Guessing Efficiently for Constrained Subspace Approximation

Adita Bhaskara* Sepideh Mahabadi† Madhusudhan Reddy Pittu‡ Ali Vakilian§
David P. Woodruff¶

Abstract

In this paper we study constrained subspace approximation problem. Given a set of n points $\{a_1, \ldots, a_n\}$ in $\mathbb{R}^d$, the goal of the *subspace approximation* problem is to find a k dimensional subspace that best approximates the input points. More precisely, for a given $p \geq 1$, we aim to minimize the pth power of the ℓ_p norm of the error vector $(\|a_1 - \boldsymbol{P}a_1\|, \ldots, \|a_n - \boldsymbol{P}a_n\|)$, where $\boldsymbol{P}$ denotes the projection matrix onto the subspace and the norms are Euclidean. In *constrained* subspace approximation (CSA), we additionally have constraints on the projection matrix $\boldsymbol{P}$. In its most general form, we require $\boldsymbol{P}$ to belong to a given subset $\mathcal{S}$ that is described explicitly or implicitly.

We introduce a general framework for constrained subspace approximation. Our approach, that we term coreset-guess-solve, yields either $(1+\varepsilon)$-multiplicative or ε-additive approximations for a variety of constraints. We show that it provides new algorithms for partition-constrained subspace approximation with applications to *fair* subspace approximation, k-means clustering, and projected non-negative matrix factorization, among others. Specifically, while we reconstruct the best known bounds for k-means clustering in Euclidean spaces, we improve the known results for the remainder of the problems.

1 Introduction

Large data sets, often represented as collections of high-dimensional points, naturally arise in fields such as machine learning, data mining, and computational geometry. Despite their high-dimensional nature, these points typically exhibit low intrinsic dimensionality. Identifying (or summarizing) this underlying low-dimensional structure is a fundamental algorithmic question with numerous applications to data analysis. We study a general formulation, that we call the *subspace approximation problem.*

In subspace approximation, given a set of n points $\{a_1, \ldots, a_n\} \in \mathbb{R}^d$ and a rank parameter k, we consider the problem of "best approximating" the input points with a k-dimensional subspace in a high-dimensional space. Here the goal is to find a rank k projection $\boldsymbol{P}$ that minimizes the projection costs $\|a_i - \boldsymbol{P}a_i\|$, aggregated over $i \in [n]$. The choice of aggregation leads to different well-studied formulations. In the ℓ_p subspace approximation problem, the objective is $(\sum_i \|a_i - \boldsymbol{P}a_i\|_2^p)$. Formally, denoting by A the $d \times n$ matrix whose ith column is a_i, the ℓ_p-subspace approximation problem asks to find a rank k projection matrix $\boldsymbol{P} \in \mathbb{R}^{d \times d}$ that minimizes $\|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p := \sum_{i=1}^n \|a_i - \boldsymbol{P}a_i\|_2^p$. For different choices of p, ℓ_p-subspace approximation captures some well-studied

*University of Utah: bhaskaraaditya@gmail.com
†Microsoft Research–Redmond: smahabadi@microsoft.com
‡Carnegie Mellon University: mpittu@andrew.cmu.edu
§Toyota Technological Institute at Chicago (TTIC): vakilian@ttic.edu
¶Carnegie Mellon University: dwoodruf@andrew.cmu.edu

problems, notably the *median hyperplane problem* (when $p = 1$), the *principal component analysis (PCA) problem* (when $p = 2$), and the *center hyperplane problem* (when $p = \infty$).

Subspace approximation for general p turns out to be NP-hard for all $p \neq 2$. For $p > 2$, semidefinite programming helps achieve a constant factor approximation (for fixed p) for the problem [DTV11]. Matching hardness results were also shown for the case $p > 2$, first assuming the Unique Games Conjecture [DTV11], and then based only on P $\neq$ NP [GRSW16]. For $p < 2$, hardness results were first shown in the work of [CW15].

Due to the ubiquitous applications of subspace approximation in various domains, several "constrained" versions of the problem have been extensively studied as well [DFK+04, YZ13, PDK13, APD14, BZMD14, CEM+15]. In the most general setting of the *constrained ℓ_p-subspace approximation* problem, we are additionally given a collection $\mathcal{S}$ of rank-k projection matrices (specified either explicitly or implicitly) and the goal is to find a projection matrix $\boldsymbol{P} \in \mathcal{S}$ minimizing the objective. I.e.,

$$\min_{\boldsymbol{P} \in \mathcal{S}} \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p. \tag{1}$$

Some examples of problems in constrained subspace approximation include the well-studied *column subset selection* [BMD09, Tro09, DR10, CMI12, GS12, BDMI14, ABF+16] where the projection matrices are constrained to project on to the span of k of the original vectors, (k, z)-*means clustering* in which the set of projection matrices can be specified by the partitioning of the points into k clusters (see [CEM+15] for a reference), and many more which we will describe in this paper.

1.1 Our Contributions and Applications

In this paper, we provide a general algorithmic framework for constrained ℓ_p-subspace approximation that yields either $(1 + \varepsilon)$-multiplicative or ε-additive error approximations to the objective (depending on the setting), with running time exponential in k. We apply the framework to several classes of constrained subspace approximation, leading to new results or results matching the state-of-the-art for these problems. Note that since the problems we consider are typically APX-hard (including k-means, and even the *unconstrained* version of ℓ_p-subspace approximation for $p > 2$), a running time exponential in k is necessary for our results, assuming the Exponential Time Hypothesis; a discussion in Section 2. Before presenting our results, we start with an informal description of the framework.

Overview of Approach. Our approach is based on coresets [FMS07] (also [CASS21, CALS+24, HLW24] and references therein), but turns out to be different from the standard approach in a subtle yet important way. Recall that a (strong) coreset for an optimization problem $\mathcal{O}$ on set of points $\boldsymbol{A}$ is a subset $\boldsymbol{B}$ such that for any solution for $\mathcal{O}$, the cost on $\boldsymbol{B}$ is approximately the same as the cost on $\boldsymbol{A}$, up to an appropriate scaling. In the formulation of ℓ_p-subspace approximation above, a coreset for a dataset $\boldsymbol{A}$ would be a subset $\boldsymbol{B}$ of its columns with $k' \ll n$ columns, such that for all k-dimensional subspaces, each defined by some $\boldsymbol{P}$, $\|\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B}\|_{2,p}^p \approx \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p$, up to scaling. Thus the goal becomes to minimize the former quantity.

In the standard coreset approach, first a coreset is obtained, and then a problem-specific enumeration procedure is used to find a near optimal solution $\boldsymbol{P}$. For example, for the k-means clustering objective, one can consider all the k-partitions of the points in the coreset $\boldsymbol{B}$; each partition leads to a set of candidate centers, and the best of these candidate solutions will be an approximate solution to the full instance. Similarly for (unconstrained) ℓ_p-subspace approximation, one observes that for an optimal solution, the columns of $\boldsymbol{P}$ must lie in the span of the vectors of $\boldsymbol{B}$, and thus one can enumerate over the combinations of the vectors of $\boldsymbol{B}$. Each combination gives a candidate $\boldsymbol{P}$, and the best of these candidate solutions is an approximate solution to the full instance.

However, this approach does not work in general for constrained subspace approximation. To see this, consider the very simple constraint of having the columns of $\boldsymbol{P}$ coming from some given subspace S. Here, the coreset for ℓ_p-subspace approximation on $\boldsymbol{A}$ will be some set $\boldsymbol{B}$ that is "oblivious" of the subspace S. Thus, enumerating over combinations of $\boldsymbol{B}$ may not yield any vectors in S!

Our main idea is to avoid enumeration over candidate solutions, but instead, we view the solution (the matrix $\boldsymbol{P} \in \mathbb{R}^{d \times k}$) as simply a set of variables. We then note that since the goal is to use $\boldsymbol{P}$ to approximate $\boldsymbol{B}$, there must be some combination of the vectors of $\boldsymbol{P}$ (equivalently, a set of k coefficients) that approximates each vector a_i in $\boldsymbol{B}$. If the coreset size is k', there are only $k \cdot k'$ coefficients in total, and we can thus hope to enumerate these coefficients in time $\exp(k \cdot k')$. For every given choice of coefficients, we can then solve an optimization problem to find the optimal $\boldsymbol{P}$. For the constraints we consider (including the simple example above), this problem turns out to be convex, and can thus be solved efficiently!

This simple idea yields ε-additive approximation guarantees for a range of problems. We then observe that in specific settings of interest, we can obtain $(1 + \varepsilon)$-multiplicative approximations by avoiding guessing of the coefficients. In these settings, once the coefficients have been guessed, there is a *closed form* for the optimal basis vectors, in the form of low degree polynomials of the coefficients. We can then use the literature on solving polynomial systems of equations (viewing the coefficients as variables) to obtain algorithms that are more efficient than guessing. The framework is described more formally in Section 3.

We believe our general technique of using coresets to reduce the number of *coefficients* needed in order to turn a constrained non-convex optimization problem into a convex one, may be of broader applicability. We note it is fundamentally different than the "guess a sketch" technique for variable reduction in [RSW16, BBB+19, BWZ19, MW20] and the techniques for reducing variables in non-negative matrix factorization [Moi16]. To support this statement, the guess a sketch technique requires the existence of a small sketch, and consequently has only been applied to approximation with entrywise p-norms for $p \leq 2$ and weighted variants [RSW16, BBB+19, MW20], whereas our technique applies to a much wider family of norms.

Relation to Prior Work. We briefly discuss the connection to prior work on binary matrix factorization using coresets. The work of [VVWZ23] addresses binary matrix factorization by constructing a strong coreset that reduces the number of distinct rows via importance sampling, leveraging the discrete structure of binary inputs. Our framework generalizes these ideas to continuous settings: we use strong coresets not merely to reduce distinct rows, but to reduce the number of variables in a polynomial system for solving continuous constrained optimization problems. This enables us to extend the approach to real-valued matrices and to more general loss functions. Overall, our framework can be seen as a generalization and unification of prior coreset-based "guessing" strategies, adapting them to significantly broader settings.

Applications. We apply our framework to the following applications. Each of these settings can be viewed as subspace approximation with a constraint on the subspace (i.e., on the projection matrix), or on properties of the associated basis vectors. Below we describe these applications, mention how they can be formulated as Constrained Subspace Approximation, and state our results for them. See Table 1 for a summary.

Problem	Running Time	Approx.	Prior Work
PC-ℓ_p-Subspace Approx.	$(\frac{\kappa}{\varepsilon})^{\mathrm{poly}(\frac{k}{\varepsilon})} \cdot \mathrm{poly}(n)$ (4.6)	$\Big(O(\varepsilon p) \cdot \\|\boldsymbol{A}\\|_{p,2}^p\Big)^+$	-
	$n^{O(\frac{k^2}{\varepsilon})} \cdot \mathrm{poly}(H)$ (4.7)	$(1+\varepsilon)^*$	-
Constrained Subspace Est.	$\mathrm{poly}(n) \cdot (\frac{1}{\delta})^{O(\frac{k^2}{\varepsilon})}$ (4.3)	$(1+\varepsilon, O(\delta \cdot \\|\boldsymbol{A}\\|_F^2))^\dagger$	$\sim$
	$O(\frac{nd\gamma}{\varepsilon})^{O(\frac{k^3}{\varepsilon})}$ (4.4)	$(1+\varepsilon)^*$	$\sim$
PNMF	$O(\frac{dk^2}{\varepsilon}) \cdot (\frac{1}{\delta})^{O(\frac{k^2}{\varepsilon})}$(4.15)	$(1+\varepsilon, O(\delta \cdot \\|\boldsymbol{A}\\|_F^2))^\dagger$	$\sim$
	$(\frac{nd\gamma}{\varepsilon})^{O(\frac{k^3}{\varepsilon})}$ (4.16)	$(1+\varepsilon)^*$	$\sim$
k-Means Clustering	$O(nnz(\boldsymbol{A}) + 2^{\widetilde{O}(\frac{k}{\varepsilon})} + n^{o(1)})$ (4.19)	$(1+\varepsilon)^*$	[FMS07]
Sparse PCA	$d^{O(\frac{k^3}{\varepsilon^2})} \cdot \frac{k^3}{\varepsilon}$ (4.21)	$(\varepsilon\\|\boldsymbol{A} - \boldsymbol{A}_k\\|_F^2)^+$	[DP22]

Table 1: Summary of the upper bound results we get using our framework. In the approximation column, we use super scripts $*, +, \dagger$ to represent if its a multiplicative, additive, or multiplicative-additive approximation respectively. In the notes on prior work column, we use tilde ($\sim$) to indicate that no prior theoretical guarantees are known (only heuristics) and hyphen ($-$) to specify that the problem is new.

1.1.1 Subspace Approximation with Partition Constraints

First, we study a generalization of ℓ_p-subspace approximation, where we have *partition constraints* on the subspace. More specifically, we consider *PC-ℓ_p-subspace approximation*, where besides the point set $\{a_1, \cdots, a_n\} \in \mathbb{R}^d$, we are given ℓ subspaces $S_1, \cdots, S_\ell$ along with capacities $k_1, \cdots, k_\ell$ such that $\sum_{i=1}^{\ell} k_i = k$. Now the set of valid projections $\mathcal{S}$ is implicitly defined to be the set of projections onto the subspaces that are obtained by selecting k_i vectors from S_i for each $i \in [\ell]$, taking their span.

PC-ℓ_p-subspace approximation can be viewed as a variant of data summarization with "fair representation". Specifically, when S_i is the span of the vectors (or points) in group i, then by setting k_i values properly (depending on the application or the choice of policy makers), PC-ℓ-subspace approximation captures the problem of finding a summary of the input data in which groups are fairly represented. This corresponds to the equitable representation criterion, a popular notion studied extensively in the fairness of algorithms, e.g., clustering [KAM19, JNN20, CKR20, HMV23].[1] We show the following results for PC-subspace approximation:

- First, in Theorem 4.6, we show for any $p \geq 1$, an algorithm for PC-ℓ_p-subspace approximation with runtime $(\frac{\kappa}{\varepsilon})^{\mathrm{poly}(k/\varepsilon)} \cdot \mathrm{poly}(n)$ that returns a solution with additive error at most $O(\varepsilon p) \cdot \|\boldsymbol{A}\|_{p,2}^p$, where κ is the condition number of the optimal choice of vectors from the given subspaces.

- For $p = 2$, which is one of the most common loss functions for PC-ℓ_p-subspace approximation, we also present a multiplicative approximation guarantee. There exists a $(1+\varepsilon)$-approximation algorithm running in time $s^{O(k^2/\varepsilon)} \cdot \mathrm{poly}(H)$ where H is the bit complexity of each element in the input and s is the sum of the dimensions of the input subspaces $S_1, \cdots, S_\ell$, i.e., $s = \sum_{j=1}^{\ell} \dim(S_j)$. The formal statement is in Theorem 4.7.

[1]We note that the fair representation definitions differ from those in the line of work on fair PCA and column subset selection [STM+18, TSS+19, MOT23, SVWZ24], where the *objective contributions* (i.e., projection costs) of different groups must either be equal (if possible) or ensure that the maximum incurred cost is minimized. We focus on the question of groups having equal, or appropriately bounded, *representation* among the chosen low-dimensional subspace (i.e., directions). This distinction is also found in algorithmic fairness studies of other problems, such as clustering.

