Low-Rank Binary Matrix Approximation in Column-Sum Norm

Fedor V. Fomin

Department of Informatics, University of Bergen, Norway
http://www.ii.uib.no/~fomin
fomin@ii.uib.no

Petr A. Golovach

Department of Informatics, University of Bergen, Norway
https://folk.uib.no/pgo041/
petr.golovach@ii.uib.no

Fahad Panolan[1]

Department of Computer Science and Engineering, IIT Hyderabad, India
https://iith.ac.in/~fahad/
fahad@cse.iith.ac.in

Kirill Simonov

Department of Informatics, University of Bergen, Norway
kirill.simonov@ii.uib.no

Abstract

We consider ℓ_1-Rank-r Approximation over GF(2), where for a binary $m \times n$ matrix $\mathbf{A}$ and a positive integer constant r, one seeks a binary matrix $\mathbf{B}$ of rank at most r, minimizing the column-sum norm $\|\mathbf{A}-\mathbf{B}\|_1$. We show that for every $\varepsilon \in (0,1)$, there is a randomized $(1+\varepsilon)$-approximation algorithm for ℓ_1-Rank-r Approximation over GF(2) of running time $m^{\mathcal{O}(1)} n^{\mathcal{O}(2^{4r}\cdot\varepsilon^{-4})}$. This is the first polynomial time approximation scheme (PTAS) for this problem.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Binary Matrix Factorization, PTAS, Column-sum norm

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2020.32

Category APPROX

Related Version A full version of the paper is available at https://arxiv.org/abs/1904.06141.

Funding *Fedor V. Fomin*: The research leading to these results have been supported by the Research Council of Norway via the project "MULTIVAL" (grant 263317). The work was done within the CEDAS center in Bergen.

1 Introduction

Low-rank matrix approximation is the method of compressing a matrix by reducing its dimension. It is the basic component of various methods in data analysis including Principal Component Analysis (PCA), one of the most popular and successful techniques used for dimension reduction in data analysis and machine learning [31, 15, 8]. In low-rank matrix approximation one seeks the best low-rank approximation of data matrix $\mathbf{A}$ with matrix $\mathbf{B}$ solving

$$\begin{aligned} &\text{minimize } \|\mathbf{A}-\mathbf{B}\|_\nu \\ &\text{subject to } \operatorname{rank}(\mathbf{B}) \le r. \end{aligned} \tag{1}$$

[1] Corresponding author

© Fedor V. Fomin, Petr A. Golovach, Fahad Panolan, and Kirill Simonov;
licensed under Creative Commons License CC-BY

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020).
Editors: Jarosław Byrka and Raghu Meka; Article No. 32; pp. 32:1–32:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Here $\|\cdot\|_\nu$ is some matrix norm. The most popular matrix norms studied in the literature are the *Frobenius* $||\mathbf{A}||_F^2 = \sum_{i,j} a_{ij}^2$ and the *spectral* $\|\mathbf{A}\|_2 = \sup_{x\neq 0} \frac{\|\mathbf{Ax}\|_2}{\|\mathbf{x}\|_2}$ norms. By the Eckart-Young-Mirsky theorem [8, 27], (1) is efficiently solvable via Singular Value Decomposition (SVD) for these two norms. The spectral norm is an "extremal" norm – it measures the worst-case stretch of the matrix. On the other hand, the Frobenius norm is "averaging". Spectral norm is usually applied in the situation when one is interested in actual columns for the subspaces they define and is of greater interest in scientific computing and numerical linear algebra. The Frobenius norm is widely used in statistics and machine learning, see the survey of Mahony [24] for further discussions.

Recently there has been considerable interest in developing algorithms for low-rank matrix approximation problems for binary (categorical) data. Such variants of dimension reduction for high-dimensional data sets with binary attributes arise naturally in applications involving binary data sets, like latent semantic analysis [4], pattern discovery for gene expression[32], or web search models [19], see [7, 17, 14, 20, 30, 37] for other applications. In many such applications it is much more desirable to approximate a binary matrix $\mathbf{A}$ with a binary matrix $\mathbf{B}$ of small (GF(2) or Boolean) rank because it could provide a deeper insight into the semantics associated with the original matrix. There is a big body of work done on binary and Boolean low-rank matrix approximation, see [2, 3, 7, 22, 25, 26, 28, 35, 34] for further discussions.

Unfortunately, SVD is not applicable for the binary case which makes such problems computationally much more challenging. For a binary matrix, its squared Frobenius norm is equal to the number of its 1-entries, that is $\|\mathbf{A}\|_F^2 = \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}$. Thus, the value $\|\mathbf{A}-\mathbf{B}\|_F^2$ measures the total Hamming distance from points (columns) of $\mathbf{A}$ to the subspace spanned by the columns of $\mathbf{B}$. For this variant of the low-rank binary matrix approximation, a number of approximation algorithms were developed, resulting in efficient polynomial time approximation schemes (EPTASes) obtained in [1, 9]. However, the algorithmic complexity of the problem for any vector-induced norm, including the spectral norm, remained open.

For binary matrices, the natural "extremal" norm to consider is the $\|\cdot\|_1$ norm, also known as *column-sum norm*, operator 1-norm, or Hölder matrix 1-norm. That is, for a matrix $\mathbf{A}$,

$$\|\mathbf{A}\|_1 = \sup_{\|\mathbf{x}\|_1\neq 0} \frac{\|\mathbf{Ax}\|_1}{\|\mathbf{x}\|_1} = \max_{1\leq j\leq n} \sum_{i=1}^{m} |a_{ij}|.$$

In other words, the column-sum norm is the maximum number of 1-entries in a column in $\mathbf{A}$, whereas the Frobenius norm is the total number of 1-entries in $\mathbf{A}$. The column-sum norm is analogous to the spectral norm, only it is induced by the ℓ_1 vector norm, not the ℓ_2 vector norm.

We consider the problem, where for an $m\times n$ binary data matrix $\mathbf{A}$ and a positive integer constant r, one seeks a binary matrix $\mathbf{B}$ optimizing

$$\begin{aligned} &\text{minimize } \|\mathbf{A}-\mathbf{B}\|_1 \\ &\text{subject to } \operatorname{rank}(\mathbf{B}) \leq r. \end{aligned} \tag{2}$$

Here, by the rank of the binary matrix $\mathbf{B}$ we mean its GF(2)-rank. We refer to the problem defined by (2) as to ℓ_1-Rank-r Approximation over GF(2). The value $\|\mathbf{A}-\mathbf{B}\|_1$ is the maximum Hamming distance from each of the columns of $\mathbf{A}$ to the subspace spanned by columns of $\mathbf{B}$ and thus, compared to approximation with the Frobenius norm, it could provide a more accurate dimension reduction.

It is easy to see by the reduction from the Closest String problem, that already for $r = 1$, ℓ_1-Rank-r Approximation over GF(2) is NP-hard. The main result of this paper is that (2) admits a polynomial time approximation scheme (PTAS). More precisely, we prove the following theorem.

▶ **Theorem 1.** *For every $\varepsilon \in (0, 1)$, there is a randomized $(1 + \varepsilon)$-approximation algorithm for ℓ_1-Rank-r Approximation over GF(2) of running time $m^{\mathcal{O}(1)} n^{\mathcal{O}(2^{4r} \cdot \varepsilon^{-4})}$.*

In order to prove Theorem 1 we obtain a PTAS for a more general problem, namely Binary Constrained k-Center. This problem has a strong expressive power and can be used to obtain PTASes for a number of problems related to ℓ_1-Rank-r Approximation over GF(2). For example, for the variant, when the rank of the matrix $\mathbf{B}$ is not over GF(2) but is Boolean. Or a variant of clustering, where we want to partition binary vectors into groups, minimizing the maximum distance in each of the group to some subspace of small dimension. We provide discussions of other applications of our work in Section 4.

Related work. The variant of (1) with both matrices $\mathbf{A}$ and $\mathbf{B}$ binary, and $\|\cdot\|_\nu$ being the Frobenius norm, is known as Low GF(2)-Rank Approximation. Due to numerous applications, various heuristic algorithms for Low GF(2)-Rank Approximation could be found in the literature [16, 17, 11, 20, 32].

When it concerns rigorous algorithmic analysis of Low GF(2)-Rank Approximation, Gillis and Vavasis [13] and Dan et al. [7] have shown that Low GF(2)-Rank Approximation is NP-complete for every $r \geq 1$. A subset of the authors studied parameterized algorithms for Low GF(2)-Rank Approximation in [10]. The first approximation algorithm for Low GF(2)-Rank Approximation is due to Shen et al. [32], who gave a 2-approximation algorithm for the special case of $r = 1$. For rank $r > 1$, Dan et al. [7] have shown that a $(r/2 + 1 + \frac{r}{2(2^r-1)})$-approximate solution can be formed from r columns of the input matrix $\mathbf{A}$. Recently, these algorithms were significantly improved in [1, 9], where efficient polynomial time approximation schemes (EPTASes) were obtained.

Also note that for general (non-binary) matrices a significant amount of work is devoted to $L1$-PCA, where one seeks a low-rank matrix $\mathbf{B}$ approximating given matrix $\mathbf{A}$ in *entrywise* ℓ_1 norm, see e.g. [33].

While our main motivation stems from low-rank matrix approximation problems, ℓ_1-Rank-r Approximation over GF(2) extends Closest String, very well-studied problem about strings. Given a set of binary strings $S = \{s_1, s_2, \ldots, s_n\}$, each of length m, the Closest String problem is to find the smallest d and a string s of length m which is within Hamming distance d to each $s_i \in S$.

A long history of algorithmic improvements for Closest String was concluded by the PTAS of running time $n^{\mathcal{O}(\epsilon^{-5})}$ by Li, Ma, and Wang [21], which running time was later improved to $n^{\mathcal{O}(\epsilon^{-2})}$ [23]. Let us note that Closest String can be seen as a special case of ℓ_1-Rank-r Approximation over GF(2) for $r = 1$. Indeed, Closest String is exactly the variant of ℓ_1-Rank-r Approximation over GF(2), where columns of $\mathbf{A}$ are strings of S and approximating matrix $\mathbf{B}$ is required to have all columns equal. Note that in a binary matrix $\mathbf{B}$ of rank 1 all non-zero columns are equal. However, it is easy to construct an equivalent instance of Closest String by attaching to each string of S a string 1^{m+1}, such that the solution to ℓ_1-Rank-r Approximation over GF(2) for $r = 1$ does not have zero columns.

