THEORY OF COMPUTING, Volume 8 (2012), pp. 597–622
www.theoryofcomputing.org

A Constant-Factor Approximation Algorithm for Co-clustering*

Aris Anagnostopoulos† Anirban Dasgupta Ravi Kumar

Received: August 4, 2011; published: December 10, 2012.

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 a lot of 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 provide constant-factor approximations to the optimum. We also show that co-clustering is NP-hard, thereby complementing our algorithmic result.

ACM Classification: F.2.0

AMS Classification: 68W25

Key words and phrases: 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 [12, 20]. 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

*A preliminary version appeared in the Proc. 27th ACM Symp. on Principles of Database Systems (PODS 2008). The present version contains complete proofs. In particular, Section 3.1 has been rewritten to be self-contained.

†Partially supported by the EU FP7 Project N. 255403 SNAPS.

DOI: 10.4086/toc.2012.v008a026

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 [22]. 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 whose 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, maybe 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* [17, 7]; other terms 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 a lot of attention in recent years, with several applications in text mining [10, 15], market-basket data analysis, image, speech, and video analysis, and bioinformatics [7, 8, 23]; see the paper by Banerjee et al. [3] and the survey by Madeira and Oliveira [25].

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 [18]. A natural generalization of the k-means algorithm to co-clustering is known to converge [3]. Apart from these, most of the algorithmic work done on co-clustering has been heuristic in nature, with no proven guarantees of performance.

Main contributions In this paper we address the problem of co-clustering from an algorithmic point of view. 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 a standard set of matrix norms 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 known results on the k-median problem. We next consider the important special case of the Frobenius norm and obtain a $(\sqrt{2}+\varepsilon)$-approximation algorithm when k and ℓ are fixed by exploiting the geometry of the space and the results on the k-means problem.

We complement these results by considering the extreme cases of $\ell = 1$ and $\ell = n^{\varepsilon}$, where the matrix is of size $m \times n$ and ε is any fixed value in $(0,1)$. We show that while the $(k,1)$-co-clustering problem can be solved exactly in time $O(mn+m^2k)$, the (k,n^{ε})-co-clustering problem is NP-hard, for $k \geq 2$ under the ℓ_1 and ℓ_2 norms.

In the data-mining and machine-learning literature there exists a variety of co-clustering problems depending on what is the precise objective function one is trying to optimize; we give some examples in Section 2. The version that we address in this paper is a natural one and at the same time it possesses such a structure that allows us to approximately solve it by decomposing it into single-dimensional clustering problems. This is not true for other types of objectives. For instance, if one considers the residue given by (2.2) (for more discussion about the relevant concepts refer to Section 2) then one can create counter-examples showing that the problem is not decomposable; solving such cases is interesting future work.

Related work Research on clustering has a long and varied history, with work ranging from approximation algorithms to axiomatic developments of the objective functions [19, 12, 22, 20, 36, 16]. The problem of co-clustering itself has found growing applications in several practical fields, for example, simultaneously clustering words and documents in information retrieval [10], clustering genes and expression data for biological data analysis [7, 34], 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 [18]. 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 [25] and Tanay, Sharan, and Shamir [32]. Banerjee et al. [3] 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 is precisely 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 [28]. These variants are often hard even to approximate to within a constant factor. Hassanpour [18] showed that a version of the co-clustering problem that finds out homogeneous submatrices is hard. Feige and Kogan [13] showed that the problem of finding the maximum biclique cannot be approximated to within $2^{(\log n)^{\delta}}$ for some $\delta = \delta(\varepsilon) > 0$ assuming that 3-SAT cannot be solved in $O(2^{n^{3/4+\varepsilon}})$ deterministic time for some fixed $\varepsilon > 0$.

Concurrently with the conference publication of our work, Puolamäki et al. [30] 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). Subsequent to the conference publication of our work, Jegelka et al. [21] showed that our algorithm can be extended to tensor clustering and to general metrics.

2 Preliminaries and problem definition

In this section we mention briefly some of the variants of the objective function 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 $\mathcal{I} = \{I_1, \dots, I_k\}$ of the set of rows $\{1, \dots, m\}$ (i. e., we want to find k sets $I_1, \dots, I_k$, $I_i \subseteq \{1, \dots, m\}$ such that $I_i \cap I_{i'} = \emptyset$ for $i \neq i'$ and $\bigcup_{i=1}^{k} I_i = \{1, \dots, m\}$) and an ℓ-partitioning $\mathcal{J} = \{J_1, \dots, J_\ell\}$ of the set of columns $\{1, \dots, n\}$.

Cho et al. [8] define two different co-clustering objectives, based on two different definitions of *residue*. For every element a_{ij} that belongs to the (I,J)-co-cluster define its *residue* h_{ij} either as

$$a_{ij} - a_{IJ}\,, \tag{2.1}$$

or as

$$a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}\,, \tag{2.2}$$

where

$$a_{IJ} = \frac{1}{|I| \cdot |J|} \sum_{i \in I, j \in J} a_{ij} \quad \text{is the average of all the entries in the co-cluster,}$$

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij} \quad \text{is the mean of all the entries in row } i \text{ whose columns belong into } J\text{, and}$$

$$a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij} \quad \text{is the mean of all the entries in column } j \text{ whose rows belong into } I.$$

Having defined the (two different) 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. [35] attempt to find (possibly overlapping) clusters that minimize the value

$$\frac{1}{|I|\,|J|} \sum_{i \in I} \sum_{j \in J} |h_{ij}|$$

using definition (2.2), where I and J define a cluster.

More generally, one can define the norm

$$|||H|||_p = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} \left|h_{ij}\right|^p \right)^{1/p}. \tag{2.3}$$

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 (2.3), for $p \geq 1$, using the residual definition of (2.1); designing approximation algorithms for residual definition corresponding to (2.2) is an open problem. Specifically, we assume that the data is given in the form of a matrix A in $\mathbb{R}^{m\times n}$. We denote the ith row of A as $A_{i\star}$ and the jth 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 the difference between A and the clustered matrix. Formally, we want to compute a k-partitioning $\mathcal{I} = \{I_1, \dots, I_k\}$ of the set of rows $\{1, \dots, m\}$ and an ℓ-partitioning $\mathcal{J} = \{J_1, \dots, J_\ell\}$ of the set of columns $\{1, \dots, n\}$. For any such partition $\mathcal{I} = \{I_1, \dots, I_k\}$ of rows (or columns) a convenient way for us to describe it will be in terms of a *clustering index matrix*, say $R = \{r_{ij}\} \in \mathbb{R}^{m\times k}$, such that 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 $i \in I$ and 0 otherwise (see Figure 1). (Note that we slightly abuse notation and we use the partition as an index. For example, the element r_{iI_s} refers to the element r_{is} of matrix R and it equals 1 if $i \in I_s$ and 0 otherwise.) We can define a similar index matrix $C = \{c_{ij}\} \in \mathbb{R}^{\ell\times n}$ for the column partition $\mathcal{J}$ in a similar fashion: $c_{Jj} = 1$, if $j \in J$ and 0 otherwise. The co-clustering solution is then completely defined by the three matrices $R \in \mathbb{R}^{m\times k}$, $M \in \mathbb{R}^{k\times \ell}$ (to be defined below), and $C \in \mathbb{R}^{\ell\times n}$ (see Figure 1). For each row cluster and 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