1.1.2 Constrained Subspace Estimation

The *Constrained Subspace Estimation* problem originates from the signal processing community [SVK+17], and aims to find a subspace V of dimension k, that best approximates a collection of experimentally measured subspaces $T_1, \cdots, T_m$, with the constraint that it intersects a model-based subspace W in at least a predetermined number of dimensions ℓ, i.e., $\dim(V \cap W) \geq \ell$. This problem arises in applications such as beamforming, where the model-based subspace is used to encode the available prior information about the problem. The paper of [SVK+17] formulates and motivates that problem, and further present an algorithm based on a semidefinite relaxation of this non-convex problem, where its performance is only demonstrated via numerical simulation.

We show in Section 4.1, that this problem can be reduced to at most k instances of PC-ℓ_2-subspace approximation, in which the number of parts is 2. This will give us the following result for the constrained subspace estimation problem.

- In Corollary 4.3, we show a $(1+\varepsilon, \delta\|A\|_F^2)$-multiplicative-additive approximation in time $\mathrm{poly}(n) \cdot (1/\delta)^{O(k^2/\varepsilon)}$.
- In Theorem 4.4, we show a $(1+\varepsilon)$ multiplicative approximation in time $O(nd\gamma/\varepsilon)^{O(k^3/\varepsilon)}$ where we assume A has integer entries of absolute value at most γ. We assume that $\gamma = \mathrm{poly}(n)$.

1.1.3 Projective Non-Negative Matrix Factorization

Projective Non-Negative Matrix Factorization (PNMF) [YO05] (see also [YYO09, YO10]) is a variant of Non-Negative Matrix Factorization (NMF), used for dimensionality reduction and data analysis, particularly for datasets with non-negative values such as images and texts. In NMF, a non-negative matrix $\boldsymbol{X}$ is factorized into the product of two non-negative matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ such that $\boldsymbol{X} \approx \boldsymbol{WH}$ where $\boldsymbol{W}$ contains basis vectors, and $\boldsymbol{H}$ represents coefficients. In PNMF, the aim is to approximate the data matrix by projecting it onto a subspace spanned by non-negative vectors, similar to NMF. However, in PNMF, the factorization is constrained to be *projective*.

Formally, PNMF can be formulated as a constrained ℓ_2-subspace approximation as follows: the set of feasible projection matrices $\mathcal{S}$, consists of all matrices that can be written as $\boldsymbol{P} = UU^T$, where U is a $d \times k$ orthonormal matrix with all non-negative entries.

We show the following results:

- In Theorem 4.15, we show a $(1+\varepsilon, \delta\|A\|_F^2)$-multiplicative-additive approximation in time $O(dk^2/\varepsilon) \cdot (1/\delta)^{O(k^2/\varepsilon)}$.
- In Theorem 4.16, we show a $(1+\varepsilon)$ multiplicative approximation in time $(nd\gamma)^{O(k^3/\varepsilon)}$, where we assume A has integer entries of absolute value at most γ.

1.1.4 k-Means Clustering

k-means is a popular clustering algorithm widely used in data analysis and machine learning. Given a set of n vectors $a_1, \cdots, a_n$ and a parameter k, the goal of k-means clustering is to partition these vectors into k clusters $\{C_1, \cdots, C_k\}$ such that the sum of the squared distances of all points to their corresponding cluster center $\sum_{i=1}^n \|a_i - \mu_{C(a_i)}\|_2^2$ is minimized, where $C(a_i)$ denotes the cluster that a_i belongs to and $\mu_{C(a_i)}$ denotes its center. It is an easy observation that once the clustering is determined, the cluster centers need to be the centroid of the points in each cluster. It is shown in [CEM+15] that this problem is an instance of constrained subspace approximation. More precisely, the set of valid projection matrices are all those that can be written as $\boldsymbol{P} = X_C X_C^T$, where X_C

is a $n \times k$ matrix where $X_C(i,j)$ is $1/\sqrt{|C_j|}$ if $C(a_i) = j$ and 0 otherwise. Note that this is an orthonormal matrix and thus $X_C X_C^T$ is an orthogonal projection matrix. Further, note that using our language we need to apply the constrained subspace approximation on the matrix A^T, i.e., $\min_{\boldsymbol{P}\in\mathcal{S}} \|A^T - \boldsymbol{P}A^T\|_F^2$.

In Theorem 4.19, we show a $(1+\varepsilon)$ approximation algorithm for k-means that runs in $O(nnz(\boldsymbol{A}) + 2^{\widetilde{O}(k/\varepsilon)} + n^{o(1)})$ time, whose dependency on k and ε matches that of [FMS07].

1.1.5 Sparse PCA

The goal of Principal Component Analysis (PCA) is to find k linear combinations of the d features (dimensions), which are called principal components, that captures most of the mass of the data. As mentioned earlier, PCA is the subspace approximation problem with $p = 2$. However, typically the obtained principal components are linear combinations of all vectors which makes interpretability of the components more difficult. As such, *Sparse PCA* which is the optimization problem obtained from PCA by adding a sparsity constraint on the principal components have been defined which provides higher data interpretability [DP22, ZHT06, CJ95, HTW15, BDMI11].

Sparse PCA can be formulated as a constrained subspace approximation problem in which the set of projection matrices are constrained to those that can be written as $P = UU^T$ where U is a $d \times k$ orthonormal matrix such that the total number of non-zero entries in the U is at most s, for a given parameter s.

We give an algorithm that runs in time $d^{O(k^3/\varepsilon^2)}\left(dk^3/\varepsilon + d\log d\right)$ that computes a $\varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2$ additive approximate solution, which translates to a $(1+\varepsilon)$-multiplicative approximate solution to one formulation the problem (see Theorem 4.21 for the exact statement).

1.1.6 Column Subset Selection with Partition Constraint

Column subset selection (CSS) is a popular data summarization technique [BZMD14, CEM+15, ABF+16], where given a matrix $\boldsymbol{A}$, the goal is to find k columns in $\boldsymbol{A}$ that best approximates all columns of $\boldsymbol{A}$. Since in CSS, a subset of columns in the matrix are picked as the summary of the matrix $\boldsymbol{A}$, enforcing partition constraints naturally captures the problem of column subset selection with fair representation. More formally, in *column subset selection with partition constraints* (*PC-column subset selection*), given a partitioning of the columns of $\boldsymbol{A}$ into ℓ groups, $\boldsymbol{A}^{(1)}, \cdots, \boldsymbol{A}^{(\ell)}$, along with capacities $k_1, \cdots, k_\ell$, where $\sum_i k_i = k$, the set of valid subspaces are obtained by picking k_i vectors from $\boldsymbol{A}^{(i)}$, and projecting onto the span of these k columns of $\boldsymbol{A}$.

In Section 5, we show that PC-column subset selection is hard to approximate to any factor f in polynomial time, even if there are only two groups, or even when we allow for violating the capacity constraint by a factor of $O(\log n)$ (see Theorem 5.4 for the formal statement). This is in sharp contrast with the standard column subset selection problem for which efficient algorithms with tight guarantees are known.

2 Preliminaries

We will heavily use standard notations for vector and matrix quantities. For a matrix $\boldsymbol{M}$, we denote by $\boldsymbol{M}_{\cdot,i}$ the ith column of $\boldsymbol{M}$ and by $\boldsymbol{M}_{i,\cdot}$ the ith row. We denote by $\|\boldsymbol{M}\|_F$ the Frobenius norm, which is simply $\sqrt{\sum_{i,j} m_{ij}^2}$, where m_{ij} is the entry in the ith row and jth column of $\boldsymbol{M}$. We also use mixed norms, where $\|\boldsymbol{M}\|_{2,p} = \left(\sum_i \|\boldsymbol{M}_{\cdot,i}\|_2^p\right)^{1/p}$. I.e., it is the ℓ_p norm of the vector whose entries are the ℓ_2 norm of the columns of $\boldsymbol{M}$.

We also use $\sigma_{\min}(\boldsymbol{M})$ to denote the least singular value of a matrix, and $\sigma_{\max}(\boldsymbol{M})$ to denote the largest singular value. The value $\kappa(\boldsymbol{M})$ is used to denote the condition number, which is the ratio of the largest to the smallest singular value.

In analyzing the running times of our algorithms, we will use the following basic primitives, the running times of which we denote as T_0 and T_1 respectively. These are standard results from numerical linear algebra; while there are several improvements using randomization, these bounds will not be the dominant ones in our running time, so we do not optimize them.

Lemma 2.1 (SVD Computation; see [GL13])**.** *Given $\boldsymbol{A} \in \mathbb{R}^{d\times n}$, computing the reduced matrix $\boldsymbol{B}$ as in Lemma 3.5 takes time $T_0 := H \cdot \min\{O(nd^2), O(nd \cdot \frac{k}{\varepsilon})\}$, where H is the maximum bit complexity of any element of $\boldsymbol{A}$.*

Lemma 2.2 (Least Squares Regression; see [GL13])**.** *Given $\boldsymbol{A} \in \mathbb{R}^{d\times n}$ and given a target matrix $\boldsymbol{B}$ with r columns, the optimization problem $\min_{\boldsymbol{C}} \|\boldsymbol{B} - \boldsymbol{A}\boldsymbol{C}\|_F^2$ can be solved in time $T_1 := O(nrd^2 \cdot H)$, where H is the maximum bit length of any entry in A, B.*

Remark on the Exponential in k Running Times. In all of our results, it is natural to ask if the exponential dependence on k is necessary. We note that many of the problems we study are APX hard, and thus obtaining *multiplicative* $(1+\varepsilon)$ factors will necessarily require exponential time in the worst case. For problems that generalize ℓ_p-subspace approximation (e.g., the PC-ℓ_p-subspace approximation problem, Section 4.2), the works of [GRSW16] and [CW15] showed APX hardness. In these reductions, we in fact have the stronger property that the YES and NO instances differ in objective value by $\frac{1}{\mathrm{poly}(k)} \cdot \|\boldsymbol{A}\|_{2,p}^p$, where $\boldsymbol{A}$ is the matrix used in the reduction. Thus, assuming the Exponential Time Hypothesis, even the additive error guarantee in general requires an exponential dependence on either k or $1/\varepsilon$.

3 Framework for Constrained Subspace Approximation

Given a $d \times n$ matrix $\boldsymbol{A}$ and a special collection $\mathcal{S}$ of rank k projection matrices, we are interested in selecting the projection matrix $\boldsymbol{P} \in \mathcal{S}$ that minimizes the sum of projection costs (raised to the p^{th} power) of the columns of $\boldsymbol{A}$ onto $\boldsymbol{P}$. More compactly, the optimization problem is

$$\min_{\boldsymbol{P}\in\mathcal{S}} : \ \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p. \tag{CSA}$$

A more geometric and equivalent interpretation is that we have a collection of n data-points $\{a_1, a_2, \ldots, a_n\} \subseteq \mathbb{R}^d$ and we would like to approximate these data points by a subspace while satisfying certain constraints on the subspace:

$$\begin{aligned} \min : &\sum_{i=1}^{n} \|a_i - \widehat{a}_i\|_2^p \\ &\widehat{a}_i \in \mathrm{ColumnSpan}(\boldsymbol{P}) \\ &\boldsymbol{P} \in \mathcal{S}. \end{aligned} \tag{CSA-geo}$$

See Lemma 3.2 for a proof of the equivalence. We provide a unified framework to obtain approximately optimal solutions for various special collections of $\mathcal{S}$. In our framework, there are three steps to obtaining an approximate solution to any instance of CSA.

1. **Build a coreset:** Reduce the size of the problem by replacing $\boldsymbol{A}$ with a different matrix $\boldsymbol{B} \in \mathbb{R}^{d \times r}$ with fewer number of columns typically $\mathrm{poly}(k, 1/\varepsilon)$. The property we need to guarantee is that the projection cost is approximately preserved possibly with an additive error $c \geq 0$ independent of $\boldsymbol{P}$:

$$\|\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B}\|_{2,p}^p \in (1, 1+\varepsilon) \cdot \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p - c \quad \forall \boldsymbol{P} \text{ with rank at most } k. \tag{2}$$

   Such a $\boldsymbol{P}$ (for $p = 2$) has been referred to as a *Projection-Cost-Preserving Sketch with one sided error* in [CEM+15] . See Definition 3.3, Theorem 3.4, and Lemma 3.5 for results obtaining such a $\boldsymbol{B}$ for various $1 \leq p < \infty$. Lemma 3.7 shows that approximate solutions to reduced instances $(\boldsymbol{B}, \mathcal{S})$ satisfying Equation (2) are also approximate solutions to the original instance $(\boldsymbol{A}, \mathcal{S})$.

2. **Guess Coefficients:** Since the projection matrix $\boldsymbol{P}$ is of rank k, it can be represented as $\boldsymbol{U}\boldsymbol{U}^T$ such that $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k$. Using this, observe that the residual matrix

$$\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B} = \boldsymbol{B} - \boldsymbol{U}(\boldsymbol{U}^T\boldsymbol{B})$$

   can be represented as $\boldsymbol{B} - \boldsymbol{U}\boldsymbol{C}$ where $\boldsymbol{C} = \boldsymbol{U}^T\boldsymbol{B}$ is a $\mathbb{R}^{k \times r}$ matrix. The norm of the i^{th} column of $\boldsymbol{C}$ can be bounded by $\|b_i\|_2$ the norm of the i^{th} column of $\boldsymbol{B}$. This allows us to guess every column of $\boldsymbol{C}$ inside a k dimensional ball of radius at most the norm of the corresponding column in $\boldsymbol{B}$. Using a net with appropriate granularity, we guess the optimal $\boldsymbol{C}$ up to an additive error.

3. **Solve:** For every fixed $\boldsymbol{C}$ in the search space above, we solve the constrained regression problem

$$\min_{\boldsymbol{U} \in \mathbb{R}^{d \times k} : \boldsymbol{U}\boldsymbol{U}^T \in \mathcal{S}} \|\boldsymbol{B} - \boldsymbol{U}\boldsymbol{C}\|_{2,p}^p$$

   exactly. If $\widehat{\boldsymbol{C}}$ is the $\boldsymbol{C}$ matrix that induces the minimum cost, and $\widehat{\boldsymbol{U}}$ is the minimizer to the constrained regression problem, we return the projection matrix $\widehat{\boldsymbol{U}}\widehat{\boldsymbol{U}}^T$.

The following lemma formalizes the framework above and can be used as a black box application for several specific instances of CSA.

Lemma 3.1. *Given an instance $(\boldsymbol{A}, \mathcal{S})$ of CSA, for $1 \leq p < \infty$,*

1. *Let T_s be the time taken to obtain a smaller instance $(\boldsymbol{B}, \mathcal{S})$ such that the approximate cost property in Equation (2) is satisfied and the number of columns in $\boldsymbol{B}$ is r.*

2. *Let T_r be the time taken to solve the constrained regression problem for any fixed $\boldsymbol{B} \in \mathbb{R}^{d \times r}$ and $\boldsymbol{C} \in \mathbb{R}^{k \times r}$*

$$\min_{\boldsymbol{U} \in \mathbb{R}^{d \times k} : \boldsymbol{U}\boldsymbol{U}^T \in \mathcal{S}} \|\boldsymbol{U}\boldsymbol{C} - \boldsymbol{B}\|_{2,p}^p. \tag{3}$$

   Then for any granularity parameter $0 < \delta < 1$, we obtain a solution $\boldsymbol{P} \in \mathcal{S}$ such that

$$\|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p \leq (1+\varepsilon) OPT + \Delta \tag{4}$$

   in time $T_s + T_r \cdot O((1/\delta)^{kr})$.

