Bipartite Correlation Clustering: Maximizing Agreements

Megasthenis Asteris
The University of Texas at Austin

Anastasios Kyrillidis
The University of Texas at Austin

Dimitris Papailiopoulos
University of California, Berkeley

Alexandros G. Dimakis
The University of Texas at Austin

Abstract

In *Bipartite Correlation Clustering* (BCC) we are given a complete *bipartite* graph G with '+' and '−' edges, and we seek a vertex clustering that maximizes the number of *agreements*: the number of all '+' edges within clusters plus all '−' edges cut across clusters. BCC is known to be NP-hard [5].

We present a novel approximation algorithm for k-BCC, a variant of BCC with an upper bound k on the number of clusters. Our algorithm outputs a k-clustering that provably achieves a number of agreements within a multiplicative $(1-\delta)$-factor from the optimal, for any desired accuracy δ. It relies on solving a combinatorially constrained bilinear maximization on the bi-adjacency matrix of G. It runs in time exponential in k and $1/\delta$, but linear in the size of the input.

Further, we show that in the (unconstrained) BCC setting, an $(1-\delta)$-approximation can be achieved by $O(\delta^{-1})$ clusters regardless of the size of the graph. In turn, our k-BCC algorithm implies an Efficient PTAS for the BCC objective of maximizing agreements.

1 Introduction

Correlation Clustering (CC) [7] considers the task of partitioning a set of objects into clusters based on their pairwise relationships. It arises naturally in several areas such as in network monitoring [1], document clustering [7] and textual similarity for data integration [10]. In its simplest form, objects are represented as vertices in a complete graph whose edges are labeled '+' or '−' to encode *similarity* or *dissimilarity* among vertices, respectively. The objective is to compute a vertex partition that maximizes the number of *agreements* with the underlying graph, *i.e.*, the total number of '+' edges in the interior of clusters plus the number of '−' edges across clusters. The number of output clusters is itself an optimization variable –not part of the input. It may be meaningful, however, to restrict the number of output clusters to be at most k. The constrained version is known as k-CC and similarly to the unconstrained problem, it is NP-hard [7, 13, 15]. A significant volume of work has focused on approximately solving the problem of maximizing the number of agreements (MaxAgree), or the equivalent –yet more challenging in terms of approximation– objective of minimizing the number of *disagreements* (MinDisagree) [7, 11, 9, 3, 19, 4].

Bipartite Correlation Clustering (BCC) Given a complete bipartite graph $G = (U, V, E)$ with edges labeled '+' or '−', the objective is once again to compute a clustering of the vertices that maximizes the number of agreements with the labeled pairs. BCC is a special case of the *incomplete* CC problem [9], where only a subset of vertices are connected and non-adjacent vertices do not affect the objective function. The output clusters may contain vertices from either one or both sides of the graph. Finally, we can define the k-BCC variant that enforces an upper bound on the number of output clusters, similar to k-CC for CC.

The task of clustering the vertices of a bipartite graph is common across many areas of machine learning. Applications include recommendation systems [18, 20], where analyzing the structure of a large sets of pairwise interactions (*e.g.*, among users and products) allows useful predictions about future interactions, gene expression data analysis [5, 16] and graph partitioning problems in data mining [12, 21].

Despite the practical interest in bipartite graphs, there is limited work on BCC and it is focused on the

Appearing in Proceedings of the 19^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 51.

theoretically more challenging MinDisagree objective: [5] established an 11-approximation algorithm, while [2] achieved a 4-approximation, the currently best known guarantee. Algorithms for incomplete CC [11, 9, 17] can be applied to BCC, but they do not leverage the structure of the bipartite graph. Moreover, existing approaches for incomplete CC rely on LP or SDP solvers which scale poorly.

Our contributions We develop a novel approximation algorithm for k-BCC with provable guarantees for the MaxAgree objective. Further, we show that under an appropriate configuration, our algorithm yields an Efficient Polynomial Time Approximation Scheme (EPTAS[1]) for the unconstrained BCC problem. Our contributions can be summarized as follows:

1. k-BCC: Given a bipartite graph $G = (U, V, E)$, a parameter k, and any constant accuracy parameter $\delta \in (0, 1)$, our algorithm computes a clustering of $U \cup V$ into at most k clusters and achieves a number of agreements that lies within a $(1 - \delta)$-factor from the optimal. It runs in time exponential in k and δ^{-1}, but linear in the size of G.
2. BCC: In the unconstrained BCC setting, the optimal number of clusters may be anywhere from 1 to $|U| + |V|$. We show that if one is willing to settle for a $(1 - \delta)$-approximation of the MaxAgree objective, it suffices to use at most $O(\delta^{-1})$ clusters, regardless of the size of G. In turn, under an appropriate configuration, our k-BCC algorithm yields an EPTAS for the (unconstrained) BCC problem.
3. Our algorithm relies on formulating the k-BCC/ MaxAgree problem as a combinatorially constrained bilinear maximization

$$\max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \quad \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}\mathbf{Y}\big)\,, \tag{1}$$

where $\mathbf{B}$ is the bi-adjacency matrix of G, and $\mathcal{X}$, $\mathcal{Y}$ are the sets of cluster assignment matrices for U and V, respectively. In Sec. 3, we briefly describe our approach for approximately solving (1) and its guarantees under more general constraints.

We note that our k-BCC algorithm and its guarantees can be extended to *incomplete*, but dense BCC instances, where the input G is *not* a complete bipartite graph, but $|E| = \Omega(|U| \cdot |V|)$ For simplicity, we restrict the description to the complete case. Finally, we supplement our theoretical findings with experimental results on synthetic and real datasets.

[1]EPTAS refers to an algorithm that approximates the solution of an optimization problem within a multiplicative $(1-\epsilon)$-factor, for any constant $\epsilon \in (0, 1)$, and has complexity that scales arbitrarily in $1/\epsilon$, but as a constant order polynomial (independent of ϵ) in the input size n. EPTAS is more efficient than a PTAS; for example, a running time of $O(n^{1/\epsilon})$ is considered a PTAS, but not an EPTAS.

1.1 Related work

There is extensive literature on CC; see [8] for a list of references. Here, we focus on bipartite variants.

BCC was introduced by Amit in [5] along with an 11-approximation algorithm for the MinDisagree objective, based on a linear program (LP) with $O(|E|^3)$ constraints. In [2], Ailon et al. proposed two algorithms for the same objective: *i*) a deterministic 4-approximation based on an LP formulation and derandomization arguments by [19], and *ii*) a randomized 4-approximation combinatorial algorithm. The latter is computationally more efficient with complexity scaling linearly in the size of the input, compared to the LP that has $O\big((|V| + |U|)^3\big)$ constraints.

