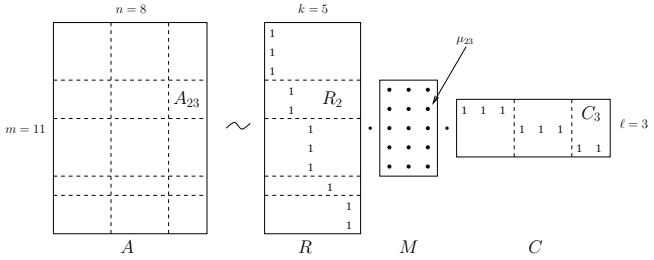

Figure 2: An example of co-clustering, where we have rows and columns that appear in the same cluster next to each other. We have $A_{IJ} \sim R_I \cdot M \cdot C_J \sim \mu_{IJ}$. For example, $A_{23} \sim R_2 \cdot M \cdot C_3 \sim \mu_{23}$.

the difference between A and the clustered matrix. More formally, we want to compute a k-partitioning $\mathcal{I} = \{I_1, \ldots, I_k\}$ of the set of rows $\{1, \ldots, m\}$ and an ℓ-partitioning $\mathcal{J} = \{J_1, \ldots, J_\ell\}$ of the set of columns $\{1, \ldots, n\}$. The two partitionings $\mathcal{I}$ and $\mathcal{J}$ naturally induce clustering index matrices (see Figure 2) $R \in \mathbb{R}^{m \times k}, M \in \mathbb{R}^{k \times \ell}, C \in \mathbb{R}^{\ell \times n}$, defined as follows: each row in R essentially corresponds to the index vector of the corresponding part in the partition $\mathcal{I}$, that is $R_{iI} = 1$, if $A_{i\star} \in I$ and 0 otherwise. Similarly the index matrix C is constructed to represent the partitioning $\mathcal{J}$, that is $C_{Jj} = 1$, if $A_{\star j} \in J$ and 0 otherwise. For each row-cluster column-cluster tuple (I, J), we refer to the set of indices in $I \times J$ to be a block.

The clustering error associated with the co-clustering $(\mathcal{I}, \mathcal{J})$ is defined to be the quantity

$$\left|\!\left|\!\left| A - RMC \right|\!\right|\!\right|_p ,$$

where M is defined as the matrix in $\mathbb{R}^{k \times \ell}$ that minimizes

$$M = \arg\min_X \left|\!\left|\!\left| A - RXC \right|\!\right|\!\right|_p .$$

Let m_I be the size of the row-cluster I and n_J denote the size of the columns cluster J. By the definition of the $\left|\!\left|\!\left| \cdot \right|\!\right|\!\right|_p$, we can write

$$\left|\!\left|\!\left| A - RMC \right|\!\right|\!\right|_p = \left(\sum_{\substack{I \in \mathcal{I} \\ J \in \mathcal{J}}} \left|\!\left|\!\left| A_{IJ} - \mu_{IJ} R_I C_J \right|\!\right|\!\right|_p^p \right)^{1/p} , \quad (4)$$

where each $A_{IJ} \in \mathbb{R}^{m_I \times n_J}$, each vector $R_I \in \mathbb{R}^{m_I \times 1}$, and each $\mu_{IJ} \in \mathbb{R}$, and vector $C_J \in \mathbb{R}^{1 \times n_J}$. Two special cases that are of interest to us are $p = 1, 2$. For the $p = 2$ case, the matrix norm $\left|\!\left|\!\left| \cdot \right|\!\right|\!\right|_p$ corresponds to the the well known Frobenius norm $\left\| \cdot \right\|_F$, and the value μ_{IJ} corresponds to a simple average of the corresponding block. For the $p = 1$ case, the norm corresponds to a simple sum over the absolute values of the entries of the matrix, and the corresponding μ_{IJ} value would be the median of the entries in that block.

5. ALGORITHM

In this section, we give a simple algorithm for co-clustering. We first present the algorithm, and then show that for the general $\left|\!\left|\!\left| \cdot \right|\!\right|\!\right|_p$ norm, the algorithm gives a constant-factor approximation. We then do a tighter analysis for the simpler case of $\left|\!\left|\!\left| \cdot \right|\!\right|\!\right|_2$, i.e., the Frobenius norm, to show that we get a $(\sqrt{2} + \epsilon)$-approximation.

Algorithm 1 `Co-Cluster`(A, k, ℓ)

Require: Matrix $A \in \mathbb{R}^{m \times n}$, number of row-clusters k, number of column-clusters ℓ.

1: Let $\hat{\mathcal{I}}$ be the α-approximate clustering of the row vectors with k clusters.
2: Let $\hat{\mathcal{J}}$ be the α-approximate clustering of the column vectors with ℓ clusters.
3: **return** $(\hat{\mathcal{I}}, \hat{\mathcal{J}})$.

5.1 Constant-Factor Approximation

We now show that the co-clustering returned by algorithm `Co-Cluster` is a constant-factor approximation to the optimum.

THEOREM 4. *Given an α-approximation algorithm for the k-means$_p$ problem, the algorithm* `Co-Cluster`(A, k, ℓ) *returns a co-clustering that is a 3α-approximation to the optimal co-clustering of A.*

PROOF. Let $\mathcal{I}^*, \mathcal{J}^*$ be the optimal co-clustering solution. Define the corresponding index matrices to be R^* and C^* respectively. Furthermore, let $\hat{\mathcal{I}}^*$ be the optimal row-clustering and $\hat{\mathcal{J}}^*$ be the optimal column-clustering. Define the index matrix $\hat{R}^*$ from the clustering $\hat{\mathcal{I}}^*$, and the index matrix $\hat{C}^*$ from the clustering $\hat{\mathcal{J}}^*$. This means that there is a matrix $\hat{M}_R^* \in \mathbb{R}^{k \times n}$ such that

$$\left|\!\left|\!\left| A - \hat{R}^* \hat{M}_R^* \right|\!\right|\!\right|_p$$

is minimized over all such index matrices representing k clusters. Similarly, there is a a matrix $\hat{M}_C^* \in \mathbb{R}^{m \times \ell}$ such that

$$\left|\!\left|\!\left| A - \hat{M}_C^* \hat{C}^* \right|\!\right|\!\right|_p$$

is minimized over all such index matrices representing ℓ clusters.