Here, $\Delta = (1+\varepsilon)\|\boldsymbol{A}\|_{2,p}^p \cdot ((1+\delta)^p - 1)$ and $OPT = \min_{\boldsymbol{P}' \in \mathcal{S}} \|\boldsymbol{A} - \boldsymbol{P}'\boldsymbol{A}\|_{2,p}^p$.

Proof. Let the optimal solution to the instance $(\boldsymbol{A}, \mathcal{S})$ be $\boldsymbol{P}^* = \boldsymbol{U}^*\boldsymbol{U}^{*T}$ and let $\boldsymbol{C}^* = \boldsymbol{U}^{*T}\boldsymbol{B}$. Since the columns of $\boldsymbol{U}^*$ are unit vectors, the norm of the i^{th} column of $\boldsymbol{C}^*$ is at most $\|b_i\|_2$ the norm of the i^{th} column of $\boldsymbol{B}$. We will try to approximately guess the columns of $\boldsymbol{C}^*$ using epsilon nets. For each i, we search for the i^{th} column of $\boldsymbol{C}$ using a $(\|b_i\|_2 \cdot \delta)$-net inside a k dimensional ball of radius $\|b_i\|_2$ centered at origin. The size of the net for each column of $\boldsymbol{C}$ is $O((1/\delta)^k)$ and hence the total search space over matrices $\boldsymbol{C}$ has $O((1/\delta)^{kr})$ possibilities.

For each $\boldsymbol{C}$, we solve the constrained regression problem in Equation (3). Let $\widehat{\boldsymbol{C}}$ be the matrix for which the cost is minimized and $\widehat{\boldsymbol{U}}$ be the corresponding minimizer to the constrained regression problem respectively. Consider the solution $\widehat{\boldsymbol{P}} = \widehat{\boldsymbol{U}}\widehat{\boldsymbol{U}}^T$. The cost of this solution on reduced instance $(\boldsymbol{B}, \mathcal{S})$ is

$$\|\boldsymbol{B} - \widehat{\boldsymbol{U}}\widehat{\boldsymbol{U}}^T\boldsymbol{B}\|_{2,p}^p \le \|\boldsymbol{B} - \widehat{\boldsymbol{U}}\widehat{\boldsymbol{C}}\|_{2,p}^p. \tag{5}$$

Let $\overline{\boldsymbol{C}}$ be the matrix in the search space such that $\|\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i}\|_2 \le \|b_i\|_2 \cdot \delta$ for every $i \in [r]$. Using the cost minimality of $\widehat{\boldsymbol{C}}$, we can imply that the above cost is

$$\begin{aligned} &\le \min_{\boldsymbol{U}\in\mathbb{R}^{d\times k}:\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S}} \|\boldsymbol{B} - \boldsymbol{U}\overline{\boldsymbol{C}}\|_{2,p}^p &(6)\\ &\le \|\boldsymbol{B} - \boldsymbol{U}^*\overline{\boldsymbol{C}}\|_{2,p}^p. &(7)\end{aligned}$$

It remains to upper bound the difference $\Delta = \|\boldsymbol{B} - \boldsymbol{U}^*\overline{\boldsymbol{C}}\|_{2,p}^p - \|\boldsymbol{B} - \boldsymbol{U}^*\boldsymbol{C}^*\|_{2,p}^p$. If we let $b_i^* := (\boldsymbol{U}^*\boldsymbol{C}^*)_{.,i}$ and $\overline{b}_i := (\boldsymbol{U}^*\overline{\boldsymbol{C}})_{.,i}$ for $i \in [r]$, then

$$\Delta = \sum_{i=1}^{r} \left(\|b_i - \overline{b}_i\|_2^p - \|b_i - b_i^*\|_2^p\right). \tag{8}$$

Using the fact that $\|\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i}\|_2 \le \|b_i\|_2 \cdot \delta$, we know that

$$\|\overline{b}_i - b_i^*\|_2 = \|\boldsymbol{U}^*(\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i})\|_2 \le \|\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i}\|_2 \le \|b_i\|_2 \cdot \delta. \tag{9}$$

This implies that each error term

$$\begin{aligned} \Delta_i &:= \|b_i - \overline{b}_i\|_2^p - \|b_i - b_i^*\|_2^p &(10)\\ &\le (\|b_i - b_i^*\|_2 + \|b_i^* - \overline{b}_i\|_2)^p - \|b_i - b_i^*\|_2^p &\text{(Triangle inequality)}\\ &\le (\|b_i - b_i^*\|_2 + \|b_i\| \cdot \delta)^p - \|b_i - b_i^*\|_2^p &(\|\overline{b}_i - b_i^*\|_2 \le \|b_i\|_2 \cdot \delta)\\ &\le \|b_i\|_2^p \cdot ((1+\delta)^p - 1). &((x+\delta)^p - x^p \text{ is increasing in } [0,1],\ \|b_i - b_i^*\|_2 \le \|b_i\|_2)\end{aligned}$$

Summing up, the total error Δ is at most $\|\boldsymbol{B}\|_{2,p}^p \cdot ((1+\delta)^p - 1) = O(\delta p) \cdot \|\boldsymbol{B}\|_{2,p}^p$ for $\delta \le 1/p$. This implies that

$$\|\boldsymbol{B} - \widehat{\boldsymbol{P}}\boldsymbol{B}\|_{2,p}^p \le \|\boldsymbol{B} - \boldsymbol{P}^*\boldsymbol{B}\|_{2,p}^p + \|\boldsymbol{B}\|_{2,p}^p \cdot ((1+\delta)^p - 1) \tag{11}$$

Using the property of $\boldsymbol{B}$ from Equation (2), we can imply that

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p \le (1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\| + \|\boldsymbol{B}\|_{2,p}^p \cdot ((1+\delta)^p - 1). \tag{12}$$

setting $\boldsymbol{P} = 0$ in Equation (2) and using the fact that $c \ge 0$ gives $\|\boldsymbol{B}\|_{2,p}^p \le (1+\varepsilon)\|\boldsymbol{A}\|_{2,p}^p$. Plugging this in the equation above gives

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p \le (1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\| + (1+\varepsilon)\|\boldsymbol{A}\|_{2,p}^p \cdot ((1+\delta)^p - 1) \tag{13}$$

The total time taken by the algorithm is $T_s + T_r \cdot O((1/\delta)^{kr})$. $\square$

Lemma 3.2. *The mathematical programs CSA and CSA-geo equivalent to the following "constrained factorization" problem:*

$$\min_{\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S},\, \boldsymbol{H}\in\mathbb{R}^{d\times n}} \|\boldsymbol{A}-\boldsymbol{U}\boldsymbol{H}\|_{2,p}^p. \tag{CSA-fac}$$

Proof. First, we will prove the equivalence between CSA and CSA-fac.

1. The easier direction to see is $\min_{\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S},\, \boldsymbol{H}\in\mathbb{R}^{d\times n}} \|\boldsymbol{A}-\boldsymbol{U}\boldsymbol{H}\|_{2,p}^p \le \min_{\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S}} \|\boldsymbol{A}-\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_{2,p}^p$ because setting $\boldsymbol{H}=\boldsymbol{U}^T\boldsymbol{A}$ in CSA-fac gives CSA.

2. For the other direction, it suffices to show that for any fixed choice of $\boldsymbol{U}$ such that $\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S}$, an optimal choice of $\boldsymbol{H}$ is $\boldsymbol{U}^T\boldsymbol{A}$. In order to see this, observe that the problem

$$\min_{\boldsymbol{H}} \|\boldsymbol{A}-\boldsymbol{U}\boldsymbol{H}\|_{2,p}^p = \min_{\boldsymbol{H}} \sum_{i=1}^n \|a_i-\boldsymbol{U}h_i\|_2^p \tag{14}$$

where a_i and h_i are the i^{th} columns of $\boldsymbol{A}$ and $\boldsymbol{H}$ respectively. Since the cost function decomposes into separate problems for each column, we can push the minimization inside.

$$= \sum_{i=1}^n \left(\min_{h_i} \|a_i-\boldsymbol{U}h_i\|_2\right)^p. \tag{15}$$

Using normal equation, the optimal choice for h_i satisfies $\boldsymbol{U}^T\boldsymbol{U}h_i=\boldsymbol{U}^Ta_i$. Since the columns of $\boldsymbol{U}$ are orthonormal, this implies that $h_i=\boldsymbol{U}^Ta_i$ for each $i\in[n]$ and hence $\boldsymbol{H}=\boldsymbol{U}^T\boldsymbol{A}$.

Now we show the equivalence between CSA-fac and CSA-geo. Observe that CSA-geo can be re-written as

$$\begin{aligned} &\min \sum_{i=1}^n \|a_i-\widehat{a}_i\|_2^p \\ &\widehat{a}_i \in \mathrm{ColumnSpan}(\boldsymbol{U}) \\ &\boldsymbol{U}\boldsymbol{U}^T\in\mathcal{S}. \end{aligned}$$

Because the column span of $\boldsymbol{P}=\boldsymbol{U}\boldsymbol{U}^T$ is identical to the column span of $\boldsymbol{U}$. Replacing $\widehat{a}_i\in \mathrm{ColumnSpan}(\boldsymbol{U})$ by $\widehat{a}_i=\boldsymbol{U}h_i$ gives CSA-fac. □

Definition 3.3 (Strong coresets; as defined in [WY24])**.** Let $1\le p<\infty$ and $0<\varepsilon<1$. Let $\boldsymbol{A}\in\mathbb{R}^{d\times n}$. Then, a diagonal matrix $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ is a $(1\pm\varepsilon)$ strong coreset for ℓ_p subspace approximation if for all rank k projection matrices $\boldsymbol{P}_F$, we have

$$\|(\boldsymbol{I}-\boldsymbol{P}_F)\boldsymbol{A}\boldsymbol{S}\|_{2,p}^p \in (1\pm\varepsilon)\|(\boldsymbol{I}-\boldsymbol{P}_F)\boldsymbol{A}\|_{2,p}^p. \tag{16}$$

The number of non-zero entries $\mathrm{nnz}(\boldsymbol{S})$ of $\boldsymbol{S}$ will be referred to as the size of the coreset.

Theorem 3.4 (Theorems 1.3 and 1.4 of [WY25])**.** *Let $p\in[1,2)\cup(2,\infty)$ and $\varepsilon>0$ be given, and let $\boldsymbol{A}\in\mathbb{R}^{d\times n}$. There is an algorithm running in $\widetilde{O}(\mathrm{nnz}(\boldsymbol{A})+d^\omega)$ time which, with probability at least $1-\delta$, constructs a strong coreset $\boldsymbol{S}$ that satisfies definition 3.3 and has size:*

$$\mathrm{nnz}(\boldsymbol{S}) = \begin{cases} \frac{k}{\varepsilon^{4/p}}(\log(k/\varepsilon\delta))^{O(1)} & \text{if } p\in[1,2), \\ \frac{k^{p/2}}{\varepsilon^p}(\log(k/\varepsilon\delta))^{O(p^2)} & \text{if } p\in(2,\infty). \end{cases} \tag{17}$$

Remark. *Note that for any* $\boldsymbol{S}$ *that satisfies the property in Definition 3.3, we can scale it up to satisfy* $\|(\boldsymbol{I}-\boldsymbol{P}_F)\boldsymbol{A}\boldsymbol{S}\|_{p,2}^p \in (1, 1+\varepsilon)\|(\boldsymbol{I}-\boldsymbol{P}_F)\boldsymbol{A}\|_{p,2}^p$ *matching the condition in Equation* (2).

For many of the applications, we have $p = 2$. For this case, the choice of the reduced matrix $\boldsymbol{B}$ that replaces $\boldsymbol{A}$ is simply the matrix of scaled left singular vectors of $\boldsymbol{A}$. More formally,

Lemma 3.5. *When* $p = 2$, *if* $\boldsymbol{A} = \sum_{i=1}^{n} \sigma_i p_i q_i^T$ *be the singular value decomposition of* $\boldsymbol{A}$ *(where* σ_i *is the* i^{th} *largest singular value and* $p_i \in \mathbb{R}^d, q_i \in \mathbb{R}^n$ *are the left singular vector and right singular vector corresponding to* σ_i*), then* $\boldsymbol{B} = \sum_{i=1}^{r} \sigma_i p_i q_i^T$ *satisfies Equation* (2) *for* $r = k + k/\varepsilon$.

Proof. For any two arbitrary projection matrices $\boldsymbol{P}$ and $\boldsymbol{P}'$ of rank $\leq k$, consider the difference

$$\begin{aligned}
&\left(\|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - \|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2\right) - \left(\|\boldsymbol{A}-\boldsymbol{P}'\boldsymbol{A}\|_F^2 - \|\boldsymbol{B}-\boldsymbol{P}'\boldsymbol{B}\|_F^2\right) && (18)\\
&= \langle \boldsymbol{A}\boldsymbol{A}^T, \boldsymbol{I}-\boldsymbol{P}\rangle - \langle \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{I}-\boldsymbol{P}\rangle - \langle \boldsymbol{A}\boldsymbol{A}^T, \boldsymbol{I}-\boldsymbol{P}'\rangle + \langle \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{I}-\boldsymbol{P}'\rangle && (19)\\
&= \langle \boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{P}'\rangle - \langle \boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{P}\rangle && (20)\\
&\leq \langle \boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{P}'\rangle && (\boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T \succeq 0,\ \boldsymbol{P} \succeq 0)\\
&\leq \sum_{i=r+1}^{r+k} \sigma_i && (\text{rank of } \boldsymbol{P}' \leq k)\\
&\leq k \cdot \sigma_r && (\sigma_r \geq \sigma_{r'},\ r' \geq r)\\
&\leq \frac{k}{r-k} \cdot \left(\sum_{i=k+1}^{r} \sigma_i\right) && (\sigma_r \leq \sigma_{r'},\ r' \leq r)\\
&\leq \frac{k}{r-k}\|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 = \varepsilon\|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2. && \left(\|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 = \sum_{i=k+1}^{d}\sigma_i\right)
\end{aligned}$$

If we let $c := \max_{\text{rank}(\boldsymbol{P})\leq k}\left(\|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - \|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2\right)$, then we have

$$c - \varepsilon\|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 \leq \|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - \|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2 \leq c$$

for any projection matrix $\boldsymbol{P}$ of rank at most k. This can we re written as

$$\|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2 \in (0, \varepsilon) \cdot \|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 + \|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - c. \tag{21}$$

Using the fact that $\|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 \leq \|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2$, we get

$$\|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2 \in (1, 1+\varepsilon) \cdot \|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - c.$$

The fact that $c \geq 0$ follows from the fact that

$$\begin{aligned}
\|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - \|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2 &= \langle \boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T, \boldsymbol{I}-\boldsymbol{P}\rangle && (22)\\
&\geq 0. && (\boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{B}\boldsymbol{B}^T \succeq 0,\ \boldsymbol{I}-\boldsymbol{P} \succeq 0)
\end{aligned}$$

□

Remark 3.6. *Notice that when* $p = 2$, *Lemma 3.5 proves the condition in Equation* (21)*:*

$$\|\boldsymbol{B}-\boldsymbol{P}\boldsymbol{B}\|_F^2 \in (0, \varepsilon) \cdot \|\boldsymbol{A}-\boldsymbol{A}_k\|_F^2 + \|\boldsymbol{A}-\boldsymbol{P}\boldsymbol{A}\|_F^2 - c$$

which is stronger than the condition in Equation (2).