For the *incomplete* CC problem, which encompasses BCC as a special case, [11] provided an LP-based $O(\log n)$-approximation for MinDisagree. A similar result was achieved by [9]. For the MaxAgree objective, [2, 17] proposed an SDP relaxation, similar to that for MAX k-CUT, and achieved a 0.7666-approximation. We are not aware of any results explicitly on the MaxAgree objective for either BCC or k-BCC. For comparison, in the k-CC setting [7] provided a simple 3-approximation algorithm for $k = 2$, while for $k \geq 2$ [13] provided a PTAS for MaxAgree and one for MinDisagree. [15] improved on the latter utilizing approximations schemes to the Gale-Berlekamp switching game. Table 1 summarizes the aforementioned results.

Finally, we note that our algorithm relies on adapting ideas from [6] for approximately solving a combinatorially constrained quadratic maximization.

2 k-BCC as a Bilinear Maximization

We describe the k-BCC problem and show that it can be formulated as a combinatorially constrained bilinear maximization on the bi-adjacency matrix of the input graph. Our k-BCC algorithm relies on approximately solving that bilinear maximization.

Maximizing Agreements An instance of k-BCC consists of an undirected, complete, bipartite graph $G = (U, V, E)$ whose edges have binary ± 1 weights, and a parameter k. The objective, referred to as MaxAgree$[k]$, is to compute a clustering of the vertices into at most k clusters, such that the total number of positive edges in the interior of clusters plus the number of negative edges across clusters is maximized.

Let E^+ and E^- denote the sets of positive and nega-

Ref.	Min./Max.	Guarantee	Complexity	D/R	Setting
[13]	MaxAgree	(E)PTAS	$n/\delta \cdot k^{O\left(\delta^{-2}\log k \log(1/\delta)\right)}$	R	k-CC
[13]	MinDisagree	PTAS	$n^{O(100^k/\delta^2)} \cdot \log n$	R	k-CC
[15]	MinDisagree	PTAS	$n^{O(9^k/\delta^2)} \cdot \log n$	R	k-CC
[11, 9]	MinDisagree	$O(\log n)$-OPT	LP	D	Inc. CC
[17]	MaxAgree	0.766-OPT	SDP	D	Inc. CC
[5]	MinDisagree	11-OPT	LP	D	BCC
[2]	MinDisagree	4-OPT	LP	D	BCC
[2]	MinDisagree	4-OPT	$\lvert E\rvert$	R	BCC
Ours	MaxAgree	(E)PTAS	$2^{O(k/\delta^2\cdot\log\sqrt{k}/\delta)} \cdot (\delta^{-2}+k)\cdot n + \mathrm{T}_{\mathsf{SVD}}(\delta^{-2})$	R	k-BCC
	MaxAgree	(E)PTAS	$2^{O(\delta^{-3}\cdot\log\delta^{-3})} \cdot O(\delta^{-2})\cdot n + \mathrm{T}_{\mathsf{SVD}}(\delta^{-2})$	R	BCC

Table 1: Summary of results on BCC and related problems. For each scheme, we indicate the problem setting, the objective (MaxAgree/MinDisagree), the guarantees (c-OPT implies a multiplicative factor approximation), and its computational complexity (n denotes the total number of vertices, LP/SDP denotes the complexity of a linear/semidefinite program, and $\mathrm{T}_{\mathsf{SVD}}(r)$ the time required to compute a rank-r truncated SVD of a $n \times n$ matrix). The D/R column indicates whether the scheme is deterministic or randomized.

tive edges, respectively. Also, let $m = |U|$ and $n = |V|$. The bipartite graph G can be represented by its weighted bi-adjacency matrix $\mathbf{B} \in \{\pm 1\}^{m\times n}$, where B_{ij} is equal to the weight of edge (i, j).

Consider a clustering $\mathcal{C}$ of the vertices $U \cup V$ into *at most* k clusters $C_1, \ldots, C_k$. Let $\mathbf{X} \in \{0, 1\}^{m\times k}$ be the cluster assignment matrix for the vertices of U associated with $\mathcal{C}$, that is, $X_{ij} = 1$ if and only if vertex $i \in U$ is assigned to cluster C_j. Similarly, let $\mathbf{Y} \in \{0, 1\}^{n\times k}$ be the cluster assignment for V.

Lemma 2.1. *For any instance $G = (U, V, E)$ of the k-BCC problem with bi-adjacency matrix $\mathbf{B} \in \{\pm 1\}^{|U|\times|V|}$, and for any clustering $\mathcal{C}$ of $U \cup V$ into k clusters, the number of agreements achieved by $\mathcal{C}$ is*

$$\textsc{Agree}(\mathcal{C}) = \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}\mathbf{Y}\big) + |E^-|,$$

where $\mathbf{X} \in \{0, 1\}^{|U|\times k}$ and $\mathbf{Y} \in \{0, 1\}^{|V|\times k}$ are the cluster assignment matrices corresponding to $\mathcal{C}$.

Proof. Let $\mathbf{B}^+$ and $\mathbf{B}^- \in \{0, 1\}^{|U|\times|V|}$ be indicator matrices for E^+ and E^-. Then, $\mathbf{B} = \mathbf{B}^+ - \mathbf{B}^-$. Given a clustering $\mathcal{C} = \{C_j\}_{j=1}^k$, or equivalently the assignment matrices $\mathbf{X} \in \{0, 1\}^{|U|\times k}$ and $\mathbf{Y} \in \{0, 1\}^{|V|\times k}$, the number of pairs of similar vertices assigned to the same cluster is equal to $\textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}^+\mathbf{Y}\big)$. Similarly, the number of pairs of dissimilar vertices assigned to different clusters is equal to $|E^-| - \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}^-\mathbf{Y}\big)$. The total number of agreements achieved by $\mathcal{C}$ is

$$\begin{aligned}\textsc{Agree}(\mathcal{C}) &= \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}^+\mathbf{Y}\big) + |E^-| - \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}^-\mathbf{Y}\big)\\ &= \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}\mathbf{Y}\big) + |E^-|,\end{aligned}$$

which is the desired result. □

It follows that computing a k-clustering that achieves the maximum number of agreements, boils down to a constrained bilinear maximization:

$$\textsc{MaxAgree}[k] = \max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \textsc{Tr}\big(\mathbf{X}^\top \mathbf{B}\mathbf{Y}\big) + |E^-|, \quad (2)$$

where

$$\begin{aligned}\mathcal{X} &\triangleq \big\{\mathbf{X} \in \{0, 1\}^{m\times k} : \|\mathbf{X}\|_{\infty,1} = 1\big\},\\ \mathcal{Y} &\triangleq \big\{\mathbf{Y} \in \{0, 1\}^{n\times k} : \|\mathbf{Y}\|_{\infty,1} = 1\big\}.\end{aligned} \quad (3)$$

Here, $\|\mathbf{X}\|_{\infty,1}$ denotes the maximum of the ℓ_1-norm of the rows of $\mathbf{X}$. Since $\mathbf{X} \in \{0, 1\}^{|U|\times k}$, the constraint ensures that each row of $\mathbf{X}$ has exactly one nonzero entry. Hence, $\mathcal{X}$ and $\mathcal{Y}$ describe the sets of valid cluster assignment matrices for U and V, respectively.