The algorithm `Co-Cluster` uses approximate solutions for the one-sided row and column-clustering problems to compute partitionings $\hat{\mathcal{I}}$ and $\hat{\mathcal{J}}$. Let $\hat{R}$ be the clustering index matrix corresponding to this row-clustering and $\hat{M}_R$ be the set of centers. Similarly, let $\hat{C}$, $\hat{M}_C$ be the corresponding matrices for the column-clustering constructed by `Co-Cluster`. By the assumptions of the theorem we have that

$$\left|\!\left|\!\left| A - \hat{R}\hat{M}_R \right|\!\right|\!\right|_p \leq \alpha \left|\!\left|\!\left| A - \hat{R}^* \hat{M}_R^* \right|\!\right|\!\right|_p , \quad (5)$$

and, similarly,

$$\left|\!\left|\!\left| A - \hat{M}_C \hat{C} \right|\!\right|\!\right|_p \leq \alpha \left|\!\left|\!\left| A - \hat{M}_C^* \hat{C}^* \right|\!\right|\!\right|_p . \quad (6)$$

For the co-clustering $(\hat{M}_R, \hat{M}_C)$ that the algorithm computes, define the center matrix $M \in \mathbb{R}^{k \times \ell}$ as follows. Each entry μ_{IJ} is defined to be

$$\mu_{IJ} = \arg\min_x \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - x|^p . \quad (7)$$

Now we will show that the co-clustering $(\hat{\mathcal{I}}, \hat{\mathcal{J}})$ with the center matrix M will be a 3α-approximate solution. First, we lower bound the cost of the optimal co-clustering solution by the optimal row-clustering and optimal column-clustering. Since $(\hat{R}^*, \hat{M}_R^*)$ is the optimal row-clustering, we

have that

$$\begin{aligned}\left|\!\left|\!\left|A - \hat{R}^* \hat{M}_R^*\right|\!\right|\!\right|_p &\leq \min_X \left|\!\left|\!\left|A - R^* X\right|\!\right|\!\right|_p \\ &\leq \left|\!\left|\!\left|A - R^* M^* C^*\right|\!\right|\!\right|_p .\end{aligned} \tag{8}$$

Similarly, since $(\hat{C}^*, \hat{M}_C^*)$ is the optimal column-clustering,

$$\begin{aligned}\left|\!\left|\!\left|A - \hat{M}_C^* \hat{C}^*\right|\!\right|\!\right|_p &\leq \min_X \left|\!\left|\!\left|A - X C^*\right|\!\right|\!\right|_p \\ &\leq \left|\!\left|\!\left|A - R^* M^* C^*\right|\!\right|\!\right|_p .\end{aligned} \tag{9}$$

Let us consider a particular block $(I, J) \in \hat{\mathcal{I}} \times \hat{\mathcal{J}}$. Note that $(\hat{R}\hat{M}_R)_{ij} = (\hat{R}\hat{M}_R)_{i'j}$ for $i, i' \in I$. We denote $\hat{r}_{Ij} = (\hat{R}\hat{M}_R)_{ij}$. Let $\hat{\mu}_{IJ}$ be the value x that minimizes

$$\hat{\mu}_{IJ} = \arg\min_x \sum_{j \in J} |\hat{r}_{Ij} - x|^p .$$

We also denote $\hat{c}_{iJ} = (\hat{M}_C \hat{C})_{ij}$. Then for all $i \in I$ we have

$$\sum_{j \in J} |\hat{r}_{Ij} - \hat{\mu}_{IJ}|^p \leq \sum_{j \in J} |\hat{r}_{Ij} - \hat{c}_{iJ}|^p ,$$

which gives

$$\begin{aligned}\left(\sum_{\substack{i \in I \\ j \in J}} |\hat{r}_{Ij} - \hat{\mu}_{IJ}|^p)\right)^{1/p} &\leq \left(\sum_{\substack{i \in I \\ j \in J}} |\hat{r}_{Ij} - \hat{c}_{iJ}|^p\right)^{1/p} \\ &\leq \left(\sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{r}_{Ij}|^p\right)^{1/p} \\ &\quad + \left(\sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{c}_{iJ}|^p\right)^{1/p} ,\end{aligned} \tag{10}$$

where the last inequality is just application of the triangle inequality.

Then we get

$$\begin{aligned}\left|\!\left|\!\left|A - \hat{R} M \hat{C}\right|\!\right|\!\right|_p &\overset{(a)}{=} \left(\sum_{I,J} \left|\!\left|\!\left|A_{IJ} - \mu_{IJ} \hat{R}_I \hat{C}_J\right|\!\right|\!\right|_p^p\right)^{1/p} \\ &= \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \mu_{IJ}|^p\right)^{1/p} \\ &\overset{(b)}{\leq} \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{\mu}_{IJ}|^p\right)^{1/p} \\ &\overset{(c)}{\leq} \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{r}_{Ij}|^p\right)^{1/p} \\ &\quad + \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |\hat{r}_{Ij} - \hat{\mu}_{IJ}|^p\right)^{1/p} \\ &\overset{(d)}{\leq} \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{r}_{Ij}|^p\right)^{1/p} \\ &\quad + \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{r}_{Ij}|^p\right)^{1/p} \\ &\quad + \left(\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} |a_{ij} - \hat{c}_{iJ}|^p\right)^{1/p} \\ &= \left|\!\left|\!\left|A - \hat{R}\hat{M}_R\right|\!\right|\!\right|_p + \left|\!\left|\!\left|A - \hat{R}\hat{M}_R\right|\!\right|\!\right|_p \\ &\quad + \left|\!\left|\!\left|A - \hat{M}_C \hat{C}\right|\!\right|\!\right|_p \\ &\overset{(e)}{\leq} \alpha \left(\left|\!\left|\!\left|A - \hat{R}^* \hat{M}_R^*\right|\!\right|\!\right|_p + \left|\!\left|\!\left|A - \hat{R}^* \hat{M}_R^*\right|\!\right|\!\right|_p \right. \\ &\quad \left. + \left|\!\left|\!\left|A - \hat{M}_C^* \hat{C}^*\right|\!\right|\!\right|_p\right) \\ &\overset{(f)}{\leq} 3\alpha \left|\!\left|\!\left|A - R^* M^* C^*\right|\!\right|\!\right|_p ,\end{aligned}$$