Lemma 3.7. *If $(\boldsymbol{A}, \mathcal{S})$ is an instance of CSA and $\boldsymbol{B} \in \mathbb{R}^{d \times r}$ is a matrix that satisfies Equation* (2)*, and*

$$\widehat{\boldsymbol{P}} := \underset{\boldsymbol{P} \in \mathcal{S}}{\arg\min} \|\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B}\|_{2,p}^p, \quad \boldsymbol{P}^* := \underset{\boldsymbol{P} \in \mathcal{S}}{\arg\min} \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_{2,p}^p, \tag{23}$$

then $\widehat{\boldsymbol{P}}$ is an $(1+\varepsilon)$-approximate solution to the instance $(\boldsymbol{A}, \mathcal{S})$ i.e.,

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p \leq (1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_{2,p}^p. \tag{24}$$

1. *More generally, if $\widehat{\boldsymbol{P}}$ is an approximate solution to $(\boldsymbol{B}, \mathcal{S})$ such that*

   $$\|\boldsymbol{B} - \widehat{\boldsymbol{P}}\boldsymbol{B}\|_{2,p}^p \leq \alpha\|\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B}\|_{2,p}^p + \beta \quad \forall \boldsymbol{P} \in \mathcal{S},$$

   for some $\alpha \geq 1$, $\beta \geq 0$, then we have

   $$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p \leq \alpha(1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_{2,p}^p + \beta.$$

2. *For the specific case when $p = 2$, if $\widehat{\boldsymbol{P}}$ is an exact solution to $(\boldsymbol{B}, \mathcal{S})$, then we have*

   $$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_F^2 \leq \|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_F^2 + \varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2.$$

Proof. 1. Using the approximate optimality of $\widehat{\boldsymbol{P}}$ for the instance $(\boldsymbol{B}, \mathcal{S})$, we have

$$\|\boldsymbol{B} - \widehat{\boldsymbol{P}}\boldsymbol{B}\|_{2,p}^p \leq \alpha\|\boldsymbol{B} - \boldsymbol{P}^*\boldsymbol{B}\|_{2,p}^p + \beta. \tag{25}$$

Using the lower-bound and upper-bound from Equation (2) for the LHS and RHS, we get

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p - c \leq \alpha(1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_{2,p}^p - \alpha c + \beta. \tag{26}$$

Since $\alpha \geq 1$ and $c \geq 0$, we get

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_{2,p}^p \leq \alpha(1+\varepsilon)\|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_{2,p}^p + \beta. \tag{27}$$

2. Using the optimality of $\widehat{\boldsymbol{P}}$ for the instance $(\boldsymbol{B}, \mathcal{S})$ for with $p = 2$, we have

$$\|\boldsymbol{B} - \widehat{\boldsymbol{P}}\boldsymbol{B}\|_F^2 \leq \|\boldsymbol{B} - \boldsymbol{P}^*\boldsymbol{B}\|_F^2. \tag{28}$$

Using Remark 3.6, we know that $\|\boldsymbol{B} - \boldsymbol{P}\boldsymbol{B}\|_F^2 \in (0, \varepsilon) \cdot \|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2 + \|\boldsymbol{A} - \boldsymbol{P}\boldsymbol{A}\|_F^2 - c$ for any rank k projection matrix $\boldsymbol{P}$ for some $c \geq 0$ independent of $\boldsymbol{P}$ (see Equation (21)). Using this, we get

$$\|\boldsymbol{A} - \widehat{\boldsymbol{P}}\boldsymbol{A}\|_F^2 - c \leq \|\boldsymbol{B} - \widehat{\boldsymbol{P}}\boldsymbol{B}\|_F^2 \leq \|\boldsymbol{B} - \boldsymbol{P}^*\boldsymbol{B}\|_F^2 \leq \|\boldsymbol{A} - \boldsymbol{P}^*\boldsymbol{A}\|_F^2 + \varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2 - c.$$

Canceling out the $-c$ gives the inequality we claimed.

$\square$

Lemma 3.8 (Lemma 4.1 in [CW09])**.** *If $n \times d$ matrix $\boldsymbol{A}$ has integer entries bounded in magnitude by γ, and has rank $\rho \geq k$, then the k^{th} singular value σ_k of $\boldsymbol{A}$ has $|\log \sigma_k| = O(\log(nd\gamma))$ as $nd \to \infty$. This implies that $\|\boldsymbol{A}\|_F / \Delta_k \leq (nd\gamma)^{O(k/(\rho-k))}$ as $nd \to \infty$. Here $\Delta_k := \|\boldsymbol{A} - \boldsymbol{A}_k\|_F$*

4 Applications

We present several applications to illustrate our framework.

4.1 Constrained Subspace Estimation [SVK$^+$17]

In constrained subspace estimation, we are given a collection of target subspaces $T_1, T_2, \ldots, T_m$ and a model subspace W. The goal is to find a subspace V of dimension k such that $\dim(V \cap W) \geq \ell$ that maximizes the average overlap between the subspace V and $T_1, \ldots, T_m$. More formally, the problem can be formulated as mathematical program:

$$\max : \langle \overline{\boldsymbol{P}}_T, \boldsymbol{P}_V \rangle \tag{CSE-max}$$

$$\dim(V) = k,\ \dim(V \cap W) \geq \ell, \tag{29}$$

$$\overline{\boldsymbol{P}}_T = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{P}_{T_i}, \tag{30}$$

$$\boldsymbol{P}_{T_i} \text{ and } \boldsymbol{P}_V \text{ are the projection matrices onto the subspaces } T_i \text{ and } V \text{ respectively.} \tag{31}$$

Let us assume that the constraint $\dim(V \cap W) \geq \ell$ is actually an exact constraint $\dim(V \cap W) = \ell$ because we can solve for $k - \ell + 1$ different cases $\dim(V \cap W) = i$ for each $\ell \leq i \leq k$. Since $\overline{\boldsymbol{P}}_T$ is a PSD matrix, let it be $\boldsymbol{A}\boldsymbol{A}^T$ for some $\boldsymbol{A} \in \mathbb{R}^{d \times d}$. Changing the optimization problem from a maximization problem to a minimization problem, we get

$$\min : \langle \boldsymbol{A}\boldsymbol{A}^T, \boldsymbol{I} - \boldsymbol{P}_V \rangle = \|\boldsymbol{A} - \boldsymbol{P}_V \boldsymbol{A}\|_F^2 \tag{CSE-min}$$

$$\boldsymbol{P}_V \text{ is the projection matrix onto } V \tag{32}$$

$$\dim(V) = k,\ \dim(V \cap W) = \ell. \tag{33}$$

Lemma 4.1. *The CSE-min problem is a special case of CSA.*

Proof. Setting $p = 2$ and $\mathcal{S}$ as the set of k dimensional projection matrices P_V such that $\dim(V \cap W) = \ell$ in CSA gives CSE-min. □

Let $\boldsymbol{B} \in \mathbb{R}^{d \times r},\ r = k + k/\varepsilon$ be the reduced matrix obtained as in Lemma 3.5. Using Lemma 3.7, it is sufficient to focus on the reduced instance with $\boldsymbol{A}$ replaced instead of $\boldsymbol{B}$.

Any subspace V such that $\dim(V) = k,\ \dim(V \cap W) = \ell$ can be represented equivalently as

$$V = \mathrm{Span}(u_1, u_2, \ldots, u_\ell, v_1, v_2, \ldots, v_{k-\ell})$$
$$u_i \in W,\ v_j \in W^\perp \quad \forall i \in [\ell],\ j \in [k - \ell].$$

Using these observations and Lemma 3.2, we can focus on the following subspace estimation program

$$\min : \|\boldsymbol{B} - \boldsymbol{U}\boldsymbol{C}\|_F^2 \tag{34}$$

$$\boldsymbol{U} \text{ is a orthogonal basis for } \mathrm{Span}(u_1, \ldots, u_\ell, v_1, \ldots, v_{k-\ell}) \tag{35}$$

$$u_i \in W,\ v_j \in W^\perp \quad \forall i \in [\ell],\ j \in [k - \ell]. \tag{36}$$

Since $\boldsymbol{C}$ is unconstrained, we can replace the condition in Equation (35) with the much simpler condition $\boldsymbol{U} = [u_1, \ldots, u_\ell, v_1, \ldots, v_\ell]$. This gives

$$\min : \|\boldsymbol{B} - \boldsymbol{U}\boldsymbol{C}\|_F^2 \tag{CSE-min-reduced}$$

$$\boldsymbol{U} = [u_1, \ldots, u_\ell, v_1, \ldots, v_\ell] \tag{37}$$

$$u_i \in W,\ v_j \in W^\perp \quad \forall i \in [\ell],\ j \in [k - \ell]. \tag{38}$$

Lemma 4.2. *For any fixed $\boldsymbol{B} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{C} \in \mathbb{R}^{k\times r}$, the Equation* (CSE-min-reduced) *can be solved exactly in* $\text{poly}(n)$ *time.*

Proof. For fixed $\boldsymbol{B}$ and $\boldsymbol{C}$, the objective is convex quadratic in $\boldsymbol{U}$ and the constraints are linear on $\boldsymbol{U}$. Linear constrained convex quadratic program can be efficiently solved. □

Corollary 4.3 (Additive approximation for CSE)**.** *Using Lemma 3.1, we can get a subspace V such that $\dim(V) = k,\ \dim(V \cap W) = \ell$ and*

$$\|\boldsymbol{A} - \boldsymbol{P}_V \boldsymbol{A}\|_F^2 \leq (1+\varepsilon)OPT + O(\delta\|\boldsymbol{A}\|_F^2)$$

for any choice of $0 < \delta < 1$ in time $\text{poly}(n) \cdot (1/\delta)^{O(k^2/\varepsilon)}$.

Lemma 3.8 gives a lower bound for OPT when the entries of the input matrix $\boldsymbol{A}$ are integers bounded in magnitude by γ.

Theorem 4.4 (Multiplicative approximation for CSE)**.** *Given an instance $(\boldsymbol{A} \in \mathbb{R}^{d\times n}, k, W)$ of constrained subspace estimation with integer entries of absolute value at most γ in $\boldsymbol{A}$, there is an algorithm that obtains a subspace V such that $\dim(V) = k,\ \dim(V \cap W) = \ell$ and*

$$\|\boldsymbol{A} - \boldsymbol{P}_V \boldsymbol{A}\|_F^2 \leq (1+\varepsilon)OPT$$

in $O(nd\gamma/\varepsilon)^{O(k^3/\varepsilon)}$ time.

Proof. Using Lemma 3.8, we know that $\|\boldsymbol{A}\|_F^2/\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2 \leq (nd\gamma)^{O(k)}$. Setting $\delta = \varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2/\|\boldsymbol{A}\|_F^2 \geq \varepsilon(nd\gamma)^{-O(k)}$ in Corollary 4.3 gives the desired time complexity. □

4.2 Partition Constrained ℓ_p-Subspace Approximation

We now consider the PC-ℓ_p-subspace approximation problem, which generalizes the subspace approximation and subspace estimation problems.

Definition 4.5 (Partition Constrained ℓ_p-Subspace Approximation)**.** In the PC-ℓ_p-subspace approximation problem, we are given a set of target vectors $\{a_1, a_2, \ldots, a_n\} \subseteq \mathbb{R}^d$ as columns of a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times n}$, a set of ℓ subspaces $S_1, \ldots, S_\ell \subseteq \mathbb{R}^d$, and a sequence of capacity constraints $k_1, \cdots, k_\ell$ where $k_1 + \cdots + k_\ell = k$. The goal is to select k vectors in total, k_i from subspace S_i, such that their span captures as much of $\boldsymbol{A}$ as possible. Formally, the goal is to select vectors $\{v_{i,t_i}\}_{i\leq \ell, t_i \leq k_i}$, such that for every $i \leq \ell$, $v_{i,1}, \ldots, v_{i,k_i} \in S_i$, so as to minimize $\sum_{i\in[n]} \|\text{proj}^{\perp}_{\text{span}(\{v_{i,t_i}\}_{i\leq\ell,t_i\leq k_i})}(a_i)\|_2^p$.

Our results will give algorithms with running times exponential in $\text{poly}(k)$ for PC-ℓ-subspace approximation. Given this goal, we can focus on the setting where $k_i = 1$, since we can replace each S_i in the original formulation with k_i copies of S_i, with a budget of 1 for each copy.

PC-ℓ-subspace approximation with Unit Capacity. Given a set of vectors $\{a_1, a_2, \ldots, a_n\} \subseteq \mathbb{R}^d$ as columns of a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times n}$ and subspaces $S_1, \ldots, S_k \subseteq \mathbb{R}^d$, select a vector $v_i \in S_i$ for $i \in [k]$ in order to minimize $\sum_{i\in[n]} \|\text{proj}^{\perp}_{\text{span}(v_1,\ldots,v_k)}(a_i)\|_2^p$, where $p \geq 1$ is a given parameter. A more compact formulation is

$$\min : \sum_{i=1}^{n} \|a_i - \widehat{a}_i\|_2^p \qquad \text{(PC-}\ell_p\text{-SA-geo)}$$

$$\widehat{a}_i \in \text{Span}(v_1, \ldots, v_k) \quad \forall i \in [n] \qquad (39)$$

$$v_j \in S_j \quad \forall j \in [k]. \qquad (40)$$

Using Lemma 3.2, the two other equivalent formulations are

$$\begin{aligned}
\min : \; & \|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_{2,p}^p && \text{(PC-}\ell_p\text{-SA)}\\
& \boldsymbol{U} \text{ is an orthogonal basis for } \mathrm{Span}(v_1, v_2, \ldots, v_k) && (41)\\
& v_i \in S_i \quad \forall i \in [k]. && (42)\\
\min : \; & \|\boldsymbol{A} - \boldsymbol{V}\boldsymbol{C}\|_{2,p}^p && \text{(PC-}\ell_p\text{-SA-fac)}\\
& \boldsymbol{V} = [v_1, \ldots, v_k] && (43)\\
& v_i \in S_i \quad \forall i \in [k]. && (44)
\end{aligned}$$

In what follows, we thus focus on the unit capacity version. We can use our general framework to derive an additive error approximation, for any p.

4.2.1 Additive Error Approximation

Theorem 4.6. *There exists an algorithm for PC-ℓ_p-subspace approximation with runtime $(\kappa/\varepsilon)^{\mathrm{poly}(k/\varepsilon)} \cdot \mathrm{poly}(n)$ which returns a solution with additive error at most $O(\varepsilon p) \cdot \|A\|_{p,2}^p$, where κ is the condition number of an optimal solution $\boldsymbol{V}^* = [v_1^*, v_2^*, \ldots, v_k^*]$ for the PC-subspace approximation problem PC-ℓ_p-SA-fac.*

The algorithm we present below assumes a given bound on κ, the condition number. In practice, we can search for it via doubling and stop when a sufficiently small approximation error is reached or a certain time complexity is reached. We also note that it may happen that the *optimal* solution uses a large κ, but there is an approximately optimal solution with small κ. In this case, our result can be applied with the smaller κ, and it gives a guarantee relative to the latter solution.

