k-PCA for (Non-Squared) Euclidean Distances: Deterministic Polynomial Time Approximation

Daniel Greenhut
University of Haifa

Dan Feldman
University of Haifa

Abstract

Given an integer $k \geq 1$ and a set P of n points in $\mathbb{R}^d$, the classic k-PCA (Principal Component Analysis) approximates the affine *k-subspace mean* of P, which is the k-dimensional affine linear subspace that minimizes its sum of squared Euclidean distances ($\ell_{2,2}$-norm) over the points of P, i.e., the mean of these distances. The *k-subspace median* is the subspace that minimizes its sum of (non-squared) Euclidean distances ($\ell_{2,1}$-mixed norm), i.e., their median. The median subspace is usually more sparse and robust to noise/outliers than the mean, but also much harder to approximate since, unlike the $\ell_{z,z}$ (non-mixed) norms, it is non-convex for $k < d-1$.

We provide the first polynomial-time deterministic algorithm whose both running time and approximation factor are not exponential in k. More precisely, the multiplicative approximation factor is $\sqrt{d}$, and the running time is polynomial in the size of the input. We expect that our technique would be useful for many other related problems, such as $\ell_{2,z}$ norm of distances for $z \notin \{1,2\}$, e.g., $z = \infty$, and handling outliers/sparsity.

Open code and experimental results on real-world datasets are also provided.

1 INTRODUCTION

In the k-subspace problem, the input is a set P of $n > d$ points in $\mathbb{R}^d$, the d-dimensional Euclidean space, along with an integer $k \in [1, d-1]$. The objective is to find a k-dimensional linear subspace (k-subspace, for short) $S \subseteq \mathbb{R}^d$ that closely represents these points according to a given loss function. This is a fundamental problem in mathematics and computational geometry with numerous applications; see surveys, e.g., in [Kim et al., 2014, Ta and Dellaert, 2014, Zhang et al., 2014, Cao et al., 2016, Kanji, 2016, Bhutta et al., 2020, Wendel and Underwood, 2016].

A natural loss function is to compute the k-subspace S (or its corresponding k-rank projection matrix Π) that minimizes its sum over each Euclidean distance $\text{dist}(p,S) := \min_{s \in S} \|p-s\|_2$ from the n input points,

$$\sum_{p \in P} \text{dist}^z(p,S) = \left\| \left(\left\| p - XX^T p \right\|_2 \right)_{p \in P} \right\|_z . \quad (1)$$

That is, the minimum is over all matrices $X \in \mathbb{R}^{d \times k}$ whose columns are mutually orthogonal unit vectors, i.e., $X^T X = I$. Here, $\Pi := XX^T \in \mathbb{R}^{d \times d}$ is the corresponding projection matrix of rank k.

For simplicity of notation, we assume that $P := (p_i)_{i=1}^n$ is an n-tuple, although the order is arbitrary. By letting $A \in \mathbb{R}^{n \times d}$ denote the matrix whose ith row is $A_{i*} := p_i$, the k-subspace median corresponds to a matrix B that minimizes

$$\|A - B\|_{2,z} := \left\| (\|A_{i*} - B_{i*}\|_2)_{i=1}^n \right\|_z , \quad (2)$$

over every k-rank matrix $B \in \mathbb{R}^{n \times d}$, where B_{i*} denotes its ith row for $i \in \{1, \cdots, n\}$. In (2) we essentially want to approximate the matrix A by a matrix B of the same size, but of rank only k. See Section A or, e.g., [Song et al., 2017, Li and Woodruff, 2016].

As discussed in Section 2, only a limited number of solvers exist for the k-subspace median problem. Most of these methods are randomized and, more critically, their running time scales exponentially with k. Such complexity is impractical for contemporary large-scale applications, particularly in the present era of data-intensive AI. Modern models often contain billions of parameters, and the corresponding value of k increases in tandem. This growth arises because both the orig-

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

inal data dimension and the dimension of the learned embedding space are very high.

Motivated by these considerations, we concentrate in this paper on the k-subspace median (kSM) problem, which corresponds to the mixed norm $\ell_{q,z} := \ell_{2,1}$. Our framework, however, is designed to naturally generalize beyond this special case, and we anticipate that it can be extended with minimal modifications to any fixed constants $q, z > 1$.

Why it is hard? Because the derivative of a quadratic objective is linear, we can treat the k-subspace mean problem (for $z = 2$) using linear algebraic tools, in close analogy to PCA and SVD. In the special case $k = d-1$, the induced optimization problem is nonlinear but remains convex, since it reduces to minimizing $\|Ax\|_2$ subject to the spherical constraint $\|x\|_2 = 1$, that is, searching over the boundary of an ellipsoid.

More generally, the $\ell_{z,z}$ norm (commonly just called the ℓ_z norm) yields a convex objective for $k < d-1$ as well, because one can swap the order of summation over rows and columns [Dasgupta et al., 2009], preserving convexity. In contrast, when we consider the mixed norm $\ell_{2,1}$ and still require $k < d-1$, convexity is lost. The difficulty arises from the non-convexity of the rank constraint inherent in this formulation, which likely accounts for the scarcity of efficient and robust solvers for this problem in both the theoretical and applied literature.

The natural open problem that we answer affirmatively in this work is thus as follows.

Is there a deterministic polynomial-time algorithm that computes the k-subspace median of a given set of points, up to an approximation factor that is not exponential in k?

In particular, we aim to approximation factors that are smaller than k, even when $k \in \Omega(d)$.

2 RELATED WORK

In this paper, we aim to provide deterministic algorithms for the case $z = 1$ and non-fixed k, such as $k \in \Omega(d)$, but give more general results in this section.

Affine subspace (flat). A straightforward extension for the k-subspace median is the *k-flat median* that minimizes the sum of distances among every *affine* k-subspace of $\mathbb{R}^d$. However, there are simple reductions to the (non-affine) linear case; see e.g. [Maalouf et al., 2020] and Section A.

The case $k = 0$. In this case, the only 0-subspace is the origin, but the 0-flat median is the median point, also called the *Fermat-Weber Point* [Hansen and Thisse, 1983], which minimizes the sum of Euclidean distances to the n input points. This is a convex problem that cannot be solved exactly [Cockayne and Melzak, 1969], but a $(1+\varepsilon)$-approximation can be computed in $O\left(nd \log^3 \frac{1}{\varepsilon} \log \frac{1}{\delta}\right)$ time with a probability of at least $1-\delta$ [Cohen et al., 2016].