$$|||A - RMC|||_p\,, \tag{2.4}$$

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

$$M = \operatorname*{argmin}_{X\in\mathbb{R}^{k\times\ell}} |||A - RXC|||_p\,.$$

Let $m_I = |I|$ be the size of the row cluster I and $n_J = |J|$ denote the size of the column cluster J. By the definition of $|||\cdot|||_p$, we can write (see also Figure 1)

$$|||A - RMC|||_p = \left(\sum_{\substack{I\in\mathcal{I}\\ J\in\mathcal{J}}} |||A_{IJ} - R_I M C_J|||_p^p\right)^{1/p} = \left(\sum_{\substack{I\in\mathcal{I}\\ J\in\mathcal{J}}} |||A_{IJ} - \mu_{IJ}\cdot \mathbf{1}_{m_I\times n_J}|||_p^p\right)^{1/p}, \tag{2.5}$$

where each $A_{IJ} = \{(a_{IJ})_{ij}\} \in \mathbb{R}^{m_I\times n_J}$ and equals

$$(a_{IJ})_{ij} = a_{\alpha_i,\beta_j}, \qquad \text{if } I = \{\alpha_1, \dots, \alpha_i, \dots, \alpha_{m_I}\} \text{ and } J = \{\beta_1, \dots, \beta_j, \dots, \beta_{n_J}\}\,,$$

each $R_I = \{(r_I)_{ij}\} \in \mathbb{R}^{m_I\times k}$ and equals

$$(r_I)_{ij} = \ 1 \text{ if } I = I_j, \text{ and 0 otherwise}$$

(which means that if I refers to the jth row cluster then the jth column of R_I is 1), each $C_J = \{c_J\} \in \mathbb{R}^{\ell\times n_J}$, and equals

$$(c_J)_{ij} = \ 1 \text{ if } J = J_i, \text{ and 0 otherwise}$$

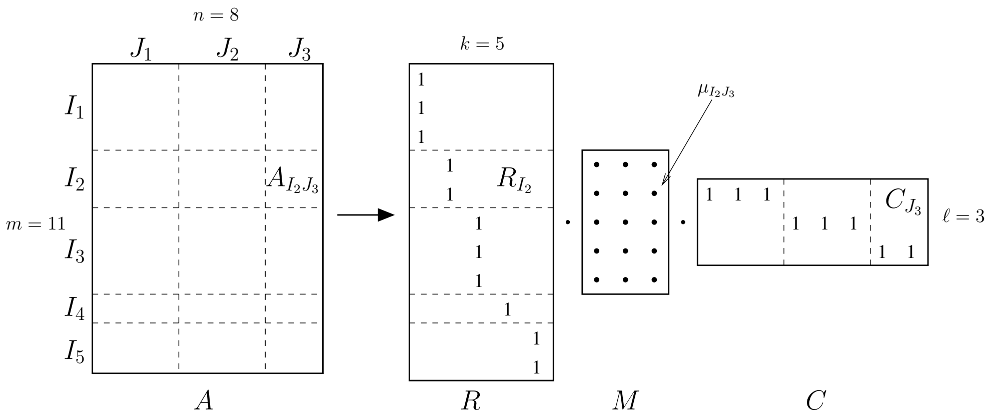

Figure 1: An example of co-clustering, where we have rows and columns that appear in the same cluster next to each other. Here, A_{IJ} is represented by $R_I M C_J = \mu_{IJ} \cdot \mathbf{1}_{m_I \times n_J}$ as in (2.5), for instance, the 2×2 submatrix $A_{I_2 J_3}$ is represented by $R_{I_2} M C_{J_3} = \mu_{I_2 J_3} \cdot \mathbf{1}_{2\times 2}$.

(which means that if J refers to the ith column cluster then the ith row of C_J is 1), each $\mu_{IJ} \in \mathbb{R}$ is the mean element associated with co-cluster (I,J), and $\mathbf{1}_{m_I \times n_J}$ is the 2-dimensional matrix of size $m_I \times n_J$ with all the elements equal to 1. Two special cases that are of interest to us are $p = 1,2$. For the $p = 2$ case, the matrix norm $|||\cdot|||_p$ corresponds to the the well known Frobenius norm $\|\cdot\|_F$, 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} values would be the median of the entries in that block.

In general the number of row and column clusters, k and ℓ, are parts of the input. In some cases, for k and ℓ constants one can obtain stronger results; when this is the case we will mention it explicitly.

3 Algorithm

In this section we give a simple algorithm for co-clustering for the objective function defined in (2.4). We first present the algorithm and then show that for the general $|||\cdot|||_p$ norm, the algorithm gives a constant-factor approximation. We then do a tighter analysis for the simpler case of $|||\cdot|||_2$ (i. e., the Frobenius norm), to show that we get a $(\sqrt{2}+\varepsilon)$-approximation when k,ℓ are constant.

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

Before analyzing the algorithm we need to show how we can solve the subproblems of steps 1 and 2. For this we use some results for "standard" one-sided clustering and we develop some new ones needed to solve the problem for any p-norm.

3.1 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 $a_1, a_2, \ldots a_n \in \mathbb{R}^d$, the standard ℓ_p 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 $I \in \mathcal{I}$, the *center* of the cluster I is defined to be the vector μ_I such that

$$\mu_I = \operatorname*{argmin}_{x \in \mathbb{R}^d} \sum_{j \in I} \|a_j - x\|_p^p . \tag{3.1}$$

The *cost* 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 $1/p$, that is,

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

The goal for the k-means$_p$ problem is to minimize this cost. In general the number of clusters k is part of the input. We also consider the case that k is constant, where we can obtain stronger results. When this is the case we mention it explicitly.

In matrix notation, the points define a matrix $A = [a_1, \ldots, a_n]^T \in \mathbb{R}^{n \times d}$. 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 2). 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

$$M = \operatorname*{argmin}_{X \in \mathbb{R}^{k \times d}} |||A - RX|||_p .$$

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

$$|||A - RM|||_p^p = \sum_{I \in \mathcal{I}} |||A_I - R_I M|||_p^p = \sum_{I \in \mathcal{I}} \sum_{i \in I} \|A_{i\star} - \mu_I\|_p^p . \tag{3.2}$$

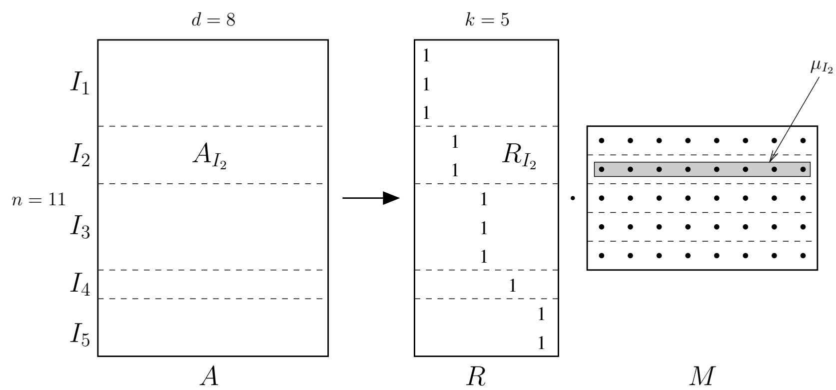

Figure 2: An example of a row clustering, where we have rows and columns that appear in the same cluster next to each other. Here, A_I is represented by $R_I \cdot M$ as in (3.2).

Of particular interest to us is the value $p = 2$. In that case, the center μ_I for each cluster is the average of all the points $A_{i\star}$ in that cluster (the average minimizes the expression (3.1) for $p = 2$). It is commonly known as the *k-means* problem and it admits a polynomial-time $(1+\varepsilon)$-approximation algorithm when k is fixed.