Cygan et al. [6] proved that the existence of an EPTAS for Closest String, that is $(1 + \varepsilon)$-approximation in time $n^{\mathcal{O}(1)} \cdot f(\varepsilon)$, for any computable function f, is unlikely, as it would imply that FPT=W[1], a highly unexpected collapse in the hierarchy of parameterized complexity classes. They also showed that the existence of a PTAS for Closest String with running time $f(\varepsilon)n^{o(1/\varepsilon)}$, for any computable function f, would contradict the Exponential Time Hypothesis. The result of Cygan et al. implies that ℓ_1-Rank-r Approximation over GF(2) also does not admit EPTAS (unless FPT=W[1]) already for $r = 1$.

A generalization of Closest String, k-closest strings is also known to admit a PTAS [18, 12]. This problem corresponds to the variant of ℓ_1-Rank-r Approximation over GF(2), where approximating matrix $\mathbf{B}$ is required to have at most k different columns. However, it is not clear how solution to this special case can be adopted to solve ℓ_1-Rank-r Approximation over GF(2).

1.1 Our approach

The usual toolbox of techniques to handle NP-hard variants of low-rank matrix approximation problems like sketching [36], sampling, and dimension reduction [5] is based on randomized linear algebra. It is very unclear whether any of these techniques can be used to solve even the simplest case of ℓ_1-Rank-r Approximation over GF(2) with $r = 1$. For example for sampling, the presence of just one outlier outside of a sample, makes all information we can deduce from the sample about the column sum norm of the matrix, completely useless. This is exactly the reason why approximation algorithms for Closest String do not rely on such techniques. On the other hand, randomized dimension reduction appears to be very helpful as a "preprocessing" procedure whose application allows us to solve ℓ_1-Rank-r Approximation over GF(2) by applying linear programming techniques similar to the ones developed for the Closest String. From a very general perspective, our algorithm consists of three steps. While each of these steps is based on the previous works, the way to combine these steps, as well as the correctness proof, is a non-trivial task. We start with a high-level description of the steps and then provide more technical explanations.

Step 1. In order to solve ℓ_1-Rank-r Approximation over GF(2), we encode it as the Binary Constrained k-Center problem. This initial step is almost identical to the encoding used in [9] for Low GF(2)-Rank Approximation. Informally, Binary Constrained k-Center is defined as follows. For a given set of binary vectors X, a positive integer k, and a set of constraints, we want to find k binary vectors $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ satisfying the constraints and minimizing $\max_{\mathbf{x}\in X} d_H(\mathbf{x}, C)$, where $d_H(\mathbf{x}, C)$ is the Hamming distance between $\mathbf{x}$ and the closest vector from C. For example, when $k = 1$ and there are no constraints, then this is just the Closest String problem over binary alphabet.

In the technical description below we give a formal definition of this encoding and in Section 4 we prove that ℓ_1-Rank-r Approximation over GF(2) is a special case of Binary Constrained k-Center. Now on, we are working with Binary Constrained k-Center.

Step 2. We give an approximate Turing reduction which allows to find a partition of vector set X into clusters $X_1, \ldots, X_k$ such that if we find a tuple of vectors $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ satisfying the constraints and minimizing $\max_{1\le i\le k, \mathbf{x}\in X_i} d_H(\mathbf{x}, \{\mathbf{c}_i\})$, then the same tuple C will be a good approximation to Binary Constrained k-Center. In order to obtain such a partition, we use the dimension reduction technique of Ostrovsky and Rabani [29]. While this provides us with important structural information, we are not done yet. Even with a given partition, the task of finding the corresponding tuple of "closest strings" C satisfying the constraints, is non-trivial.

Step 3. In order to find the centers, we implement the approach used by Li, Ma, and Wang in [21] to solve CLOSEST STRING. By brute-forcing, it is possible to reduce the solution of the problem to special instances, which loosely speaking, have a large optimum. Moreover, BINARY CONSTRAINED k-CENTER has an Integer Programming (IP) formulation. Similar to [21], for the reduced instance of BINARY CONSTRAINED k-CENTER (which has a "large optimum") it is possible to prove that the randomized rounding of the corresponding Linear Program (LP) relaxation of this IP, provides a good approximation.

Now we give a more technical description of the algorithm.

Step 1. Binary Constrained k-Center. Note that the BINARY CONSTRAINED k-CENTER problem is nearly identical to BINARY CONSTRAINED CLUSTERING defined in [9], except for the cost function. Still, for completeness we define BINARY CONSTRAINED k-CENTER formally next. First, we need to define some notations. A k-ary relation R is a set of binary k-tuples with elements from $\{0,1\}$. A k-tuple $t = (t_1, \ldots, t_k)$ *satisfies* R, we write $t \in R$, if t is equal to one of the k-tuples in R.

▶ **Definition 2** (Vectors satisfying $\mathcal{R}$). *Let $\mathcal{R} = (R_1, \ldots, R_m)$ be a tuple of k-ary relations. We say that a tuple $C = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k)$ of binary m-dimensional vectors* satisfies $\mathcal{R}$ *and write $< C, \mathcal{R} >$, if $(\mathbf{c}_1[i], \ldots, \mathbf{c}_k[i]) \in R_i$ for all $i \in \{1, \ldots, m\}$.*

For example, for $m=2$, $k=3$, $R_1=\{(0,0,1),(1,0,0)\}$, and $R_2=\{(1,1,1),(1,0,1),(0,0,1)\}$, the tuple of vectors

$$\mathbf{c}_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \mathbf{c}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \mathbf{c}_3 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

satisfies $\mathcal{R} = (R_1, R_2)$ because $(\mathbf{c}_1[1], \mathbf{c}_2[1], \mathbf{c}_3[1]) = (0,0,1) \in R_1$ and $(\mathbf{c}_1[2], \mathbf{c}_2[2], \mathbf{c}_3[2]) = (1,0,1) \in R_2$.

Let us recall that the *Hamming distance* between two vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^m$, where $\mathbf{x} = (x_1, \ldots, x_m)^\intercal$ and $\mathbf{y} = (y_1, \ldots, y_m)^\intercal$, is $d_H(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^m |x_i - y_i|$ or, in words, the number of positions $i \in \{1, \ldots, m\}$ where x_i and y_i differ. Recall that for a set of vectors $C \subseteq \{0,1\}^m$ and a vector $\mathbf{x} \in \{0,1\}^m$, $d_H(\mathbf{x}, C) = \min_{\mathbf{c} \in C} d_H(\mathbf{x}, \mathbf{c})$. For sets $X, C \subset \{0,1\}^m$, we define $\mathrm{cost}(X, C) = \max_{\mathbf{x} \in X} d_H(\mathbf{x}, C)$.

Now we define BINARY CONSTRAINED k-CENTER formally.

> BINARY CONSTRAINED k-CENTER
> **Input:** A set $X \subseteq \{0,1\}^m$ of n vectors, a positive integer k, and a tuple of k-ary relations $\mathcal{R} = (R_1, \ldots, R_m)$.
> **Task:** Among all tuples $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ of vectors from $\{0,1\}^m$ satisfying $\mathcal{R}$, find a tuple C minimizing $\mathrm{cost}(X, C)$.

As in the case of LOW GF(2)-RANK APPROXIMATION in [9], we prove that ℓ_1-RANK-r APPROXIMATION OVER GF(2) is a special case of BINARY CONSTRAINED k-CENTER, where $k = 2^r$. For completeness, this proof and other applications of BINARY CONSTRAINED k-CENTER are given in Section 4. Thus, to prove Theorem 1, it is enough to design a PTAS for BINARY CONSTRAINED k-CENTER.

▶ **Theorem 3.** *There is an algorithm for BINARY CONSTRAINED k-CENTER that given an instance $J = (X, k, \mathcal{R})$ and $0 < \varepsilon < 1$, runs in time $m^{\mathcal{O}(1)} n^{\mathcal{O}((k/\varepsilon)^4)}$, and outputs a $(1+\varepsilon)$-approximate solution with probability at least $1 - 2n^{-2}$.*

By the argument above, Theorem 1 is an immediate corollary of Theorem 3.

Step 2: Dimension reduction. Let $J = (X, k, \mathcal{R} = (R_1, \ldots, R_m))$ be an instance of Binary Constrained k-Center and $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ be a solution to J, that is, a tuple of vectors satisfying $\mathcal{R}$. Then, the cost of C is $\text{cost}(X, C)$. Given the tuple C, there is a natural way we can partition the set of vectors X into k parts $X_1 \uplus \cdots \uplus X_k$ such that

$$\text{cost}(X, C) = \max_{i \in \{1,\ldots,k\}, \mathbf{x} \in X_i} d_H(\mathbf{x}, \mathbf{c}_i).$$

Thus, for each vector $\mathbf{x}$ in X_i, the closest to $\mathbf{x}$ vector from C is $\mathbf{c}_i$. We call such a partition $X_1 \uplus \cdots \uplus X_k$ the *clustering of X induced by C* and refer to the sets $X_1, \ldots, X_k$ as the *clusters corresponding to C*. We use $\mathsf{OPT}(J)$ to denote the cost of an optimal solution to J. That is, $\mathsf{OPT}(J) = \min\{\text{cost}(X, C) \mid \ < C, \mathcal{R} >\}$. In fact, even if we know the clustering of X induced by a hypothetical optimal solution, finding a good solution is not trivial as the case when $k = 1$ is the same as the Closest String problem.

As mentioned before, our approach is to reduce to a version of Binary Constrained k-Center, where we know the partition of X, and solve the corresponding problem. That is, we design an approximation scheme for the following partitioned version of the problem.

Binary Constrained Partition Center
Input: A positive integer k, a set $X \subseteq \{0,1\}^m$ of n vectors partitioned into $X_1 \uplus \ldots \uplus X_k$, and a tuple of k-ary relations $\mathcal{R} = (R_1, \ldots, R_m)$.
Task: Among all tuples $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ of vectors from $\{0,1\}^m$ satisfying $\mathcal{R}$, find a tuple C minimizing $\max_{i \in \{1,\ldots,k\}, \mathbf{x} \in X_i} d_H(\mathbf{x}, \mathbf{c}_i)$.