Proof. As a first step, we will find an additive approximation to the smaller instance obtained by replacing the $\boldsymbol{A}$ matrix with the smaller $\boldsymbol{B}$ matrix as in Lemma 3.5. Our proof mimics the argument from Lemma 3.1, but we need a slight change in the analysis because $\{v_j\}$ are not orthogonal. Note that we can assume without loss of generality that the columns of $\boldsymbol{V}^*$ are unit vectors, $\sigma_{\max}(\boldsymbol{V}^*) \geq 1$ and $\sigma_{\min}(\boldsymbol{V}^*) \leq 1$, and thus $\kappa \geq 1$. Given a bound on κ, the algorithm is simply the following: we first create a δ-net for the Ball of radius κ in $\mathbb{R}^k$, with $\delta = \varepsilon/\kappa$, and for each i, we form a guess for the coefficient vector $\boldsymbol{C}_{.,i}$ as $\|b_i\|_2 \cdot u$, where u is a vector from the net and b_i is the i^{th} column of $\boldsymbol{B}$. For each guess $\widehat{\boldsymbol{C}}$, we solve for $\boldsymbol{V}$ that minimizes $\left\|\boldsymbol{B} - \boldsymbol{V}\widehat{\boldsymbol{C}}\right\|_{2,p}$ subject to $v_j \in S_j$. Note that we can drop the unit vector constraints at this point; this makes the above optimization problem convex (specifically, it is the well-studied problem of ℓ_p regression [AKPS24]), which can be solved in polynomial time.

To bound the error, we first note that the optimum coefficients $\boldsymbol{C}^*$ satisfy the condition that for each i,

$$\left\|\boldsymbol{C}^*_{.,i}\right\| \leq \frac{\|b_i\|_2}{\sigma_{\min}(\boldsymbol{V}^*)} \leq \|b_i\|_2 \cdot \kappa.$$

Now suppose we focus on one target vector b_i. By choice, in one of our guessed solutions, say $\overline{\boldsymbol{C}}$, we will have $\left\|\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i}\right\| \leq \|b_i\|_2 \cdot \delta$. Thus, we have

$$\left\|\bar{b}_i - b_i^*\right\|_2 \leq \left\|\boldsymbol{V}^*(\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i})\right\|_2 \leq \sigma_{\max}(\boldsymbol{V}^*) \left\|\overline{\boldsymbol{C}}_{.,i} - \boldsymbol{C}^*_{.,i}\right\|_2 \leq \kappa \cdot \|b_i\|_2 \cdot \delta.$$

Thus, analyzing the error Δ_i as in the proof of Lemma 3.1, we obtain

$$\Delta_i \leq \|b_i\|_2^p \cdot \left((1 + \varepsilon/p)^p - 1\right).$$

This yields the desired additive guarantee to the reduced instance. Using the coreset property from Equation (2), we know that the cost of the solution we find is at most $(1+\varepsilon)\mathrm{OPT}+O(\varepsilon p)\cdot\|\boldsymbol{A}\|_{2,p}^{p}$. Using the fact that $\mathrm{OPT}\le\|\boldsymbol{A}\|_{2,p}^{p}$ completes the proof. □

4.2.2 Multiplicative Approximation Using Polynomial System Solving

For the special case of $p=2$, it turns out that we can obtain a $(1+\varepsilon)$-multiplicative approximation, using a novel idea.

As described in our framework, we start by constructing the reduced instance $\boldsymbol{B},\mathcal{S}$, where $\boldsymbol{B}=\{b_1,b_2,\dots,b_r\}\subset\mathbb{R}^d$ is a set of target vectors and $\mathcal{S}=\{S_1,S_2,\dots,S_k\}$ is the given collection of subspaces of $\mathbb{R}^d$. We define $\boldsymbol{P}_j$ to be some fixed orthonormal basis for the space S_j. Recall that any solution to PC-ℓ_2-subspace approximation is defined by (a) the vector x_j that expresses the chosen v_j as $v_j=\boldsymbol{P}_j x_j$ (we have one x_j for each $j\in[k]$), and (b) a set of combination coefficients c_{ij} used to represent the vectors b_i using the vectors $\{v_j\}_{j=1}^{k}$. We collect the vectors x_j into one long vector $\boldsymbol{x}$ and the coefficients c_{ij} into a matrix $\boldsymbol{C}$.

Theorem 4.7. *Let $\boldsymbol{B},\mathcal{S}$ be an instance of PC-ℓ_2-subspace approximation, where $\boldsymbol{B}=\{b_1,b_2,\dots b_r\}$, and suppose that the bit complexity of each element in the input is bounded by H. Suppose there exists an (approximately) optimal solution is defined by the pair $(\boldsymbol{x}^*,\boldsymbol{C}^*)$ with bit complexity poly(n,H). There exists an algorithm that runs in time $n^{O(k^2/\varepsilon)}\cdot\mathrm{poly}(H)$ and outputs a solution whose objective value is within a $(1+\varepsilon)$ factor of the optimum objective value. We denote $s=\sum_{j=1}^{k}s_j$ and $s_j=\dim(S_j)$; n for this result can be set to $\max(s,d,k/\varepsilon)$.*

Algorithm Overview. Recall that $\boldsymbol{P}_j$ specifies an orthonormal basis for S_j. Let $\boldsymbol{P}_{ij}:=c_{ij}\boldsymbol{P}_j$, where c_{ij} are variables. Define $\boldsymbol{P}$ to be the $\mathbb{R}^{rd\times s}$ matrix consisting of $r\times k$ blocks; the $(i,j)^{\text{th}}$ block is $\boldsymbol{P}_{ij}$ and we let $\boldsymbol{x},\boldsymbol{b}$ be the vectors representing all the x_j,b_i stacked vertically respectively as shown below:

$$\boldsymbol{P}=\left[\begin{array}{c|c|c|c}\boldsymbol{P}_{1,1}&\boldsymbol{P}_{1,2}&\cdots&\boldsymbol{P}_{1,k}\\\hline\boldsymbol{P}_{2,1}&\boldsymbol{P}_{2,2}&\cdots&\boldsymbol{P}_{2,k}\\\hline\vdots&\vdots&\ddots&\vdots\\\hline\boldsymbol{P}_{r,1}&\boldsymbol{P}_{r,2}&\cdots&\boldsymbol{P}_{r,k}\end{array}\right],\quad\boldsymbol{x}=\left[\begin{array}{c}x_1\\\hline x_2\\\hline\vdots\\\hline x_k\end{array}\right],\quad\boldsymbol{b}=\left[\begin{array}{c}b_1\\\hline b_2\\\hline\vdots\\\hline b_r\end{array}\right].$$

The problem PC-ℓ_2-subspace approximation can now be expressed as the regression problem:

$$\min_{\boldsymbol{C},\boldsymbol{x}}:\|\boldsymbol{P}\boldsymbol{x}-\boldsymbol{b}\|_2^2.\tag{45}$$

Written this way, it is clear that for any $\boldsymbol{C}$, the optimization problem with respect to $\boldsymbol{x}$ is simply a regression problem. For the sake of exposition, suppose that for the optimal solution $(\boldsymbol{C}^*,\boldsymbol{x}^*)$, the matrix $\boldsymbol{P}$ turns out to have a full column rank (i.e., $\boldsymbol{P}^T\boldsymbol{P}$ is invertible). In this case, the we can write down the normal equation $\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{x}=\boldsymbol{P}^T\boldsymbol{b}$ and solve it using Cramer's rule! More specifically, let $\boldsymbol{D}=\boldsymbol{P}^T\boldsymbol{P}$ and $\boldsymbol{D}_j^{(i)}$ be the matrix obtained by replacing the i^{th} column in the j^{th} column block of $\boldsymbol{D}$ with the column $\boldsymbol{P}^T\boldsymbol{b}$ for $j\in[k],i\in[s_j]$. Using Cramer's rule, we have $x_j^{(i)}=\det(\boldsymbol{D}_j^{(i)})/\det(\boldsymbol{D})$.

The key observation now is that substituting this back into the objective yields an optimization problem over (the variables) $\boldsymbol{C}$. First, observe that using the normal equation, the objective can be simplified as

$$\|\boldsymbol{P}\boldsymbol{x}-\boldsymbol{b}\|_2^2=\boldsymbol{x}^T\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{x}-\boldsymbol{x}^T\boldsymbol{P}^T\boldsymbol{b}-\boldsymbol{b}^T\boldsymbol{P}\boldsymbol{x}+\|\boldsymbol{b}\|^2=\|\boldsymbol{b}\|^2-\boldsymbol{b}^T\boldsymbol{P}\boldsymbol{x}.$$

Suppose t is a real valued parameter that is a guess for the objective value. We then consider the following feasibility problem:

$$\|\boldsymbol{P}\boldsymbol{x}-\boldsymbol{b}\|_2^2 = \|\boldsymbol{b}\|_2^2 - \boldsymbol{b}^T\boldsymbol{P}\boldsymbol{x} \le t \tag{46}$$

$$\iff \|\boldsymbol{b}\|_2^2 - t \le \sum_{j\in[k], i\in[s_j]} (\boldsymbol{b}^T\boldsymbol{P})_j^{(i)} \det(\boldsymbol{D}_j^{(i)}), \quad \det(\boldsymbol{D}) = 1. \tag{47}$$

The idea is to solve this feasibility problem using the literature on solving polynomial systems. This leaves two main gaps: guessing t, and handling the case of $\boldsymbol{P}$ not having a full column rank in the optimal solution. We handle the former issue using known quantitative bounds on the solution value to polynomial systems, and the latter using a pre-multiplication with random matrices of different sizes.

Core Solver. We begin by describing the details of solving (47), assuming a feasible guess for t, and assuming that $\boldsymbol{P}$ has full column rank. Note that this is an optimization problem in the variables c_{ij}, and so the number of variables is $rk = O(k^2/\varepsilon)$. Furthermore, the degree of the polynomials is $O(s)$, and the bit sizes of all the coefficients is $\mathrm{poly}(n, H)$.

We can thus use a well-known result on solving polynomial systems over the reals:

Theorem 4.8 ([Ren92a, Ren92b, BPR96])**.** *Given a real polynomial system $P(x_1,\ldots,x_v)$ having v variables and m polynomial constraints $f_i(x_1,\ldots,x_v)\Delta_i 0$, where $\Delta_i \in \{\ge, =, \le\}$, where d is the maximum degree of all polynomials, and H is the maximum bit-size of the coefficients of the polynomials, one can find a solution to P (or declare infeasibility) in time $(md)^{O(v)}\mathrm{poly}(H)$.*

Applying this Theorem to our setting, we obtain a running time of $s^{O(rk)} \cdot \mathrm{poly}(H) = n^{O(k^2/\varepsilon)} \cdot \mathrm{poly}(H)$, where H is the maximum bit-size of the entries in the matrices $\boldsymbol{P}_j$ and the target vectors $\boldsymbol{b}$.

Guessing t. The solver step assumes that we are able to guess a feasible value of t. In order to perform a binary search, we need some guarantees on the range of the objective value. First, we note that the value of the objective is in the range $[0, \|\boldsymbol{b}\|_2^2]$, so we have an obvious upper bound. We can also test if the problem is feasible for $t = 0$. If $t = 0$ is infeasible, we need a non-trivial lower bound on t. Fortunately, this problem has been well-studied in the literature on polynomial systems. We will use the following Theorem:

Theorem 4.9 ([JPT13])**.** *Let $T = \{x \in \mathbb{R}^v | f_1(x) \ge 0, \ldots, f_\ell(x) \ge 0, f_{\ell+1}(x) = 0, \ldots, f_m(x) = 0\}$ be the feasibility set for a polynomial system, where $f_1, \ldots, f_m \in \mathbb{Z}[x_1, \ldots, x_v]$ are polynomials with degrees bounded by an even integer d and coefficients of absolute value at most G, and let C be a compact connected component of T. Let $g \in \mathbb{Z}[x_1,\ldots,x_v]$ be a polynomial of degree at most d and coefficients of absolute value bounded by H. Then the minimum value that g takes over C if not zero, has absolute value at least*

$$(2^{4-v/2}\tilde{G}d^v)^{-v2^vd^v},$$

where $\tilde{G} = \max\{G, 2v+2m\}$.

We can use Theorem 4.9 to obtain a lower bound on the minimum non-zero value attainable for the polynomial

$$\min_{\boldsymbol{C}} : \|\boldsymbol{b}\|_2^2 - \sum_{j\in[k], i\in[s_j]} (\boldsymbol{b}^T\boldsymbol{P})_j^{(i)} \det(\boldsymbol{D}_j^{(i)}), \quad \text{over the set defined by } \det(\boldsymbol{D}) = 1. \tag{48}$$

Using our assumption that the bit complexity of the (near-) optimal solution $\boldsymbol{C}^*$ is $\Delta = \text{poly}(n, H)$, we can add box constraints for each of the variables, making the feasible set compact. Thus, we have polynomials of degree $O(s)$ defining the constraints and the objective, and the number of variables is $rk = O(k^2/\varepsilon)$. Using Theorem 4.9, we have that if the minimum value is non-zero, it must be at least

$$\left(2^{\text{poly}(n,H)} s^v\right)^{-v2^v s^v}, \text{ where } v = \frac{k^2}{\varepsilon}.$$

This implies that a binary search takes time $O\left((2s)^{k^2/\varepsilon}\text{poly}(n, H)\frac{k^4}{\varepsilon^2}\log s\right)$. Note that this time roughly matches the complexity of the core Solver procedure.

Column Rank of $\boldsymbol{P}$. The algorithm above requires the matrix $\boldsymbol{D} = \boldsymbol{P}^T\boldsymbol{P}$ to have full rank, for a (near) optimum $\boldsymbol{C}^*$. If this does not hold, the idea will be to search over all the possible values of the rank. One natural idea is to solve the problem for all subsets of the columns $[s]$, but note that this takes time $\exp(s)$, which is much larger than our target running time $\exp(\text{poly}(k))$. We thus perform a randomized procedure: for every guess j for the column rank of $\boldsymbol{P}$, we take a random matrix $R \in \mathbb{R}^{s\times j}$, and consider the regression problem

$$\min_{\boldsymbol{C},\boldsymbol{x}} \|(\boldsymbol{P}R)\boldsymbol{x} - \boldsymbol{b}\|^2 \le t. \tag{49}$$

Let $\boldsymbol{M}$ be the $\boldsymbol{P}$ matrix corresponding to an optimal solution $\boldsymbol{C}^*$, and suppose j is its rank. In Lemma 4.10, we show that if the entries of R are drawn IID from $\mathcal{N}(0, 1)$, then with probability at least 3/4, the matrix $\boldsymbol{M}R$ has full column rank. Thus, if we were to solve (49) with the optimal value of t as our guess using the Solver discussed above, we would obtain an approximately optimal solution.

This leads to the following overall algorithm:

- For $j = 1, 2, \ldots, s$:
  - Sample a random matrix $R \in \mathbb{R}^{s\times j}$ with entries drawn i.i.d. from $\mathcal{N}(0, 1)$.
  - Guess value of t for the formulation (49) as discussed above.
  - If the determinant is not identically zero, call the core Solve subroutine and check obtained solution for feasibility.
- Return the solution found as above with the least t.

Note that we can boost the success probability by sampling multiple R for each guess of j. This completes the proof of our result, Theorem 4.7, modulo Lemma 4.10 which we prove below.