Theorem 3.1 ([24])**.** *For any fixed values of k and $\varepsilon > 0$ there is a polynomial-time algorithm that achieves a $(1+\varepsilon)$-factor approximation for the k-means problem, if dimension is part of the input.*

In Theorem 3.4 we show that there exists a constant approximation algorithm for the k-means$_p$ problem for any p (and k). Our proof will be based on the analysis of the similar k-median problem by Charikar et al. [6]. In the *k-median* problem we are given a set $A = \{a_1, \ldots, a_n\}$ of points, the cost $d(a_i, a_j)$ of assigning point a_i to point a_j, and an integer $k > 0$, and the goal is to find a k-partitioning $\mathfrak{I}$ of $\{1, \ldots, n\}$ to minimize the cost

$$\sum_{I \in \mathfrak{I}} \sum_{i \in I} d(a_i, \mu_I^{\text{med}}),$$

where the k-median center μ_I^{med} of cluster I is given by

$$\mu_I^{\text{med}} = \operatorname*{argmin}_{a \in A} \sum_{i \in I} d(a_i, a). \tag{3.3}$$

In the metric case, $d(\cdot,\cdot)$ is assumed to be symmetric and to satisfy the triangle inequality. Unfortunately, in our case we use the norm ℓ_p^p and the triangle inequality does not hold for $p > 1$. However, as we show next in Lemma 3.3, it does hold approximately. This allows us to use the following result of Charikar et al. [6], which is a simple extension of their constant-factor approximation for the metric k-median problem.[1]

[1]While Charikar et al. [6] study the metric case, they mention that their results can be extended to the case that the triangle inequality holds only approximately, as described in Theorem 3.2. Following their proof, shows that the approximation ratio can be bounded by $4(\delta^2 + \delta) \leq 8\delta^2$, for $\delta \geq 1$.

Theorem 3.2 ([6]). *There is a polynomial-time algorithm that achieves an $8\delta^2$-approximation to the metric k-median problem if the costs satisfy a δ-approximate triangle inequality, for $\delta \geq 1$, that is, $d(a_i, a_o) \leq \delta(d(a_i, a_j) + d(a_j, a_o))$ for each i, j, o.*

Next we prove Lemma 3.3, which shows that the distance measure induced by ℓ_p^p satisfies an approximate version of the triangle inequality.

Lemma 3.3. *Let $u, v, w \in \mathbb{R}^d$ and let $p \geq 1$. Then, $\|u-v\|_p^p \leq 2^{p-1}(\|u-w\|_p^p + \|w-v\|_p^p)$.*

Proof. Note that the function $f(x) = |x|^p$ is convex for $p \geq 1$ and hence

$$\left(\frac{|a+b|}{2}\right)^p \leq \frac{|a|^p + |b|^p}{2} \quad \text{or, equivalently,} \quad |a+b|^p \leq 2^{p-1}\left(|a|^p + |b|^p\right).$$

Let $u = (u_1, \ldots, u_d)$, and similarly for v, w. We have

$$\begin{aligned}
\|u-v\|_p^p &= \sum_{i=1}^d |u_i - v_i|^p \\
&= \sum_{i=1}^d |(u_i - w_i) + (w_i - v_i)|^p \\
&\leq \sum_{i=1}^d 2^{p-1}\left(|u_i - w_i|^p + |w_i - v_i|^p\right) \\
&= 2^{p-1}\left(\|u-w\|_p^p + \|w-v\|_p^p\right).
\end{aligned}$$

□

Now we can state and prove the theorem for the one-sided clustering.

Theorem 3.4. *There is a polynomial-time algorithm that achieves a 16-approximation to the k-means$_p$ problem for $p \geq 1$.*

Proof. The proof is based on the result by Charikar et al., described in Theorem 3.2. We note two issues that make our problem different from the standard metric k-median: (i) we have $d(a_i, a_j) = \left\|a_i - a_j\right\|_p^p$, but it does not satisfy the triangle inequality and (ii) the centers in k-median are chosen from the given set A (i. e., $\mu_I^{\text{med}} \in A$) whereas the centers in k-means$_p$ are not necessarily from A. Note also a small asymmetry between the cost functions of the k-means$_p$ and the k-median objectives, as in the former the sum of the distances is raised to the power $1/p$, unlike in the latter.

To address issue (i), as we showed in Lemma 3.3, the distance measure $\|\cdot\|_p^p$ satisfies an approximate triangle inequality and hence we appeal to Theorem 3.2. We address issue (ii) as follows. Let $\mathcal{I}^*$ be the optimal clustering according to the k-means$_p$ objective (i. e., the cluster centers can be arbitrary) and let $\mathcal{J}^*$ be the optimal clustering according to the k-median objective (i. e., the cluster centers belong to A). In Lemma 3.5, proven directly after this proof, we show that the loss of the requirement of cluster centers belonging to A can be bounded.

Define $c(\mathcal{I}^*)$ to be the cost

$$c(\mathcal{I}^*) = \sum_{I \in \mathcal{I}^*} \sum_{i \in I} \|a_i - \mu_I\|_p^p,$$

with μ_I defined as in (3.1). Notice that this is the cost of the k-means$_p$ objective raised to the pth power. Define also $c(\mathcal{J}^*)$ to be the cost of the k-median objective:

$$c(\mathcal{J}^*) = \sum_{I \in \mathcal{I}^*} \sum_{i \in I} \left\| a_i - \mu_I^{\text{med}} \right\|_p^p ,$$

with μ_I^{med} defined as in (3.3).

Using $\delta = 2^{p-1}$ from Lemma 3.3, the algorithm in Theorem 3.2 outputs a clustering $\tilde{\mathcal{J}}$ whose k-means$_p$ cost is bounded by $\left(2^{2p+1} c(\mathcal{J}^*)\right)^{1/p}$. Using Lemma 3.5, we then conclude that the k-means$_p$ cost of $\mathcal{I}^*$ is bounded by

$$\left(2^{3p+1} c(\mathcal{I}^*)\right)^{1/p} \leq 16 \left(c(\mathcal{I}^*)\right)^{1/p} . \qquad \square$$

We now prove the lemma used in the preceding theorem.

Lemma 3.5. *If the costs are in ℓ_p^p, then $c(\mathcal{J}^*) \leq 2^p c(\mathcal{I}^*)$.*

Proof. First, we claim that for each cluster $I \in \mathcal{I}^*$, there is some $a \in A$ such that

$$\sum_{i \in I} \|a_i - a\|_p^p \leq 2^p \sum_{i \in I} \|a_i - \mu_I\|_p^p .$$

The proof of the claim is by a standard averaging argument over the points in cluster I. By Lemma 3.3, we have

$$\begin{aligned}
\sum_{j \in I} \sum_{i \in I} \left\| a_i - a_j \right\|_p^p &\leq 2^{p-1} \left(\sum_{j \in I} \sum_{i \in I} \|a_i - \mu_I\|_p^p + \left\| \mu_I - a_j \right\|_p^p \right) \\
&\leq 2^{p-1} \left(\sum_{j \in I} \sum_{i \in I} \|a_i - \mu_I\|_p^p + \sum_{j \in I} \sum_{i \in I} \left\| a_j - \mu_I \right\|_p^p \right) \\
&= 2^p \sum_{j \in I} \sum_{i \in I} \|a_i - \mu_I\|_p^p \\
&= 2^p \, |I| \sum_{i \in I} \|a_i - \mu_I\|_p^p ,
\end{aligned}$$

which means that

$$\frac{1}{|I|} \sum_{j \in I} \sum_{i \in I} \left\| a_i - a_j \right\|_p^p \leq 2^p \sum_{i \in I} \|a_i - \mu_I\|_p^p .$$