For an instance $J' = (k, X = X_1 \uplus \ldots \uplus X_k, \mathcal{R})$ of Binary Constrained Partition Center, we use $\mathsf{OPT}(J')$ to denote the cost of an optimal solution to J'. That is,

$$\mathsf{OPT}(J') = \min_{C = (\mathbf{c}_1, \ldots, \mathbf{c}_k) \text{ s.t. } <C, \mathcal{R}>} \left\{ \max_{i \in \{1,\ldots,k\}, \mathbf{x} \in X_i} d_H(\mathbf{x}, \mathbf{c}_i) \right\}.$$

Clearly, for an instance $J = (X, k, \mathcal{R})$ of Binary Constrained k-Center and a partition of X into $X_1 \uplus \ldots \uplus X_k$, any solution to the instance $J' = (k, X = X_1 \uplus \ldots X_k, \mathcal{R})$ of Binary Constrained Partition Center, of cost d, is also a solution to J with cost at most d. We prove that there is a randomized polynomial time algorithm that given an instance $J = (X, k, \mathcal{R})$ of Binary Constrained k-Center and $0 < \epsilon \leq \frac{1}{4}$, outputs a collection $\mathcal{I}$ of Binary Constrained Partition Center instances $J' = (k, X = X_1 \uplus \ldots \uplus X_k, \mathcal{R})$ such that the cost of at least one instance in $\mathcal{I}$ is at most $(1 + 4\epsilon)\mathsf{OPT}(J)$ with high probability.

▶ **Lemma 4.** *There is an algorithm that given an instance $J = (X, k, \mathcal{R})$ of Binary Constrained k-Center, $0 < \epsilon \leq \frac{1}{4}$, and $\gamma > 0$, runs in time $m^2 n^{\mathcal{O}(k/\epsilon^4)}$, and outputs a collection $\mathcal{I}$ of $m \cdot n^{\mathcal{O}(k/\epsilon^4)}$ instances of Binary Constrained Partition Center such that each instance in $\mathcal{I}$ is of the form $(k, X = X_1 \uplus \ldots \uplus X_k, \mathcal{R})$, and there exists $J' \in \mathcal{I}$ such that $\mathsf{OPT}(J') \leq (1 + 4\epsilon)\mathsf{OPT}(J)$ with probability at least $1 - n^{-\gamma}$.*

To prove Lemma 4, we use the dimension reduction technique of Ostrovsky and Rabani from [29]. Loosely speaking, this technique provides a linear map ψ with the following properties. For any $\mathbf{y} \in \{0,1\}^m$, $\psi(\mathbf{y})$ is a 0-1 vector of length $\mathcal{O}(\log n/\epsilon^4)$, and for any set Y of $n + k$ vectors, Hamming distances between any pair of vectors in $\psi(Y)$ are relatively preserved with high probability. So we assume that ψ is "a good map" for the set of vectors $X \cup C$, where $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ is a hypothetical optimal solution to J. Then, we guess the potential tuples of vectors $(\phi(\mathbf{c}_1), \ldots, \phi(\mathbf{c}_k))$ for the hypothetical optimal solution $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$, and use these choices for $(\phi(\mathbf{c}_1), \ldots, \phi(\mathbf{c}_k))$ to construct partitions of X, and thereby construct instances in $\mathcal{I}$. Lemma 4 is proved in Section 2.

Step 3: LP relaxation. Because of Lemma 4, to prove Theorem 3, it is enough to design a PTAS for BINARY CONSTRAINED PARTITION CENTER which is more challenging part in our algorithm. So we prove the following lemma.

▶ **Lemma 5.** *There is an algorithm for* BINARY CONSTRAINED PARTITION CENTER *that given an instance* $J = (k, X = X_1^i \uplus \ldots \uplus X_k^i, \mathcal{R})$ *and* $0 < \epsilon < 1/2$*, runs in time* $m^{\mathcal{O}(1)} n^{\mathcal{O}((k/\epsilon)^4)}$*, and outputs a solution of cost at most* $(1 + \epsilon)\mathit{OPT}(J)$ *with probability at least* $1 - n^{-2}$*.*

Towards the proof of Lemma 5, we encode BINARY CONSTRAINED PARTITION CENTER using an Integer programming (IP) formulation (see (6) in Section 3). We show that the randomized rounding using the solution of the linear programming relaxation of this IP provides a good approximation if the optimum value is large. Here we follow the approach similar to the one used by Li, Ma, and Wang in [21] to solve CLOSEST STRING. We prove that there exist $Y_1 \subseteq X_1, \ldots, Y_k \subseteq X_k$, each of size $r = 1 + \frac{4}{\epsilon}$, with the following property. Let Q be the set of positions in $\{1, \ldots, m\}$ such that for each $i \in \{1, \ldots, k\}$ and $j \in Q$, all the vectors in Y_i agree at the position j, and for each $j \in Q$, $(\mathbf{y}_1[j], \ldots, \mathbf{y}_k[j]) \in R_j$, where $\mathbf{y}_i \in Y_i$ for all $i \in \{1, \ldots, k\}$. Then, for any solution of J such that for each $j \in Q$ the entries at the position j coincide with $(\mathbf{y}_1[j], \ldots, \mathbf{y}_k[j])$, the cost of this solution restricted to Q deviates from the cost of an optimal solution restricted to Q by at most $\frac{1}{r-1}\mathsf{OPT}(J)$. Moreover, the subproblem of J restricted to $\{1, \ldots, m\} \setminus Q$ has large optimum value and we could use linear programming to solve the subproblem. Lemma 5 is proved in Section 3.

Putting together. Next we explain how to prove Theorem 3 using Lemmata 4 and 5. Let $J = (X, k, \mathcal{R})$ be the input instance of BINARY CONSTRAINED k-CENTER and $0 < \varepsilon < 1$ be the given error parameter. Let $\beta = \frac{\varepsilon}{8}$. Since $\varepsilon < 1$, $\beta < \frac{1}{4}$. Now, we apply Lemma 4 on J, β, and $\gamma = 2$. As a result, we get a collection $\mathcal{I}$ of instances of BINARY CONSTRAINED PARTITION CENTER such that each instance in $\mathcal{I}$ is of the form $(k, X = X_1 \uplus \ldots \uplus X_k, \mathcal{R})$, and there exists $J' \in \mathcal{I}$ such that $\mathsf{OPT}(J') \leq (1 + 4\beta)\mathsf{OPT}(J)$ with probability at least $1 - n^{-2}$. From now on, we assume that this event happened. Next, for each instance $\widehat{J} \in \mathcal{I}$, we apply Lemma 5 with the error parameter β, and output the best solution among the solutions produced. Let $J' \in \mathcal{I}$ be the instance such that $\mathsf{OPT}(J') \leq (1+4\beta)\mathsf{OPT}(J) \leq (1+\frac{\varepsilon}{2})\mathsf{OPT}(J)$. Any solution to $\widehat{J} \in \mathcal{I}$ of cost d, is also a solution to J of cost at most d. Therefore, because of Lemmas 4 and 5, our algorithm outputs a solution of J with cost at most $(1+\beta)\mathsf{OPT}(J') = (1 + \frac{\varepsilon}{8})(1 + \frac{\varepsilon}{2})\mathsf{OPT}(J) \leq (1+\varepsilon)\mathsf{OPT}(J)$ with probability at least $1-2n^{-2}$, since both Lemmas 4 and 5 have the success probability of at least $1 - n^{-2}$. The running time of the algorithm follows from Lemmata 4 and 5.

As Theorem 3 is already proved using Lemmas 4 and 5, the rest of the paper is devoted to the proofs of Lemmata 4 and 5, and to the examples of the expressive power of BINARY CONSTRAINED k-CENTER, including ℓ_1-RANK-r APPROXIMATION OVER GF(2). In Sections 2 and 3, we prove Lemmata 4 and 5, respectively. In Section 4, we give applications of Theorem 3.

2 Proof of Lemma 4

In this section we prove Lemma 4. The main idea is to map the given instance to a low-dimensional space while approximately preserving distances, then try all possible tuples of centers in the low-dimensional space, and construct an instance of BINARY CONSTRAINED PARTITION CENTER by taking the optimal partition of the images with respect to a fixed tuple of centers back to the original vectors.

To implement the mapping, we employ the notion of (δ, ℓ, h)-distorted maps, introduced by Ostrovsky and Rabani [29]. Intuitively, a (δ, ℓ, h)-distorted map approximately preserves distances between ℓ and h, does not shrink distances larger than h too much, and does not expand distances smaller than ℓ too much. In what follows we make the definitions formal.

A metric space is a pair (P, d) where P is a set (whose elements are called points), and d is a distance function $d : P \times P \to \mathbb{R}$ (called a metric), such that for every $p_1, p_2, p_3 \in P$ the following conditions hold: (i) $d(p_1, p_2) \geq 0$, (ii) $d(p_1, p_2) = d(p_2, p_1)$, (iii) $d(p_1, p_2) = 0$ if and only if $p_1 = p_2$, and (iv) $d(p_1, p_2) + d(p_2, p_3) \geq d(p_1, p_3)$. Condition (iv) is called the triangle inequality. The pair $(\{0, 1\}^m, d_H)$, binary vectors of lentgh m and the Hamming distance, is a metric space.

▶ **Definition 6** ([29]). *Let (P, d) and (P', d') be two metric spaces. Let $X, Y \subseteq P$. Let δ, ℓ, h be such that $\delta > 0$ and $h > \ell \geq 0$. A mapping $\psi : P \to P'$ is (δ, ℓ, h)-distorted on (X, Y) if and only if there exists $\alpha > 0$ such that for every $x \in X$ and $y \in Y$, the following conditions hold.*

1. *If $d(x, y) < \ell$, then $d(\psi(x), \psi(y)) < (1 + \delta)\alpha\ell$.*
2. *If $d(x, y) > h$, then $d(\psi(x), \psi(y)) > (1 - \delta)\alpha h$.*
3. *If $\ell \leq d(x, y) \leq h$, then $(1 - \delta)\alpha d(x, y) \leq d(\psi(x), \psi(y)) \leq (1 + \delta)\alpha d(x, y)$.*

If $X = Y$, then we say that ψ is (δ, ℓ, h)-distorted on X.

For any $r, r' \in \mathbb{N}$ and $\varepsilon > 0$, $\mathcal{A}_{r,r'}(\varepsilon)$ denotes a distribution over $r' \times r$ binary matrices $M \in \{0, 1\}^{r' \times r}$, where entries are independent, identically distributed, random 0/1 variables with $\Pr[1] = \varepsilon$.