Lemma 4.10. *Let $M \in \mathbb{R}^{d'\times s}$ be a matrix of rank j, and $R \in \mathbb{R}^{s\times j}$ be a random matrix with entries drawn i.i.d. from $\mathcal{N}(0, 1)$. Then the rank of MR is equal to j with probability* $\ge 3/4$.

Note that since the ranks are equal, the column spans of M and MR must be the same.

Proof. We will prove a quantitative version of the statement. Suppose we denote the jth largest singular value of M as $\tau = \sigma_j(M)$. We will show that $\sigma_j(MR) \ge \frac{\sigma_j}{4j^{3/2}}$ with probability $\ge 3/4$.

Let $\mathcal{S}$ be the span of the columns of M. For a random x whose entries are drawn from $\mathcal{N}(0, 1)$, note that the distribution of Mx is a (non-spherical) Gaussian on $\mathcal{S}$. Moreover the covariance matrix has all eigenvalues $\ge \tau^2$. Let us now consider the matrix MR. We use a leave-one-out argument to show linear independence of its columns. I.e., we claim that every column of MR has a projection of length at least $\frac{\tau}{4j}$ orthogonal to the span of the other columns, with probability at least 3/4. This implies (e.g., see [RV09]) that $\sigma_j(MR) \geq \frac{\sigma_j}{4j^{3/2}}$ with probability at least 3/4.

To see the claim, suppose we condition on all but the ith column of the matrix R, for some $1 \leq i \leq j$. Then by the earlier observation, the ith column of MR (denoted by V_i) is distributed as a non-spherical Gaussian on $\mathcal{S}$ whose covariance matrix has all eigenvalues $\geq \tau^2$. Thus its projection to the space orthogonal to the span of the remaining columns of MR (which are all fixed after conditioning) behaves as a Gaussian with at least one dimension and standard deviation $\geq \tau$. Thus by using the anti-concentration bound for a Gaussian (which states that the probability mass in any $\delta\tau$ sized interval is $\leq \delta$), V_i's projection orthogonal to the span of the other columns is at least $\frac{1}{4j} \cdot \tau$, with probability $1 - \frac{1}{4j}$. We can now take a union bound over all $1 \leq i \leq j$ to obtain a success probability of 3/4. This completes the proof. □

Remark about precision. The result above assumes that we use infinite precision for R. We now show how to avoid this assumption, with a slight loss in the parameters. Note that the key step in the above argument is showing that conditioned on the randomness in all but the ith column of R, the vector $V_i = \sum_{l=1}^{s} R_{il} M_l$ has a sufficiently large norm in the direction orthogonal to the span of the other columns in MR (that are fixed due to the conditioning). For convenience, let Π be the projector orthogonal to the span of the other columns, and denote $v_l = \Pi M_l$, and $X_l := R_{il}$. By the assumption on the least singular value, for any Π, we have that $\sum_l \|v_l\|^2 \geq \tau^2$. This implies that there exists a coordinate $t \in [d']$ such that $\sum_l v_{lt}^2 \geq \frac{\tau^2}{d'}$. Just focusing on this coordinate, we could note that $\sum_l v_{lt} X_l$ is distributed as a Gaussian and thus conclude that the probability of the coordinate being $< \frac{\tau}{4j\sqrt{d'}}$ is $< 1/4j$. This leads to a slightly weaker (by a $\sqrt{d'}$ factor) bound on the least singular value, with the same success probability. However, this argument is more flexible, it lets us use "discretized" Gaussians.

Lemma 4.11. *For any $\delta, \eta \in (0, 1/2)$, there exists a centered distribution $\mathcal{Y}$ with the following properties:*

1. *The support of $\mathcal{Y}$ and the probability masses at each point in the support are all rational numbers of bit length $b = O(\log \frac{s}{\delta\eta})$.*

2. *For all $\{a_i\}_{i=1}^s$ with $\sum_i a_i^2 = 1$, if $Y_1, Y_2, \ldots, Y_s$ are independent random variables drawn from $\mathcal{Y}$,*

$$\Pr[|\sum_i a_i Y_i| < \delta] \leq 2\delta + \eta.$$

Proof. The proof goes by a discretization of the Gaussian distribution and a sequence of reductions. First, let X_i be independent random variables distributed as $\mathcal{N}(0,1)$. Let X_i' be independent random variables distributed as a *truncated normal*, the distribution obtained from $\mathcal{N}(0,1)$ by removing the mass at points $> \sqrt{\log(s/\eta)}$ and rescaling. We begin by noting that

$$\Pr[|\sum_i a_i X_i'| < \delta] \leq (1+\eta) \Pr[|\sum_i a_i X_i| < \delta]. \tag{50}$$

To see this, let us write $\boldsymbol{X}$ to be the vector $(X_1, X_2, \ldots, X_s)$. If $f(\boldsymbol{X})$ and $f'(\boldsymbol{X})$ denote the probability density functions of the multi-dimensional normal and the truncation version respectively, by the choice of the truncation, we have, for all $\boldsymbol{X}$,

$$f'(\boldsymbol{X}) \leq \left(1 + \frac{\eta}{s}\right)^s f(\boldsymbol{X}) \leq (1+\eta) f(\boldsymbol{X}).$$

Furthermore, if any of the coordinates of $\boldsymbol{X}$ is $> \sqrt{\log(s/\eta)}$, we have $f'(\boldsymbol{X}) = 0$. Next, note that the LHS of (50) can be written out as an integral, and every term also appears on the probability on the right, albeit with the measure f' replaced with f. Thus, using the bound above, (50) follows.

Next, we discretize the interval $[-\lceil\sqrt{\log(s/\eta)}\rceil, \lceil\sqrt{\log(s/\eta)}\rceil]$ into integral multiples of $1/M$, where M is a parameter we will choose later. To the point i/M, we assign the mass that the truncated Gaussian assigns to the interval $[i - \frac{1}{2M}, i + \frac{1}{2M}]$. We call this discrete distribution $\mathcal{D}$. Let $Y_1, Y_2, \ldots, Y_s$ be IID samples from $\mathcal{D}$. We can write $Y_i = X_i' + Z_i$, where X_i' is drawn from the truncated Gaussian, and $|Z_i| \le \frac{1}{2M}$. Thus, for $M > \frac{s}{\delta}$, we claim that

$$\Pr[|\sum_i a_i Y_i| < \delta] \le \Pr[|\sum_i a_i X_i'| < 2\delta].$$

This follows because $|\sum_{i\in[s]} a_i Z_i| \le \frac{s}{2M} < \delta$. Using the earlier bounds, this implies that

$$\Pr[|\sum_i a_i Y_i| < \delta] \le (1+\eta)\cdot 2\delta.$$

This almost completes the proof, because we have obtained a discrete distribution with the desired anti-concentration bound. But as such, note that the probability values can require very high precision. This turns out to be easy to correct: we can take $\mathcal{Y}$ to be any distribution on the same support as $\mathcal{D}$ with $d_{TV}(\mathcal{Y}, \mathcal{D}) < \epsilon$, and we can conclude (using a coupling argument and a union bound), that if $W_1, W_2, \ldots, W_s$ are drawn IID from $\mathcal{Y}$,

$$\Pr[|\sum_i a_i W_i| < \delta] \le \Pr[|\sum_i a_i Y_i| < \delta] + \epsilon s.$$

To complete the argument, we need to ensure that $\epsilon < \frac{\eta}{s}$; this can be achieved using probability values with bit complexity only $O(\log(s/\eta))$, thus completing the proof. □

4.3 Projective Non-negative Matrix Factorization

In projective non-negative matrix factorization, the basis matrix $\boldsymbol{U} \in \mathbb{R}^{d\times k}$ is constrained to have non negative entries. More formally, the mathematical program formulation for Projective Non-negative Matrix Factorization (NMF) is

$$\begin{aligned} \min : \|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \quad & \text{(NMF)} \\ \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k,\ \boldsymbol{U} \in \mathbb{R}^{d\times k}_{\ge 0}. \quad & (51) \end{aligned}$$

There is a alternate formulation of NMF that better aligns with the name of the problem and is well suited to apply our framework:

$$\begin{aligned} \min : \|\boldsymbol{A} - \boldsymbol{W}\boldsymbol{H}\|_F^2 \quad & \text{(NMF-alternate)} \\ \boldsymbol{W} \in \mathbb{R}^{d\times k}_{\ge 0} \text{ has orthogonal columns .} \quad & (52) \end{aligned}$$

Lemma 4.12. *The programs NMF and NMF-alternate are equivalent.*

Proof. Using Lemma 3.2, we know that NMF is equivalent to

$$\begin{aligned} \min : \|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{H}\|_F^2 \\ \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k,\ \boldsymbol{U} \in \mathbb{R}^{d\times k}_{\ge 0}. \end{aligned}$$

Since $\boldsymbol{H}$ is unconstrained, it can absorb the normalization of the columns of $\boldsymbol{U}$. This gives

$$\begin{gathered}\min : \|\boldsymbol{A} - \boldsymbol{W}\boldsymbol{H}\|_F^2 \\ \boldsymbol{W} \in \mathbb{R}_{\geq 0}^{d\times k} \text{ has orthogonal columns}\end{gathered}$$

which is exactly Equation (NMF-alternate). □

Lemma 4.13. *The set of matrices*

$$\mathcal{W} := \{\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{d\times k} : \boldsymbol{W} \text{ has orthogonal columns } \} \tag{53}$$

is equal to the set

$$\overline{\mathcal{W}} := \{\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{d\times k} : \|\boldsymbol{W}_{i,\cdot}\|_0 \leq 1 \quad \forall i \in [d]\}. \tag{54}$$

Proof. For any $\boldsymbol{W} \in \mathcal{W}$, if there exists a row $i \in [d]$ and two distinct indices $j, j' \in [k]$ such that $\boldsymbol{W}_{i,j}, \boldsymbol{W}_{i,j'} \neq 0$, then by the non-negativity constraint, these non zero values are in fact strictly positive. The dot product of the columns $\boldsymbol{W}_{\cdot,j}$ and $\boldsymbol{W}_{\cdot,j'}$is at least $\boldsymbol{W}_{i,j} \cdot \boldsymbol{W}_{i,j'} > 0$ which contradicts the orthogonality of the columns of $\boldsymbol{W}$. This implies that there is at most one non-zero entry in every row of $\boldsymbol{W}$ which further implies that $\boldsymbol{W} \in \overline{\mathcal{W}}$.

For any $\boldsymbol{W} \in \overline{\mathcal{W}}$, the orthogonality of the columns is straight forward because for any two distinct indices $j, j' \in [k]$, the dot product of the columns $\boldsymbol{W}_{\cdot,j}$ and $\boldsymbol{W}_{\cdot,j'}$ is zero because either of $\boldsymbol{W}_{i,j}, \boldsymbol{W}_{i,j'}$ is equal to zero for every $i \in [d]$. □

Lemma 4.14. *For any given $\boldsymbol{B} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{H} \in \mathbb{R}^{k\times r}$, we can solve the program*

$$\begin{gathered}\min : \|\boldsymbol{B} - \boldsymbol{W}\boldsymbol{H}\|_F^2 \\ \boldsymbol{W} \in \mathbb{R}_{\geq 0}^{d\times k} \text{ has orthogonal columns}\end{gathered}$$

exactly in time $O(dkr)$.

Proof. Using Lemma 4.13, we can re-write the optimization problem as

$$\begin{gathered}\min : \|\boldsymbol{B} - \boldsymbol{W}\boldsymbol{H}\|_F^2 \\ \|\boldsymbol{W}\|_0 \leq 1 \quad \forall i \in [d],\ \boldsymbol{W} \in \mathbb{R}_{\geq 0}^{d\times k}.\end{gathered}$$

Since the rows of $\boldsymbol{W}$ are independent variables, we can decompose the problem into

$$\sum_{i=1}^{d} \min_{w_i \in \mathbb{R}_{\geq 0}^{k},\, \|w_i\|_0 \leq 1} : \|b_i - \boldsymbol{H}^T w_i\|_2^2$$

where b_i, w_i are the i^{th} columns of $\boldsymbol{B}^T$ and $\boldsymbol{W}^T$ respectively. Each problem $\min_{w_i \in \mathbb{R}_{\geq 0}^{k},\, \|w_i\|_0 \leq 1} :$ $\|b_i - \boldsymbol{H}^T w_i\|_2^2$ can be solved by looking at the k cases for the non-zero entry if w_i. For every choice of non-zero entry, say $j \in [k]$, the resulting minimization problem is $min_{\lambda \geq 0} : \|b_i - \lambda h_j\|_2^2$ where h_j is the j^{th} column of $\boldsymbol{H}^T$. The optimal choice of λ is $\max(0, \langle b_i, h_j\rangle / \|h_j\|_2^2)$. The only computation we had to do is to evaluate the dot products between b_i, h_j for $i \in [d], j \in [k]$ which takes $O(dkr)$ time. □

Theorem 4.15 (Additive approximation for NMF)**.** *Given an instance $\boldsymbol{A} \in \mathbb{R}^{d\times n}$ of Non-negative matrix factorization, there is an algorithm that computes a $\boldsymbol{U} \in \mathbb{R}^{d\times k}_{\geq 0}$, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k$ such that*

$$\|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \leq (1+\varepsilon)\cdot OPT + O(\delta \cdot \|\boldsymbol{A}\|_F^2)$$

in time $O(dk^2/\varepsilon)\cdot(1/\delta)^{O(k^2/\varepsilon)}$. For any $0 < \delta < 1$.

Proof. Let $\boldsymbol{U}^*$ be the optimal solution to NMF. This implies that $\boldsymbol{H} = \boldsymbol{U}^{T^*}\boldsymbol{A}$ is an optimal solution to NMF-alternate. We first replace the instance with a smaller instance $\boldsymbol{B}$. Then we search for every row of $\boldsymbol{H}$ exactly as in the proof of Lemma 3.1 to obtain a solution $\boldsymbol{U} \in \mathbb{R}^{d\times k}_{\geq 0}$, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k$ such that

$$\|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \leq (1+\varepsilon)\cdot \mathrm{OPT} + O(\delta \cdot \|\boldsymbol{A}\|_F^2)$$

in time $T_0 + T_1 \cdot (1/\delta)^{O(rk)}$. Where T_0 is the time required to obtain $\boldsymbol{B}$ and T_1 is the time required to solve for the optimal $\boldsymbol{W}$ in the program

$$\begin{gathered}\min : \|\boldsymbol{B} - \boldsymbol{W}\boldsymbol{H}\|_F^2 \\ \boldsymbol{W} \in \mathbb{R}^{d\times k}_{\geq 0} \text{ has orthogonal columns}\end{gathered}$$

We know that $T_1 = O(dkr)$ using Lemma 4.14 and $T_0 = O(nrd^2 \cdot H)$ from Lemma 2.1. We hide T_0 as it is negligible. $\square$

Theorem 4.16 (Multiplicative approximation for NMF)**.** *Given an instance $\boldsymbol{A} \in \mathbb{R}^{d\times n}$ of Non-negative matrix factorization with integer entries of absolute value at most γ in $\boldsymbol{A}$, there is an algorithm that computes a $\boldsymbol{U} \in \mathbb{R}^{d\times k}_{\geq 0}$, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k$ such that*

$$\|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \leq (1+\varepsilon)\cdot OPT$$

in time $(nd\gamma/\varepsilon)^{O(k^3/\varepsilon)}$.