In the sequel, we briefly describe our approach for approximately solving the maximization in (2) under more general constraints, and subsequently apply it to the k-BCC/MaxAgree problem.

3 Bilinear Maximization Framework

We describe a simple algorithm for computing an approximate solution to the constrained maximization

$$\max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \quad \textsc{Tr}\big(\mathbf{X}^\top \mathbf{A}\mathbf{Y}\big), \quad (4)$$

where the input argument $\mathbf{A}$ is a real $m \times n$ matrix, and $\mathcal{X}$, $\mathcal{Y}$ are norm-bounded sets. Our approach is exponential in the rank of the argument $\mathbf{A}$, which can be prohibitive in practice. To mitigate this effect, we solve the maximization on a low-rank approximation $\widetilde{\mathbf{A}}$ of $\mathbf{A}$, instead. The quality of the output depends on the spectrum of $\mathbf{A}$ and the rank r of the surrogate matrix. Alg. 1 outlines our approach for solving (4), operating directly on the rank-r matrix $\widetilde{\mathbf{A}}$.

If we knew the optimal value for variable $\mathbf{Y}$, then the

optimal value for variable $\mathbf{X}$ would be the solution to

$$\mathrm{P}_{\mathcal{X}}(\mathbf{L}) = \arg\max_{\mathbf{X}\in\mathcal{X}} \mathrm{Tr}\big(\mathbf{X}^\top \mathbf{L}\big)\,, \tag{5}$$

for $\mathbf{L} = \widetilde{\mathbf{A}}\mathbf{Y}$. Similarly, with $\mathbf{R} = \widetilde{\mathbf{A}}^\top \mathbf{X}$ for a given $\mathbf{X}$,

$$\mathrm{P}_{\mathcal{Y}}(\mathbf{R}) = \arg\max_{\mathbf{Y}\in\mathcal{Y}} \mathrm{Tr}\big(\mathbf{R}^\top \mathbf{Y}\big) \tag{6}$$

is the optimal value of $\mathbf{Y}$ for that $\mathbf{X}$. Our algorithm requires that such a linear maximization oracles $\mathrm{P}_{\mathcal{X}}(\cdot)$ and $\mathrm{P}_{\mathcal{Y}}(\cdot)$ exist.[2] Of course, the optimal value for either of the two variables is not known. It is known, however, that the columns of the $m\times k$ matrix $\mathbf{L} = \widetilde{\mathbf{A}}\mathbf{Y}$ lie in the r-dimensional range of $\widetilde{\mathbf{A}}$ for all feasible $\mathbf{Y} \in \mathcal{Y}$. Alg. 1 effectively operates by sampling candidate values for $\mathbf{L}$. More specifically, it considers a large –exponential in $r \cdot k$– collection of $r \times k$ matrices $\mathbf{C}$ whose columns are points of an ϵ-net of the unit ℓ_2-ball $\mathbb{B}_2^{r-1}$. Each matrix $\mathbf{C}$ is used to generate a matrix $\mathbf{L}$ whose k columns lie in the range of the input matrix $\widetilde{\mathbf{A}}$ and in turn to produce a feasible solution pair $\mathbf{X}, \mathbf{Y}$ by successively solving (5) and (6). Due to the properties of the ϵ-net, one of the computed feasible pairs is guaranteed to achieve an objective value in (4) close to the optimal one (for argument $\widetilde{\mathbf{A}}$).

Lemma 3.2. *For any real $m \times n$, rank-r matrix $\widetilde{\mathbf{A}}$, and sets $\mathcal{X} \subset \mathbb{R}^{m\times k}$ and $\mathcal{Y} \subset \mathbb{R}^{n\times k}$, let*

$$\big(\widetilde{\mathbf{X}}_\star, \widetilde{\mathbf{Y}}_\star\big) \triangleq \arg\max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \mathrm{Tr}\big(\mathbf{X}^\top \widetilde{\mathbf{A}}\mathbf{Y}\big).$$

If there exist linear maximization oracles $\mathrm{P}_{\mathcal{X}}$ and $\mathrm{P}_{\mathcal{Y}}$ as in (5) and (6), then Alg. 1 with input $\widetilde{\mathbf{A}}$ and accuracy $\epsilon \in (0,1)$ outputs $\widetilde{\mathbf{X}} \in \mathcal{X}$ and $\widetilde{\mathbf{Y}} \in \mathcal{Y}$ such that

$$\mathrm{Tr}\big(\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{Y}}\big) \geq \mathrm{Tr}\big(\widetilde{\mathbf{X}}_\star^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{Y}}_\star\big) - 2\epsilon\sqrt{k}\cdot\|\widetilde{\mathbf{A}}\|_2 \cdot \mu_{\mathcal{X}} \cdot \mu_{\mathcal{Y}},$$

where $\mu_{\mathcal{X}} \triangleq \max_{\mathbf{X}\in\mathcal{X}} \|\mathbf{X}\|_{\mathrm{F}}$ and $\mu_{\mathcal{Y}} \triangleq \max_{\mathbf{Y}\in\mathcal{Y}} \|\mathbf{Y}\|_{\mathrm{F}}$, in time $O\big(\big(2\sqrt{r}/\epsilon\big)^{r\cdot k}\cdot\big(\mathrm{T}_{\mathcal{X}} + \mathrm{T}_{\mathcal{Y}} + (m+n)r\big)\big) + \mathrm{T}_{\mathsf{SVD}}(r)$. Here, $\mathrm{T}_{\mathsf{SVD}}(r)$ denotes the time required to compute the truncated SVD of $\widetilde{\mathbf{A}}$, while $\mathrm{T}_{\mathcal{X}}$ and $\mathrm{T}_{\mathcal{Y}}$ denote the running time of the two oracles.

The crux of Lemma 3.2 is that if there exists an efficient procedure to solve the simpler maximizations (5) and (6), then we can approximately solve the bilinear maximization (4) in time that depends exponentially on the intrinsic dimension r of the input $\widetilde{\mathbf{A}}$, but polynomially on the size of the input. A formal proof is given in Appendix Sec. A.

Recall that $\widetilde{\mathbf{A}}$ is only a low-rank approximation of the potentially full rank original argument $\mathbf{A}$. The guarantees of Lemma 3.2 can be translated to guarantees for the original matrix introducing an additional error term to account for the extra level of approximation.