Letting $f(a) = \sum_{i \in I} \|a_i - a\|_p^p$, we have that

$$\frac{1}{|I|} \sum_{j \in I} f(a_j) \leq 2^p \sum_{i \in I} \|a_i - \mu_I\|_p^p .$$

Since the average (over the elements in cluster $I \in \mathcal{I}^*$) cost of $f(a)$ is bounded by at most

$$2^p \sum_{i \in I} \|a_i - \mu_I\|_p^p$$

it means that there exists an element $a_j \in I$ for which

$$f(a_j) \leq 2^p \sum_{i \in I} \|a_i - \mu_I\|_p^p ,$$

and thus we have found the desired element $a \in A$ for cluster I. By summing this over all the clusters in $\mathcal{I}^*$, we have shown that each cluster center in $\mathcal{I}^*$ can be replaced by some point in A without increasing the clustering cost by too much. The proof then follows from the optimality of $\mathcal{J}^*$. □

3.2 Constant-factor approximation

We now show that the co-clustering returned by Algorithm 1 is a constant-factor approximation to the optimum.

Theorem 3.6. *Given an algorithm for obtaining an α-approximation to the k-means$_p$ problem, the algorithm* `Co-cluster` *(Algorithm 1) returns a co-clustering that is a 3α-approximation to an optimal co-clustering of A.*

Proof. Let $(\mathcal{I}^*, \mathcal{J}^*)$ be an optimal co-clustering solution. Define the corresponding index matrices to be R^* and C^* respectively. Furthermore, let $\hat{\mathcal{I}}^*$ be an optimal row clustering and $\hat{\mathcal{J}}^*$ be an 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

$$\left|\left|\left| A - \hat{R}\hat{M}_R \right|\right|\right|_p \leq \alpha \left|\left|\left| A - \hat{R}^* \hat{M}_R^* \right|\right|\right|_p , \tag{3.4}$$

and, similarly,

$$\left|\left|\left| A - \hat{M}_C\hat{C} \right|\right|\right|_p \leq \alpha \left|\left|\left| A - \hat{M}_C^* \hat{C}^* \right|\right|\right|_p . \tag{3.5}$$

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

$$\mu_{IJ} = \operatorname*{argmin}_x \sum_{\substack{i \in I \\ j \in J}} \left| a_{ij} - x \right|^p . \tag{3.6}$$

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

$$\left\|\left|A-\hat{R}^*\hat{M}_R^*\right|\right\|_p \le \min_X \||A-R^*X|\|_p \le \||A-R^*M^*C^*|\|_p \, . \tag{3.7}$$

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

$$\left\|\left|A-\hat{M}_C^*\hat{C}^*\right|\right\|_p \le \min_X \||A-XC^*|\|_p \le \||A-R^*M^*C^*|\|_p \, . \tag{3.8}$$

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

$$\hat{\mu}_{IJ}=\operatorname*{argmin}_x \sum_{j\in J}\left|\hat{r}_{Ij}-x\right|^p .$$

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

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

which gives

$$\left(\sum_{\substack{i\in I\\ j\in J}}\left|\hat{r}_{Ij}-\hat{\mu}_{IJ}\right|^p)\right)^{1/p} \le \left(\sum_{\substack{i\in I\\ j\in J}}\left|\hat{r}_{Ij}-\hat{c}_{iJ}\right|^p\right)^{1/p} \le \left(\sum_{\substack{i\in I\\ j\in J}}\left|a_{ij}-\hat{r}_{Ij}\right|^p\right)^{1/p} + \left(\sum_{\substack{i\in I\\ j\in J}}\left|a_{ij}-\hat{c}_{iJ}\right|^p\right)^{1/p} , \tag{3.9}$$

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

Then we get

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

where (a) follows from (2.5), (b) follows from (3.6), (c) from the triangle inequality, (d) from (3.9), (e) from (3.4) and (3.5), and (f) follows from (3.7) and (3.8).

□

By combining the above with Theorem 3.4 we obtain the following.

Corollary 3.7. *There is a polynomial-time algorithm that returns a (k,ℓ)-co-clustering that is a 48-factor approximation to the optimum under the $|||\cdot|||_p$ norm.*

3.3 A tighter analysis for the Frobenius norm

A commonly used instance of our objective function is the case of $p=2$, the Frobenius norm. The results of the previous section give us a $(3+\varepsilon)$-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 $i \in I$ and 0 otherwise, and $C_{J,j}=1$, if $j \in J$ and 0 otherwise (see Section 2 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, that is, $M=\{\mu_{IJ}\}$ 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 3.8. *Given an α-approximation algorithm for the k-means clustering problem, the algorithm* `Co-cluster` *(Algorithm 1) computes a $\sqrt{2}\alpha$-approximate solution to the co-clustering problem with the $\|\cdot\|_F$ objective function.*

Proof. Define $\bar{R}=\{\bar{r}_{ij}\} \in \mathbb{R}^{m\times k}$ similarly to R, but with the values scaled down according to the clustering. Specifically, $\bar{r}_{i,I}=1/\sqrt{m_I}$, if $i\in I$ and 0 otherwise. Likewise, define $\bar{C}=\{\bar{c}_{ij}\}\in\mathbb{R}^{\ell\times n}$, with $\bar{c}_{J,j}=1/\sqrt{n_J}$, if $j\in J$ and 0 otherwise. Then notice that we can write $RMC=\bar{R}\bar{R}^TA\bar{C}^T\bar{C}$.

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

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 $P_RP_R=P_R$, hence P_R is a projection matrix. Define as $P_R^\perp=(\mathbf{I}_m-P_R)$ the projection orthogonal to P_R. Similarly we define the projection matrices $P_C=\bar{C}^T\bar{C}$ and $P_C^\perp=(\mathbf{I}_n-P_C)$. 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 $\|A-P_RAP_C\|_F^2$, 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

$$\left\|A-\hat{R}\hat{M}_R\right\|_F^2 \le \alpha^2 \left\|A-\hat{R}^*\hat{M}_R^*\right\|_F^2 \tag{3.10}$$

and

$$\left\|A-\hat{M}_C\hat{C}\right\|_F^2 \le \alpha^2 \left\|A-\hat{M}_C^*\hat{C}^*\right\|_F^2 . \tag{3.11}$$

We can write

$$A = P_{\hat{R}}A + P_{\hat{R}}^{\perp}A = P_{\hat{R}}AP_{\hat{C}} + P_{\hat{R}}AP_{\hat{C}}^{\perp} + P_{\hat{R}}^{\perp}AP_{\hat{C}} + P_{\hat{R}}^{\perp}AP_{\hat{C}}^{\perp} ,$$

and thus

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

Then,

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

where (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 (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 $\left\|P_{\hat{R}}AP_{\hat{C}}^{\perp}\right\|_F^2 \ge \left\|P_{\hat{R}}^{\perp}AP_{\hat{C}}\right\|_F^2$ (otherwise we can consider A^T). Then,

$$\begin{aligned}
\left\|A - P_{\hat{R}}AP_{\hat{C}}\right\|_F^2 &\le 2\left(\left\|P_{\hat{R}}AP_{\hat{C}}^{\perp}\right\|_F^2 + \left\|P_{\hat{R}}^{\perp}AP_{\hat{C}}^{\perp}\right\|_F^2\right) \\
&= 2\left(\left\|P_{\hat{R}}AP_{\hat{C}}^{\perp} + P_{\hat{R}}^{\perp}AP_{\hat{C}}^{\perp}\right\|_F^2\right) = 2\left\|AP_{\hat{C}}^{\perp}\right\|_F^2 = 2\left\|A - AP_{\hat{C}}\right\|_F^2 ,
\end{aligned} \tag{3.12}$$

where the first equality follows once again from the Pythagorean theorem. By combining (3.11) and (3.12) we get

$$\left\|A - P_{\hat{R}}AP_{\hat{C}}\right\|_F^2 \le 2\left\|A - AP_{\hat{C}}\right\|_F^2 \le 2\alpha^2\left\|A - AP_{\hat{C}^*}\right\|_F^2 . \tag{3.13}$$