where (a) follows from Equation (4), (b) follows from Equation (7), (c) from the triangle inequality, (d) from Equation (10), (e) from Equations (5) and (6), and (f) follows from Equations (8) and (9). □

By combining the above with Theorems 2 and 3 we obtain the following corollaries.

COROLLARY 1. *For any constant values of k, ℓ there exists an algorithm that returns a (k, ℓ)-co-clustering that is a $(3+\epsilon)$-approximation to the optimum, for any $\epsilon > 0$, under the $\left|\!\left|\!\left|\cdot\right|\!\right|\!\right|_1$ norm.*

COROLLARY 2. *For any k, ℓ there is an algorithm that returns a (k, ℓ)-co-clustering that is a 72-approximation to the optimum, for any $\epsilon > 0$.*

5.2 A $(\sqrt{2}+\epsilon)$-Factor Approximation for the $\|\cdot\|_F$ Norm

A commonly used instance of our objective function is the case of $p = 2$, i.e., the Frobenius norm. The results of the previous section give us a $(3+\epsilon)$-approximation in this particular case, when k, ℓ are constants. But it turns out that in this case, we can actually exploit the particular structure of the Frobenius norm and give a better approximation factor.

To restate the problem, we want to compute clustering matrices $R \in \mathbb{R}^{m \times k}, C \in \mathbb{R}^{\ell \times n}$, such that $R_{i,I} = 1$, if $A_{i\star} \in I$ and 0 otherwise, and $C_{J,j} = 1$, if $A_{\star j} \in J$ and 0 otherwise (see Section 4 for more details) such that $\|A - RMC\|_F$ is minimized, where $M \in \mathbb{R}^{k \times \ell}$ and M contains the averages of the cluster, i.e. $M = \{\mu_{IJ}\}$ where

$$\mu_{IJ} = \frac{1}{m_I \cdot n_J} \sum_{\substack{i \in I \\ j \in J}} a_{ij},$$

where m_I is the size of row-cluster I and n_J is the size of column-cluster J. We show the following theorem.

THEOREM 5. *Given an α-approximation algorithm for the k-means clustering problem, the algorithm* `Co-Cluster` *gives a $\sqrt{2}\alpha$-approximate solution to the co-clustering problem with the $\|\cdot\|_F$ objective function.*

PROOF. Define $\bar{R} \in \mathbb{R}^{m \times k}$ similarly to R, but with the values scaled down according to the clustering. Specifically,

$\bar{R}_{i,I} = (m_I)^{-1/2}$, if $i \in I$ and 0 otherwise. Similarly, define $\bar{C}_{J,j} = (n_J)^{-1/2}$, if $j \in J$ and 0 otherwise. Then notice that we can write $RMC = \bar{R}\bar{R}^T A \bar{C}^T \bar{C}$.

If we consider also the one-sided clusterings (RM_R and $M_C C$) then we can also write $RM_R = \bar{R}\bar{R}^T A$ and $M_C C = A\bar{C}^T\bar{C}$.

We define $P_R = \bar{R}\bar{R}^T$. Then P_R is a projection matrix. To see why this is the case, notice first that $\bar{R}$ has orthogonal columns:

$$(\bar{R}^T \cdot \bar{R})_{II} = \sum_{i \in I} \frac{1}{m_I} = 1,$$

and $(\bar{R}^T \cdot \bar{R})_{IJ} = 0$, for $I \neq J$, thus $\bar{R}^T \cdot \bar{R} = \mathbf{I}_k$. Therefore $P_R P_R = P_R$, hence P_R is a projection matrix. Define as $P_R^\perp = (\mathbf{I} - P_R)$ the projection orthogonal to P_R. Similarly we define the projection matrices $P_C = \bar{C}^T\bar{C}$ and $P_C^\perp = (\mathbf{I} - P_C)$. In general, in the rest of the section, P_X and $P_X^\perp$ refer to the projection matrices that correspond to clustering matrix X.

We can then state the problem as finding the projections of the form $P_R = \bar{R}\bar{R}^T$ and $P_C = \bar{C}^T\bar{C}$ that minimize $\|A - P_R A P_C\|_F^2$, under the constraint that $\bar{R}$ and $\bar{C}$ are of the form that we described previously.