Proof. Using Lemma 3.8, we know that $\|\boldsymbol{A}\|_F^2/\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2 \leq (nd\gamma)^{O(k)}$. Setting $\delta = \varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2/\|\boldsymbol{A}\|_F^2 \geq \varepsilon(nd\gamma)^{-O(k)}$ in Corollary 4.3 gives the desired time complexity. $\square$

4.4 k-means clustering [CEM+15]

In the k-means problem, we are given a collection of data points $a_1, \ldots, a_n \in \mathbb{R}^d$. The objective is to find k-centers $c_1, \ldots, c_k \in \mathbb{R}^d$ and an assignment $\pi : [n] \to [k]$ that minimizes:

$$\sum_{i=1}^{n} \|a_i - c_{\pi(i)}\|_2^2. \tag{k-means}$$

Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{C} \in \mathbb{R}^{k\times d}$ matrices with a_i and c_i as their i^{th} rows respectively (note that this differs from the notation we used for previous applications). Let $\boldsymbol{\Pi} \in \mathbb{R}^{n\times k}$ be the matrix such that $\boldsymbol{\Pi}_{i,j} = \mathbb{1}[\pi(i) = j]$. Using this notation, the k-means problem can be written as

$$\begin{gathered}\min : \|\boldsymbol{A} - \boldsymbol{\Pi}\boldsymbol{C}\|_F^2 \\ \text{Each row of } \boldsymbol{\Pi} \text{ is a standard basis vector.}\end{gathered} \tag{k-means-matrix}$$

Observe that k-means-matrix is a special case of NMF-alternate where $\boldsymbol{W}$ is additionally constrained to have all non-zero entries to be equal to 1. Also, the k-means and k-means-matrix correspond to the CSA-geo and CSA-fac formulations of the same problem. The corresponding CSA version is

$$\begin{aligned} \min : & \|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \\ \boldsymbol{U}_{i,j} &= 1/\sqrt{\|U_{.,j}\|_0} \quad \forall i \in [n],\, j \in [k]. \end{aligned} \tag{k-means-CSA}$$

The three main steps in our algorithm are:

1. **Reduction:** The first step is to reduce the number of rows and columns of the target matrix $\boldsymbol{A}$.
   (a) **Columns:** Replace the matrix $\boldsymbol{A}$ with the matrix $\boldsymbol{B}$ as in Lemma 3.5. This reduces the number of columns (dimension of the data-points) to $r = k + k/\varepsilon$.
   (b) **Rows:** For any fixed set of centers (selected from the rows of) $\boldsymbol{C}$, the cost induced by the centers is defined as

   $$\mathrm{Cost}(\boldsymbol{A}, \boldsymbol{C}) := \sum_{i=1}^{n} \mathrm{dist}(a_i, \boldsymbol{C})^2.$$

   Where $\mathrm{dist}(a, \boldsymbol{C}) := \min_{c \in \boldsymbol{C}} \|a - c\|_2$. A strong coreset for the k-means instance $\boldsymbol{A}$ is a subset $S \subseteq [n]$ of indices and weights w_i corresponding to each index $i \in S$ such that for any set of centers $\boldsymbol{C}$, we have

   $$\mathrm{Cost}_{w,S}(\boldsymbol{A}, \boldsymbol{C}) := \sum_{i \in S} w_i \mathrm{dist}(a_i, \boldsymbol{C}) \in (1 \pm \varepsilon) \cdot \mathrm{Cost}(\boldsymbol{A}, \boldsymbol{C}).$$

   Coresets for k-means of optimal size $\widetilde{O}(k\varepsilon^{-2} \min\{\sqrt{k}, \varepsilon^{-2}\})$ are known (See [CALS+24] for upper-bound and [HLW24] for matching lower-bound). Any algorithm that efficiently computes a coreset of size $q = \mathrm{poly}(k/\varepsilon)$ can be used as a black box for our purposes. After using such a coreset, the new formulation is

   $$\min : \|\boldsymbol{B} - \boldsymbol{\Pi}\boldsymbol{C}\|_F^2 \tag{k-means-reduced}$$
   $$\text{Each row of } \boldsymbol{\Pi} \text{ has exactly one non-zero entry equal to } w_i. \tag{55}$$

   where $\boldsymbol{B} \in \mathbb{R}^{q \times r}$, $\boldsymbol{\Pi} \in \mathbb{R}^{q \times k}$ with their rows indexed by the set S defined by the coreset and the rows of $\boldsymbol{B}$ are the scaled rows of $\boldsymbol{A}$ according to the weight defined by the coreset for that row.

2. **Enumeration:** A naive approach is to simply enumerate all the k^q possible $\boldsymbol{\Pi}$ matrices by choosing the position of the non-zero element in each row. Simply put, we go through all possible k-clusterings of the coreset elements. Optimal choice of centers can be computed as the weighted mean of the coreset elements in each cluster. This allows us to identify the optimal $\boldsymbol{\Pi}$.

   Let $\boldsymbol{\Pi}^*$ be the optimal choice of $\boldsymbol{\Pi}$ in the k-means-reduced program. Using Lemma 4.17 and Lemma 4.18, we enumerate over the $O(\log n \cdot k \cdot \mathrm{poly}(k/\varepsilon))^{O(k \log k + k/\varepsilon)} = O(k/\varepsilon \cdot \log n)^{\widetilde{O}(k/\varepsilon)}$ number of possible pairs of matrices $\boldsymbol{S}\boldsymbol{B}$ and $\boldsymbol{S}\boldsymbol{\Pi}$. For each such pair, we find the $\boldsymbol{C}$ that minimizes $\|\boldsymbol{S}\boldsymbol{B} - \boldsymbol{S}\boldsymbol{\Pi}\boldsymbol{C}\|_F^2$. For every such $\boldsymbol{C}$, evaluate the cost induced by these centers with the coreset (w, S). Let $\overline{\boldsymbol{C}}$ be the set of centers that has the lowest cost with respect to the coreset from the enumeration described before. The cost induced by this set of centers is

$$
\begin{aligned}
\min_{\boldsymbol{\Pi}} \|\boldsymbol{B} - \boldsymbol{\Pi}\overline{\boldsymbol{C}}\|_F^2 &\leq \min_{\boldsymbol{\Pi}} \|\boldsymbol{B} - \boldsymbol{\Pi}\widehat{\boldsymbol{C}}\|_F^2 && (56)\\
&\leq \|\boldsymbol{B} - \boldsymbol{\Pi}^*\widehat{\boldsymbol{C}}\|_F^2 && (57)\\
&\leq (1+\varepsilon)\|\boldsymbol{B} - \boldsymbol{\Pi}^*\boldsymbol{C}^*\|_F^2 && (58)\\
&\leq (1+\varepsilon)^2 \cdot \mathrm{OPT}. && (59)
\end{aligned}
$$

Using the coreset property, we imply that the cost of the centers $\overline{C}$ on the original instance $\boldsymbol{A}$ is at most $(1+\varepsilon)^3 \cdot \mathrm{OPT}$.

$$ \tag{60} $$

We start with the following known result (see Theorem 38 of [CW17]).

Lemma 4.17. *Given matrices $\boldsymbol{B} \in \mathbb{R}^{q \times r}$ and $\boldsymbol{\Pi}^* \in \mathbb{R}^{q \times k}$, there exists a matrix $\boldsymbol{S} \in \mathbb{R}^{t \times q}$ and such that*

1. *Each row of $\boldsymbol{S}$ contains exactly one positive non-zero element from the set $W = \{2^i : 0 \leq i \leq N\}$.*
2. *If $\boldsymbol{C}^* = \arg\min_{\boldsymbol{C} \in \mathbb{R}^{k \times r}} \|\boldsymbol{B} - \boldsymbol{\Pi}^*\boldsymbol{C}\|_F^2$ and $\widehat{\boldsymbol{C}} = \arg\min_{\boldsymbol{C} \in \mathbb{R}^{k \times r}} \|\boldsymbol{S}\boldsymbol{B} - \boldsymbol{S}\boldsymbol{\Pi}^*\boldsymbol{C}\|_F^2$, then*
$$\|\boldsymbol{B} - \boldsymbol{\Pi}^*\widehat{\boldsymbol{C}}\|_F^2 \leq (1+\varepsilon) \cdot \|\boldsymbol{B} - \boldsymbol{\Pi}^*\boldsymbol{C}^*\|_F^2. \tag{61}$$
3. *$t = O(k \log k + k/\varepsilon)$.*

Note that [CW17] do not require the non-zero element of $\boldsymbol{S}$ to come from W. Indeed, it will be proportional to the leverage score. However, note that we can "discretize" the leverage scores (while keeping a factor two approximation to each one), and still obtain all the guarantees that we require. Finally, since the leverage scores add up to the matrix dimension, we have the bound $N = O(\log n)$.

Lemma 4.18. *Given a matrix $\boldsymbol{B} \in \mathbb{R}^{q \times r}$, the number of possible matrices*

1. *of the form $\boldsymbol{S}\boldsymbol{B}$ is at most $O(Nq)^t$.*
2. *of the form $\boldsymbol{S}\boldsymbol{\Pi}$ where $\boldsymbol{\Pi}$ satisfies Equation (55) is at most $O(Nkq)^t$.*

where $\boldsymbol{S}$ satisfies property 1 in Lemma 4.17.

Proof. Each row of $\boldsymbol{S}\boldsymbol{B}$ is simply a row of $\boldsymbol{B}$ that is scaled by 2^i for some $0 \leq i \leq N$. This leaves Nq choices for each of the t rows of $\boldsymbol{S}\boldsymbol{B}$ which is $((N+1)q)^t$ possibilities. Each row of $\boldsymbol{S}\boldsymbol{\Pi}$ is a row of $\boldsymbol{\Pi}$ scaled by 2^i for some $0 \leq i \leq N$. The choices to make for each row of $\boldsymbol{S}\boldsymbol{\Pi}$ is a row of $\boldsymbol{\Pi}$ (which includes choices for non-zero element and weight w_j for $j \in [q]$) and a scaling factor from $\boldsymbol{S}$. This leaves $(N+1)kq$ choices for each of the t rows of $\boldsymbol{S}\boldsymbol{\Pi}$. □

Theorem 4.19. *Given an instance $\boldsymbol{A} \in \mathbb{R}^{n \times d}$ of k-means, there is an algorithm that computes a $(1+\varepsilon)$-approximate solution to k-means in $O(nnz(\boldsymbol{A}) + 2^{\widetilde{O}(k/\varepsilon)} + n^{o(1)})$ time.*

Proof. The time complexity of the three step procedure is dominated by the enumeration step which takes time $O(\log n \cdot k/\varepsilon)^{\widetilde{O}(k/\varepsilon)}$ time. If $\log n \leq (k/\varepsilon)^2$, then this running time is $O(k/\varepsilon)^{\widetilde{O}(k/\varepsilon)} = 2^{\widetilde{O}(k/\varepsilon)}$. Otherwise, if $\log n \geq (k/\varepsilon)^2$, then the running time is $(\log n)^{\widetilde{O}(\sqrt{\log n})} = n^{o(1)}$. □

4.5 Sparse-PCA [DP22]

The sparse PCA problem is a well-studied variant of PCA in which the components found are required to be sparse. In other words, the basis matrix $\boldsymbol{U} \in \mathbb{R}^{d\times k}$ is constrained to have sparsity requirements. There are two natural ways to formalize this question: the first is by requiring $\boldsymbol{U}$ to have at most s non-zero entries in total. Another is to require the number of non-zero *rows* of $\boldsymbol{U}$ to be bounded by a parameter s. In the popular case of $d = 1$, both of these definitions coincide. Let us focus on the first variant for now.[2] More formally, the mathematical program formulation we consider is

$$\max : \langle \boldsymbol{A}\boldsymbol{A}^T, \boldsymbol{U}\boldsymbol{U}^T\rangle \tag{sparse-PCA-max}$$
$$\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k,\ \sum_{j\in[k]} \|\boldsymbol{U}_{\cdot,j}\|_0 \le s. \tag{62}$$

Program sparse-PCA-max can also be formulated as a minimization version

$$\min : \|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\|_F^2 \tag{sparse-PCA-min}$$
$$\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k,\ \sum_{j\in[k]} \|\boldsymbol{U}_{\cdot,j}\|_0 \le s. \tag{63}$$

The following theorem from [DP22] shows how to find an optimal solution to sparse-PCA-max (and hence also to sparse-PCA-min) when $\operatorname{rank}(\boldsymbol{A}\boldsymbol{A}^T) = \operatorname{rank}(\boldsymbol{A}) = t$.

Theorem 4.20 (Theorem 1 in [DP22])**.** *There is an algorithm that finds an optimal solution to sparse-PCA-max in*

$$O\left(d^{\min\{k,t\}(t^2+t)}\left(\min\{k,t\}dt^2 + d\log d\right)\right), \tag{64}$$

where t denotes the rank of the matrix $\boldsymbol{A}$.

Theorem 4.21. *Given an instance $(\boldsymbol{A} \in \mathbb{R}^{d\times n}, k, s)$ of sparse-PCA, there is an algorithm that runs in time*

$$O\left(d^{kr^2+kr}\left(dkr^2 + d\log d\right)\right) \tag{65}$$

with $r = k + k/\varepsilon$ that computes a $\varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2$ additive approximate solution to both sparse-PCA-max and sparse-PCA-min. This is guarantees as a $(1+\varepsilon)$-approximate solution to sparse-PCA-min because $\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2$ is a lower bound to sparse-PCA-min.

Proof. First step is to replace $\boldsymbol{A}$ with the matrix $\boldsymbol{B}$ as in Lemma 3.5. This step takes time T_0. Solve for Equation (sparse-PCA-max) exactly using Theorem 4.20, with $t = k + k/\varepsilon$. This step takes time $O\left(d^{kr^2+kr}\left(dkr^2 + d\log d\right)\right)$. Using Lemma 3.7, the solution obtained is a $(1 + \varepsilon)$-approximate solution to sparse-PCA-min. In fact, for $p = 2$, we know that the error is at most $\varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2$ which implies that this also gives an additive approximation of $\varepsilon\|\boldsymbol{A} - \boldsymbol{A}_k\|_F^2$ to both the minimization and maximization versions. Because the objective for the maximization is the negative of the minimization objective added with $\|\boldsymbol{A}\|_F^2$. $\square$

[2] Our result follows via a black-box application of algorithms from [DP22]; since their algorithms work for both variants, so do our results.

5 Hardness of Column Subset Selection with Partition Constraint

In this section, we show that PC-CSS is at least as hard as the well-studied *sparse regression* problem [Nat95, FKT15, HPIM18, GV21]. In particular, our hardness implies that PC-column subset selection remains hard even if the number of groups is only two, or if we allow violating the given partition capacity constraints by a logarithmic factor. First, we define the PC-column subset selection problem and the sparse regression problem formally.

Definition 5.1 (Column Subset Selection with a Partition Constraint)**.** In an instance of the *PC-column subset selection* (*PC-CSS*) problem, we are given a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and a partition matroid $\mathcal{P} = ([n] = P_1 \uplus \cdots \uplus P_\ell,\ \mathcal{I})$, $\mathcal{I} = \{S \subseteq [n] : |S \cap P_t| \le k_t,\ \forall t \in [\ell]\}$ defined on the set of column indices $[n]$.