▶ **Proposition 7** ([29]). *Let $m, \ell \in \mathbb{N}$, and let $X \subseteq \{0, 1\}^m$ be a set of n vectors. For every $0 < \epsilon \leq 1/2$, there exists a mapping $\phi : X \to \{0, 1\}^{m'}$, where $m' = \mathcal{O}(\log n/\epsilon^4)$, which is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on X (with respect to the Hamming distance in both spaces). More precisely, for every $\gamma > 0$ there exists $\lambda > 0$, such that, setting $m' = \lambda \log n/\epsilon^4$, the linear map $\mathbf{x} \mapsto A\mathbf{x}$, where A is a random matrix drawn from $\mathcal{A}_{m,m'}(\epsilon^2/\ell)$, is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on X with probability at least $1 - n^{-\gamma}$.*

Now we are ready to prove Lemma 4. We restate it for convenience.

▶ **Lemma 4.** *There is an algorithm that given an instance $J = (X, k, \mathcal{R})$ of* Binary Constrained *k-*Center*, $0 < \epsilon \leq \frac{1}{4}$, and $\gamma > 0$, runs in time $m^2 n^{\mathcal{O}(k/\epsilon^4)}$, and outputs a collection $\mathcal{I}$ of $m \cdot n^{\mathcal{O}(k/\epsilon^4)}$ instances of* Binary Constrained Partition Center *such that each instance in $\mathcal{I}$ is of the form $(k, X = X_1 \uplus \ldots \uplus X_k, \mathcal{R})$, and there exists $J' \in \mathcal{I}$ such that $\mathsf{OPT}(J') \leq (1 + 4\epsilon)\mathsf{OPT}(J)$ with probability at least $1 - n^{-\gamma}$.*

Proof. Without loss of generality, we may assume $\mathsf{OPT}(J) > 0$. If $\mathsf{OPT}(J) = 0$, there are at most k distinct vectors in X, and we trivially construct a single instance of Binary Constrained Partition Center by grouping equal vectors together.

Let $n = |X|$ and $n' = n + k$. Let $\lambda = \lambda(\gamma)$ be the constant mentioned in Proposition 7, and $m' = \lambda \log n'/\epsilon^4$. Then, for each $\ell \in [m]$[2], we construct the collection $\mathcal{I}_\ell$ of $n^{\mathcal{O}(k/\epsilon^4)}$ Binary Constrained Partition Center instances as follows.

- Start with $\mathcal{I}_\ell := \emptyset$.
- Randomly choose a matrix A^ℓ from the distribution $\mathcal{A}_{m,m'}(\epsilon^2/\ell)$.

[2] For an integer $n \in \mathbb{N}$, we use $[n]$ as a shorthand for $\{1, \ldots, n\}$.

- For each choice of k vectors $\mathbf{c}'_1, \dots, \mathbf{c}'_k \in \{0,1\}^{m'}$, construct a partition $X_1 \uplus \dots \uplus X_k$ of X such that for each $\mathbf{x} \in X_i$, $\mathbf{c}'_i$ is one of the closest vectors to $A^\ell \mathbf{x}$ among $C' = \{\mathbf{c}'_1, \dots, \mathbf{c}'_k\}$. Then, add $(k, X = X^i_1 \uplus \dots X^i_k, \mathcal{R})$ to $\mathcal{I}_\ell$.

Finally, our algorithm outputs $\mathcal{I} = \bigcup_{\ell \in [m]} \mathcal{I}_\ell$ as the required collection of Binary Constrained Partition Center instances. Notice that for any $\ell \in [m]$, $|\mathcal{I}_\ell| = 2^{m'k} = n^{\mathcal{O}(k/\epsilon^4)}$. This implies that the cardinality of $\mathcal{I}$ is upper bounded by $m \cdot n^{\mathcal{O}(k/\epsilon^4)}$, and the construction of $\mathcal{I}_\ell$ takes time $m \cdot n^{\mathcal{O}(k/\epsilon^4)}$. Thus, the total running time of the algorithm is $m^2 \cdot n^{\mathcal{O}(k/\epsilon^4)}$.

Next, we prove the correctness of the algorithm. Let $\ell = \mathsf{OPT}(J)$ and $C = (\mathbf{c}_1, \dots, \mathbf{c}_k)$ be an optimum solution of J. Let $Y_1, \dots, Y_k$ be the clusters corresponding to C. Consider the step in the algorithm where we constructed $\mathcal{I}_\ell$. By Proposition 7, the map $\psi \colon \mathbf{x} \mapsto A^\ell \mathbf{x}$ is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on $X \cup C$ with probability at least $1 - n^{-\gamma}$. In the rest of the proof, we assume that this event happened. Let $\mathbf{c}'_1 = A^\ell \mathbf{c}_1, \dots, \mathbf{c}'_k = A^\ell \mathbf{c}_k$. Consider the Binary Constrained Partition Center instance constructed for the choice of vectors $\mathbf{c}'_1, \dots, \mathbf{c}'_k$. That is, let $X_1, \dots, X_k$ be the partition of X such that for each $\mathbf{x} \in X_i$, $\mathbf{c}'_i$ is one of the closest vector to $A^\ell \mathbf{x}$ from $C' = \{\mathbf{c}'_1, \dots, \mathbf{c}'_k\}$. Let J' be the instance $(k, X = X^i_1 \uplus \dots X^i_k, \mathcal{R})$ of Binary Constrained Partition Center.

Now, we claim that C is a solution to J' with cost at most $(1+4\epsilon)\ell = (1+4\epsilon)\mathsf{OPT}(J)$. Since C satisfies $\mathcal{R}$, C is a solution of J'. To prove $\mathsf{OPT}(J') \le (1+4\epsilon)\ell$, it is enough to prove that for each $i \in [k]$ and $\mathbf{x} \in X_i$, $d_H(\mathbf{x}, \mathbf{c}_i) \le (1+4\epsilon)\ell$. Fix an index $i \in [k]$ and $\mathbf{x} \in X_i$. Suppose $\mathbf{x} \in Y_i$. Since C is an optimum solution of J with corresponding clusters $Y_1, \dots Y_k$, we have that $d_H(\mathbf{y}, \mathbf{c}_i) \le \ell$ for all $\mathbf{y} \in Y_i \cap X_i$. Thus, $d_H(\mathbf{x}, \mathbf{c}_i) \le \ell$. So, now consider the case $\mathbf{x} \in Y_j$ for some $j \neq i$. Notice that if $d_H(\mathbf{x}, \mathbf{c}_i) \le \ell$, then we are done. We have the following two subcases.

Case 1: $d_H(\mathrm{x}, \mathrm{c}_i) \le \frac{\ell}{2\epsilon}$. We know that the map $\psi \colon \mathbf{x} \mapsto A^\ell \mathbf{x}$ is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on $X \cup C$, and let $\alpha > 0$ be the number such that conditions of Definition 6 hold. Since $\mathbf{x} \in X_i$, we have that (a) $d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_i)) \le d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_j))$. Since $d_H(\mathbf{x}, \mathbf{c}_j) \le \ell$ (because $\mathbf{x} \in Y_j$) and ψ is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on $X \cup C$, we have that (b) $d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_j)) \le (1+\epsilon)\alpha\ell$. Since $\ell < d_H(\mathbf{x}, \mathbf{c}_i) \le \frac{\ell}{2\epsilon}$, and ψ is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on $X \cup C$, we have that (c) $(1-\epsilon)\alpha d_H(\mathbf{x}, \mathbf{c}_i) \le d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_i))$. The statements (a), (b), and (c) imply that

$$d_H(\mathbf{x}, \mathbf{c}_i) \le \frac{1+\epsilon}{1-\epsilon}\ell \le (1+4\epsilon)\ell,$$

where the last inequality holds since $\epsilon \le 1/4$.

Case 2: $d_H(\mathrm{x}, \mathrm{c}_i) > \frac{\ell}{2\epsilon}$. We prove that this case is impossible by showing a contradiction. Since $\epsilon \le 1/4$, in this case, we have that $d_H(\mathbf{x}, \mathbf{c}_i) > 2\ell$. Since ψ is $(\epsilon, \ell/4, \ell/2\epsilon)$-distorted on $X \cup C$, $d_H(\mathbf{x}, \mathbf{c}_i) > 2\ell$, and $d_H(\mathbf{x}, \mathbf{c}_j) \le \ell$, we have that

$$(1-\epsilon)\alpha \cdot 2\ell \le d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_i)) \le d_H(\psi(\mathbf{x}), \psi(\mathbf{c}_j)) \le (1+\epsilon)\alpha \cdot \ell.$$

Then $2(1-\epsilon) \le (1+\epsilon)$ and thus $\epsilon \ge 1/3$, which contradicts the assumption that $\epsilon \le 1/4$. This completes the proof of the lemma. ◀

3 Proof of Lemma 5

For a set of positions $P \subset [m]$, let us define the Hamming distance restricted to P by

$$d^P_H(\mathbf{x}, \mathbf{y}) = \sum_{i \in P} |x_i - y_i|.$$

We use the following lemma in our proof.

▶ **Lemma 8.** *Let $Y = \{\mathbf{y}_1, \cdots, \mathbf{y}_l\} \subset \{0,1\}^m$ be a set of vectors and $\mathbf{c}^* \in \{0,1\}^m$ be a vector. Let $d^* = \mathrm{cost}(Y, \{\mathbf{c}^*\}) = \max_{\mathbf{y}\in Y} d_H(\mathbf{y}, \mathbf{c}^*)$. For any $r \in \mathbb{N}$, $r \geq 2$, there exist indices $i_1, \ldots, i_r$ such that for any $\mathbf{x} \in Y$*

$$d_H^P(\mathbf{x}, \mathbf{y}_{i_1}) - d_H^P(\mathbf{x}, \mathbf{c}^*) \leq \frac{1}{r-1} d^*,$$

where P is any subset of $Q_{i_1,\ldots,i_r}$ and $Q_{i_1,\ldots,i_r}$ is the set of positions where all of $\mathbf{y}_{i_1}, \ldots, \mathbf{y}_{i_r}$ coincide (i.e., $Q_{i_1,\ldots,i_r} = \{j \in [m] \colon \mathbf{y}_{i_1}[j] = \mathbf{y}_{i_2}[j] = \ldots = \mathbf{y}_{i_r}[j]\}$).