The case $k = d-1$. In this case, the required subspace is a hyperplane, the orthogonal vector to this hyperplane is a vector, and the problem is convex since $\|Ax\|_{2,1}$ is a mixed norm of x, which is a convex function. Hence, techniques such as the Ellipsoid method might give an $O(d)$-approximation in polynomial time.

The case $k \in \{1, \ldots, d-2\}$. Here, the k-subspace median problem is neither convex nor concave. We could not find any deterministic approximation algorithms for the case where k (and thus also $d > k$) is large; i.e., part of the input for the subspace-median problem.

In their paper, Shyamalkumar and Varadarajan [Shyamalkumar and Varadarajan, 2007] suggested a randomized algorithm to compute an $(1+\varepsilon)$-approximation for the k-subspace median that takes time $O\left(\frac{ndk}{\varepsilon} \ln \frac{k}{\varepsilon}\right) = nd\left(\frac{k}{\varepsilon}\right)^{O(1)}$, with a probability of success $1/2^{O\left(\frac{k^2}{\varepsilon} \ln^2 \frac{k}{\varepsilon}\right)} = 1/2^{(k/\varepsilon)^{O(1)}}$. Hence, to obtain a probability of success larger than, say, $\frac{1}{2}$, using amplification [HromkoviČ, 2005], we need to run the algorithm a number of times that is exponential in both k and $\frac{1}{\varepsilon}$.

Later, Deshpande and Varadarajan [Deshpande and Varadarajan, 2007] refined this result, showing that it is possible in $nd\left(\frac{k}{\varepsilon}\right)^{O(1)}$ time to produce a subset of $r = \left(\frac{k}{\varepsilon}\right)^{O(1)}$ points, whose span contains a $(1+\varepsilon)$-approximation to the k-subspace median. By projecting the n input points onto the span of these r points, one can find the k-subspace median in time exponential in the smaller dimension r.

Feldman et al. [Feldman et al., 2010] suggested a randomized $(1+\varepsilon)$ approximation that runs in

$$nd\left(\frac{k}{\varepsilon}\right)^{O(1)} + (n+d)\exp\left(\left(\frac{k}{\varepsilon}\right)^{O(1)}\right).$$

where $\exp(x) := e^x$, by applying the algorithm of Shyamalkumar and Varadarajan on a coreset that will be defined later in this section. Later, Clarkson and Woodruff [Clarkson and Woodruff, 2015] im-

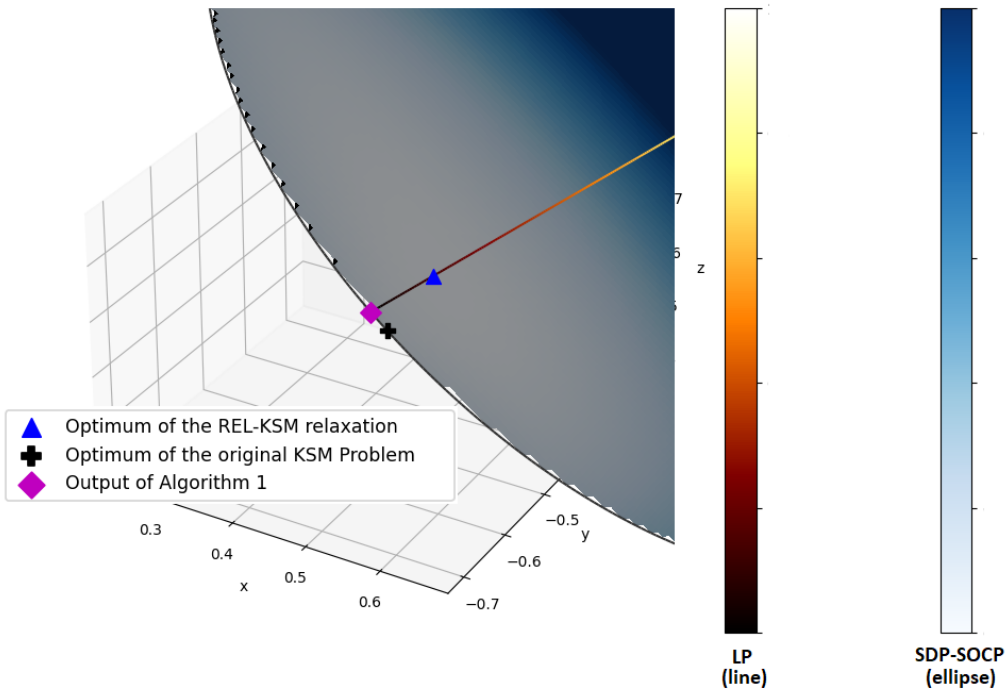


Figure 1: An illustration of the search space in Algorithm 1 for the case $d = 2$ and $k = 1$ (line-median in the plane). Every point is a matrix in $\mathbb{R}^{2\times 2}$. The gray ellipsoid consists of $\mathbb{S}_2^+ \subseteq \mathbb{R}^{2\times 2}$, which is the search space of Problem (7), and its (non-convex) boundary consists of those matrices whose rank is $d - k = 1$ (projection matrices), which is the search space of the original kSM problem (5). The values of the function of the mixed SDP-SOCP optimization problem (7) are manifested in the blue bar, while the red bar manifests the cost function of the Linear Programming (10). The projection matrix of the (optimal) 1-subspace median is the black cross, $X^* \in \mathbb{S}_2^+$ (blue triangle) minimizes the relaxed problem (7), and the pink square denote its "closest" projection matrix E, which is the output of the algorithm.

proved the running time to

$$O\left(\operatorname{nnz}(P)\right) + (n+d)\left(\frac{k}{\varepsilon}\right)^{O(1)} + \exp\left(\left(\frac{k}{\varepsilon}\right)^{O(1)}\right),$$

where $\operatorname{nnz}(P) \leq nd$ is the number of non-zero entries in the input set.