Let R^* and C^* be the optimal co-clustering solution, $\hat{R}^*$ and $\hat{C}^*$ be the optimal one-sided clusterings, and $\hat{R}$ and $\hat{C}$ be the one-sided row and column-clusterings that are α-approximate to the optimal ones. We have

$$\left\|A - \hat{R}\hat{M}_R\right\|_F^2 \leq \alpha^2 \left\|A - \hat{R}^* \hat{M}_R^*\right\|_F^2, \tag{11}$$

and

$$\left\|A - \hat{M}_C\hat{C}\right\|_F^2 \leq \alpha^2 \left\|A - \hat{M}_C^*\hat{C}^*\right\|_F^2. \tag{12}$$

We can write

$$\begin{aligned} A &= P_{\hat{R}} A + P_{\hat{R}}{}^\perp A \\ &= P_{\hat{R}} A P_{\hat{C}} + P_{\hat{R}} A P_{\hat{C}}{}^\perp + P_{\hat{R}}{}^\perp A P_{\hat{C}} + P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp, \end{aligned}$$

and thus

$$A - P_{\hat{R}} A P_{\hat{C}} = P_{\hat{R}} A P_{\hat{C}}{}^\perp + P_{\hat{R}}{}^\perp A P_{\hat{C}} + P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp.$$

Then,

$$\begin{aligned} \|A - P_{\hat{R}} A P_{\hat{C}}\|_F^2 &= \left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp + P_{\hat{R}}{}^\perp A P_{\hat{C}} + P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp\right\|_F^2 \\ &= \left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp + P_{\hat{R}}{}^\perp (A P_{\hat{C}} + A P_{\hat{C}}{}^\perp)\right\|_F^2 \\ &\overset{(a)}{=} \left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp\right\|_F^2 + \left\|P_{\hat{R}}{}^\perp (A P_{\hat{C}} + A P_{\hat{C}}{}^\perp)\right\|_F^2 \\ &= \left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp\right\|_F^2 + \left\|P_{\hat{R}}{}^\perp A P_{\hat{C}} + P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp\right\|_F^2 \\ &\overset{(b)}{=} \left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp\right\|_F^2 + \left\|P_{\hat{R}}{}^\perp A P_{\hat{C}}\right\|_F^2 \\ &\qquad + \left\|P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp\right\|_F^2, \end{aligned}$$

where equalities (a) follows from the Pythagorean theorem (we apply it to every column separately and the square of the Frobenius norm is just the sum of the column lengths squared) and the fact that the projection matrices $P_{\hat{R}}$ and $P_{\hat{R}}{}^\perp$ are orthogonal to each other, and equality (b) again from the Pythagorean theorem and the orthogonality of $P_{\hat{C}}$ and $P_{\hat{C}}{}^\perp$.

Without loss of generality we assume that

$$\left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp\right\|_F^2 \geq \left\|P_{\hat{R}}{}^\perp A P_{\hat{C}}\right\|_F^2$$

(otherwise we can consider A^T). Then,

$$\begin{aligned} \|A - P_{\hat{R}} A P_{\hat{C}}\|_F^2 &\leq 2\left(\left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp\right\|_F^2 + \left\|P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp\right\|_F^2\right) \\ &= 2\left(\left\|P_{\hat{R}} A P_{\hat{C}}{}^\perp + P_{\hat{R}}{}^\perp A P_{\hat{C}}{}^\perp\right\|_F^2\right) \\ &= 2\left\|A P_{\hat{C}}{}^\perp\right\|_F^2 \\ &= 2\|A - A P_{\hat{C}}\|_F^2, \end{aligned} \tag{13}$$

where the first equality follows once again from the Pythagorean theorem. By applying Equations (12) and (13) we get

$$\begin{aligned} \|A - P_{\hat{R}} A P_{\hat{C}}\|_F^2 &\leq 2\|A - A P_{\hat{C}}\|_F^2 \\ &\leq 2(1+\epsilon')\|A - A P_{\hat{C}^*}\|_F^2. \end{aligned} \tag{14}$$

It remains to show that the error of the optimal one-sided clustering is bounded by the error of the optimal co-clustering:

$$\begin{aligned} \|A - A P_{\hat{C}^*}\|_F^2 &\overset{(a)}{\leq} \|A - A P_{C^*}\|_F^2 \\ &= \left\|A P_{C^*}{}^\perp\right\|_F^2 \\ &\leq \left\|A P_{C^*}{}^\perp\right\|_F^2 + \left\|P_{R^*}{}^\perp A P_{C^*}\right\|_F^2 \\ &\overset{(b)}{=} \left\|A P_{C^*}{}^\perp + P_{R^*}{}^\perp A P_{C^*}\right\|_F^2 \\ &= \left\|A - A P_{C^*} + P_{R^*}{}^\perp A P_{C^*}\right\|_F^2 \\ &= \left\|A - (\mathbf{I} - P_{R^*}{}^\perp) A P_{C^*}\right\|_F^2 \\ &= \|A - P_{R^*} A P_{C^*}\|_F^2, \end{aligned} \tag{15}$$

where (a) follows from the fact that $P_{\hat{C}^*}$ corresponds to the optimal column-clustering, and (b) follows from the Pythagorean theorem and the orthogonality of P_{C^*} and $P_{C^*}{}^\perp$.

Combining Equations (14) and (15) gives

$$\|A - P_{\hat{R}} A P_{\hat{C}}\|_F^2 \leq 2\alpha^2 \|A - P_{R^*} A P_{C^*}\|_F^2.$$

Thus we can obtain a $\sqrt{2}\alpha$-approximation to the optimal co-clustering solution, under the Frobenius norm. □

We can now use Theorems 1 and 3 and obtain the following corollaries.

COROLLARY 3. *For any constant values of k, ℓ there exists an algorithm that returns a (k,ℓ)-co-clustering that is a $(\sqrt{2}+\epsilon)$-approximation to the optimum, for any $\epsilon > 0$, under the $\|\|\cdot\|\|_2$ norm.*

COROLLARY 4. *For any k, ℓ there is an algorithm that returns a (k,ℓ)-co-clustering that is a $24\sqrt{2}$-approximation to the optimum, for any $\epsilon > 0$.*

5.3 Solving the $(k,1)$-Co-Clustering

In this section we show how we can solve exactly the problem in the case that we only want one column-cluster (note that this is different from the one-sided clustering; the latter is equivalent to having n column-clusters). While this case is not of significant interest, we include it for completeness and to show that even in that case the problem is nontrivial (although it is polynomial). In particular, while we can solve exactly the problem under the Frobenius norm, it is

not clear whether we can solve it for all the norms of the form of Equation (3).