Proof. For a vector $\mathbf{x} = \mathbf{y}_{\ell'} \in Y$ and $P \subseteq Q_{i_1,\ldots,i_r}$, let

$$\begin{aligned} J_P(\ell') &= \{j \in P \colon \mathbf{y}_{i_1}[j] \neq \mathbf{x}[j] \text{ and } \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}, \text{ and} \\ J(\ell') &= \{j \in Q_{i_1,\ldots,i_r} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{x}[j] \text{ and } \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}. \end{aligned}$$

To prove the lemma it is enough to prove that $|J_P(\ell')| \leq \frac{1}{r-1} d^*$. Also, since $J_P(\ell') \subseteq J(\ell')$, to prove the lemma, it is enough to prove that $|J(\ell')| \leq \frac{1}{r-1} d^*$. Recall that for any $s \in [\ell]$ and $1 \leq i_1, \ldots, i_s \leq \ell$, $Q_{i_1,\ldots,i_s}$ is the set of positions where all of $\mathbf{y}_{i_1}, \ldots, \mathbf{y}_{i_s}$ coincide. For any $2 \leq s \leq r+1$ and $1 \leq i_1, \ldots, i_s \leq \ell$, let $p_{i_1,\ldots,i_s}$ be the number of mismatches between $\mathbf{y}_{i_1}$ and $\mathbf{c}^*$ at the positions in $Q_{i_1,\ldots,i_s}$. Let

$$\rho_s = \min_{1 \leq i_1,\ldots,i_s \leq n} \frac{p_{i_1,\ldots,i_s}}{d^*}.$$

Notice that for any $2 \leq s \leq r+1$, $\rho_s \leq 1$.

▷ Claim 9 (Claim 2.2 [21]). [3] For any s such that $2 \leq s \leq r$, there are indices $1 \leq i_1, i_2, \ldots, i_r \leq \ell$ such that for any $\mathbf{x} = \mathbf{y}_{\ell'} \in Y$, $|J(\ell')| \leq (\rho_s - \rho_{s+1}) d^*$.

Proof. Consider indices $1 \leq i_1, \ldots, i_s \leq \ell$ such that $p_{i_1,\ldots,i_s} = \rho_s \cdot d^*$. Next arbitrarily pick $r - s$ indices $i_{s+1}, i_{s+2}, \ldots, i_r$ from $[\ell] \setminus \{i_1, \ldots, i_s\}$. Next we prove that $i_1, i_2, \ldots, i_r$ are the required set of indices. Towards that, fix $\mathbf{x} = \mathbf{y}_{\ell'} \in Y$,

$$\begin{aligned} J(\ell') &= |\{j \in Q_{i_1,\ldots,i_r} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{x}[j] \text{ and } \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| \\ &\leq |\{j \in Q_{i_1,\ldots,i_s} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{x}[j] \text{ and } \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| \\ &\qquad \text{(Because } Q_{i_1,\ldots,i_r} \subseteq Q_{i_1,\ldots,i_s}\text{)} \\ &= |\{j \in Q_{i_1,\ldots,i_s} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\} \setminus \{j \in Q_{i_1,\ldots,i_s} \colon \mathbf{y}_{i_1}[j] = \mathbf{x}[j] \wedge \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| \\ &= |\{j \in Q_{i_1,\ldots,i_s} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\} \setminus \{j \in Q_{i_1,\ldots,i_s,\ell'} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| \\ &\qquad \text{(Since } \mathbf{x} = \mathbf{y}_{\ell'}\text{)} \\ &= |\{j \in Q_{i_1,\ldots,i_s} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| - |\{j \in Q_{i_1,\ldots,i_s,\ell'} \colon \mathbf{y}_{i_1}[j] \neq \mathbf{c}^*[j]\}| \qquad (3) \\ &= p_{i_1,\ldots,i_s} - p_{i_1,\ldots,i_s,\ell'} \qquad \text{(By definition)} \\ &\leq (\rho_s - \rho_{s+1}) d^* \qquad (4) \end{aligned}$$

The equality (3) holds since $Q_{i_1,\ldots,i_s} \supseteq Q_{i_1,\ldots,i_s,\ell'}$. The inequality (4) holds because $p_{i_1,\ldots,i_s} = \rho_s \cdot d^*$ by the choice of $i_1, \ldots, i_s$, and $\rho_{s+1} d^* \leq p_{i_1,\ldots,i_s,\ell'}$ by definition. ◁

Notice that $(\rho_2 - \rho_3) + (\rho_3 - \rho_4) + \ldots + (\rho_r - \rho_{r+1}) = (\rho_2 - \rho_{r+1}) \leq \rho_2 \leq 1$. Thus, one of $(\rho_2 - \rho_3), (\rho_3 - \rho_4), \ldots, (\rho_r - \rho_{r+1})$ is at most $1/(r-1)$. This completes the proof. ◀

[3] We remark that Claim 2.2 in [21] is stated for a vector $\mathbf{c}$ such that $d^* = \mathrm{cost}(Y, \{\mathbf{c}\}) = \min_{\mathbf{c}'} \mathrm{cost}(Y, \{\mathbf{c}'\})$. But the steps of the same proof work in our case as well.

Consider the instance $J = (k, X = X_1^i \uplus \dots X_k^i, \mathcal{R})$ of BINARY CONSTRAINED PARTITION CENTER. Let $C^* = (\mathbf{c}_1^*, \cdots, \mathbf{c}_k^*) \subset \{0,1\}^m$ be an optimum solution to J. Let $d_{opt} = \mathsf{OPT}(J) = \max_{i\in[k],\mathbf{x}\in X_i} d_H(\mathbf{x}, \mathbf{c}_i^*)$. For each $i \in [k]$ and $r \geq 2$, by Lemma 8, there exist r elements $\mathbf{x}_i^{(1)}, \dots, \mathbf{x}_i^{(r)}$ of X_i such that for any $\mathbf{x} \in X_i$,

$$d_H^P(\mathbf{x}, \mathbf{x}_i^{(1)}) - d_H^P(\mathbf{x}, \mathbf{c}_i^*) \leq \frac{1}{r-1} d_{opt}, \tag{5}$$

where P is any subset of Q_i, and Q_i is the set of coordinates on which $\mathbf{x}_i^{(1)}, \dots, \mathbf{x}_i^{(r)}$ agree. Let us denote as Q the intersection of all Q_i from which the positions not satisfying $\mathcal{R}$ are removed. That is,

$$Q = \left\{ j \in \bigcap_{i\in[k]} Q_i \colon (\mathbf{x}_1^{(1)}[j], \mathbf{x}_2^{(1)}[j], \dots, \mathbf{x}_k^{(1)}[j]) \in R_j \right\}.$$

Because of (5), there is an approximate solution where the coordinates $j \in Q$ are identified using $\mathbf{x}_1^{(1)}, \dots, \mathbf{x}_k^{(1)}$. Let $\overline{Q} = [m] \setminus Q$. Now the idea is to solve the problem restricted to $\overline{Q}$ separately, and then complement the solution on Q by the values of $\mathbf{x}_i^{(1)}$. We prove that for the "subproblem" restricted on $\overline{Q}$, the optimum value is *large*. Towards that we first prove the following lemma.

▶ **Lemma 10** (⋆).[4] *Let $J = (k, X = X_1^i \uplus \dots X_k^i, \mathcal{R})$ be an instance of BINARY CONSTRAINED PARTITION CENTER. Let $(\mathbf{c}_1^*, \dots, \mathbf{c}_k^*)$ be an optimal solution for J, and $r \geq 2$ be an integer. Then, there exist $\{\mathbf{x}_1^{(1)}, \dots, \mathbf{x}_1^{(r)}\} \subset X_1, \dots, \{\mathbf{x}_k^{(1)}, \dots, \mathbf{x}_k^{(r)}\} \subset X_k$ with the following properties. For each $i \in [k]$, let Q_i be the set of coordinates on which $\mathbf{x}_i^{(1)}, \dots, \mathbf{x}_i^{(r)}$ agree, $Q = \left\{ j \in \bigcap_{i\in[k]} Q_i \colon (\mathbf{x}_1^{(1)}[j], \mathbf{x}_2^{(1)}[j], \dots, \mathbf{x}_k^{(1)}[j]) \in R_j \right\}$, and $\overline{Q} = [m] \setminus Q$.*

- *For any $i \in [k]$ and $\mathbf{x} \in X_i$, $d_H^Q(\mathbf{x}, \mathbf{x}_i^{(1)}) - d_H^Q(\mathbf{x}, \mathbf{c}_i^*) \leq \frac{1}{r-1} \mathsf{OPT}(J)$, and*
- *$|\overline{Q}| \leq rk \cdot \mathsf{OPT}(J)$.*

As mentioned earlier, we fix the entries of our solution in positions j of Q with values in $\mathbf{x}_1^{(1)}[j], \dots, \mathbf{x}_k^{(1)}[j]$. Towards finding the entries of our solution in positions of $\overline{Q}$, we define the following problem and solve it.

> BINARY CONSTRAINED PARTITION CENTER⋆
> **Input:** A positive integer k, a set $X \subseteq \{0,1\}^m$ of n vectors partitioned into $X_1 \uplus \dots \uplus X_k$, a tuple of k-ary relations $\mathcal{R} = (R_1, \dots, R_m)$, and for all $\mathbf{x} \in X$, $d_{\mathbf{x}} \in \mathbb{N}$
> **Task:** Among all tuples $C = (\mathbf{c}_1, \dots, \mathbf{c}_k)$ of vectors from $\{0,1\}^m$ satisfying $\mathcal{R}$, find a tuple C that minimizes the integer d such that for all $i \in [k]$ and $\mathbf{x} \in X_i$, $d_H(\mathbf{x}, \mathbf{c}_i) \leq d - d_{\mathbf{x}}$.

▶ **Lemma 11.** *Let $J' = (k, X = X_1 \uplus \dots X_k, \mathcal{R}, (d_{\mathbf{x}})_{\mathbf{x}\in X})$ be an instance of BINARY CONSTRAINED PARTITION CENTER⋆, $\mathsf{OPT}(J') \geq \frac{m}{c}$ for some integer c, and $0 < \delta < 1/c$. Then, there is an algorithm which runs in time $m^{\mathcal{O}(1)} n^{\mathcal{O}(c^2 k/\delta^2)}$, and outputs a solution C of J', of cost at most $(1+\delta)\mathsf{OPT}(J')$ with probability at least $1 - n^{-2}$.*

Before proving Lemma 11, we explain how all these result put together to form a proof of Lemma 5. We restate Lemma 5 for the convenience of the reader.