[2] In the case of the k-BCC problem (2), where $\mathcal{X}$ and $\mathcal{Y}$ correspond to the sets of cluster assignment matrices defined in (3), such optimization oracles exist (see Alg. 2).

Algorithm 1 `BiLinearLowRankSolver`

input : $m \times n$ real matrix $\widetilde{\mathbf{A}}$ of rank r, $\epsilon \in (0,1)$
output $\widetilde{\mathbf{X}} \in \mathcal{X}$, $\widetilde{\mathbf{Y}} \in \mathcal{Y}$ {See Lemma 3.2.}

```
C ← {}                                              {Candidate solutions}
Ũ, Σ̃, Ṽ ← SVD(Ã)                                   { Σ̃ ∈ R^{r×r} }
for each C ∈ (ε-net of B_2^{r-1})^{⊗k} do
    L ← ŨΣ̃C                                         {L ∈ R^{m×k}}
    X ← P_X(L)
    R ← X^⊤ Ã                                       {R ∈ R^{k×n}}
    Y ← P_Y(R)
    C ← C ∪ {(X, Y)}
end for
(X̃, Ỹ) ← arg max_{(X,Y)∈C} Tr(X^⊤ ŨΣ̃Ṽ^⊤ Y)
```

Algorithm 2 $\mathrm{P}_{\mathcal{X}}(\cdot)$, $\mathcal{X} \triangleq \{\mathbf{X} \in \{0,1\}^{m\times k} : \|\mathbf{X}\|_{\infty,1} = 1\}$

input : $m \times k$ real matrix $\mathbf{L}$
output $\widetilde{\mathbf{X}} = \arg\max_{\mathbf{X}\in\mathcal{X}} \mathrm{Tr}\big(\mathbf{X}^\top \mathbf{L}\big)$

```
X̃ ← 0_{m×k}
for i = 1, ..., m do
    j_i ← arg max_{j∈[k]} L_{ij}
    X̃_{i j_i} ← 1
end for
```

Lemma 3.3. *For any $\mathbf{A} \in \mathbb{R}^{m\times n}$, let*

$$\big(\mathbf{X}_\star, \mathbf{Y}_\star\big) \triangleq \arg\max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \mathrm{Tr}\big(\mathbf{X}^\top \mathbf{A}\mathbf{Y}\big),$$

where $\mathcal{X}$ and $\mathcal{Y}$ satisfy the conditions of Lemma 3.2. Let $\widetilde{\mathbf{A}}$ be a rank-r approximation of $\mathbf{A}$, and $\widetilde{\mathbf{X}} \in \mathcal{X}$, $\widetilde{\mathbf{Y}} \in \mathcal{Y}$ be the output of Alg. 1 with input $\widetilde{\mathbf{A}}$ and accuracy ϵ. Then,

$$\begin{aligned}&\mathrm{Tr}\big(\mathbf{X}_\star^\top \mathbf{A}\mathbf{Y}_\star\big) - \mathrm{Tr}\big(\widetilde{\mathbf{X}}^\top \mathbf{A}\widetilde{\mathbf{Y}}\big)\\ &\quad\leq 2\cdot\Big(\epsilon\sqrt{k}\cdot\|\widetilde{\mathbf{A}}\|_2 + \|\mathbf{A}-\widetilde{\mathbf{A}}\|_2\Big)\cdot\mu_{\mathcal{X}}\cdot\mu_{\mathcal{Y}}.\end{aligned}$$

Lemma 3.3 follows from Lemma 3.2. A formal proof is deferred to Appendix Sec. A. This concludes the brief discussion on our bilinear maximization framework.

4 Our k-BCC Algorithm

The k-BCC/MaxAgree problem on a bipartite graph $G = (U, V, E)$ can be written as a constrained bilinear maximization (2) on the bi-adjacency matrix $\mathbf{B}$ over the sets of valid cluster assignment matrices (3) for V and U. Adopting the framework of the previous section to approximately solve the maximization in (2) we obtain our k-BCC algorithm.

The missing ingredient is a pair of efficient procedures $\mathrm{P}_{\mathcal{X}}(\cdot)$ and $\mathrm{P}_{\mathcal{Y}}(\cdot)$, as described in Lemma 3.2 for solving (5) and (6), respectively, in the case where $\mathcal{X}$ and

$\mathcal{Y}$ are the sets of valid cluster assignment matrices (3). Such a procedure exists and is outlined in Alg. 2.

Lemma 4.4. *For any $\mathbf{L} \in \mathbb{R}^{m\times k}$, Algorithm 2 outputs*

$$\arg\max_{\mathbf{X}\in\mathcal{X}} \mathrm{Tr}\big(\mathbf{X}^\top \mathbf{L}\big),$$

where $\mathcal{X}$ is defined in (3)*, in time $O(k \cdot m)$.*

A proof is provided in Appendix, Sec. B.

Note that in our case, Alg. 2 is used as both $\mathrm{P}_{\mathcal{X}}(\cdot)$ and $\mathrm{P}_{\mathcal{Y}}(\cdot)$. Putting the pieces together, we obtain the core of our k-BCC algorithm, outlined in Alg. 3. Given a bipartite graph G, with bi-adjacency matrix $\mathbf{B}$, we first compute a rank-r approximation $\widetilde{\mathbf{B}}$ of $\mathbf{B}$ via the truncated SVD. Using Alg. 1, equipped with Alg. 2 as a subroutine, we approximately solve the bilinear maximization (2) with argument $\widetilde{\mathbf{B}}$. The output is a pair of valid cluster assignment matrices for U and V.

Alg. 3 exposes the configuration parameters r and ϵ that control the performance of our bilinear solver, and in turn the quality of the output k-clustering. To simplify the description, we also create a k-BCC "wrapper" procedure outlined in Alg. 4: given a bound k on the number of clusters and a single accuracy parameter $\delta \in (0,1)$, Alg. 4 configures and invokes Alg. 3.

Theorem 1. (k-BCC.) *For any instance $\big(G = (U,V,E), k\big)$ of the k-BCC problem with bi-adjacency matrix $\mathbf{B}$ and for any desired accuracy parameter $\delta \in (0,1)$, Algorithm 4 computes a clustering $\widetilde{\mathcal{C}}^{(k)}$ of $U \cup V$ into at most k clusters, such that*

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}^{(k)}\big) \geq \big(1-\delta\big) \cdot \textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big),$$

where $\mathcal{C}_\star^{(k)}$ is the optimal k-clustering, in time $2^{O(k/\delta^2 \cdot \log\sqrt{k}/\delta)} \cdot (\delta^{-2}+k)\cdot(|U|+|V|) + \mathrm{T}_{\mathsf{SVD}}(\delta^{-2})$.