First we begin by stating a simple result for the case that $A \in \mathbb{R}^{m\times 1}$. Then the problem is easy, for any metric of the form of Equation (3).

LEMMA 1. *Let $A \in \mathbb{R}^{m\times 1}$ and consider any norm $|||\cdot|||_p$. There is an algorithm that can $(k,1)$-cluster matrix A optimally in time $O(m^2k)$ and space $O(mk)$.*

PROOF SKETCH. The idea is the following: A is just a set of real values, and $(k,1)$ clustering A corresponds to the partition of those values into k clusters. Note that if the optimal cluster contains points a_i and a_j then it should contain also all the points in between. This fact implies that we can solve the problem using dynamic programming. Assume that the sorted values of A are $\{a_1, a_2, \ldots, a_m\}$. Then we can define $C(i,r)$ the optimal r-clustering solution of $\{a_1, \ldots, a_i\}$. Knowing $C(j, r-1)$ for $j \leq i$ allows us to compute $C(i,r)$. The time required is $O(m^2k)$ and the space needed is $O(mk)$. Further details and the complete proof are omitted. □

We now use this lemma to solve optimally for general A, under the norm $\|\cdot\|_F$. The algorithm is simple. Assume that $A = \{a_{ij}\}$ and let $\mu_i = \frac{1}{n}\sum_{j=1}^{n} a_{ij}$ be the mean of row i. Also write $a_{ij} = \mu_i + \varepsilon_{ij}$, and note that for all i we have $\sum_{j=1}^{n} \varepsilon_{ij} = 0$. The algorithm then runs the dynamic-programming algorithm on the vector of the means and returns the clustering produced.

Algorithm 2 `Co-Cluster`$(A, k, 1)$

Require: Matrix $A \in \mathbb{R}^{m\times n}$, number of row-clusters k.
1: Create the vector $v = (\mu_1, \mu_2, \ldots, \mu_m)$, where $\mu_i = \frac{1}{n}\sum_{j=1}^{n} a_{ij}$.
2: Use the dynamic-programming algorithm of Lemma 1 and let $\mathcal{I}$ be the resulting k-clustering.
3: **return** $(\mathcal{I}, \{1, \ldots, n\})$.

THEOREM 6. *Let $A \in \mathbb{R}^{m\times n}$. Let $\mathcal{I}$ be the clustering produced under the $\|\cdot\|_F$ norm. Then $\mathcal{I}$ has optimal cost. The running time of the algorithm is $O(mn + m^2k)$.*

PROOF. Let us see the cost of a given cluster. For notational simplicity, assume a cluster containing rows 1 to r. The mean of the cluster equals

$$\mu = \frac{1}{rn}\sum_{i=1}^{r}\sum_{j=1}^{n} a_{ij} = \sum_{i=1}^{r} \mu_i,$$

and let

$$S = \sum_{i=1}^{r} \mu_i = r\mu.$$

The cost of the cluster is

$$\begin{aligned}
\sum_{i=1}^{r}\sum_{j=1}^{n}(a_{ij}-\mu)^2 &= \sum_{i=1}^{r}\sum_{j=1}^{n} a_{ij}^2 + rn\mu^2 - 2\mu\sum_{i=1}^{r}\sum_{j=1}^{n} a_{ij} \\
&= \sum_{i=1}^{r}\sum_{j=1}^{n}(\mu_i+\varepsilon_{ij})^2 + \frac{nS^2}{r} - 2\frac{S}{r}nr\mu \\
&= \sum_{i=1}^{r}\sum_{j=1}^{n}\mu_i^2 + \sum_{i=1}^{r}\sum_{j=1}^{n}\varepsilon_{ij}^2 \\
&\qquad + 2\sum_{i=1}^{r}\mu_i\sum_{j=1}^{n}\varepsilon_{ij} - \frac{nS^2}{r} \\
&= n\sum_{i=1}^{r}\mu_i^2 + \sum_{i=1}^{r}\sum_{j=1}^{n}\varepsilon_{ij}^2 - \frac{nS^2}{r},
\end{aligned}$$

since $\sum_{j=1}^{r}\varepsilon_{ij} = 0$, for all i.

Therefore, the cost of the entire clustering $\mathcal{I} = \{I_1 \ldots, I_k\}$ equals

$$n\sum_{i=1}^{m}\mu_i^2 + \sum_{i=1}^{m}\sum_{j=1}^{n}\varepsilon_{ij}^2 - n\sum_{I\in\mathcal{I}}\frac{S_I^2}{m_I}, \tag{16}$$

where m_I is the number of rows in cluster I and $S_I = \sum_{i\in I}\mu_i$.

Consider now the one-dimensional problem of $(k,1)$ clustering only the row means μ_i. The cost of a given cluster is (again assume the cluster contains rows 1 to r):

$$\begin{aligned}
\sum_{i=1}^{r}(\mu_i-\mu)^2 &= \sum_{i=1}^{r}\mu_i^2 + r\mu^2 - 2\mu\sum_{i=1}^{r}\mu_i \\
&= \sum_{i=1}^{r}\mu_i^2 - \frac{S^2}{r}.
\end{aligned}$$

Thus the cost of the clustering is

$$\sum_{i=1}^{m}\mu_i^2 - \sum_{I\in\mathcal{I}}\frac{S_I^2}{m_I}.$$

Compare the cost of this clustering with that of Equation (16). Note that in both cases the optimal row-clustering is the one that maximizes the term $\sum_{I\in\mathcal{I}}\frac{S_I^2}{m_I}$, as all the other terms are independent of the clustering. Thus we can optimally solve the problem for $A \in \mathbb{R}^{m\times n}$ by solving the problem simply on the means vector. The time needed to create the vector of means is $O(mn)$, and by applying Lemma 1 we conclude that we can solve the problem in time $O(mn + m^2k)$. □

6. HARDNESS OF THE OBJECTIVE FUNCTION

In this section, we show that the problem of co-clustering an $m \times n$ matrix A is NP-hard when the number of clusters on the column side, is at least n^ϵ, for any $\epsilon > 0$. While there are several results in the literature that show hardness of similar problems [28, 15, 5, 26], we are not aware of any previous result that proves the hardness of the co-clustering for the objectives that we study in this paper.