Because for any $x \in \mathbb{R}^m$ we have $\frac{1}{\sqrt{m}}\|x\|_1 \leq \|x\|_2 \leq \|x\|_1$, the k-dimensional subspace that minimizes the sum of squared Euclidean distances (k-PCA) provides a $\sqrt{n}$-approximation to the k-subspace median. Consequently, applying a data reduction from n to m points for the k-subspace median problem (see Section C), incurring a multiplicative loss of at most $1+\varepsilon$, results in a $(1+\varepsilon)\sqrt{m}$-approximation to the k-subspace median. The randomized algorithm for such a subset of $m \in \tilde{O}(k/\varepsilon^4)$ weighted input points ("strong coreset"), where $\tilde{O}$ hides factors that are poly-logarithmic in n, was recently suggested by Woodruff and Yasuda [Woodruff and Yasuda, 2024], following earlier results [Feldman and Langberg, 2011, Varadarajan and Xiao, 2012, Huang and Vishnoi, 2020], and a deterministic construction whose time is exponential in k by Woodruff and Sohler in [Sohler and Woodruff, 2018]; see [Woodruff and Yasuda, 2024] for details. In addition to randomness and the factor of m that may be larger than d, it is not clear when the algorithm succeeds, so it is not clear how to achieve a bounded expected running time or the Las-Vegas algorithm [Hoos and Stützle, 1998]. This is a known disadvantage of the randomized coreset construction: it is unclear how to verify that the construction succeeded.

$\ell_{z,z}$ norm. As stated in (9), the subspace median aims to compute the k-rank matrix B that minimizes $\|A-B\|_{2,1}$ for a given matrix $A \in \mathbb{R}^{n\times d}$. For the case $\|A-B\|_{1,1}$, Markopoulos et al. [Markopoulos et al., 2014] showed an algorithm that minimizes the sum of ℓ_1 norms from the points to the k-subspace, a $\sqrt{d}$-approximation to kSM. However, their running time is exponential in the dimension d. Randomized algorithms for polynomial

time approximation for the ℓ_1 norm were suggested in [Chierichetti et al., 2017, Song et al., 2017].

For the problem of minimizing $\|A - B\|_{p,p}$, we refer to the results of Dasgupta et al. [Dasgupta et al., 2009], who provide polynomial-time algorithms that achieve polynomial-factor approximations. Extensions to other matrix norms, including Schatten norms and the spectral norm, have been investigated in subsequent work, notably by Bakshi et al. [Bakshi et al., 2022] and by Fack and Kosaki [Fack and Kosaki, 1986], where related approximation and structural results are developed.

Software. Although only a handful of algorithms for the k-subspace median come with provable guarantees, we anticipated finding more available software and heuristic methods; however, this turned out not to be the case. Most of the packages we identified address restricted forms of the $\ell_{2,2}$ norm, for instance with regularization, or focus on the $\ell_{1,1}$ norm; see the references in [Markopoulos et al., 2017]. For example, we attempted to run our experiments using the new $\ell_{1,1}$ toolbox [Markopoulos, 2025], but the computations did not finish even after several nights, except when either the number n of points or the subspace dimension k was very small.

3 OUR CONTRIBUTION

This paper resolves the open problem stated in Section 1 in the positive. Specifically, we describe a deterministic algorithm that, for any given set of points, computes an approximate k-subspace median and provides an approximation guarantee that is independent of both n and k. In fact, the approximation factor is a multiplicative $\sqrt{d}$, which remains sublinear in k even in the regime where $k = \Omega(d)$. Furthermore, the running time is fully polynomial in all relevant parameters—namely n, d, and any choice of $k \geq 1$. A detailed and formal statement of these guarantees is provided in Corollary 5.2.1.

The running time can be accelerated by applying the proposed algorithm to one of the suitable coresets or sketches from Section C, yielding a linear or near-linear dependence on n, at the cost of turning our deterministic algorithm into a randomized one. Likewise, the dimensionality reduction methods from the previous section allow us to replace our $\sqrt{d}$ approximation factor by $k^{O(1)}$, although this factor becomes larger in regimes such as $k = \Omega(d)$.

Finally, we expect that our simple technique would hold for other mixed $\ell_{2,z}$ norms such as the k-subspace center ($z = \infty$), but leave this for future work.

Implementations and experiments. As discussed in the previous section, in contrast to the situation for PCA/SVD and their approximations, we were unable to locate any implementations—either for existing provable algorithms or for empirical heuristics—for computing kSM or even its close variants. Consequently, we developed and implemented these methods ourselves, alongside our new algorithm, and released all of them as open-source code. A detailed experimental evaluation, demonstrating substantial improvements in running time, approximation quality, or both, is presented in Section 6.

In contrast to many related theoretical works, we devote considerable effort to describing how to implement our algorithm efficiently and to providing open, fully reproducible code and experiments. To the best of our knowledge, this constitutes the first publicly available practical implementation for this problem. Given the widespread use of PCA in data analysis and machine learning, we anticipate that our implementation will see substantial adoption in practice.

Novelty. Existing approaches for minimizing mixed $\ell_{z,z}$ norms largely rely on the fact that these two norms coincide, which allows one to interchange summation of errors (i.e., entries of the distance matrix) across columns and rows; see, for instance, [Dasgupta et al., 2009]. Under this assumption, the problem reduces to the convex formulation $\|Ax\|_z$, as discussed in the previous section. In contrast, we were unable to identify a way to extend this methodology to $\ell_{2,z}$ norms when $z \neq 2$ (as arises in k-PCA/SVD).

To address these challenges, we propose a novel optimization approach, illustrated in Fig. 1. Our method differs from much of the existing literature in that the optimization is performed over the space of symmetric $d \times d$ matrices rather than directly in the original space $\mathbb{R}^d$ of d-dimensional vectors. Concretely, for $d = 2$, the search is carried out over symmetric matrices of the form $\begin{pmatrix} x & y \\ y & z \end{pmatrix}$ instead of over points in $\mathbb{R}^2$.

A second key component of our method is a strategy for addressing the inherent difficulty of the k-subspace median problem. This challenge arises because the feasible set of projection matrices is neither convex nor concave, which precludes the use of many standard convex optimization techniques. Geometrically, this nonconvexity manifests in the shape of the admissible region, as illustrated by the boundary of the ellipsoid in Fig. 1.

To address this issue, we instead solved the related convex optimization problem over the convex hull of the feasible set (the filled ellipsoid in Fig. 1). While the

true optimum does not lie on this boundary, we show that a close approximation of the optimal solution is feasible; see △ in Fig. 1.

Finally, we explain how to convert this infeasible solution into a feasible one; see △ and ◇ in Fig. 1, respectively. This is done by carefully constructing a subspace of the matrix space that passes through the infeasible point and intersects the feasible region (ellipsoid) in such a way that the resulting point is both feasible and still a provably good approximation. Equivalently, we approximate a general PSD matrix by projecting it onto the nearest rank-k matrix with respect to the sum of (non-squared) distances.