[4] The proofs of results marked with ⋆ are deferred to the full version of the paper.

▶ **Lemma 5.** *There is an algorithm for* Binary Constrained Partition Center *that given an instance $J = (k, X = X_1^i \uplus \ldots \uplus X_k^i, \mathcal{R})$ and $0 < \epsilon < 1/2$, runs in time $m^{\mathcal{O}(1)} n^{\mathcal{O}((k/\epsilon)^4)}$, and outputs a solution of cost at most $(1+\epsilon)OPT(J)$ with probability at least $1 - n^{-2}$.*

Proof. Let $J = (k, X = X_1^i \uplus \ldots X_k^i, \mathcal{R})$ be the input instance of Binary Constrained Partition Center, and $0 < \epsilon < \frac{1}{2}$ be the error parameter. Let $(\mathbf{c}_1^*, \ldots, \mathbf{c}_k^*)$ be an optimal solution for J. Let $r \geq 2$ be an integer which we fix later. First, for each $i \in [k]$ we obtain r vectors $\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(r)} \in X_i$ which satisfy the conditions of Lemma 10. Their existence is guaranteed by Lemma 10, and we guess them in time $n^{\mathcal{O}(rk)}$ over all $i \in [k]$. For each $i \in [k]$, let Q_i be the set of coordinates on which $\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(r)}$ agree, $Q = \left\{ j \in \bigcap_{i \in [k]} Q_i \colon (\mathbf{x}_1^{(1)}[j], \mathbf{x}_2^{(1)}[j], \ldots, \mathbf{x}_k^{(1)}[j]) \in R_j \right\}$, and $\overline{Q} = [m] \setminus Q$. Next, we construct a solution $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ as follows. For each $i \in [k]$ and $j \in Q$, we set $\mathbf{c}_i[j] = \mathbf{x}_i^{(1)}[j]$.

Towards finding the entries of vectors $\mathbf{c}_1, \ldots, \mathbf{c}_k$ at the coordinates in $\overline{Q}$, we use Lemma 11. Let J' be the instance of Binary Constrained Partition Center*, which is a natural restriction of J to $\overline{Q}$. That is, $J' = (k, X' = X_1' \uplus \ldots X_k', \mathcal{R}|_{\overline{Q}}, (d_{\mathbf{x}|_{\overline{Q}}})_{\mathbf{x} \in X'})$, where for each $i \in [k]$, $X_i' = \{\mathbf{x}|_{\overline{Q}} \colon \mathbf{x} \in X_i\}$ and for each $\mathbf{x} \in X_i$, $d_{\mathbf{x}|_{\overline{Q}}} = d_H^Q(\mathbf{x}, \mathbf{x}_i^{(1)})$. By the second condition in Lemma 10, we have that $|\overline{Q}| \leq rk \cdot \mathsf{OPT}(J)$.

▷ Claim 12. $\mathsf{OPT}(J) \leq \mathsf{OPT}(J') \leq \left(1 + \frac{1}{r-1}\right) \mathsf{OPT}(J)$.

Proof. First, we prove that $\mathsf{OPT}(J) \leq \mathsf{OPT}(J')$. Towards that we show that we can transform a solution $C' = (\mathbf{c}_1', \cdots, \mathbf{c}_k')$ of J' with the objective value d to a solution C of J with the same objective value. For each $i \in [k]$, consider $\widehat{\mathbf{c}}_i$ which is equal to $\mathbf{x}_i^{(1)}$ restricted to Q, and to $\mathbf{c}_i'$ restricted to $\overline{Q}$, and the solution $\widehat{C} = (\widehat{\mathbf{c}}_1, \cdots, \widehat{\mathbf{c}}_k)$. Clearly, $\widehat{C}$ satisfies $\mathcal{R}$ since on $\overline{Q}$ it is guaranteed by C' being a solution to J', and on Q by construction of Q. The objective value of C is

$$\begin{aligned}
\max_{i \in [k], \mathbf{x} \in X_i} d_H(\mathbf{x}, \mathbf{c}_i) &= \max_{i \in [k], \mathbf{x} \in X_i} \left(d_H^{\overline{Q}}(\mathbf{x}, \mathbf{c}_i) + d_H^Q(\mathbf{x}, \mathbf{c}_i) \right) \\
&= \max_{i \in [k], \mathbf{x} \in X_i} \left(d_H(\mathbf{x}|_{\overline{Q}}, \mathbf{c}_i') + d_H^Q(\mathbf{x}, \mathbf{x}_i^{(1)}) \right) \\
&= \max_{i \in [k], \mathbf{x} \in X_i} \left(d_H(\mathbf{x}|_{\overline{Q}}, \mathbf{c}_i') + d_{\mathbf{x}|_{\overline{Q}}} \right) = d.
\end{aligned}$$

Thus, $\mathsf{OPT}(J) \leq \mathsf{OPT}(J')$.

Next, we prove that $\mathsf{OPT}(J') \leq \left(1 + \frac{1}{r-1}\right) \mathsf{OPT}(J)$. Recall that $(\mathbf{c}_1^*, \ldots, \mathbf{c}_k^*)$ is an optimal solution for J. Then, $(\mathbf{e}_1^*, \ldots, \mathbf{e}_k^*)$, where each $\mathbf{e}_i^*$ is the restriction of $\mathbf{c}_i^*$ on $\overline{Q}$, is a solution for J'. For each $i \in [k]$ and $\mathbf{x} \in X_i$,

$$\begin{aligned}
d_H(\mathbf{x}|_{\overline{Q}}, \mathbf{e}_i^*) + d_{\mathbf{x}|_{\overline{Q}}} &= d_H^{\overline{Q}}(\mathbf{x}, \mathbf{c}_i^*) + d_H^Q(\mathbf{x}, \mathbf{x}_i^{(1)}) \\
&\leq d_H^{\overline{Q}}(\mathbf{x}, \mathbf{c}_i^*) + d_H^Q(\mathbf{x}, \mathbf{c}_i^*) + \frac{1}{r-1} \mathsf{OPT}(J) \qquad \text{(By Lemma 10)} \\
&\leq d_H(\mathbf{x}, \mathbf{c}_i^*) + \frac{1}{r-1} \mathsf{OPT}(J) \\
&\leq \left(1 + \frac{1}{r-1}\right) \mathsf{OPT}(J)
\end{aligned}$$

This completes the proof of the claim. ◁

Since $|\overline{Q}| \leq rk \cdot \mathsf{OPT}(J)$ and by Claim 12, we have that $\mathsf{OPT}(J') \geq \frac{|\overline{Q}|}{rk} = \frac{|\overline{Q}|}{c}$, where $c = rk$. Let $0 < \delta < \frac{1}{c}$ be a number which we fix later.

Now we apply Lemma 11 on the input J' and δ, and let $C' = (\mathbf{c}'_1, \ldots, \mathbf{c}'_k)$ be the solution for J' obtained. We know that the cost d' of $\mathbf{c}'$ is at most $(1+\delta)\mathsf{OPT}(J')$ with probability at least $1 - n^{-2}$. For the rest of the proof we assume that the cost $d' \leq (1+\delta)\mathsf{OPT}(J')$. Recall that we have partially computed the entries of the solution $\mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ for the instance J. That is, for each $j \in Q$ and $i \in [k]$, we have already set the value of $\mathbf{c}_i[j]$. Notice that $C' \subseteq \{0,1\}^{|\overline{Q}|}$. Since J' is obtained from J by restricting to $\overline{Q}$, there is a natural bijection f from $\overline{Q}$ to $[|\overline{Q}|]$ such that for each $\mathbf{x} \in X$ and $j \in \overline{Q}$, $\mathbf{x}[j] = \mathbf{y}[f(j)]$, where $\mathbf{y} = \mathbf{x}|_{\overline{Q}}$. Now for each $i \in [k]$ and $j \in \overline{Q}$, we set $\mathbf{c}_i[j] = \mathbf{c}'_i[f(j)]$.

In Claim 12, we have proven that the solution C of J obtained in this way has cost at most d'. By Lemma 11, we know that $d' \leq (1+\delta)\mathsf{OPT}(J')$. By Claim 12, $\mathsf{OPT}(J') \leq (1 + \frac{1}{r-1})\mathsf{OPT}(J)$. Thus, we have that the cost of the solution C of J is at most $(1+\delta)(1 + \frac{1}{r-1})\mathsf{OPT}(J)$. Now we fix $r = (1 + \frac{4}{\epsilon})$ and $\delta = \frac{\epsilon}{(2\epsilon+8)k}$. Then the cost of C is at most $(1+\epsilon)\mathsf{OPT}(J)$.

Running time analysis. The number of choices for $\{\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_1^{(r)}\} \subset X_1, \ldots, \{\mathbf{x}_k^{(1)}, \ldots, \mathbf{x}_k^{(r)}\} \subset X_k$ is at most $n^{\mathcal{O}(rk)} = n^{\mathcal{O}(k/\epsilon)}$. For each such choice, we run the algorithm of Lemma 11 which takes time at most $m^{\mathcal{O}(1)} n^{\mathcal{O}(c^2k/\delta^2)} = m^{\mathcal{O}(1)} n^{\mathcal{O}((k/\epsilon)^4)}$. Thus, the total running time is $m^{\mathcal{O}(1)} n^{\mathcal{O}((k/\epsilon)^4)}$. ◀

Now the only piece left is the proof of Lemma 11. We use the following tail inequality (a variation of Chernoff bound) in the proof of Lemma 11.

▶ **Proposition 13** (Lemma 1.2 [21]). *Let $X_1, \ldots, X_n$ be n independent 0-1 random variables, $X = \sum_{i=1}^n X_i$, and $0 < \epsilon \leq 1$. Then,* $\Pr[X > E[X] + \epsilon n] \leq e^{-\frac{1}{3}n\epsilon^2}$.

Finally, we prove Lemma 11.