THEOREM 7. *The problem of finding a (k,ℓ) co-clustering for a matrix $A \in \mathbb{R}^{m\times n}$ is NP-hard for $(k,\ell) = (k, n^\epsilon)$ for any $k \geq 2$ and any $\epsilon > 0$, under the ℓ_1 norm.*

PROOF. The proof contains several steps. First we reduce the one-sided k-median problem (where $k = n/3$) under the ℓ_1 norm to the $(2, n/3)$-co-clustering when $A \in \mathbb{R}^{2\times n}$. We reduce the latter problem to the case of $A \in \mathbb{R}^{m\times n}$ and $(k, n/3)$, and this, finally, to the case of (k, n^ϵ)-co-clustering. We now proceed with the details.

Megiddo and Supowit [23] show that the (one-sided) k-median problem is NP-hard under the ℓ_1 norm in $\mathbb{R}^2$. By looking carefully the pertinent proof we can see that the problem is hard even if we restrict it to the case of $k = n/3+o(n)$ (n is the number of points). Let us assume that we have such a problem instance of n points $\{a_j\}$, $j = 1, \ldots, n$ and we want to assign them into ℓ clusters, $\ell = n/3 + o(n)$, so as to minimize the ℓ_1 norm. Specifically, we want to compute a partition $\mathcal{J} = \{J_1, \ldots, J_\ell\}$ of $\{1, \ldots, n\}$, and points $\mu_1, \ldots, \mu_\ell$ such that the objective

$$\sum_{J\in\mathcal{J}} \sum_{j\in J} \|a_j - \mu_j\|_1 \tag{17}$$

is minimized.

We construct a co-clustering instance by constructing the matrix A where we set $A_{ij} = a_{ji}$, for $i = 1, 2$ and $j = 1, \ldots, n$:

$$A = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n1} \end{bmatrix},$$

which we want to $(2, \ell)$-co-cluster. Solving this problem is equivalent to solving the one-sided clustering problem. To provide all the details, there is only one row-clustering, $\mathcal{I} = \{\{1\}, \{2\}\}$, and consider the column-clustering $\mathcal{J} = \{J_1, \ldots, J_\ell\}$. and the corresponding center matrix $M \in \mathbb{R}^{2\times\ell}$. The cost of the solution equals

$$\sum_{I,J} \sum_{\substack{i\in I \\ j\in J}} |A_{ij} - M_{IJ}| = \sum_{J\in\mathcal{J}} \sum_{j\in J} |a_{j1} - M_{1J}| + |a_{j2} - M_{2J}| \tag{18}$$

Note that this expression is minimized when (M_{1J}, M_{2J}) is the median of the points a_j, $j \in J$, in which case the cost equals to that of Equation (17). Thus a solution to the co-clustering problem induces a solution to the one-sided problem. Therefore, solving the $(2, \ell)$-co-clustering problem in $\mathbb{R}^{2\times n}$ is NP-hard.

The next step is to show that it is hard to (k, ℓ)-co-cluster a matrix for any k and $\ell = n/3 + o(n)$. This follows from the previous $(2, \ell)$-co-clustering in $\mathbb{R}^{2\times n}$, by adding to A rows of some value B, where B is some large value (say $B > 2mn \max |a_{ij}|$):

$$A = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n1} \\ B & B & \cdots & B \\ \vdots & \vdots & \ddots & \vdots \\ B & B & \cdots & B \end{bmatrix}.$$

Indeed, we can achieve a solution with the same cost as Equation (18) by the same column partitioning $\mathcal{J}$ and a row partitioning that puts each of rows 1 and 2 to each own cluster and cluster the rest of the rows (where all the values equal B) arbitrarily. Notice that this is an optimal solution since any other row-cluster will place at least one value a_{ij} and B in the same co-cluster, in which case the cost just of that co-cluster will be at least $|B - |a_{ij}||$, which is larger than that of Equation (18).

The final step is to reduce a problem instance of finding a (k, ℓ')-co-clustering of a matrix $A' \in \mathbb{R}^{m\times n'}$, with $\ell' = n'/3 + o(n')$ to a problem instance of finding a (k, ℓ)-co-clustering of a matrix $A \in \mathbb{R}^{m\times n}$, with $\ell = n^\epsilon$, for any $\epsilon > 0$.

The construction is similar as before. Let $A' = \{A'_{ij}\}$. Define $n = (\ell' + 1)^{1/\epsilon}$ and let $A \in \mathbb{R}^{m\times n}$. For $1 \le j \le n'$ (assume that ϵ is sufficiently small so that $n \ge n'$), define $A_{ij} = A'_{ij}$ and for any $j > n'$, define $A_{ij} = B$, where B is some sufficiently large value, (e.g., $B > 2mn \max |a_{ij}|$):

$$A = \begin{bmatrix} A'_{11} & A'_{12} & \cdots & A'_{1d} & B & B & \cdots & B \\ A'_{21} & A'_{22} & \cdots & A'_{2d} & B & B & \cdots & B \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A'_{m1} & A'_{m2} & \cdots & A'_{md} & B & B & \cdots & B \end{bmatrix}.$$

Now, we only need to prove that the optimal solution of a $(k, \ell' + 1) = (k, n^\epsilon)$-co-clustering of A corresponds to the optimal solution of the (k, ℓ')-co-clustering of A'.