This nontrivial projection, whose specifics are presented in Section 11, can be viewed as an adaptation of the classical Eckart–Young–Mirsky [Wang, 2015] and Courant–Fischer [Ikebe et al., 1987] theorems for approximating a matrix by a k-rank matrix. In our setting, however, the Frobenius norm is replaced by the mixed $\ell_{2,1}$ norm, and instead of simply retaining the top k singular values, we employ a more sophisticated procedure. Although in Section F we apply the standard central-path method to solve our SDP over the second-order cone, an additional difficulty lies in constructing an initialization matrix that, as shown in Lemma E.3, is provably close to the optimum.

We anticipate that this approach can be extended to other mixed norms and to convex objectives defined over rank-k matrices.

4 NOTATIONS AND DEFINITIONS

Throughout this work, we assume that we are given integers $d \geq 2$ and $n \geq d$ that represent the dimension and cardinality of the input set of points, respectively. We define $\mathbb{R}^d$ as the set of real d-dimensional column vectors, where a single column is denoted by $x := (x_1, \cdots, x_d) \in \mathbb{R}^d$. The set of all $n \times d$ real matrices is $\mathbb{R}^{n \times d}$. The determinant of a matrix $X \in \mathbb{R}^{n \times n}$ is $|X| := \det X$. For $\Delta > 1$, we define $\mathbb{R}_\Delta = \left\{ \frac{n}{\Delta} \in \mathbb{R} : n \in \mathbb{Z}, -\Delta^2 \leq n \leq \Delta^2, n \neq 0 \right\}$.

We define $\mathbb{1}_n := (1, \cdots, 1)$ and $\mathbb{0}_n := (0, \cdots, 0)$ to be the n-dimensional column vectors of ones and zeros, respectively, and $\mathbb{0}_{n \times n}$ to be the $n \times n$ zero matrix. The $d \times d$ identity matrix is denoted by I_d, or I if d is clear from the context.

The rank and trace of a matrix A are denoted by $\operatorname{rank}(A)$ and $\operatorname{trace}(A)$, respectively. Furthermore, we define $[n] := \{1, \ldots, n\}$, $\mathbb{R}_+ := \{x \in \mathbb{R} | x \geq 0\}$, $\mathbb{R}_{++} := \{x \in \mathbb{R} | x > 0\}$. For a set $\{k_i\}_{i=1}^n$ of positive integers and $v_i \in \mathbb{R}^{k_1}$, we define $v := (v_1 \mid \ldots \mid v_n) \in \mathbb{R}^{\sum_{i=1}^n k_i}$ as the column vector of their concatenation. For a vector $v \in \mathbb{R}^n$, we define $\operatorname{diag}(v) \in \mathbb{R}^{n \times n}$ as the diagonal matrix whose diagonal is v, and for a matrix $X \in \mathbb{R}^{n \times n}$ we denote by $\operatorname{diag}(X) \in \mathbb{R}^n$ the vector of the values on the diagonal of X.

The running time it takes to invert a matrix in $\mathbb{R}^{n \times n}$ is denoted by $O(n^\omega)$.

4.1 Convex Cones

The *positive semi-definite (PSD) cone* is denoted by

$$\mathbb{S}_d^+ := \{X \in \mathbb{R}^{d \times d} \mid X^T = X, \quad \forall v \in \mathbb{R}^d, v^T X v \geq 0\}, \tag{3}$$

its interior is

$$\mathbb{S}_d^{++} := \left\{X \in \mathbb{R}^{d \times d} \middle| X^T = X,\ \forall v \in \mathbb{R}^d, v^T X v > 0\right\},$$

as proved in [Boyd et al., 2004]. The *second order cone* is denoted by

$$\mathbb{Q}_d^+ := \left\{(v, t) \in \mathbb{R}^d \times [0, \infty) \mid \|v\|_2 \leq t\right\}, \tag{4}$$

and its interior, as explained in [Boyd et al., 2004] is

$$\mathbb{Q}_d^{++} := \left\{(v, t) \in \mathbb{R}^d \times [0, \infty) \mid \|v\|_2 < t\right\}.$$

For every pair $X, Y \in \mathbb{S}_d^+$, we define the partial order

$$X \preceq Y \iff Y - X \in \mathbb{S}^+.$$

4.2 k-Subspace Median

Every matrix $Y \in \mathbb{S}_d^+$ has an *eigen decomposition* $Y = VDV^T$ such that $V^T V = I$, and $D \in \mathbb{S}_d^+$ is a diagonal matrix; see [Golub and Van Loan, 1996]. If $D \in \{0, 1\}^{d \times d}$ is a binary diagonal matrix, i.e., it has $k \geq 0$ diagonal entries of ones and $(d - k)$ zeros, then Y is called a k-rank *projection matrix* and satisfies $Y^2 = Y$. Its *corresponding k-linear subspace* (or k-*subspace*, for brevity) is the column space of Y (or V).

The k-*median subspace* (or kSM, for short) of an n-tuple $P = (p_i)_{i=1}^n$ of n points in $\mathbb{R}^d$, minimizes the sum of Euclidean distances to its points, over every k-subspace of $\mathbb{R}^d$. It is the null space of a $(d-k)$-rank projection matrix X, which is an optimal solution to the following problem:

$$\begin{aligned} \min_{X \in \mathbb{R}^{d \times d}} \quad & \sum_{i=1}^n \|X p_i\|_2 \\ s.t. \quad & X^2 = X^T = X \\ & \operatorname{rank}(X) = d - k. \end{aligned} \tag{5}$$

We denote its optimal solution by $\operatorname{ksm}(P, k)$. More formally,

$$\operatorname{ksm} : \left((\mathbb{R}^d)^n \times [d-1]\right) \to \mathbb{R}_+$$

is the function that maps every pair $(P, k) \in (\mathbb{R}^d)^n \times [d-1]$ to the sum of Euclidean distances to a k-subspace median of the n-tuple P of points.

For a given $\alpha \geq 1$, a k-subspace $S \subseteq \mathbb{R}^d$ is an α-*approximation* to the kSM of P, if its sum of Euclidean distances to the points is the same as $\text{ksm}(P, k)$, up to a multiplicative factor of α. Formally,

$$\sum_{i=1}^{n} \text{dist}(p_i, S) \leq \alpha \text{ksm}(P, k). \tag{6}$$

5 MAIN ALGORITHM

This section introduces our main approximation algorithm. The input is an n-tuple $P = (p_i)_{i=1}^n$ of points in $\mathbb{R}^d$ together with an integer parameter $k \in [d-1]$, and the output is a projection matrix of rank $(d-k)$ whose null space is a k-dimensional subspace of $\mathbb{R}^d$. We establish that this algorithm achieves a $\sqrt{d}$-approximation for $\text{ksm}(P, k)$, meaning that it produces a k-subspace whose sum of Euclidean distances to the points in P is within a multiplicative factor of $\sqrt{d}$ of the optimal value, as formalized in (6).