In the remainder of this section, we prove Theorem 1. We begin with the core Alg. 3, which invokes the low-rank bilinear solver Alg. 1 with accuracy parameters ϵ and r on the rank-r matrix $\widetilde{\mathbf{B}}$ and computes a pair of valid cluster assignment matrices $\widetilde{\mathbf{X}}^{(k)}$, $\widetilde{\mathbf{Y}}^{(k)}$. By Lemma 3.2, and taking into account that $\sigma_1(\widetilde{\mathbf{B}}) = \sigma_1(\mathbf{B}) \leq \|\mathbf{B}\|_{\mathrm{F}} \leq \sqrt{mn}$, and the fact that $\|\mathbf{X}\|_F = \sqrt{m}$ and $\|\mathbf{Y}\|_F = \sqrt{n}$ for all valid cluster assignment matrix pairs, the output pair satisfies

$$\mathrm{Tr}\big(\widetilde{\mathbf{X}}_\star^{(k)\top}\widetilde{\mathbf{B}}\widetilde{\mathbf{Y}}_\star^{(k)}\big) - \mathrm{Tr}\big(\widetilde{\mathbf{X}}^{(k)\top}\widetilde{\mathbf{B}}\widetilde{\mathbf{Y}}^{(k)}\big) \leq 2\cdot\epsilon\cdot\sqrt{k}\cdot mn,$$

where $\big(\widetilde{\mathbf{X}}_\star^{(k)}, \widetilde{\mathbf{Y}}_\star^{(k)}\big) \triangleq \max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \mathrm{Tr}\big(\mathbf{X}^\top\widetilde{\mathbf{B}}\mathbf{Y}\big)$.

In turn, by Lemma 3.3 taking into account that $\|\mathbf{B}-\widetilde{\mathbf{B}}\|_2 \leq \sqrt{mn}/\sqrt{r}$ (see Cor. 1), on the original bi-adjacency matrix $\mathbf{B}$ the output pair satisfies

$$\mathrm{Tr}\big(\mathbf{X}_\star^{(k)\top}\mathbf{B}\mathbf{Y}_\star^{(k)}\big) - \mathrm{Tr}\big(\widetilde{\mathbf{X}}^{(k)\top}\mathbf{B}\widetilde{\mathbf{Y}}^{(k)}\big) \leq 2\epsilon\sqrt{k}\cdot mn + 2(r+1)^{-1/2}\cdot mn, \quad (7)$$

where $\big(\mathbf{X}_\star^{(k)}, \mathbf{Y}_\star^{(k)}\big) \triangleq \max_{\mathbf{X}\in\mathcal{X},\mathbf{Y}\in\mathcal{Y}} \mathrm{Tr}\big(\mathbf{X}^\top\mathbf{B}\mathbf{Y}\big)$. The unknown optimal pair $\mathbf{X}_\star^{(k)}, \mathbf{Y}_\star^{(k)}$ represents the optimal k-clustering $\mathcal{C}_\star^{(k)}$, *i.e.*, the clustering that achieves the maximum number of agreements using at most k clusters. Similarly, let $\widetilde{\mathcal{C}}^{(k)}$ be the clustering induced by the computed pair $\widetilde{\mathbf{X}}^{(k)}$, $\widetilde{\mathbf{Y}}^{(k)}$. From (7) and Lemma 2.1, it immediately follows that

$$\textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big) - \textsc{Agree}\big(\widetilde{\mathcal{C}}^{(k)}\big) \leq 2\cdot(\epsilon\sqrt{k}+\sqrt{r+1})\cdot mn. \quad (8)$$

Eq. 8 establishes the guarantee of Alg. 3 as a function of its accuracy parameters r and ϵ. Substituting those parameters by the values assigned by Alg. 4, we obtain

$$\textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big) - \textsc{Agree}\big(\widetilde{\mathcal{C}}^{(k)}\big) \leq 2^{-1}\cdot\delta\cdot mn. \quad (9)$$

Fact 1. *For any* BCC *instance (or k-*BCC *with $k \geq 2$), the optimal clustering achieves at least $mn/2$ agreements.*

Proof. If more than half of the edges are labeled '+', a single cluster containing all vertices achieves at least $nm/2$ agreements. Otherwise, two clusters corresponding to U and V achieve the same result. □

In other words, Fact 1 states that

$$\textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big) \geq mn/2, \quad \forall k \geq 2. \quad (10)$$

Combining (10) with (9), we obtain

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}^{(k)}\big) \geq (1-\delta)\cdot\textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big),$$

which is the desired result. Finally, the computational complexity is obtained substituting the appropriate values in that of Alg. 1 as in Lemma 3.2, and the time complexity of subroutine Alg. 2 given in Lemma 4.4. This completes the proof of Theorem 1.

5 An Efficient PTAS for BCC

We provide an efficient polynomial time approximation scheme (EPTAS) for BCC/MaxAgree, *i.e.*, the unconstrained version with no upper bound on the number of output clusters. We rely on the following key observation: *any constant factor approximation of the* MaxAgree *objective, can be achieved by constant number of clusters.*[3] Hence, the desired approximation algorithm for BCC can be obtained by invoking the k-BCC algorithm under the appropriate configuration.

Recall that in the unconstrained BCC setting, the number of output clusters is an optimization variable; the optimal clustering may comprise any number of clusters, from a single cluster containing all vertices of

[3] A similar result for CC/MaxAgree exists in [7].

Algorithm 3 k-BCC via Low-rank Bilinear Maximization

input : • Bi-adjacency matrix $\mathbf{B} \in \{\pm 1\}^{m,n}$ of bipartite graph $G = (U, V, E)$,
• Target number of clusters k, • Approximation rank r, • Accuracy parameter $\epsilon \in (0, 1)$.
output Clustering $\widetilde{\mathcal{C}}^{(k)}$ of $U \cup V$ into k clusters –
Equivalently, cluster membership matrices $\widetilde{\mathbf{X}}^{(k)} \in \mathcal{X}$ and $\widetilde{\mathbf{Y}}^{(k)} \in \mathcal{Y}$ for U and V, respectively.
1: $\widetilde{\mathbf{B}} \leftarrow \mathsf{SVD}(\mathbf{B}, r)$ {Truncated SVD.}
2: Approx. solve the bilinear maximization (2) on $\widetilde{\mathbf{B}}$, over the sets $\mathcal{X}$, $\mathcal{Y}$ of valid cluster assignment matrices (Eq. 3):
$\widetilde{\mathbf{X}}^{(k)}, \widetilde{\mathbf{Y}}^{(k)} \leftarrow \texttt{BiLinearLowRankSolver}\big(\widetilde{\mathbf{B}}, \epsilon, r\big)$ {Alg. 1, using Alg. 2 as $\mathrm{P}_{\mathcal{X}}(\cdot)$ and $\mathrm{P}_{\mathcal{Y}}(\cdot)$.}