Proof of Lemma 11. First, assume that $m < 9c^2 \log n/\delta^2$. If this is the case, we enumerate all possible solutions for J' and output the best solution. The number of solutions is at most $2^{k \cdot m} = n^{\mathcal{O}(c^2k/\delta^2)}$. Thus, in this case the algorithm is exact and deterministic, and the running time bound holds. For the rest of the proof we assume that $m \geq 9c^2 \log n/\delta^2$.

Binary Constrained Partition Center* can be formulated as a 0-1 optimization problem as explained below. For each $j \in [m]$ and tuple $t \in R_j$, we use a 0-1 variable $y_{j,t}$ to indicate whether the j^{th} entries of a solution form a tuple $t \in R_j$ or not. For any $i \in [k]$, $\mathbf{x} \in X_i$, $j \in [m]$ and $t \in R_j$, denote $\chi_i(\mathbf{x}[j], t) = 0$ if $\mathbf{x}[j] = t[i]$ and $\chi_i(\mathbf{x}[j], t) = 1$ if $\mathbf{x}[j] \neq t[i]$. Now Binary Constrained Partition Center* can be defined as the following 0-1 optimization problem.

$$
\begin{aligned}
&\min d \\
&\text{subject to} \\
&\sum_{t \in R_j} y_{j,t} = 1, && \text{for all } j \in [m]; \qquad (6)\\
&\sum_{j \in [m]} \sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot y_{j,t} \leq d - d_{\mathbf{x}}, && \text{for all } i \in [k] \text{ and } \mathbf{x} \in X_i \\
&y_{j,t} \in \{0,1\}, && \text{for all } j \in [m] \text{ and } t \in R_j.
\end{aligned}
$$

Any solution $y_{j,t}$ ($j \in [m]$ and $t \in R_j$) to (6) corresponds to the solution $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ where for all $j \in [m]$ and $t \in R_j$ such that $y_{j,t} = 1$, we have $(\mathbf{c}_1[j], \ldots, \mathbf{c}_k[j]) = t$.

Now, we solve the above optimization problem using linear programming relaxation and obtain a fractional solution $y^{\star}_{j,t}$ ($j \in [m]$ and $t \in R_j$) with cost d'. Clearly, $d' \leq d_{opt} = \mathsf{OPT}(J')$. Now, for each $j \in [m]$, independently with probability $y^{\star}_{j,t}$, we set $y'_{j,t} = 1$ and $y'_{j,t'} = 0$, for any $t' \in R_j \setminus \{t\}$. Then $y'_{j,t}$ ($j \in [m]$ and $t \in R_j$) form a solution to (6). Next we construct the solution $C = (\mathbf{c}_1, \ldots, \mathbf{c}_k)$ to Binary Constrained Partition Center*, corresponding to $y'_{j,t}$ ($j \in [m]$ and $t \in R_j$). That is, for all $j \in [m]$ and $t \in R_j$ such that $y_{j,t} = 1$, we have $(\mathbf{c}_1[j], \ldots, \mathbf{c}_k[j]) = t$.

For the running time analysis, notice that solving the linear program and performing the random rounding takes polynomial time in the size of the problem (6). And the size of (6) is polynomial in the size of J', so the running time bound is satisfied. It remains to show that the constructed solution has cost at most $(1+\delta)\mathsf{OPT}(J')$ with probability at least $1 - n^{-2}$.

For any $j \in [m]$, the above random rounding procedure ensures that there is exactly one tuple $t \in R_j$ such that $y'_{j,t} = 1$. This implies that for any $j \in [m]$, $i \in [k]$ and $\mathbf{x} \in X_i$, $\sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot y'_{j,t}$ is a 0-1 random variable. Since for each $j \in [m]$ the rounding procedure is independent, we have that for any $i \in [k]$ and $\mathbf{x} \in X_i$ the random variables $(\sum_{t \in R_1} \chi_i(\mathbf{x}[1], t) \cdot y'_{1,t}), \ldots, (\sum_{t \in R_m} \chi_i(\mathbf{x}[m], t) \cdot y'_{j,t})$ are independent. Hence, for any $i \in [k]$ and $\mathbf{x} \in X_i$, the Hamming distance between $\mathbf{x}$ and $\mathbf{c}_i$, $d_H(\mathbf{x}, \mathbf{c}_i) = \sum_{j \in [m]} \sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot y'_{j,t}$, is the sum of m independent 0-1 random variables. For each $i \in [k]$ and $\mathbf{x} \in X_i$, we upper bound the expected value of $d_H(\mathbf{x}, \mathbf{c}_i)$ as follows.

$$
\begin{aligned}
E[d_H(\mathbf{x}, \mathbf{c}_i)] &= E\left[\sum_{j \in [m]} \sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot y'_{j,t}\right] \\
&= \sum_{j \in [m]} \sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot E[y'_{j,t}] \\
&= \sum_{j \in [m]} \sum_{t \in R_j} \chi_i(\mathbf{x}[j], t) \cdot y^{\star}_{j,t} \\
&\leq d' - d_{\mathbf{x}} \qquad\qquad \text{(By the constraints of (6))}
\end{aligned}
$$

Fix $\epsilon = \frac{\delta}{c}$. Then, by Proposition 13, for all $i \in [k]$, and $\mathbf{x} \in X_i$,

$$\Pr[d_H(\mathbf{x}, \mathbf{c}_i) > d' - d_{\mathbf{x}} + \epsilon m] \leq e^{-\frac{1}{3} m \epsilon^2}.$$

Therefore, by the union bound,

$$\Pr[\text{There exist } i \in [k] \text{ and } \mathbf{x} \in X_i \text{ such that } d_H(\mathbf{x}, \mathbf{c}_i) > d' - d_{\mathbf{x}} + \epsilon m] \leq n \cdot e^{-\frac{1}{3} m \epsilon^2} \tag{7}$$

We remind that $m \geq 9c^2 \log n / \delta^2 = 9 \log n / \epsilon^2$ and so $n \cdot e^{-\frac{1}{3} m \epsilon^2} \leq n^{-2}$. Thus, by (7),

$$\Pr[\text{There exist } i \in [k] \text{ and } \mathbf{x} \in X_i \text{ such that } d_H(\mathbf{x}, \mathbf{c}_i) > d' - d_{\mathbf{x}} + \epsilon m] \leq n^{-2}. \tag{8}$$

Since $d' \leq \mathsf{OPT}(J')$ and $\mathsf{OPT}(J') \geq m/c$, $d' + \epsilon m \leq (1 + c\epsilon)\mathsf{OPT}(J')$. Then, the probability that there exist $i \in [k]$ and $\mathbf{x} \in X_i$ such that $d_H(\mathbf{x}, \mathbf{c}_i) > (1 + c\epsilon)\mathsf{OPT}(J') - d_{\mathbf{x}}$ is at most n^{-2} by (8). Since $c\epsilon = \delta$, the proof is complete. ◀

4 Applications

In this section we explain the impact of Theorem 3 about Binary Constrained k-Center to other problems around low-rank matrix approximation. We would like to mention that Binary Constrained k-Center is very similar to the Binary Constrained Clustering problem from [9]. In Binary Constrained k-Center we want to minimize the maximum distance of a vector from the input set of vectors to the closest center, whereas in Binary Constrained Clustering the sum of distances is minimized. While these problems are different, the reduction we explain here, except a few details, are identical to the ones described in [9]. For reader's convenience, we give one reduction (Lemma 14) in full details and skip all other reductions, which are similar.

In the following lemma we show that ℓ_1-Rank-r Approximation over GF(2) is a special case of Binary Constrained k-Center.

▶ **Lemma 14** ($\star$). *There is an algorithm that given an instance $(\mathbf{A}, r)$ of ℓ_1-Rank-r Approximation over GF(2), where $\mathbf{A}$ is an $m \times n$-matrix and r is an integer, runs in time $\mathcal{O}(m + n + 2^{2r})$, and outputs an instance $J = (X, k = 2^r, \mathcal{R})$ of Binary Constrained k-Center with the following property. Given any α-approximate solution C to J, an α-approximate solution $\mathbf{B}$ to $(\mathbf{A}, r)$ can be constructed in time $\mathcal{O}(rmn)$ and vice versa.*

Thus, Theorem 1 follows from Theorem 3 and Lemma 14.

Low Boolean-Rank Approximation. Let $\mathbf{A}$ be a binary $m \times n$ matrix. Now we consider the elements of $\mathbf{A}$ to be *Boolean* variables. The *Boolean rank* of $\mathbf{A}$ is the minimum r such that $\mathbf{A} = \mathbf{U} \wedge \mathbf{V}$ for a Boolean $m \times r$ matrix $\mathbf{U}$ and a Boolean $r \times n$ matrix $\mathbf{V}$, where the product is Boolean, that is, the logical $\wedge$ plays the role of multiplication and $\vee$ the role of sum. Here $0 \wedge 0 = 0$, $0 \wedge 1 = 0$, $1 \wedge 1 = 1$, $0 \vee 0 = 0$, $0 \vee 1 = 1$, and $1 \vee 1 = 1$. Thus the matrix product is over the Boolean semi-ring $(0, 1, \wedge, \vee)$. This can be equivalently expressed as the normal matrix product with addition defined as $1 + 1 = 1$. Binary matrices equipped with such algebra are called *Boolean matrices.*

In Boolean ℓ_1-Rank-r Approximation, we are given an $m \times n$ binary data matrix $\mathbf{A}$ and a positive integer r, and we seek a binary matrix $\mathbf{B}$ optimizing

$$\begin{aligned} &\text{minimize } \|\mathbf{A} - \mathbf{B}\|_1 \\ &\text{subject to } \operatorname{rank}(\mathbf{B}) \le r. \end{aligned}$$

Here, by the rank of binary matrix $\mathbf{B}$ we mean its Boolean rank, and norm $\|\cdot\|_1$ is the *column sum norm.* Similar to Lemma 14, one can prove that Boolean ℓ_1-Rank-r Approximation is a special case of Binary Constrained k-Center, where $k = 2^r$. Thus, we get the following corollary from Theorem 3.