A detailed proof of the algorithm's correctness is provided in Section D, and Section E contains a detailed examination of its computational complexity.

The main technical result is a relaxation of the kSM problem to the following mixed SDP-SOCP minimization problem, with the same input P and k:

$$\begin{aligned} \min_{\substack{X \in \mathbb{R}^{d \times d} \\ y \in \mathbb{R}^n}} \quad & \mathbb{1}_n^T y \\ s.t. \quad & X^T = X \\ & \text{trace}(X) = d - k \\ & \mathbb{0}_{d \times d} \preceq X \preceq I_d \\ & \|X p_i\|_2 \leq y_i \quad \forall i \in [n]. \end{aligned} \tag{7}$$

The optimal solution of this problem is denoted by relaxksm(P, k). That is, the function

$$\text{relaxksm} : (\mathbb{R}^d)^n \times [d-1] \to \mathbb{R}_+, \tag{8}$$

maps every pair $(P, k) \in (\mathbb{R}^d)^n \times [d-1]$ to the minimizers of Problem (7).

To demonstrate the intuition behind Algorithm 1, consider the simplest case where the input points are on the plane ($d = 2$), and the 1-subspace median is a line through the origin ($k = 1$). Each 3-dimensional point (x, y, z) in Fig. 1 represents a symmetric matrix $X := \begin{pmatrix} x & y \\ y & z \end{pmatrix}$ in $\mathbb{R}^{2\times 2}$. The search space of Algorithm 1 is the union over every symmetric matrix, and

Algorithm 1: kSM-Approx(P, k); see Theorem 5.1.

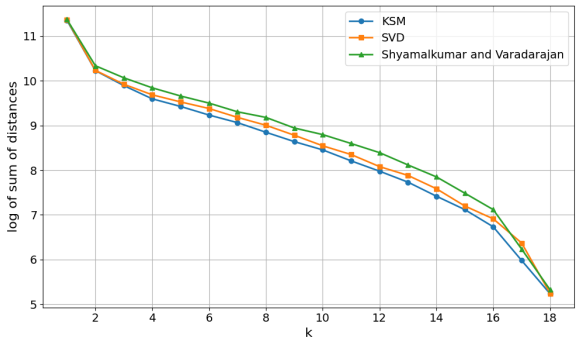
Input : A set P of n points in $\mathbb{R}^d$, and an integer $k \geq 1$.
Output: A k-dimensional linear subspace $S \subseteq \mathbb{R}^d$

1 Compute a minimizer $(X^*, y^*) \in \mathbb{S}_d^+ \times \mathbb{R}^n$ of the convex problem (7).
// See Proposition E.2 and Lemma E.3.
2 Compute the eigen decomposition $X^* = VDV^T$ of X^*.
// i.e., D is diagonal and $V^TV = I_d$.
3 **for** *every* $i \in [n]$ **do**
4 Set $v^{(i)} \leftarrow Vp_i$
5 Let $q \in [0, \infty)^d$ be the vector whose jth entry is $q_j := \sum_{i=1}^n \left| v_j^{(i)} \right|$, for every $j \in [d]$.
// $v_j^{(i)}$ is the jth entry of $v^{(i)}$
6 Set $\zeta \leftarrow \mathbb{0}_d$
7 Let Ind $\subseteq [d]$ denote the indices of the $d-k$ smallest entries of q; ties broken arbitrarily.
8 Set $\zeta \in \{0, 1\}^d$ such that $\zeta_i = 1$ if and only if $i \in$ Ind
9 Set $E \leftarrow V^T \text{diag}(\zeta) V$
10 Let $S \subseteq \mathbb{R}^d$ denote the corresponding k-subspace of the projection matrix $I - E$.
// S is the column space of $(I-E)V^T$.
11 **return** S

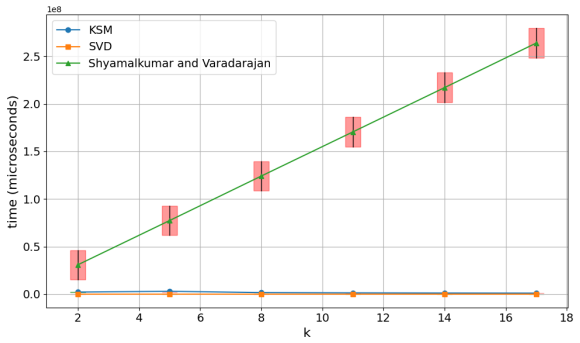
is denoted by the gray ellipsoid in Fig. 1. In contrast, each point on the boundary of the ellipsoid consists of a $(d-k)$-rank projection matrix, which is the search space of the original kSM problem in (5). The hardness of solving the kSM problem arises from the non-convexity of the feasible set in Problem (5). This feasible set is the set of $(d-k)$-rank matrices that corresponds to the non-convex boundary of the (convex) ellipsoid, which is an empty ellipsoid.

Instead of solving the NP-hard kSM problem in (5) directly, and computing the optimal point in the empty (non-convex) ellipsoid, which is denoted by a black cross in Fig. 1, we adopt an alternative approach. We compute the optimal solution X^* over the (full, convex) ellipsoid, which is denoted by a blue triangle in the figure. This is the minimizer of Problem (7), which is solved in Line 1 of Algorithm 1. This problem is a convex mixed SDP-SOCP relaxation of Problem (5), and returns a 3-dimensional (more generally, $\frac{d(d+1)}{2}$-dimensional) point $X^* \in \mathbb{S}_2^+$ in the space of positive semi-definite 2×2 matrices inside the ellipsoid (the blue triangle).