Algorithm 4 k-BCC/MaxAgree

input : Bi-adjacency matrix $\mathbf{B} \in \{\pm 1\}^{m,n}$,
Target number of k,
Accuracy $\delta \in (0, 1)$.
output Clustering $\widetilde{\mathcal{C}}^{(k)}$ of $U \cup V$ such that
$\textsc{Agree}\big(\widetilde{\mathcal{C}}^{(k)}\big) \geq \big(1 - \delta\big) \cdot \textsc{Agree}\big(\mathcal{C}_\star^{(k)}\big)$.
1: Set up parameters:
$\epsilon \leftarrow 2^{-3} \cdot \delta \cdot k^{-1/2}, \quad r \leftarrow 2^6 \cdot \delta^{-2} - 1$.
2: Return output of Alg. 3 for input $(\mathbf{B}, k, r, \epsilon)$.

Algorithm 5 A PTAS for BCC/MaxAgree

input : Bi-adjacency matrix $\mathbf{B} \in \{\pm 1\}^{m,n}$,
Accuracy $\delta \in (0, 1)$.
output Clustering $\widetilde{\mathcal{C}}$ of $U \cup V$ (into at most
$2^3 \cdot \delta^{-1}$ clusters) such that
$\textsc{Agree}\big(\widetilde{\mathcal{C}}\big) \geq \big(1 - \delta\big) \cdot \textsc{Agree}\big(\mathcal{C}_\star\big)$.
1: Set up parameters:
$k \leftarrow 2^3 \cdot \delta^{-1}, \quad \epsilon \leftarrow 2^{-6} \cdot \delta^2, \quad r \leftarrow 2^8 \cdot \delta^{-2} - 1$.
2: Return output of Alg. 3 for input $(\mathbf{B}, k, r, \epsilon)$.

the graph, up to $|U| + |V|$ singleton clusters. In principle, one could solve MaxAgree$[k]$ for all possible values of the parameter k to identify the best clustering, but this approach is computationally intractable.

On the contrary, if one is willing to settle for an approximately optimal solution, then a constant number of clusters may suffice. More formally,

Lemma 5.5. *For any* BCC *instance, and* $0 < \epsilon \leq 1$, *there exists a clustering* $\mathcal{C}$ *with at most* $k = 2 \cdot \epsilon^{-1} + 2$ *clusters such that* $\textsc{Agree}(\mathcal{C}) \geq \textsc{Agree}(\mathcal{C}_\star) - \epsilon \cdot nm$, *where* $\mathcal{C}_\star$ *denotes the optimal clustering.*

We defer the proof to the end of the section.

In conjunction with Fact 1, Lemma 5.5 suggests that to obtain a constant factor approximation for the unconstrained BCC/MaxAgree problem, for any constant arbitrarily close to 1, it suffices to solve a k-BCC instance for a sufficiently large –but constant– number of clusters k. In particular, to obtain an $(1 - \delta)$-factor approximation for any constant $\delta \in (0, 1)$, it suffices to solve the k-BCC problem with an upper bound $k = O(\delta^{-1})$ on the number of clusters.

Alg. 5 outlines the approximation scheme for BCC. For a given accuracy parameter δ, it invokes Alg. 3 under an appropriate configuration of the parameters k, r and ϵ, yielding the following guarantees:

Theorem 2. (PTAS for BCC.) *For any instance* $G = (U, V, E)$ *of the* BCC *problem with bi-adjacency matrix* $\mathbf{B}$ *and for any desired accuracy parameter* $\delta \in (0, 1)$, *Algorithm 5 computes a clustering* $\widetilde{\mathcal{C}}$ *of* $U \cup V$ *into (at most)* $2^3 \cdot \delta^{-1}$ *clusters, such that*

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}\big) \geq \big(1 - \delta\big) \cdot \textsc{Agree}\big(\mathcal{C}_\star\big),$$

where $\mathcal{C}_\star$ *is an optimal clustering (with no constraint on the number of clusters), in time*

$$2^{O(\delta^{-3} \cdot \log \delta^{-3})} \cdot \delta^{-2} \cdot (m + n) + \mathrm{T}_{\mathsf{SVD}}(\delta^{-2}).$$

The proof of Theorem 2 follows from the guarantees of Alg. 3 in (8) substituting the values of the parameters k, r and ϵ with the values specified by Alg 5. The core k-BCC Alg. 3 returns a clustering $\widetilde{\mathcal{C}}$ with at most $k_0 = 2^3 \cdot \delta^{-1}$ clusters that satisfies

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}\big) \geq \textsc{Agree}\big(\mathcal{C}_\star^{(k_0)}\big) - \tfrac{\delta}{4} \cdot mn, \tag{11}$$

where $\mathcal{C}_\star^{(k_0)}$ is the best among the clusterings using at most k_0 clusters. Also, for $k = k_0$, Lemma 5.5 implies that $\textsc{Agree}(\mathcal{C}_\star^{(k_0)}) \geq \textsc{Agree}(\mathcal{C}_\star) - \delta/4 \cdot nm$, where $\mathcal{C}_\star$ is the optimal clustering without any constraint on the number of output clusters. Hence,

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}\big) \geq \textsc{Agree}\big(\mathcal{C}_\star\big) - \delta/2 \cdot mn. \tag{12}$$

Continuing from (12), and taking into account Fact 1, we conclude that the output $\widetilde{\mathcal{C}}$ of Alg. 5 satisfies

$$\textsc{Agree}\big(\widetilde{\mathcal{C}}\big) \geq (1 - \delta) \cdot \textsc{Agree}\big(\mathcal{C}_\star\big), \tag{13}$$

which is the desired result. Finally, the computational complexity of Alg. 5 follows from that of Alg. 3 substituting the parameter values in Lemma 3.2, taking into account the running time $\mathrm{T}_{\mathcal{X}} = O(\delta^{-2} \cdot m)$ of of Alg. 2 used as subroutine $\mathrm{P}_{\mathcal{X}}(\cdot)$ and similarly $\mathrm{T}_{\mathcal{Y}} = O(\delta^{-2} \cdot n)$ for $\mathrm{P}_{\mathcal{Y}}(\cdot)$. This concludes the proof of Theorem 2.

For any desired constant accuracy $\delta \in (0, 1)$, Alg. 5 outputs a clustering that achieves a number of agreements within a $(1 - \delta)$-factor from the optimal, in time

that grows exponentially in δ^{-1}, but linearly in the size mn of the input. In other words, Alg. 5 is an EPTAS for BCC/MaxAgree.