Assume that the optimal solution for matrix A' is given by the partitions $\mathcal{I}' = \{I'_1, \ldots, I'_k\}$ and $\mathcal{J}' = \{J'_1, \ldots, \mathcal{J}'_{\ell'}\}$. The cost of the solution is

$$C'(\mathcal{I}', \mathcal{J}') = \sum_{\substack{I\in\mathcal{I}' \\ J\in\mathcal{J}'}} \sum_{\substack{i\in I \\ j\in J}} \left|A'_{ij} - M_{IJ}\right|,$$

where M_{IJ} is defined as the median of the values $\{A'_{ij};\ i \in I, j \in J\}$.

Let us compute the optimal solution for the $(k, \ell' + 1)$-co-clustering of A. First note that we can compute a solution $(\mathcal{I}, \mathcal{J})$ with cost $C'(\mathcal{I}', \mathcal{J}')$. We let $\mathcal{I} = \mathcal{I}'$, and for $\mathcal{J} = \{J_1, \ldots, \mathcal{J}_{\ell'+1}\}$ we set $J_j = J'_j$ for $j \le \ell'$, and $J_{\ell'+1} = \{n' + 1, n'+2, \ldots, n\}$. For the centering matrix M we have $M_{IJ_j} = M'_{IJ_j}$ for $j \le \ell'$ and $M_{IJ_{\ell'+1}} = B$. The cost $C(\mathcal{I}, \mathcal{J})$ of the co-clustering equals

$$\begin{aligned} C(\mathcal{I}, \mathcal{J}) &= \sum_{\substack{I\in\mathcal{I} \\ J\in\mathcal{J}}} \sum_{\substack{i\in I \\ j\in J}} |A_{ij} - M_{IJ}| \\ &= \sum_{\substack{I\in\mathcal{I} \\ J\in\mathcal{J}}} \sum_{\substack{i\in I \\ j\in J}} |A_{ij} - M_{IJ}| \\ &= \sum_{\substack{I\in\mathcal{I} \\ J\in\mathcal{J}'}} \sum_{\substack{i\in I \\ j\in J}} |A_{ij} - M_{IJ}| + \sum_{I\in\mathcal{I}} \sum_{\substack{i\in I \\ j\in J_{\ell'+1}}} |A_{ij} - M_{IJ}| \\ &= \sum_{\substack{I\in\mathcal{I}' \\ J\in\mathcal{J}'}} \sum_{\substack{i\in I \\ j\in J}} \left|A'_{ij} - M'_{IJ}\right| + \sum_{I\in\mathcal{I}} \sum_{\substack{i\in I \\ j\in J_{\ell'+1}}} |B - B| \\ &= C'(\mathcal{I}', \mathcal{J}'). \end{aligned}$$

Now, we have to show that the optimal solution to the co-clustering problem has to have the above structure, that is, if $\mathcal{J} = \{J_1, J_2, \ldots, J_{\ell'+1}\}$ are the column-clusters, then it has to be the case that, modulo a permutation of cluster indices, $J_j = J'_j$ for $j \le \ell'$ and $J_{\ell'+1} = \{n' + 1, \ldots, n\}$ and $\mathcal{I} = \mathcal{I}'$. Suppose not, then we consider two cases. The first is that there exists a column $A_{\star j}$ for $j > n'$ that is put into the same cluster (say cluster J) as a column $A_{\star y}$ for $y \le \ell'$. In this case we show that the resulting co-clustering cost will be much more than $c(I_1^{\text{opt}}, I_2^{\text{opt}})$. To show this, just consider the error from the two coordinates A_{1j} and A_{1y}, for instance. The value of the center for this row, is some $M_{1J} = x$.

Now, if $x > B/2$, then since (trivially) $A_{1y} < B/4$, we have that $|A_{1y} - x| > B/4 > C'(\mathcal{I}', \mathcal{J}')$. On the other hand if $x \leq B/2$ then $|A_{1j} - x| > B/4 > C'(\mathcal{I}', \mathcal{J}')$. Thus the cost of this solution is much larger than the cost of the optimal solution.

Assume now that this is not the case. Then we can assume that there exists a column-cluster containing all the columns greater than n': $\mathcal{J}_{\ell'+1} = \{n'+1, \ldots, n\}$ (there can be more than one clusters but this will only increase the total cost), and note that the cost of the corresponding co-clusters is 0. Thus the total cost is equal to the cost of the (k, ℓ')-co-clustering of the submatrix of A, $i = 1, \ldots, m,\ j = 1, \ldots, n'$. This is exactly the original problem of co-clustering matrix A'. Thus, the solution $(\mathcal{I}, \mathcal{J})$ is optimal.

Note that $\ell' + 1 = n^\epsilon$. Thus, solving the $(k, \ell' + 1) = (k, n^\epsilon)$-co-clustering problem on the new matrix gives us a solution to the original k-median problem. Hence the (k, ℓ)-co-clustering problem under the ℓ_1 norm is NP-hard, for any $k > 1$ and $\ell = n^\epsilon$. □

Note that while we showed hardness for the ℓ_1 norm, our reduction can show hardness of co-clustering from hardness of one-sided clustering. So, for example, hardness for the k-means objective [9] implies hardness for the co-clustering under the Frobenius norm.

7. DISCUSSION AND FUTURE WORK

In this paper we consider the problem of co-clustering. We obtain the first algorithms for this problem with provable performance guarantees. Our algorithms are simple and achieve constant-factor approximations with respect to the optimum. We also show that the co-clustering problem is NP-hard, for a wide range of the input parameters. Finally, as a byproduct, we introduce the k-means$_p$ problem, which generalizes the k-median and k-means problems, and give a constant-factor approximation algorithm.

Our work leads to several interesting questions. In Section 6 we showed that the co-clustering problem is hard if $\ell = \Omega(n^\epsilon)$ under the ℓ_1 norm. It seems that the hardness should hold for any ℓ_p norm, $p \geq 1$. It would also be interesting to show that it is hard for any combination of k, ℓ. In particular, even the hardness questions for the $(2, 2)$ or the $(O(1), O(1))$ cases are, as far as we know, unresolved. While we conjecture that these cases are hard, we do not have yet a proof for this. As we noted at the end of Section 6 the NP-hardness of the k-median problem in low-dimensional Euclidean spaces (and with small number of clusters) would give further hardness results for the co-clustering problem. During our research in the pertinent literature we were surprised to discover that while there are several publications on approximation algorithms for k-means and k-median in low-dimensional Euclidean spaces their complexity is still open, especially when the number of clusters is $o(n)$. Thus any hardness result in that direction would be of great interest.