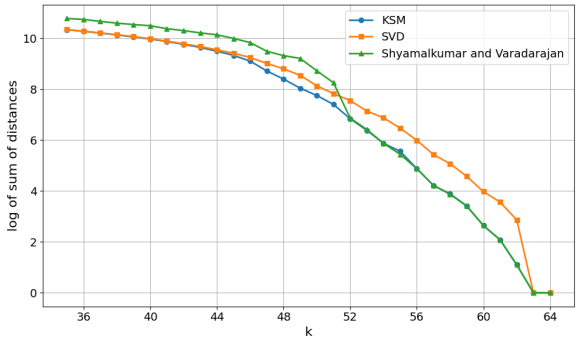
Although the minimizer (matrix) X^* obtained from Problem (7) provides an approximation to the objec-



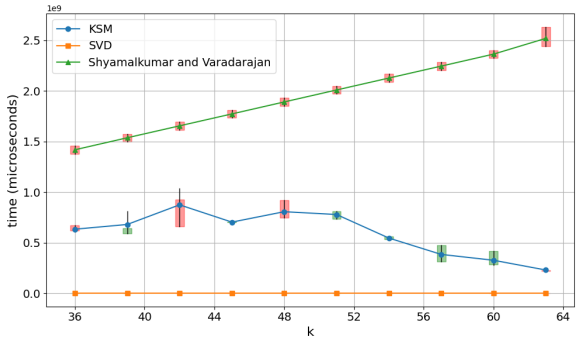
(a) Log of sum of distances from "Statlog (Vehicle Silhouettes)"



(b) Running time on "Statlog (Vehicle Silhouettes)"



(c) Log of sum of distances from "Optical Recognition of Handwritten Digits"



(d) Running time on "Optical Recognition of Handwritten Digits"

Figure 2: Comparison of Algorithm 1 with SVD and the method by Shyamalkumar and Varadarajan on UC Irvine datasets. Figures 2a and 2c: Log of the sum of distances from the data points to the subspace. Figures 2b and 2d: Computation time in microseconds, shown with Japanese candlesticks and a batch size of 3.

tive function of the original problem (5), it does not constitute a feasible solution to that problem. Specifically, the matrix X^* lies strictly inside the ellipsoid, and therefore belongs to $\mathbb{S}_2^+$, but it does not, in general, have rank $(d-k)$ and thus fails to be a projection matrix, which would require it to lie on the boundary of the ellipsoid. To convert X^* into a projective matrix, we map it onto the ellipsoid's boundary by performing a suitable projection operation. Geometrically, this step corresponds to the pink triangle in Fig. 1, representing the image of the blue triangle after being projected onto the (initially) empty ellipsoid.

In the algorithm, this projection is obtained in Line 2, where we perform the eigen-decomposition $X^* = V^T D V$. Because X^* is PSD, all eigenvalues in the diagonal matrix D are non-negative, whereas a projection matrix is characterized by having eigenvalues only in $\{0,1\}$. In Lines 3–9, we therefore construct, in an appropriate sense, the $(d-k)$-dimensional projection matrix that is "closest" to X^*. To this end, we carefully "round" each diagonal entry of D to either 0 or 1. This yields a diagonal matrix $\operatorname{diag}(\zeta) \in \{0,1\}^{2\times 2}$, and Line 7 enforces that exactly $d-k$ of these diagonal entries are equal to 1.

Once the diagonal entries of $\operatorname{diag}(D)$ are rounded to a binary vector ζ, the matrix $E := V^T \operatorname{diag}(\zeta) V$ becomes a projection onto a k-dimensional subspace generated by some subset of k columns of V. However, this projected matrix is no longer an optimal solution to Problem (7). Consequently, it is necessary to design a rounding procedure that provides a rigorous guarantee on the approximation ratio of E relative to the optimal solution X^*.

Without loss of generality, we may assume $V = I_d$; if not, we can simply rotate the coordinate system. Under this assumption, the nonzero coordinates of ζ select a subset of k out of the d axes, and these axes span the output k-dimensional subspace. In Line 4, we calculate the projection $v_j^{(i)}$ of the ith input point onto the jth axis. In Line 4, q_j is defined as the sum of these projections. Geometrically, q_j represents the sum of distances of the input points to the hyperplane orthogonal to the jth axis. This should be contrasted

with the matrix D, whose diagonal entries give the sum of *squared* distances to each corresponding hyperplane. Thus, q_j should not be mistaken for $\sqrt{D_j}$, which is instead the square root of the sum of squared distances to those hyperplanes. Because the k-subspace median is defined as the subspace that minimizes the sum of distances to the data points, it is natural that in Line 7 we select as the output the subspace spanned by the k axes. These k axes are those that minimize this total distance.

The correctness of the Algorithm 1 is formalized in Theorem 5.1.

Theorem 5.1 (Correctness of the Algorithm 1)**.** *Let $k \in [d-1]$ be an integer and $P = (p_i)_{i=1}^n \in (\mathbb{R}^d)^n$. Let $S \subseteq \mathbb{R}^{d\times d}$ be the output of a call to kSM-Approx(P,k); see Algorithm 1. Then S is a $\sqrt{d}$-approximation to the k-subspace median of P. That is,*

$$\sum_{i=1}^{n} \operatorname{dist}(p_i, S) \leq \sqrt{d} \cdot \operatorname{ksm}(P,k).$$

Proof. See Theorem D.2. □

Theorem 5.2 proves that Algorithm 1 is fully polynomial in all its parameters n, d and any $k \geq 1$.

Theorem 5.2 (running time)**.** *Let $k \in [d-1]$ and $\Delta \geq 1$ be integers, and $P := (p_i)_{i=1}^n \subseteq \mathbb{R}^d_\Delta$. Let $S \subseteq \mathbb{R}^d$ be the output of a call to kSM-Approx(P,k); see Algorithm 1. Then the projection matrix of S can be computed in time*

$$O(n+d^2)^\omega (n+d) \ln\big(\Delta \cdot (n+2d)\big).$$

Proof. See Theorem E.4. □

Our main result then follows by combining Theorem 5.2 and Theorem 5.1 as follows.

Corollary 5.2.1 (k-subspace median)**.** *Let $k \in [d-1]$ and $\Delta \geq 1$ be integers, and $P := (p_i)_{i=1}^n$ be an n-tuple of n points in $\mathbb{R}^d_\Delta$. Let $S \subseteq \mathbb{R}^{d\times d}$ be the output of a call to kSM-Approx(P,k); see Algorithm 1. Then S is a $\sqrt{d}$-approximation to the k-subspace median of P. Moreover, the projection matrix of S can be computed in $(nd \ln \Delta)^{O(1)}$ time, i.e., polynomial in the size of the input P and k.*

6 EXPERIMENTAL RESULTS

We evaluated our dimensionality reduction approach on two benchmark datasets from the UC Irvine repository: the "Statlog (Vehicle Silhouettes)" dataset [Mowforth and Shepherd,], which contains $n = 847$ samples in $d = 19$ dimensions, and the "Optical Recognition of Handwritten Digits" dataset [Alpaydin and Kaynak, 1998], consisting of $n = 5621$ samples in $d = 65$ dimensions. Following the perspective of Siebert [Siebert, 1987] and Kaynak and Cenk [Kaynak, 1995], dimensionality reduction is regarded as a key step in designing effective classifiers for high-dimensional feature spaces, both to mitigate the curse of dimensionality and to improve generalization. All computational experiments were executed on an AWS EC2 r5b.4xlarge instance equipped with 16 vCPUs and 128 GiB of RAM, ensuring sufficient resources for handling the datasets and repeated runs of the algorithms.