We complete the section with a proof for Lemma 5.5, which stated that a constant number of clusters suffices to obtain an approximately optimal solution.

5.1 Proof of Lemma 5.5

It suffices to consider $\epsilon > 2/(m+n-2)$ as the lemma holds trivially otherwise. Without loss of generality, we focus on clusterings whose all but at most two clusters contain vertices from both U and V; one of the two remaining clusters can contain vertices only from U and the other only from V. To verify that, consider an arbitrary clustering $\mathcal{C}$ and let $\mathcal{C}'$ be the clustering obtained by merging all clusters of $\mathcal{C}$ containing only vertices of U into a single cluster, and those containing only vertices of V into another. The number of agreements is not affected, *i.e.*, $\textsc{Agree}(\mathcal{C}') = \textsc{Agree}(\mathcal{C})$.

We show that a clustering $\mathcal{C}$ with the properties described in the lemma exists by modifying $\mathcal{C}_\star$. Let $\mathcal{S}_U \subseteq \mathcal{C}_\star$ be the set of clusters that contain at most $\epsilon m/2$ vertices of U, and $\mathcal{S}_V \subseteq \mathcal{C}_\star$ those containing at most $\epsilon n/2$ vertices of V. ($\mathcal{S}_U$ and $\mathcal{S}_V$ may not be disjoint). Finally, let $\mathcal{B}$ be the set of vertices contained in clusters of $\mathcal{S}_U \cup \mathcal{S}_V$. We construct $\mathcal{C}$ as follows. Clusters of $\mathcal{C}_\star$ not in $\mathcal{S}_U \cup \mathcal{S}_V$ are left intact. Each such cluster has size at least $\epsilon(n+m)/2$. Further, all vertices in $\mathcal{B}$ are rearranged into two clusters: one for the vertices in $\mathcal{B} \cap U$ and one those in $\mathcal{B} \cap V$.

The above rearrangement can reduce the number of agreements by at most ϵnm. To verify that, consider a cluster C in $\mathcal{S}_U$; the cluster contains at most $\epsilon m/2$ vertices of U. Let $t \triangleq |V \cap C|$. Splitting C into two smaller clusters $C \cap U$ and $C \cap V$ can reduce the number of agreements by $\frac{1}{2}\epsilon mt$, *i.e.*, the total number of edges in C. Note that agreements on $-$ edges are not affected by splitting a cluster. Performing the same operation on all clusters in $\mathcal{S}_U$ incurs a total reduction

$$\tfrac{1}{2}\epsilon m \sum_{C \in \mathcal{S}_U} |V \cap C| \leq \tfrac{1}{2}\epsilon m |V| = \tfrac{1}{2}\epsilon mn.$$

Repeating on $\mathcal{S}_V$ incurs an additional cost of at most $\epsilon mn/2$ agreements, while merging all clusters containing only vertices from U (similarly for V) into a single cluster does not reduce the agreements. We conclude that $\textsc{Agree}(\mathcal{C}) \geq \textsc{Agree}(\mathcal{C}_\star) - \epsilon nm$.

Finally, by construction all but at most two clusters in $\mathcal{C}$ contain at least $\frac{1}{2}\epsilon(n+m)$. In turn,

$$n + m = \sum_{C \in \mathcal{C}} |C| \geq (|\mathcal{C}| - 2) \cdot \tfrac{1}{2}\epsilon(n+m),$$

from which the desired result follows. □

6 Experiments

We evaluate our algorithm on synthetic and real data and compare with PivotBiCluster [2] and the SDP-based approach of [17][4] Although [5, 9, 11] provide algorithms applicable to BCC, those rely on LP or SDP formulations with a high number of constraints which imposes limitations in practice.

We run our core k-BCC algorithm (Alg. 3) to obtain a k-clustering. Disregarding the theoretical guarantees, we apply a threshold on the execution time which can only hurt the performance; equivalently, the iterative procedure of Alg. 1 is executed for an arbitrary subset of the points in the ϵ-net (Alg. 1, step 3).

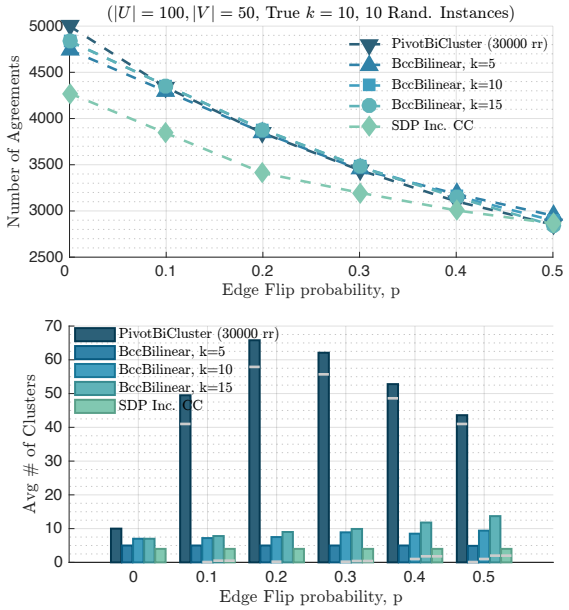
6.1 Synthetic Data

We generate synthetic BCC instances as follows. We arbitrarily set $m = 100$, $n = 50$, and number of clusters $k' = 5$, and construct a complete bipartite graph $G = (U, V, E)$ with $|U| = m$, $|V| = n$ and assign binary ± 1 labels to the edges according to a random k'-clustering of the vertices. Then, we modify the sign of each label independently with probability p.

We consider multiple values of p in the range $[0, 0.5]$. For each value, we generate 10 random instances as described above and compute a vertex clustering using all algorithms. PivotBiCluster returns a clustering of arbitrary size; no parameter k is specified and the algorithm determines the number of clusters in an attempt to maximize the number of agreements. The majority of the output clusters are singletons which can in principle be merged into two clusters (see Subsec. 5.1). The algorithm is substantially faster that the other approaches on examples of this scale. Hence, we run PivotBiCluster with $3 \cdot 10^4$ random restarts on each instance and depict best results to allow for comparable running times. Contrary to PivotBiCluster, for our algorithm, which is reffered to as BccBilinear, we need to specify the target number k of clusters. We run it for $k = 5, 10$ and 15 and arbitrarily set the parameter $r = 5$ and terminate our algorithm after 10^4 random samples/rounds. Finally, the SDP-based approach returns 4 clusters by construction, and is configure to outpute best results among 100 random pivoting steps. Fig. 1 depicts the number of agreements achieved by each algorithm for each value of p, the number of output clusters, and the execution times. All numbers are averages over multiple random instances.