The objective is to select a (index) subset $S \in \mathcal{I}$ of columns of $\boldsymbol{A}$ in order to minimize the squared projection cost of all the column vectors of $\boldsymbol{A}$ onto the span of the column space induced by the subset of columns corresponding to S

$$\mathrm{cost}_S(\boldsymbol{A}) := \sum_{i\in[n]} \|\mathrm{proj}_{\mathrm{span}^\perp(S)}(a_i)\|_2^2 = \|\boldsymbol{A} - \boldsymbol{A}_S\boldsymbol{A}_S^+\boldsymbol{A}\|_F^2, \tag{66}$$

where a_i is the column vector corresponding to column index i in $\boldsymbol{A}$ and $\boldsymbol{A}_S$ is the matrix with columns from $\boldsymbol{A}$ corresponding to S.

Definition 5.2 ((g,h)-Sparse Regression)**.** Given a matrix $\boldsymbol{B} \in \mathbb{R}^{m\times n}$ and a positive integer s, for which there exists an unknown vector $x^* \in \mathbb{R}^n$ such that $\|x^*\|_0 \le s$ and $\boldsymbol{B}x^* = \mathbb{1}$, the goal is to output an $x \in \mathbb{R}^n$ with $\|x\|_0 \le s \cdot g(n)$ such that $\|\boldsymbol{B}x - \mathbb{1}\|_2^2 \le h(m,n)$.

The sparse regression problem is known to be computationally hard. In particular,

Theorem 5.3 ([FKT15])**.** *Let $0 < \delta < 1$. If there is a deterministic polynomial time algorithm $\mathcal{A}$ for (g,h)-sparse regression, for which $g(n) = (1-\delta)\ln n$ and $h(m,n) = m^{1-\delta}$, then* SAT $\in$ DTIME$(n^{O(\log\log n)})$.

Next, we prove our main hardness of approximation result for PC-column subset selection.

Theorem 5.4. *Assuming* SAT $\notin$ DTIME$(n^{O(\log\log n)})$*, the PC-column subset selection problem is hard to approximate to any multiplicative factor f, even in the following special cases:*

(i) The case of $\ell = 2$ groups, where the capacities on all the groups are the same parameter s.

(ii) The case where the capacities on all the groups are the same parameter s, and we allow a solution to violate the capacity by a factor $g(n) = o(\log n)$, where n is the total number of columns in the instance.

Proof. The proof is via a reduction from sparse regression. First, we show a hardness for two groups (part (a) of the Theorem). Consider an instance of sparse regression, given by an $m \times n$ matrix B and parameter s. Now consider a matrix A whose columns are $A_1 \cup A_2$, defined as follows. A_1 is an $(m+s)\times n$ matrix whose ith column is the ith column of B appended with s zeros. A_2 is an $(m+s)\times(s+1)$ matrix whose columns we denote by $u_1, u_2, \ldots, u_{s+1}$. We set $u_i = C \cdot e_{m+i}$ for $1 \le i \le s$, and $u_{s+1} = D \cdot (\mathbf{1} \oplus \mathbf{0}_s)$, for appropriately chosen parameters $C > D$.[3]

Consider any solution that chooses exactly s columns from A_1 and A_2. In the YES case of sparse regression, where there exists an s-sparse x^* with $Bx^* = \mathbf{1}$, by choosing the columns corresponding to the support of x^* from A_1 along with the columns $u_1, \ldots, u_s$ from A_2, we obtain an approximation

[3] As is standard, $\mathbf{1} \oplus \mathbf{0}_s$ is simply the all ones vector (here in m dimensions) with s zeros appended.

error at most $\|A_1\|_F^2$. Consider the NO case of sparse regression. We will choose C large enough, so that even if one of the u_i for $i \le s$ is not chosen, the error is $\ge C$. Next, suppose all the $\{u_i\}_{i\in[s]}$ are chosen. For any choice of s columns from A_1, the error on the column u_{s+1} is at least $D \cdot h(m,n)$, by assumption. Thus in either case, the approximation error is $\ge \min(C, D \cdot h(m,n))$. We can now choose C, D large enough (e.g., $> f \cdot n\|A_1\|_F^2$), and obtain the desired hardness of approximation.

Next, suppose we are allowed to choose αs columns from each group, for some slack parameter α (assumed to be an integer ≥ 1 and $< g(n)$, where the latter function comes from the hardness for sparse regression). Now let T be a parameter we will choose later (integer ≥ 1), and consider an instance of PC-column subset selection where we have $(T+1)$ groups of vectors (matrices), $A_1, A_2, \dots, A_{T+1}$, and the vectors (columns) have dimension Tm. We view each column vector as consisting of T blocks of size m. For $1 \le j \le T$, the columns of A_j are identical to those of B in the jth block, and zero everywhere else. The matrix A_{T+1} has T columns, denoted $u_1, \dots, u_T$, where u_j is the vector that has $\mathbf{1}$ in the jth block and zero everywhere else, scaled by parameter D.

As before, in the YES case of sparse regression, the approximation error is $\le T\|A_1\|_F^2$. In the NO case, consider any solution that chooses at most αs vectors from each A_j. By assumption, the error in the jth block (of u_j) is at least $h(m,n)$, for any vector u_j that is not picked from A_{T+1}. If we set $T > 2\alpha s$, then at least $(T/2)$ of the vectors u_j cannot be picked, and so the total error is at least $D \cdot (T/2)h(m,n)$. Again, we can choose D large enough to obtain the desired hardness. □

This strong hardness of approximation further motivates the study of a relaxed variant, in which the set of vectors in the small-size summary S, rather than being a subset of $\boldsymbol{A}^{(1)}, \cdots, \boldsymbol{A}^{(\ell)}$, are instead required to belong to the *subspaces* spanned by the columns in each group $\boldsymbol{A}^{(1)}, \cdots, \boldsymbol{A}^{(\ell)}$. This is precisely our PC-subspace approximation problem.

Acknowledgments: Aditya Bhaskara was supported by NSF CCF-2047288. David P. Woodruff was supported by a Simons Investigator Award and Office of Naval Research award number N000142112647.

References

[ABF$^+$16] Jason Altschuler, Aditya Bhaskara, Gang Fu, Vahab Mirrokni, Afshin Rostamizadeh, and Morteza Zadimoghaddam. Greedy column subset selection: New bounds and distributed algorithms. In *International Conference on Machine Learning*, pages 2539–2548, 2016.

[AKPS24] Deeksha Adil, Rasmus Kyng, Richard Peng, and Sushant Sachdeva. Fast algorithms for ℓ_p-regression. *J. ACM*, 71(5), October 2024.

[APD14] Megasthenis Asteris, Dimitris Papailiopoulos, and Alexandros Dimakis. Nonnegative sparse pca with provable guarantees. In *International Conference on Machine Learning*, pages 1728–1736. PMLR, 2014.

[BBB$^+$19] Frank Ban, Vijay Bhattiprolu, Karl Bringmann, Pavel Kolev, Euiwoong Lee, and David P. Woodruff. A PTAS for ℓ_p-low rank approximation. In Timothy M. Chan, editor, *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.

[BDMI11] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Sparse features for pca-like linear regression. *Advances in Neural Information Processing Systems*, 24, 2011.

[BDMI14] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

[BMD09] Christos Boutsidis, Michael W Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 968–977, 2009.

[BPR96] Saugata Basu, Richard Pollack, and Marie-Françoise Roy. On the combinatorial and algebraic complexity of quantifier elimination. *J. ACM*, 43(6):1002–1045, 1996.

[BWZ19] Frank Ban, David P. Woodruff, and Qiuyi (Richard) Zhang. Regularized weighted low rank approximation. *CoRR*, abs/1911.06958, 2019.

[BZMD14] Christos Boutsidis, Anastasios Zouzias, Michael W Mahoney, and Petros Drineas. Randomized dimensionality reduction for k-means clustering. *IEEE Transactions on Information Theory*, 61(2):1045–1062, 2014.

[CALS+24] Vincent Cohen-Addad, Kasper Green Larsen, David Saulpic, Chris Schwiegelshohn, and Omar Ali Sheikh-Omar. Improved coresets for euclidean k-means. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2024.

[CASS21] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 169–182, New York, NY, USA, 2021. Association for Computing Machinery.

[CEM+15] Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 163–172, 2015.

[CJ95] Jorge Cadima and Ian T Jolliffe. Loading and correlations in the interpretation of principle compenents. *Journal of applied Statistics*, 22(2):203–214, 1995.

[CKR20] Ashish Chiplunkar, Sagar Kale, and Sivaramakrishnan Natarajan Ramamoorthy. How to solve fair k-center in massive data models. In *International Conference on Machine Learning*, pages 1877–1886, 2020.

[CMI12] Ali Civril and Malik Magdon-Ismail. Column subset selection via sparse approximation of SVD. *Theoretical Computer Science*, 421:1–14, 2012.

[CW09] Kenneth L. Clarkson and David P. Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 205–214, New York, NY, USA, 2009. Association for Computing Machinery.

[CW15] Kenneth L. Clarkson and David P. Woodruff. Input sparsity and hardness for robust subspace approximation. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 310–329, 2015.

[CW17] Kenneth L Clarkson and David P Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)*, 63(6):1–45, 2017.

[DFK$^+$04] Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and Vishwanathan Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56:9–33, 2004.

[DP22] Alberto Del Pia. Sparse pca on fixed-rank matrices. *Math. Program.*, 198(1):139–157, January 2022.

[DR10] Amit Deshpande and Luis Rademacher. Efficient volume sampling for row/column subset selection. In *2010 ieee 51st annual symposium on foundations of computer science*, pages 329–338, 2010.

[DTV11] Amit Deshpande, Madhur Tulsiani, and Nisheeth K. Vishnoi. Algorithms and hardness for subspace approximation. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, page 482–496, USA, 2011. Society for Industrial and Applied Mathematics.

[FKT15] Dean Foster, Howard Karloff, and Justin Thaler. Variable selection is hard. In *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pages 696–709. PMLR, 2015.

[FMS07] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A ptas for k-means clustering based on weak coresets. In *Proceedings of the twenty-third annual symposium on Computational geometry*, pages 11–18, 2007.

[GL13] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 4th edition, 2013.

[GRSW16] Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing ugc from some optimal geometric inapproximability results. *ACM Trans. Algorithms*, 12(1), feb 2016.

[GS12] Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1207–1214, 2012.

[GV21] Aparna Gupte and Vinod Vaikuntanathan. The fine-grained hardness of sparse linear regression. *arXiv preprint arXiv:2106.03131*, 2021.

[HLW24] Lingxiao Huang, Jian Li, and Xuan Wu. On optimal coreset construction for euclidean (k, z)-clustering. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, page 1594–1604, 2024.

[HMV23] Sedjro Salomon Hotegni, Sepideh Mahabadi, and Ali Vakilian. Approximation algorithms for fair range clustering. In *International Conference on Machine Learning*, pages 13270–13284. PMLR, 2023.

[HPIM18] Sariel Har-Peled, Piotr Indyk, and Sepideh Mahabadi. Approximate sparse linear regression. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[HTW15] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. Statistical learning with sparsity. *Monographs on statistics and applied probability*, 143(143):8, 2015.

[JNN20] Matthew Jones, Huy Nguyen, and Thy Nguyen. Fair k-centers via maximum matching. In *International Conference on Machine Learning*, pages 4940–4949, 2020.

[JPT13] Gabriela Jeronimo, Daniel Perrucci, and Elias Tsigaridas. On the minimum of a polynomial function on a basic closed semialgebraic set and applications. *SIAM Journal on Optimization*, 23(1):241–255, 2013.

[KAM19] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. Fair k-center clustering for data summarization. In *International Conference on Machine Learning*, pages 3448–3457, 2019.

[Moi16] Ankur Moitra. An almost optimal algorithm for computing nonnegative rank. *SIAM J. Comput.*, 45(1):156–173, 2016.

[MOT23] Antonis Matakos, Bruno Ordozgoiti, and Suhas Thejaswi. Fair column subset selection. *arXiv preprint arXiv:2306.04489*, 2023.

[MW20] Arvind V. Mahankali and David P. Woodruff. Optimal ℓ_1 column subset selection and a fast PTAS for low rank approximation. *CoRR*, abs/2007.10307, 2020.

[Nat95] Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[PDK13] Dimitris Papailiopoulos, Alexandros Dimakis, and Stavros Korokythakis. Sparse pca through low-rank approximations. In *International Conference on Machine Learning*, pages 747–755. PMLR, 2013.

[Ren92a] James Renegar. On the computational complexity and geometry of the first-order theory of the reals. part i: Introduction. preliminaries. the geometry of semi-algebraic sets. the decision problem for the existential theory of the reals. *Journal of symbolic computation*, 13(3):255–299, 1992.

[Ren92b] James Renegar. On the computational complexity and geometry of the first-order theory of the reals. part ii: The general decision problem. preliminaries for quantifier elimination. *Journal of Symbolic Computation*, 13(3):301–327, 1992.

[RSW16] Ilya Razenshteyn, Zhao Song, and David P. Woodruff. Weighted low rank approximations with provable guarantees. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, page 250–263, 2016.

[RV09] Mark Rudelson and Roman Vershynin. The smallest singular value of a random rectangular matrix, 2009.

[STM+18] Samira Samadi, Uthaipon Tantipongpipat, Jamie H Morgenstern, Mohit Singh, and Santosh Vempala. The price of fair PCA: One extra dimension. In *Advances in neural information processing systems*, pages 10976–10987, 2018.

[SVK+17] Ignacio Santamaria, Javier Vía, Michael Kirby, Tim Marrinan, Chris Peterson, and Louis Scharf. Constrained subspace estimation via convex optimization. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1200–1204. IEEE, 2017.

[SVWZ24] Zhao Song, Ali Vakilian, David Woodruff, and Samson Zhou. On socially fair regression and low-rank approximation. In *Advances in Neural Information Processing Systems*, 2024.

[Tro09] Joel A Tropp. Column subset selection, matrix factorization, and eigenvalue optimization. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 978–986. SIAM, 2009.

[TSS+19] Uthaipon Tantipongpipat, Samira Samadi, Mohit Singh, Jamie H Morgenstern, and Santosh Vempala. Multi-criteria dimensionality reduction with applications to fairness. In *Advances in Neural Information Processing Systems*, pages 15135–15145, 2019.

[VVWZ23] Ameya Velingker, Maximilian Vötsch, David P. Woodruff, and Samson Zhou. Fast $(1+\varepsilon)$-approximation algorithms for binary matrix factorization. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[WY24] David P. Woodruff and Taisuke Yasuda. Nearly linear sparsification of ℓ_p subspace approximation, 2024.

[WY25] David P. Woodruff and Taisuke Yasuda. Ridge leverage score sampling for ℓ_p subspace approximation, 2025.

[YO05] Zhijian Yuan and Erkki Oja. Projective nonnegative matrix factorization for image compression and feature extraction. In *Image Analysis: 14th Scandinavian Conference, SCIA 2005, Joensuu, Finland, June 19-22, 2005. Proceedings 14*, pages 333–342. Springer, 2005.

[YO10] Zhirong Yang and Erkki Oja. Linear and nonlinear projective nonnegative matrix factorization. *IEEE Transactions on Neural Networks*, 21:734–749, 2010.

[YYO09] Zhijian Yuan, Zhirong Yang, and Erkki Oja. Projective nonnegative matrix factorization : Sparseness , orthogonality , and clustering. 2009.

[YZ13] Xiao-Tong Yuan and Tong Zhang. Truncated power method for sparse eigenvalue problems. *Journal of Machine Learning Research*, 14(4), 2013.

[ZHT06] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.