By (3.8) we know that the error of the optimal one-sided clustering is bounded by the error of the optimal co-clustering. Considering (3.8) for $p = 2$, taking the square, and rewriting with the notation of this section we have that

$$\left\|A - AP_{\hat{C}^*}\right\|_F^2 \le \left\|A - P_{R^*}AP_{C^*}\right\|_F^2 . \tag{3.14}$$

Combining (3.13) and (3.14) gives

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

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

We can now use Theorem 3.1 and obtain the following corollary.

Corollary 3.9. *For any fixed values of k, ℓ and $\varepsilon > 0$ there is a polynomial time algorithm that returns a (k, ℓ)-co-clustering that is a $(\sqrt{2} + \varepsilon)$-approximation to the optimum, under the Frobenius norm.*

We also improve slightly the approximation ratio when k and ℓ are part of the input by using Theorem 3.4:

Corollary 3.10. *There is a polynomial-time algorithm that returns a (k, ℓ)-co-clustering that is a 32-factor approximation to the optimum under the Frobenius norm.*

3.4 Solving $(k, 1)$-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 practical interest, we include it for completeness and to show that even in that case the problem can be nontrivial. 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 (2.3), including even the case that $p = 1$. To summarize the cases that we study in this section, one can consider the following cases of $(k, 1)$-co-clustering:

- The problem of $(k, 1)$-co-clustering for a matrix of dimension $m \times 1$, for any $p \geq 1$. This case is known to be solvable in polynomial time using dynamic programming and we present the algorithm and the analysis in Lemma 3.11 for completeness.

- The problem of $(k, 1)$-co-clustering for a matrix of dimension $m \times n$, for any $n > 1$ and for $p = 2$. In Theorem 3.12 we show how we can solve this problem in polynomial time based on Lemma 3.11.

- The problem of $(k, 1)$-co-clustering for a matrix of dimension $m \times n$, for any $n > 1$ and for $p \neq 2$. It remains open whether this case can be solved exactly in polynomial time.

First we begin by considering the first case ($A \in \mathbb{R}^{m \times 1}$). Then the problem is easy, for any norm of the form of (2.3). We state the following simple result.

Lemma 3.11. *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 $O(m^2 k)$ and space $O(mk)$.*

Proof. The proof is based on Brucker's algorithm for one-dimensional clustering [5]; we present it for completeness. Since A is just a set of real values, then $(k,1)$ clustering A corresponds to the partition of those values into k clusters. The proof is based on the following fact. If a cluster in the optimal solution contains values a_i and a_j then it should contain also all the values in between. The proof is by contradiction. Assume that there are two clusters such that the values are not separated. Then we can replace these two clusters with two clusters with the values separated, and the cost will be lower.

This fact implies that we can solve the problem using dynamic programming. Assume that the sorted values of A are $\{a_1, a_2, \dots, a_m\}$. We define $C(i,r)$ to be the optimal r-clustering solution of $\{a_1, \dots, a_i\}$. Then, knowing $C(j,r-1)$ for $j \le i$ allows us to compute $C(i,r)$, by considering all the possible positions that the rth cluster begins:

$$C(i,r) = \min_{j \in \{r, r+1, \dots, i\}} \{C(j-1, r-1) + c(j,i)\},$$

where $c(j,i)$ is the cost of the single cluster containing the values $\{a_j, a_{j+1}, \dots, a_i\}$.

We need to compute $O(mk)$ values $C(i,r)$ so the space needed is $O(mk)$, and to compute each of them we require $O(m)$ time, so the total time required is $O(m^2 k)$. $\square$

We now use this lemma to solve optimally for general A, under the Frobenius norm. The algorithm is simple. Assume that

$$A = \{a_{ij}\} \quad \text{and let} \quad \mu_i = \frac{1}{n} \sum_{j=1}^{n} a_{ij}$$

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-DP`(A,k)

Require: Matrix $A \in \mathbb{R}^{m \times n}$, number of row clusters k.
1: Create the vector $v = (\mu_1, \mu_2, \dots, \mu_m)$, where $\mu_i = \frac{1}{n}\sum_{j=1}^{n} a_{ij}$.
2: Use the dynamic programming algorithm of Lemma 3.11 and let $\mathcal{I}$ be the resulting k-clustering.
3: **return** $(\mathcal{I}, \{1, \dots, n\})$.

Theorem 3.12. *Let $A \in \mathbb{R}^{m \times n}$. Let $\mathcal{I}$ be the clustering produced under the $\|\cdot\|_F$ norm by Algorithm 2. Then $\mathcal{I}$ has optimal cost. The running time of the algorithm is $O(mn + m^2 k)$.*

Proof. Let us see the cost of a given cluster I, with $|I| = m_I$ rows. The mean μ_I of the cluster equals

$$\mu_I = \frac{1}{m_I n} \sum_{i \in I} \sum_{j=1}^{n} a_{ij} = \frac{1}{m_I} \sum_{i \in I} \mu_i .$$

The cost of the cluster is

$$\begin{aligned}\sum_{i\in I}\sum_{j=1}^{n}(a_{ij}-\mu_I)^2 &= \sum_{i\in I}\sum_{j=1}^{n}a_{ij}^2+m_In\mu_I^2-2\mu_I\sum_{i\in I}\sum_{j=1}^{n}a_{ij}\\ &= \sum_{i\in I}\sum_{j=1}^{n}(\mu_i+\varepsilon_{ij})^2+m_In\mu_I^2-2\mu_I^2m_In\\ &= \sum_{i\in I}\sum_{j=1}^{n}\mu_i^2+\sum_{i\in I}\sum_{j=1}^{n}\varepsilon_{ij}^2+2\sum_{i\in I}\mu_i\sum_{j=1}^{n}\varepsilon_{ij}-m_In\mu_I^2\\ &= n\sum_{i\in I}\mu_i^2+\sum_{i\in I}\sum_{j=1}^{n}\varepsilon_{ij}^2-m_In\mu_I^2,\end{aligned}$$

since $\sum_{j=1}^{n}\varepsilon_{ij}=0$, for all i, as we mentioned previously.