▶ **Corollary 15.** *There is an algorithm for Boolean ℓ_1-Rank-r Approximation that given an instance $I = (\mathbf{A}, r)$ and $0 < \varepsilon < 1$, runs in time $m^{\mathcal{O}(1)} n^{\mathcal{O}(2^{4r}/\varepsilon^4)}$, and outputs a $(1 + \varepsilon)$-approximate solution with probability at least $1 - 2n^{-2}$.*

Projective k-center. The Binary Projective k-Center problem is a variation of the Binary k-Center problem, where the centers of clusters are linear subspaces of bounded dimension r. (For $r = 1$ this is Binary k-Center and for $k = 1$ this is ℓ_1-Rank-r Approximation over GF(2).) Formally, in Binary Projective k-Center we are given a set $X \subseteq \{0, 1\}^m$ of n vectors and positive integers k and r. The objective is to find a family of r-dimensional linear subspaces $C = \{C_1, \ldots, C_k\}$ over GF(2) minimizing $\max_{\mathbf{x} \in X} d_H(\mathbf{x}, \bigcup_{i=1}^k C)$.

To see that Binary Projective k-Center is a special case of Binary Constrained k-Center, we observe that the condition that C_i is an r-dimensional subspace over GF(2) can be encoded (as in Lemma 14) by 2^r constraints. This observation leads to the following lemma.

▶ **Lemma 16.** *There is an algorithm that given an instance (X, r, k) of* Binary Projective k-Center*, runs in time $\mathcal{O}(m + n + 2^{\mathcal{O}(rk)})$, and outputs an instance $J = (X, k' = 2^{kr}, \mathcal{R})$ of* Binary Constrained k-Center *with the following property. Given any α-approximate solution C to J, an α-approximate solution C' to (X, r, k) can be constructed in time $\mathcal{O}(rkmn)$ and vice versa.*

Combining Theorem 3 and Lemma 16 together, we get the following corollary.

▶ **Corollary 17.** *There is an algorithm for* Binary Projective k-Center *that given an instance $I = (X, r, k)$ and $0 < \varepsilon < 1$, where $X \subseteq \{0,1\}^m$ is a set of n vectors and $r, k \in \mathbb{N}$, runs in time $m^{\mathcal{O}(1)} n^{\mathcal{O}(2^{4kr}/\varepsilon^4)}$, and outputs a $(1+\varepsilon)$-approximate solution with probability at least $1 - 2n^{-2}$.*

5 Conclusion

In this paper we gave a randomized PTAS for the Binary Constrained k-Center problem. This yields the first approximation scheme for ℓ_1-Rank-r Approximation over GF(2) and its Boolean variant. This paper leaves several interesting open problems. The running time of our $(1+\varepsilon)$-approximation algorithm is $m^{\mathcal{O}(1)} n^{\mathcal{O}(2^{4r} \cdot \varepsilon^{-4})}$. How far is this running time from being optimal? A simple adaptation of the result of Cygan et al. [6] for Closest String, yields that already for $r = 1$, an $(1+\varepsilon)$-approximation in time $n^{\mathcal{O}(1)} \cdot f(\varepsilon)$, for any computable function f, would imply that FPT=W[1]. Also the existence of a PTAS for $r = 1$ with running time $f(\varepsilon) n^{o(1/\varepsilon)}$, for any computable function f, would contradict the Exponential Time Hypothesis [6]. But these results do not exclude the opportunity of having an algorithm of running time $f(r, \varepsilon) \cdot (nm)^{poly(1/\varepsilon)}$ for some function f. Even the existence of an algorithm for ℓ_1-Rank-r Approximation over GF(2) of running time $m^{\mathcal{O}(1)} n^{poly(r,\varepsilon)}$ is an interesting open question.

References

1 Frank Ban, Vijay Bhattiprolu, Karl Bringmann, Pavel Kolev, Euiwoong Lee, and David P. Woodruff. A PTAS for p-low rank approximation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 747–766. SIAM, 2019. doi:10.1137/1.9781611975482.47.

2 Eduard Bartl, Radim Belohlávek, and Jan Konecny. Optimal decompositions of matrices with grades into binary and graded matrices. *Annals of Mathematics and Artificial Intelligence*, 59(2):151–167, June 2010. doi:10.1007/s10472-010-9185-y.

3 Radim Belohlávek and Vilém Vychodil. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *J. Computer and System Sciences*, 76(1):3–20, 2010. doi:10.1016/j.jcss.2009.05.002.

4 Michael W Berry, Susan T Dumais, and Gavin W O'Brien. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

5 Avrim Blum, John Hopcroft, and Ravi Kannan. *Foundations of Data Science*. Cambridge University Press, 2020. URL: https://books.google.co.in/books?id=koHCDwAAQBAJ.

6 Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Lower Bounds for Approximation Schemes for Closest String. In *Proceedings of the 15th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 53 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:10, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SWAT.2016.12.

7 Chen Dan, Kristoffer Arnsfelt Hansen, He Jiang, Liwei Wang, and Yuchen Zhou. On low rank approximation of binary matrices. *CoRR*, abs/1511.01699, 2015. arXiv:1511.01699.

8 Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

9 Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Approximation schemes for low-rank binary matrix approximation problems. *CoRR*, abs/1807.07156, 2018. `arXiv:1807.07156`.

10 Fedor V. Fomin, Petr A. Golovach, and Fahad Panolan. Parameterized low-rank binary matrix approximation. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107 of *LIPIcs*, pages 53:1–53:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. `doi:10.4230/LIPIcs.ICALP.2018.53`.

11 Yinghua Fu, Nianping Jiang, and Hong Sun. Binary matrix factorization and consensus algorithms. In *Proceedings of the International Conference on Electrical and Control Engineering (ICECE)*, pages 4563–4567. IEEE, 2010.

12 Leszek Gasieniec, Jesper Jansson, and Andrzej Lingas. Approximation algorithms for hamming clustering problems. *J. Discrete Algorithms*, 2(2):289–301, 2004. `doi:10.1016/S1570-8667(03)00079-0`.

13 Nicolas Gillis and Stephen A. Vavasis. On the complexity of robust PCA and ℓ_1-norm low-rank matrix approximation. *CoRR*, abs/1509.09236, 2015. `arXiv:1509.09236`.

14 Harold W. Gutch, Peter Gruber, Arie Yeredor, and Fabian J. Theis. ICA over finite fields - separability and algorithms. *Signal Processing*, 92(8):1796–1808, 2012. `doi:10.1016/j.sigpro.2011.10.003`.

15 Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

16 Peng Jiang and Michael T. Heath. Mining discrete patterns via binary matrix factorization. In *ICDM Workshops*, pages 1129–1136. IEEE Computer Society, 2013.

17 Peng Jiang, Jiming Peng, Michael Heath, and Rui Yang. A clustering approach to constrained binary matrix factorization. In *Data Mining and Knowledge Discovery for Big Data: Methodologies, Challenge and Opportunities*, pages 281–303. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

18 Yishan Jiao, Jingyi Xu, and Ming Li. On the k-closest substring and k-consensus pattern problems. In *Proceedings of the 15th Annual Symposium on Combinatorial Pattern (CPM)*, volume 3109 of *Lecture Notes in Comput. Sci.*, pages 130–144. Springer, 2004.

19 Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999. `doi:10.1145/324133.324140`.

20 Mehmet Koyutürk and Ananth Grama. Proximus: A framework for analyzing very high dimensional discrete-attributed datasets. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 147–156, New York, NY, USA, 2003. ACM. `doi:10.1145/956750.956770`.

21 Ming Li, Bin Ma, and Lusheng Wang. On the closest string and substring problems. *J. ACM*, 49(2):157–171, 2002. `doi:10.1145/506147.506150`.

22 Haibing Lu, Jaideep Vaidya, Vijayalakshmi Atluri, and Yuan Hong. Constraint-aware role mining via extended boolean matrix decomposition. *IEEE Trans. Dependable Sec. Comput.*, 9(5):655–669, 2012. `doi:10.1109/TDSC.2012.21`.

23 Bin Ma and Xiaoming Sun. More efficient algorithms for closest string and substring problems. *SIAM J. Computing*, 39(4):1432–1443, 2009. `doi:10.1137/080739069`.

24 Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011. `doi:10.1561/2200000035`.

25 Pauli Miettinen, Taneli Mielikäinen, Aristides Gionis, Gautam Das, and Heikki Mannila. The discrete basis problem. *IEEE Trans. Knowl. Data Eng.*, 20(10):1348–1362, 2008. `doi:10.1109/TKDE.2008.53`.

26 Pauli Miettinen and Jilles Vreeken. Model order selection for boolean matrix factorization. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 51–59. ACM, 2011. `doi:10.1145/2020408.2020424`.

27 L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *Quart. J. Math. Oxford Ser. (2)*, 11:50–59, 1960. doi:10.1093/qmath/11.1.50.

28 Barsha Mitra, Shamik Sural, Jaideep Vaidya, and Vijayalakshmi Atluri. A survey of role mining. *ACM Comput. Surv.*, 48(4):50:1–50:37, February 2016. doi:10.1145/2871148.

29 Rafail Ostrovsky and Yuval Rabani. Polynomial-time approximation schemes for geometric min-sum median clustering. *J. ACM*, 49(2):139–156, 2002. doi:10.1145/506147.506149.

30 Amichai Painsky, Saharon Rosset, and Meir Feder. Generalized independent component analysis over finite alphabets. *IEEE Trans. Information Theory*, 62(2):1038–1053, 2016. doi:10.1109/TIT.2015.2510657.

31 Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

32 Bao-Hong Shen, Shuiwang Ji, and Jieping Ye. Mining discrete patterns via binary matrix factorization. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 757–766, New York, NY, USA, 2009. ACM. doi:10.1145/1557019.1557103.

33 Zhao Song, David P. Woodruff, and Peilin Zhong. Low rank approximation with entrywise ℓ_1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 688–701. ACM, 2017. doi:10.1145/3055399.3055431.

34 Jaideep Vaidya. Boolean matrix decomposition problem: Theory, variations and applications to data engineering. In *Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE)*, pages 1222–1224. IEEE Computer Society, 2012. doi:10.1109/ICDE.2012.144.

35 Jaideep Vaidya, Vijayalakshmi Atluri, and Qi Guo. The role mining problem: finding a minimal descriptive set of roles. In *Proceedings of the 12th ACM Symposium on Access Control Models and (SACMAT)*, pages 175–184, 2007. doi:10.1145/1266840.1266870.

36 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014. doi:10.1561/0400000060.

37 Arie Yeredor. Independent component analysis over Galois fields of prime order. *IEEE Trans. Information Theory*, 57(8):5342–5359, 2011. doi:10.1109/TIT.2011.2145090.