Another question is whether the problem becomes easy for matrices A having a particular structure. For instance, if A is symmetric, and $k = \ell$, is it the case that the optimal co-clustering is also symmetric? The answer turns out to be negative, even if we are restricted to 0/1-matrices, and the counterexample reveals some of the difficulty in co-clustering. Consider the matrix

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

We are interested in a $(2, 2)$-co-clustering, say using $\|\cdot\|_F$. There are are three symmetric solutions, $\mathcal{I} = \mathcal{J} = \{\{1, 2\}, \{3\}\}$, $\mathcal{I} = \mathcal{J} = \{\{2, 3\}, \{1\}\}$, and $\mathcal{I} = \mathcal{J} = \{\{1, 3\}, \{2\}\}$, and all have a cost of 1. Instead, the nonsymmetric solution $(\mathcal{I}, \mathcal{J}) = (\{\{1\}, \{2, 3\}\},\ \{\{1, 2\}, \{3\}\})$, has cost of $\sqrt{3}/4$. Therefore, even for symmetric matrices, one-sided clustering cannot be used to obtain the optimal co-clustering.

A further interesting direction is to find approximation algorithms for other commonly used objective functions for the co-clustering problem. It appears that our techniques cannot be directly applied to any of those. As we mentioned before, the work by Dhillon et al. [4] unifies a number of such objectives and gives an expectation maximization style heuristic for such merit functions. It would be interesting to see if given an approximation algorithm for solving the clustering problem for a Bregman divergence, we can construct a co-clustering approximation algorithm from it. Another objective function for which our approach is not immediately applicable is Equation (3) using the residual definition of Equation (2). For several problems this class of objective functions might be more appropriate than the one that we analyze here.

Finally one can wonder what happens when the matrix to be clustered has more than two dimensions. For example, what happens when $A \in \mathbb{R}^{m \times n \times o}$? Is there a version of our algorithm (or any algorithm) that can solve this problem?

8. REFERENCES

[1] D. Agarwal and S. Merugu. Predictive discrete latent factor models for large scale dyadic data. In *Proc. of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 26–35, 2007.

[2] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, June 2004.

[3] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proc. of the 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

[4] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to Bregman co-clustering and matrix approximation. *Journal of Machine Learning Research*, 8:1919–1986, 2007.

[5] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[6] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. of the 8th International Conference on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.

[7] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum sum-squared residue co-clustering of gene expression data. In *Proc. of the 4th SIAM International Conference on Data Mining*. SIAM, 2004.

[8] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274, 2001.

[9] P. Drineas, A. M. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine Learning*, 56(1-3):9–33, 2004.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2000.

[11] U. Feige and S. Kogan. Hardness of approximation of the balanced complete bipartite subgraph problem, 2004.

[12] B. Gao, T. Liu, X. Zheng, Q. Cheng, and W. Ma. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *Proc. of the 11th ACM Conference on Knowledge Discovery and Data Mining*, pages 41–50, 2005.

[13] S. Gollapudi, R. Kumar, and D. Sivakumar. Programmable clustering. In *Proc. 25th ACM Symposium on Principles of Database Systems*, pages 348–354, 2006.

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

[15] S. Hassanpour. Computational complexity of bi-clustering. Master's thesis, University of Waterloo, 2007.

[16] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[17] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001.

[18] M. Jambyu and M. O. Lebeaux. *Cluster Analysis and Data Analysis*. North-Holland, 1983.

[19] J. Kleinberg. An impossibility theorem for clustering. In *Advances in Neural Information Processing Systems 15*, pages 446–453, 2002.

[20] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Research*, 13:703–716, 2003.

[21] A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time $(1 + \epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proc. of the 45th IEEE Symposium on Foundations of Computer Science*, pages 454–462, 2004.

[22] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[23] N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1):182–196, 1984.

[24] N. Mishra, D. Ron, and R. Swaminathan. On finding large conjunctive clusters. In *Proc. of the 16th Annual Conference on Computational Learning Theory*, pages 448–462, 2003.

[25] N. Mishra, D. Ron, and R. Swaminathan. A new conceptual clustering framework. *Machine Learning*, 56(1-3):115–151, 2004.

[26] R. Peeters. The maximum edge biclique problem is NP-complete. *Discrete Applied Mathematics*, 131(3):651–654, 2003.

[27] K. Puolamäki, S. Hanhijärvi, and G. C. Garriga. An approximation ratio for biclustering. *CoRR*, abs/0712.2682, 2007.

[28] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182, 2004.

[29] H. Takamura and Y. Matsumoto. Co-clustering for text categorization. *Information Processing Society of Japan Journal*, 2003.

[30] A. Tanay, R. Sharan, and R. Shamir. Biclustering algorithms: A survey. In E. by Srinivas Aluru, editor, *In Handbook of Computational Molecular Biology*. Chapman & Hall/CRC, Computer and Information Science Series, 2005.

[31] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.

[32] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *Proc. of the 3rd IEEE Conference on Bioinformatics and Bioengineering*, pages 321–327, 2003.

[33] J. Yang, W. Wang, H. Wang, and P. S. Yu. *delta*-clusters: Capturing subspace correlation in a large data set. In *Proc. of the 18th International Conference on Data Engineering*, pages 517–528, 2002.

[34] H. Zhou and D. P. Woodruff. Clustering via matrix powering. In *Proc. of the 23rd ACM Symposium on Principles of Database Systems*, pages 136–142, 2004.