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

$$n\sum_{i=1}^{m}\mu_i^2+\sum_{i=1}^{m}\sum_{j=1}^{n}\varepsilon_{ij}^2-n\sum_{I\in\mathcal{I}}m_I\mu_I^2\,. \tag{3.15}$$

Consider now the one-dimensional problem of $(k,1)$ clustering only the row means μ_i. The cost of a given cluster I is:

$$\sum_{i\in I}(\mu_i-\mu_I)^2=\sum_{i\in I}\mu_i^2+m_I\mu_I^2-2\mu_I\sum_{i\in I}\mu_i=\sum_{i\in I}\mu_i^2-m_I\mu_I^2\,.$$

Thus the cost of the clustering is

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

Compare the cost of this clustering with that of (3.15). Note that in both cases the optimal row clustering is the one that maximizes the term $\sum_{I\in\mathcal{I}}m_I\mu_I^2$, 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 3.11 we conclude that we can solve the problem in time $O(mn+m^2k)$. □

4 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 fixed $0<\varepsilon<1$. While there are several results in the literature that show hardness of similar problems [31, 18, 4, 29], 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 4.1. *The problem of finding an optimal (k,ℓ) co-clustering under the ℓ_1 norm for a matrix $A\in\mathbb{R}^{m\times n}$ is NP-hard for $(k,\ell)=(k,n^{\varepsilon})$, for any $k\geq 2$ and any fixed $0<\varepsilon<1$.*

Proof. The proof contains several steps. First we reduce the one-sided s-median problem (where $s=n/3+o(n)$) under the ℓ_1 norm to the $(2,n/3+o(n))$-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 $(k,n/3+o(n))$, and this, finally, to the case of (k,n^{ε})-co-clustering. We now proceed with the details.

Hardness of $(2, n/3 + o(n))$ Megiddo and Supowit [27] show that the (one-sided) continuous (i. e., cluster centers can be arbitrary points in space) p-median problem is NP-hard under the ℓ_1 norm in $\mathbb{R}^2$. By looking carefully at the pertinent proof we can see that the problem is hard even if we restrict the number of clusters to be $n/3 + o(n)$; here, n is the number of points. Let us assume that we have such a problem instance of n points $\{b_j = (b_{j1}, b_{j2});\ j = 1, \dots, n\}$ 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 $\mathcal{J} = \{J_1, \dots, J_\ell\}$ of $\{1, \dots, n\}$, and points $\mu_1, \dots, \mu_\ell \in \mathbb{R}^2$ such that the objective

$$\sum_{J \in \mathcal{J}} \sum_{j \in J} \left\| b_j - \mu_J \right\|_1 = \sum_{J \in \mathcal{J}} \sum_{j \in J} \left| b_{j1} - \mu_{J1} \right| + \left| b_{j2} - \mu_{J2} \right| \tag{4.1}$$

is minimized.

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

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \end{bmatrix} = \begin{bmatrix} b_{11} & b_{21} & \cdots & b_{n1} \\ b_{12} & b_{22} & \cdots & b_{n2} \end{bmatrix},$$

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, $\mathcal{I} = \{\{1\}, \{2\}\}$ (strictly speaking, there is also the clustering $\{\{1, 2\}, \{\}\}$, which has higher cost unless $b_{j1} = b_{j2}$ for all j), and consider the column clustering $\mathcal{J} = \{J_1, \dots, J_\ell\}$ and the corresponding center matrix $M \in \mathbb{R}^{2 \times \ell}$. The cost of the solution equals

$$\sum_{I,J} \sum_{\substack{i \in I \\ j \in J}} \left| A_{ij} - M_{IJ} \right| = \sum_{J \in \mathcal{J}} \sum_{j \in J} \left| b_{j1} - M_{1J} \right| + \left| b_{j2} - M_{2J} \right|. \tag{4.2}$$

Note that this expression is minimized when the point (M_{1J}, M_{2J}) is the median of the points $b_j = (b_{j1}, b_{j2})$, $j \in J$, in which case the cost equals to that of (4.1), if the partitioning $\mathcal{J}$ is the same. 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.

Hardness of $(k, n/3 + o(n))$**,** $k > 2$ The next step is to show that it is NP-hard to (k, ℓ)-co-cluster any matrix of dimensions $m \times n$ for any $m, n, k > 2$ and $\ell = n/3 + o(n)$. To prove it we will use the previous hardness result for the $(2, \ell)$-co-clustering in $\mathbb{R}^{2 \times n}$. We start with an instance of the $(2, \ell)$-co-clustering problem in $\mathbb{R}^{2 \times n}$ given by a matrix A as described previously and we create a matrix $\tilde{A} = \{\tilde{A}_{ij}\} \in \mathbb{R}^{m \times n}$, by distorting one of the two dimensions and by adding to the previous matrix A, $m - 2$ rows of some value $4B$, where B is sufficiently large (say $B > 8m \max\{|A_{ij}|\}$):

$$\tilde{A} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} + B & A_{22} + B & \cdots & A_{2n} + B \\ 4B & 4B & \cdots & 4B \\ \vdots & \vdots & \ddots & \vdots \\ 4B & 4B & \cdots & 4B \end{bmatrix}.$$

Indeed, we can achieve a solution with the same cost as (4.2) by the same column partitioning $\mathcal{J}$ and a row partitioning that puts each of rows 1 and 2 to its own cluster and cluster the rest of the rows (where all the values equal $4B$) arbitrarily. Notice that this is an optimal solution since any other row cluster is of the following form: either it puts rows 1 and 2 in the same cluster, or it puts either row 1 or 2 with any of the other rows in the same cluster. In the first case, let I with $1,2 \in I$ be the row cluster and J be any column cluster. Then, if M_{IJ} is the median value of the entries in the cluster, we have that $M_{IJ} > B/2$, since half of the entries are at least B. Thus, there will be a cell $(1, j)$, $j \in J$ for which the residue will be

$$\left|A_{1j} - M_{IJ}\right| \;\geq\; M_{IJ} - \left|A_{1j}\right| \;>\; \frac{B}{2} - \max\{\left|A_{ij}\right|\} \;>\; \frac{B}{4},$$

since $B > 4\max\{\left|A_{ij}\right|\}$. In the second case, since half of the elements of a cluster have value at least $4B$, the median is at least $2B$, therefore there is a residue of a cell $\tilde{A}_{ij}$ for $i \in \{1,2\}$, $j \in J$ that is at least

$$\left|\tilde{A}_{ij} - M_{IJ}\right| \;\geq\; M_{IJ} - \left|\tilde{A}_{ij}\right| \;\geq\; 2B - B - \max\{\left|A_{ij}\right\}\right| \;>\; \frac{B}{4}.$$

Thus, in both cases, we incur a penalty of at least $B/4$, which is larger than that of (4.2), for $B > 8m\max\{\left|A_{ij}\right|\}$.

Hardness of (k, n^{ε}) 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 $\ell = n^{\varepsilon}$, for any fixed $0 < \varepsilon < 1$.

The construction is similar as before. Let $A' = \{A'_{ij}\}$. Define $n = (\ell'+1)^{1/\varepsilon}$ and let $A \in \mathbb{R}^{m\times n}$. For $n' > 3^{1/(1-\varepsilon)}$, $n > n'$. Furthermore, $n = O({n'}^{1/\varepsilon})$, hence the reduction is polynomial time. For $1 \leq j \leq n'$, define $A_{ij} = A'_{ij}$ and for any $j > n'$, define $A_{ij} = B'$, where B' is some sufficiently large value (e. g., $B' > 4mn\max|A'_{ij}|$):

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

Now, we only need to prove that the optimal solution of a $(k,\ell'+1) = (k,n^{\varepsilon})$-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 $j \leq \ell'$, and $J_{\ell'+1} = \{n'+1, n'+2,\ldots,n\}$. For matrix M we have $M_{IJ_j} = M'_{IJ_j}$ for $j \leq \ell'$ and $M_{IJ_{\ell'+1}} = B'$.