We note that in some cases and contrary to intuition, our algorithm performs better for lower values of k, the target number of clusters. We attribute this phe-

[4] All algorithms are prototypically implemented in Matlab. [17] was implemented using CVX.



	$p = 0.0$	0.1	0.2	0.3	0.4	0.5
PivotBiCluster [2]	124	83	66	54	45	39
BccBil. (k=5)	69	70	70	71	71	71
BccBil. (k=10)	73	74	75	75	75	75
BccBil. (k=15)	78	79	79	80	80	80
SDP Inc CC [17]	198	329	257	258	266	314

Average runtimes/instance (in seconds).

Figure 1: Synthetic data. We generate an $m \times n$ bipartite graph with edge weights ± 1 according to a random vertex k-clustering and subsequently flip the sign of each edge with probability p. We plot the average number of agreements achieved by each method over 10 random instances for each p value. The bar plot depicts the average number of output clusters for each scheme/p-value pair. Within each bar, a horizontal line marks the number of singleton clusters.

nomenon to the fact that for higher values of k, we should typically use higher approximation rank and consider larger number of samples.

6.2 MovieLens Dataset

The MovieLens datasets [14] are sets of movie ratings: each of m users assigns scores in $\{1, \ldots, 5\}$ to a small subset of the n movies. Table 2 lists the dimensions of the datasets. From each dataset, we generate an instance of the (incomplete) BCC problem as follows: we construct the bipartite graph on the user-movie pairs using the ratings as edge weights, compute the average weight and finally set weights higher than the average to $+1$ and the others to -1.

We run our algorithm to obtain a clustering of the vertices. For a reference, we compare to PivotBiCluster with 50 random restarts. Note, however, that PivotBiCluster is not designed for the incomplete BCC problem; to apply the algorithm, we effectively treat missing edges as edges of negative weight. Finally, the SDP approach of [17], albeit suitable for the incomplete CC problem, does not scale to this size of input. Table 3 lists the number of agreements achieved by each method on each one of the three datasets and the corresponding execution times.

Dataset	m (Users)	n (Movies)	Ratings
MovieLens100K	1000	1700	10^5
MovieLens1M	6000	4000	10^6
MovieLens10M	72000	10000	10^7

Table 2: Summary of MovieLens datasets [14].

7 Conclusions

We presented the first algorithm with provable approximation guarantees for k-BCC/MaxAgree. Our approach relied on formulating k-BCC as a constrained bilinear maximization over the sets of cluster assignment matrices and developing a simple framework to approximately solve that combinatorial optimization.

In the unconstrained BCC setting, with no bound on the number of output clusters, we showed that any constant multiplicative factor approximation for the MaxAgree objective can be achieved using a constant number of clusters. In turn, under the appropriate configuration, our k-BCC algorithm yields an Efficient PTAS for BCC/MaxAgree.

Acknowledgments This research has been supported by NSF Grants CCF 1344179, 1344364, 1407278, 1422549 and ARO YIP W911NF-14-1-0258. DP is supported by NSF awards CCF-1217058 and CCF-1116404 and MURI AFOSR grant 556016.

MovieLens	100K	1M	10M
PivotBiCluster	46134	429277	5008577
	(27.95)	(651.13)	$(1.5 \cdot 10^5)$
BccBilinear	68141	694366	6857509
	(6.65)	(19.50)	$(1.2 \cdot 10^3)$

Table 3: Number of agreements achieved by the two algorithms on incomplete (k)-BCC instances obtained from the MovieLens datasets [14]. For PivotBiCluster we present best results over 50 random restarts. Our algorithm was arbitrarily configured with $r = 4$, $k = 10$ and a limit of 10^4 samples (iterations). We also note in parenteses the runtimes in seconds.

References

[1] KookJin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation clustering in data streams. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2237–2246, 2015.

[2] Nir Ailon, Noa Avigdor-Elgrabli, Edo Liberty, and Anke Van Zuylen. Improved approximation algorithms for bipartite correlation clustering. *SIAM Journal on Computing*, 41(5):1110–1121, 2012.

[3] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23, 2008.

[4] Nir Ailon and Edo Liberty. Correlation clustering revisited: The "true" cost of error minimization problems. In *Automata, Languages and Programming*, pages 24–36. Springer, 2009.

[5] Noga Amit. The bicluster graph editing problem. Master's thesis, Tel Aviv University, 2004.

[6] Megasthenis Asteris, Dimitris Papailiopoulos, Anastasios Kyrillidis, and Alexandros G Dimakis. Sparse pca via bipartite matchings. In *Advances in Neural Information Processing Systems 28*, pages 766–774. Curran Associates, Inc., 2015.

[7] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[8] Francesco Bonchi, David Garcia-Soriano, and Edo Liberty. Correlation clustering: from theory to practice. In *Proceedings of the 20th ACM International Conference on Knowledge Discovery and Data mining*, pages 1972–1972. ACM, 2014.

[9] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 524–533. IEEE, 2003.

[10] William W Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data mining*, pages 475–480. ACM, 2002.

[11] Erik D Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187, 2006.

[12] Xiaoli Zhang Fern and Carla E Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the 21st International Conference on Machine Learning*, page 36. ACM, 2004.

[13] Ioannis Giotis and Venkatesan Guruswami. Correlation clustering with a fixed number of clusters. In *Proceedings of the 17th annual ACM-SIAM Symposium on Discrete algorithms*, pages 1167–1176. Society for Industrial and Applied Mathematics, 2006.

[14] University of Minnesota GroupLens Lab. Movielens datasets. `http://grouplens.org/datasets/movielens/`. Accessed: 2015-10-03.

[15] Marek Karpinski and Warren Schudy. Linear time approximation schemes for the gale-berlekamp game and related minimization problems. In *Proceedings of the 41st annual ACM Symposium on Theory of Computing*, pages 313–322. ACM, 2009.

[16] Sara C Madeira and Arlindo L Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[17] Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *Proceedings of the 15th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 526–527. Society for Industrial and Applied Mathematics, 2004.

[18] Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos Papadopoulos, and Yannis Manolopoulos. Nearest-biclusters collaborative filtering with constant values. In *Advances in web mining and web usage analysis*, pages 36–55. Springer, 2007.

[19] Anke Van Zuylen and David P Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. *Mathematics of Operations Research*, 34(3):594–620, 2009.

[20] Michail Vlachos, Francesco Fusco, Charalambos Mavroforakis, Anastasios Kyrillidis, and Vassilios G Vassiliadis. Improving co-cluster quality with application to product recommendations. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, pages 679–688. ACM, 2014.

[21] Hongyuan Zha, Xiaofeng He, Chris Ding, Horst Simon, and Ming Gu. Bipartite graph partitioning and data clustering. In *Proceedings of the 10th ACM International Conference on Information and Knowledge Management*, pages 25–32. ACM, 2001.