In each experiment, we benchmark Algorithm 1 against the classical SVD algorithm, as well as against 100 iterations of the approach proposed by Shyamalkumar and Varadarajan; see [Golub and Van Loan, 1996] and [Shyamalkumar and Varadarajan, 2007]. We also made an effort to incorporate the L1PCA algorithm [Markopoulos et al., 2014] into our comparisons; however, in practice, the publicly available implementations [ktountas, 2021] could not be executed at the required scale for our datasets and were therefore excluded from the experiments.

In Fig. 2, the X-axis shows k, the dimension of the subspace. Figures 2a and 2c show the log of the sum of distances from the data points to the subspace, while figures 2b and 2d show the computation time in microseconds, with Japanese candlesticks and a batch size of 3.

Our proposed algorithm demonstrates superior performance compared to both the SVD and the method introduced by Shyamalkumar and Varadarajan. As illustrated in Fig. 2a, Algorithm 1 achieves distance sums that are reduced by a factor of up to 1.5 when compared to the SVD, and by a factor of up to 1.55 when compared to Shyamalkumar's Algorithm. Furthermore, in Fig. 2c, Algorithm 1 presents sums that are diminished by up to 8.2 times relative to the SVD and by up to 3.2 times in relation to Shyamalkumar's Algorithm. Only for $k = 55$, the method proposed by Shyamalkumar and Varadarajan achieves distance sums that are reduced by a factor of up to 1.13 compared to our algorithm. In summary, Algorithm 1 delivers mainly consistently superior results over both the SVD and a single iteration of the method proposed by Shyamalkumar and Varadarajan.

Figures 2b and 2d report the computational run-time results. While our proposed method is slower than the SVD-based approach, its execution time remains practically acceptable. In contrast, the algorithm of Shyamalkumar and Varadarajan requires substan-

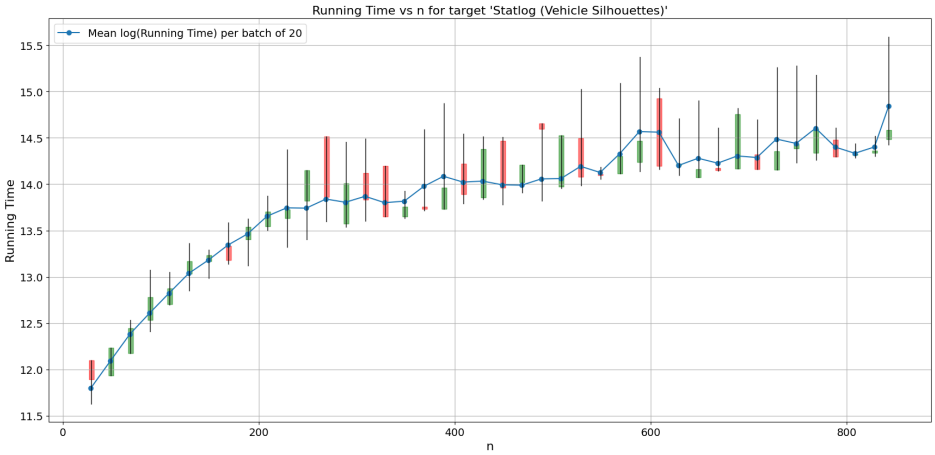


Figure 3: Logarithmic running time of Algorithm 1 on 'Statlog (Vehicle Silhouettes)' as the number of points increases (fixed $d = 19$, $k = 9$). The Japanese candlesticks are for 20-point batches.

tially more time than Algorithm 1, even when restricted to just 100 evaluations. Achieving an $(1+\varepsilon)$-approximate solution with their method demands an exponential number of iterations, which renders it unsuitable for large-scale datasets.

For Algorithm 1, the run-time shows a slight decrease as k grows. This behavior is likely linked to the algorithm's sensitivity to the initialization of the mixed SOCP-SDP formulation in (7), where different values of k may yield more favorable starting points. A more detailed discussion of these computational aspects, including formal complexity guarantees and a thorough analysis of the run-time behavior, is provided in Section E and Theorem 5.2.

To empirically substantiate the findings from Section E, we conducted a runtime study on the "Statlog (Vehicle Silhouettes)" dataset, varying the number of data points while keeping $d = 19$ and $k = 9$ fixed. The outcomes are depicted in Fig. 3, where the horizontal axis corresponds to the number of points and the vertical axis reports the logarithm of the running time of Algorithm 1 (measured in microseconds). The resulting plot closely follows a logarithmic trend, which is consistent with a polynomial dependence on n and thus empirically supports the claim that Algorithm 1 operates in polynomial time with respect to n, in agreement with Theorem 5.2.

Acknowledgements

We thank the reviewers for their useful and inspiring comments and our colleagues who contributed to the ideas.

This work was supported by the Center for Cyber Law & Policy at the University of Haifa in conjunction with the Israel National Cyber Directorate in the Prime Minister's Office.

References

[Alpaydin and Kaynak, 1998] Alpaydin, E. and Kaynak, C. (1998). Optical Recognition of Handwritten Digits. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C50P49.

[Bakshi et al., 2022] Bakshi, A., Clarkson, K. L., and Woodruff, D. P. (2022). Low-rank approximation with $1/\epsilon^{1/3}$ matrix-vector products. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1130–1143.

[Beck, 2017] Beck, A. (2017). *First-order methods in optimization*. SIAM.

[Ben-Tal and Nemirovski, 2001] Ben-Tal, A. and Nemirovski, A. (2001). *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. SIAM.

[Bhutta et al., 2020] Bhutta, M. U. M., Aslam, S., Yun, P., Jiao, J., and Liu, M. (2020). Smart-inspect: micro scale localization and classification of smartphone glass defects for industrial automation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2860–2865. IEEE.

[Boyd et al., 2004] Boyd, S., Boyd, S. P., and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

[Braverman et al., 2016] Braverman, V., Feldman, D., Lang, H., Statman, A., and Zhou, S. (2016). New frameworks for offline and streaming coreset constructions. *arXiv preprint arXiv:1612.00889*.

[Cao et al., 2016] Cao, Z., Sheikh, Y., and Banerjee, N. K. (2016). Real-time scalable 6dof pose estimation for textureless objects. In *2016 IEEE International conference on Robotics and Automation (ICRA)*, pages 2441–2448. IEEE.