The cost $C(\mathcal{I},\mathcal{J})$ of the co-clustering equals

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

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 $y\le n'$. In this case we show that the resulting co-clustering cost will be much more than $C'(\mathcal{I}',\mathcal{J}')$. 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 (by the assumption on the value of B') $A_{1y}<B'/4$, we have that $|A_{1y}-x|>B'/4>C'(\mathcal{I}',\mathcal{J}')$. On the other hand if $x\le B'/2$ then $\left|A_{1j}-x\right|>B'/4>C'(\mathcal{I}',\mathcal{J}')$. Thus the cost of this solution is much larger than the cost of the optimal solution.

We have now established that any column $j>n'$ is not placed in the same cluster as $y\le n'$. Now, note that since the columns $\{n'+1,\ldots,n\}$ are clustered amongst themselves, their contribution to the total cost is 0, irrespectively of the number of column clusters used. Suppose we use $\ell''\le\ell'$ column clusters for the columns $\{1,\ldots,n'\}$, and $\ell'+1-\ell''$ clusters for $\{n'+1,\ldots,n\}$. Thus the total cost is equal to the cost of finding a (k,ℓ'')-co-clustering, $\ell''\le\ell'$, of the submatrix of $A[1,\ldots,m,1,\ldots,n']$. This is thus maximized by using $\ell''=\ell'$. Thus, the solution $(\mathcal{I},\mathcal{J})$ is optimal.

Note that $\ell'+1=n^{\varepsilon}$. Thus, solving the $(k,\ell'+1)=(k,n^{\varepsilon})$-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 $\ell=n^{\varepsilon}$. □

Observe that while we showed hardness for the ℓ_1 norm, our reduction can show hardness of co-clustering from hardness of one-sided clustering. The following corollary about co-clustering under the Frobenius norm is immediate.

Corollary 4.2. *For any fixed $\varepsilon>0$, any $k>1$ and $\ell>n^{\varepsilon}$, finding a (k,ℓ)-co-clustering that is optimal under the Frobenius norm is NP-hard.*

The proof follows by the above theorem and by hardness of the k-means objective shown in [11, 9, 8].

5 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 4 we showed that the co-clustering problem is NP-hard if $\ell = \Omega(n^\varepsilon)$ under the ℓ_1 and ℓ_2 norms. The restriction on ℓ arises from the fact that the base result on k-median hardness by Megiddo and Supowit [27] only works for $\Omega(n)$ clusters. Recent results by Dasgupta [9] and Aloise et al. [2] showed that k-means is hard for $k = 2$ when the number of dimensions is part of the input. Mahajan et al. [26] and Vattani [33] showed hardness for constant dimension and k being part of the input. Similar results for k-median are unresolved. For co-clustering, 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. It should also be interesting to extend the hardness results to any ℓ_p norm, $p \geq 1$; relevant to this are the hardness result for the k-center [14]. Similarly it would be interesting to see if we can obtain PTAS for co-clustering for some norms.

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 $\sqrt{3}/2$. Therefore, even for symmetric matrices, one-sided clustering cannot be used to obtain the optimal co-clustering.

Another 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 Banerjee et al. [3] 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 (2.3) using the residual definition of (2.2). For several problem domains this class of objective functions might be more appropriate than the one that we analyze here.

6 Acknowledgments

We would like to thank the anonymous reviewers of this article, whose constructive comments improved significantly the presentation of our results.

References

[1] DEEPAK AGARWAL AND SRUJANA MERUGU: Predictive discrete latent factor models for large scale dyadic data. In *Proc. 13th Internat. Conf. on Knowledge Discovery and Data Mining (KDD'07)*, pp. 26–35, 2007. [doi:10.1145/1281192.1281199] 599

[2] DANIEL ALOISE, AMIT DESHPANDE, PIERRE HANSEN, AND PREYAS POPAT: NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009. [doi:10.1007/s10994-009-5103-0] 617

[3] ARINDAM BANERJEE, INDERJIT S. DHILLON, JOYDEEP GHOSH, SRUJANA MERUGU, AND DHARMENDRA S. MODHA: A generalized maximum entropy approach to Bregman co-clustering and matrix approximation. *Journal of Machine Learning Research*, 8:1919–1986, 2007. Preliminary version in KDD'04. [ACM:1314498.1314563] 598, 599, 617

[4] NIKHIL BANSAL, AVRIM BLUM, AND SHUCHI CHAWLA: Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004. Preliminary version in FOCS'02. [doi:10.1023/B:MACH.0000033116.57574.95] 613

[5] PETER BRUCKER: On the complexity of clustering problems. In *Optimization and Operations Research: Proc. of a Workshop held at the University of Bonn*, pp. 45–54. Springer, 1977. 612

[6] MOSES CHARIKAR, SUDIPTO GUHA, ÉVA TARDOS, AND DAVID B. SHMOYS: A constant-factor approximation algorithm for the k-median problem. *J. Comput. System Sci.*, 65(1):129–149, 2002. Preliminary version in STOC'99. [doi:10.1006/jcss.2002.1882] 604, 605

[7] YIZONG CHENG AND GEORGE M. CHURCH: Biclustering of expression data. In *Proc. 8th Internat. Conf. on Intelligent Systems for Molecular Biology (ISMB'00)*, pp. 93–103. AAAI Press, 2000. [ACM:645635.660833] 598, 599

[8] HYUK CHO, INDERJIT S. DHILLON, YUQIANG GUAN, AND SUVRIT SRA: Minimum sum-squared residue co-clustering of gene expression data. In *Proc. 4th SIAM Internat. Conf. on Data Mining (SDM'04)*, pp. 114–125. SIAM, 2004. Accessible at http://siam.org/proceedings/datamining/2004/dm04_011choh.pdf. 598, 600, 616

[9] SANJOY DASGUPTA: The hardness of k-means clustering. Technical Report CS2008-0916, University of California, San Diego, 2008. Accessible at http://csetechrep.ucsd.edu/Dienst/UI/2.0/Describe/ncstrl.ucsd_cse/CS2008-0916. 616, 617

[10] INDERJIT S. DHILLON: Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. 7th Internat. Conf. on Knowledge Discovery and Data Mining (KDD'01)*, pp. 269–274. ACM Press, 2001. [doi:10.1145/502512.502550] 598, 599

[11] PETROS DRINEAS, ALAN M. FRIEZE, RAVI KANNAN, SANTOSH VEMPALA, AND V. VINAY: Clustering large graphs via the singular value decomposition. *Machine Learning*, 56(1–3):9–33, 2004. Preliminary version in SODA'99. [doi:10.1023/B:MACH.0000033113.59016.96] 616

[12] RICHARD O. DUDA, PETER E. HART, AND DAVID G. STORK: *Pattern Classification*. Wiley Interscience, 2nd edition, 2000. 597, 599

[13] URIEL FEIGE AND SHIMON KOGAN: Hardness of approximation of the balanced complete bipartite subgraph problem. Technical Report MCS04-04, Department of Computer Science and Applied Math., The Weizmann Institute of Science, 2004. Accessible at http://www.wisdom.weizmann.ac.il/math/reports/years/2004.shtml. 599

[14] ROBERT J. FOWLER, MIKE PATERSON, AND STEVEN L. TANIMOTO: Optimal packing and covering in the plane are NP-complete. *Inform. Process. Lett.*, 12(3):133–137, 1981. [doi:10.1016/0020-0190(81)90111-3] 617

[15] BIN GAO, TIE-YAN LIU, XIN ZHENG, QIAN-SHENG CHENG, AND WEI-YING MA: Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *Proc. 11th Internat. Conf. on Knowledge Discovery and Data Mining (KDD'05)*, pp. 41–50, 2005. [doi:10.1145/1081870.1081879] 598

[16] SREENIVAS GOLLAPUDI, RAVI KUMAR, AND D. SIVAKUMAR: Programmable clustering. In *Proc. 25th Symp. on Principles of Database Systems (PODS'06)*, pp. 348–354, 2006. [doi:10.1145/1142351.1142400, ACM:1142351.1142400] 599

[17] JOHN A. HARTIGAN: Direct clustering of a data matrix. *J. Amer. Stat. Assoc.*, 67(337):123–129, 1972. [doi:10.1080/01621459.1972.10481214] 598

[18] SAEED HASSANPOUR: Computational complexity of bi-clustering. Master's thesis, University of Waterloo, 2007. Accessible at http://uwspace.uwaterloo.ca/bitstream/10012/3359/1/Thesis.pdf. 598, 599, 613

[19] ANIL K. JAIN, M. NARASIMHA MURTY, AND PATRICK J. FLYNN: Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, 1999. [doi:10.1145/331499.331504] 599

[20] MICHEL JAMBU AND MARIE-ODILE LEBEAUX: *Cluster Analysis and Data Analysis*. Elsevier Science Ltd., 1983. 597, 599

[21] STEFANIE JEGELKA, SUVRIT SRA, AND ARINDAM BANERJEE: Approximation algorithms for tensor clustering. In *Proc. 20th Internat. Conf. on Algorithmic Learning Theory (ALT'09)*, pp. 368–383. Springer, 2009. [doi:10.1007/978-3-642-04414-4_30] 599

[22] JON KLEINBERG: An impossibility theorem for clustering. In *Proc. Advances in Neural Information Processing Systems (NIPS'02)*, pp. 446–453. MIT Press, 2002. 598, 599

[23] YUVAL KLUGER, RONEN BASRI, JOSEPH T. CHANG, AND MARK GERSTEIN: Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Research*, 13:703–716, 2003. [doi:10.1101/gr.648603] 598

[24] AMIT KUMAR, YOGISH SABHARWAL, AND SANDEEP SEN: A simple linear time $(1+\varepsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proc. 45th FOCS*, pp. 454–462. IEEE Comp. Soc. Press, 2004. [doi:10.1109/FOCS.2004.7] 604

[25] SARA C. MADEIRA AND ARLINDO L. OLIVEIRA: Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 1(1):24–45, 2004. [doi:10.1109/TCBB.2004.2] 598, 599

[26] MEENA MAHAJAN, PRAJAKTA NIMBHORKAR, AND KASTURI VARADARAJAN: The planar k-means problem is NP-hard. *Theoret. Comput. Sci.*, 442:13–21, 2012. Preliminary version in WALCOM'09. [doi:10.1016/j.tcs.2010.05.034] 617

[27] NIMROD MEGIDDO AND KENNETH J. SUPOWIT: On the complexity of some common geometric location problems. *SIAM J. Comput.*, 13(1):182–196, 1984. [doi:10.1137/0213014] 614, 617

[28] NINA MISHRA, DANA RON, AND RAM SWAMINATHAN: A new conceptual clustering framework. *Machine Learning*, 56(1-3):115–151, 2004. Preliminary version in COLT'03. [doi:10.1023/B:MACH.0000033117.77257.41] 599

[29] RENÉ PEETERS: The maximum edge biclique problem is NP-complete. *Discr. Appl. Math.*, 131(3):651–654, 2003. [doi:10.1016/S0166-218X(03)00333-0] 613

[30] KAI PUOLAMÄKI, SAMI HANHIJÄRVI, AND GEMMA C. GARRIGA: An approximation ratio for biclustering. *Inform. Process. Lett.*, 108(2):45–49, 2008. [doi:10.1016/j.ipl.2008.03.013] 599

[31] RON SHAMIR, RODED SHARAN, AND DEKEL TSUR: Cluster graph modification problems. *Discr. Appl. Math.*, 144(1–2):173–182, 2004. Preliminary version in WG'02. [doi:10.1016/j.dam.2004.01.007] 613

[32] AMOS TANAY, RODED SHARAN, AND RON SHAMIR: Biclustering algorithms: A survey. In SRINIVAS ALURU, editor, *Handbook of Computational Molecular Biology*. Chapman & Hall/CRC, Computer and Information Science Series, 2005. 599

[33] ANDREA VATTANI: The hardness of k-means clustering in the plane. Manuscript, accessible at http://cseweb.ucsd.edu/~avattani/papers/kmeans_hardness.pdf. 617

[34] JIONG YANG, HAIXUN WANG, WEI WANG, AND PHILIP S. YU: Enhanced biclustering on expression data. In *Proc. 3rd IEEE Conf. on Bioinformatics and Bioengineering (BIBE'03)*, pp. 321–327. IEEE Comp. Soc. Press, 2003. [doi:10.1109/BIBE.2003.1188969] 599

[35] JIONG YANG, WEI WANG, HAIXUN WANG, AND PHILIP S. YU: δ-clusters: Capturing subspace correlation in a large data set. In *Proc. 18th Internat. Conf. on Data Engineering (ICDE'02)*, pp. 517–528. IEEE Comp. Soc. Press, 2002. [doi:10.1109/ICDE.2002.994771] 600

[36] HANSON ZHOU AND DAVID P. WOODRUFF: Clustering via matrix powering. In *Proc. 23rd Symp. on Principles of Database Systems (PODS'04)*, pp. 136–142. ACM Press, 2004. [doi:10.1145/1055558.1055579] 599

AUTHORS

Aris Anagnostopoulos
Assistant Professor
Department of Computer, Control, and Management Engineering
Sapienza University of Rome, Italy

aris@dis.uniroma1.it
http://aris.me

Anirban Dasgupta
Yahoo!
Sunnyvale, CA

anirban.dasgupta@gmail.com
http://sites.google.com/site/anirbandasgupta

Ravi Kumar
Google
Mountain View, CA

ravi.k53@gmail.com
https://sites.google.com/site/ravik53

ABOUT THE AUTHORS

ARIS ANAGNOSTOPOULOS is an assistant professor at the Department of Computer, Control, and Management Engineering of the Sapienza University of Rome, Italy. Before joining the department he was a postdoctoral fellow at the Yahoo! Research labs in Santa Clara, CA. He graduated in 2000 in computer engineering and informatics from the University of Patras, Greece and in 2006 he obtained a Ph. D. in computer science from Brown University in the area of analysis of stochastic processes in computer science, under the supervision of Eli Upfal. His research interests lie in the broad areas of algorithms and probabilistic analysis, with emphasis on data-mining, web-mining, and social-network applications.

ANIRBAN DASGUPTA did his undergraduate studies at the Computer Science department of IIT Kharagpur, and joined the Cornell CS department as a graduate student in 2000. Anirban finished his Ph. D. in 2006 under the supervision of John Hopcroft, having worked on spectral methods for learning mixtures of distributions. Since then, Anirban has been employed as a scientist at Yahoo! Research. His research interests span linear algebraic techniques for information retrieval, algorithmic game theory, modeling of and algorithms for social networks, and the design and analysis of randomized and approximation algorithms in general.

RAVI KUMAR has been a senior staff research scientist at Google since June 2012. Prior to this, he was a research staff member at the IBM Almaden Research Center and a principal research scientist at Yahoo! Research. He obtained his Ph. D. in Computer Science from Cornell University in 1998, under the guidance of Ronitt Rubinfeld; his thesis topic was on program checking. His primary interests are web and data mining, social networks, algorithms for large data sets, and theory of computation.