[Chierichetti et al., 2017] Chierichetti, F., Gollapudi, S., Kumar, R., Lattanzi, S., Panigrahy, R., and Woodruff, D. P. (2017). Algorithms for ℓ_p low-rank approximation. In *International Conference on Machine Learning*, pages 806–814. PMLR.

[Clarkson and Woodruff, 2015] Clarkson, K. L. and Woodruff, D. P. (2015). Input sparsity and hardness for robust subspace approximation. In *2015 IEEE 56th annual symposium on foundations of computer science*, pages 310–329. IEEE.

[Cockayne and Melzak, 1969] Cockayne, E. J. and Melzak, Z. A. (1969). Euclidean constructibility in graph-minimization problems. *Mathematics Magazine*, 42(4):206–208.

[Cohen et al., 2016] Cohen, M. B., Lee, Y. T., Miller, G., Pachocki, J., and Sidford, A. (2016). Geometric median in nearly linear time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 9–21.

[Cohen-Addad et al., 2021] Cohen-Addad, V., Saulpic, D., and Schwiegelshohn, C. (2021). Improved coresets and sublinear algorithms for power means in Euclidean spaces. *Advances in Neural Information Processing Systems*, 34:21085–21098.

[Dasgupta et al., 2009] Dasgupta, A., Drineas, P., Harb, B., Kumar, R., and Mahoney, M. W. (2009). Sampling algorithms and coresets for l_p regression. *SIAM Journal on Computing*, 38(5):2060–2078.

[Deshpande et al., 2006] Deshpande, A., Rademacher, L., Vempala, S. S., and Wang, G. (2006). Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(1):225–247.

[Deshpande et al., 2011] Deshpande, A., Tulsiani, M., and Vishnoi, N. K. (2011). Algorithms and hardness for subspace approximation. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 482–496. SIAM.

[Deshpande and Varadarajan, 2007] Deshpande, A. and Varadarajan, K. (2007). Sampling-based dimension reduction for subspace approximation. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 641–650.

[Deshpande and Vempala, 2006] Deshpande, A. and Vempala, S. (2006). Adaptive sampling and fast low-rank matrix approximation. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 292–303. Springer.

[Ding et al., 2006] Ding, C., Zhou, D., He, X., and Zha, H. (2006). R_1-PAC: rotational invariant L_1-norm principal component analysis for robust subspace factorization. In *Proceedings of the 23rd international conference on Machine learning*, pages 281–288.

[Drineas et al., 2006a] Drineas, P., Kannan, R., and Mahoney, M. W. (2006a). Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on computing*, 36(1):158–183.

[Drineas et al., 2006b] Drineas, P., Mahoney, M. W., and Muthukrishnan, S. (2006b). Polynomial time algorithm for column-row based relative-error low-rank matrix approximation. Technical report, Technical report 2006-04, DIMACS.

[Fack and Kosaki, 1986] Fack, T. and Kosaki, H. (1986). Generalized s-numbers of τ-measurable operators. *Pacific Journal of Mathematics*, 123(2):269–300.

[Feldman and Langberg, 2011] Feldman, D. and Langberg, M. (2011). A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing (STOC)*, pages 569–578.

[Feldman et al., 2010] Feldman, D., Monemizadeh, M., Sohler, C., and Woodruff, D. P. (2010). Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 630–649. SIAM.

[Feldman et al., 2020] Feldman, D., Schmidt, M., and Sohler, C. (2020). Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657.

[Feng et al., 2021] Feng, Z., Kacham, P., and Woodruff, D. (2021). Dimensionality reduction for the sum-of-distances metric. In *International Conference on Machine Learning*, pages 3220–3229. PMLR.

[Golub and Van Loan, 1996] Golub, G. H. and Van Loan, C. F. (1996). Matrix computations, johns hopkins u. *Math. Sci., Johns Hopkins University Press, Baltimore, MD*.

[Grötschel et al., 2012] Grötschel, M., Lovász, L., and Schrijver, A. (2012). *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media.

[Hansen and Thisse, 1983] Hansen, P. and Thisse, J.-F. (1983). Recent advances in continuous location theory. *Sistemi Urbani*, 1:33–54.

[Har-Peled, 2014] Har-Peled, S. (2014). Low rank matrix approximation in linear time. *arXiv preprint arXiv:1410.8802*.

[Hoos and Stützle, 1998] Hoos, H. and Stützle, T. (1998). On the empirical evaluation of las vegas algorithms-position paper. Technical report, Technical report, Computer Science Department, University of British Columbia.

[HromkoviČ, 2005] HromkoviČ, J. (2005). Success amplification and random sampling. In *Design and Analysis of Randomized Algorithms*, pages 153–182. Springer.

[Huang and Vishnoi, 2020] Huang, L. and Vishnoi, N. K. (2020). Coresets for clustering in euclidean spaces: importance sampling is nearly optimal. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1416–1429.

[Ikebe et al., 1987] Ikebe, Y., Inagaki, T., and Miyamoto, S. (1987). The monotonicity theorem, cauchy's interlace theorem, and the courant-fischer theorem. *The American Mathematical Monthly*, 94(4):352–354.

[Kanji, 2016] Kanji, T. (2016). Self-localization from images with small overlap. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4497–4504. IEEE.

[Kaynak, 1995] Kaynak, C. (1995). Methods of combining multiple classifiers and their application to handwritten digit recognition. Master's thesis, Fen Bilimleri Enstitüsü.

[Kim et al., 2014] Kim, E., Choi, S., and Oh, S. (2014). A robust autoregressive gaussian process motion model using l_1-norm based low-rank kernel matrix approximation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4396–4401. IEEE.

[ktountas, 2021] ktountas (2021). L1-Norm-Algorithms. https://github.com/ktountas/L1-Norm-Algorithms.git.

[Li and Woodruff, 2016] Li, Y. and Woodruff, D. P. (2016). Tight bounds for sketching the operator norm, schatten norms, and subspace embeddings. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2016)*, pages 39–1.

[Lopuhaa and Rousseeuw, 1991] Lopuhaa, H. P. and Rousseeuw, P. J. (1991). Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, pages 229–248.

[Maalouf et al., 2020] Maalouf, A., Statman, A., and Feldman, D. (2020). Tight sensitivity bounds for smaller coresets. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2051–2061.

[Markopoulos, 2025] Markopoulos, P. (2025). L1-PCA Toolbox. https://www.mathworks.com/matlabcentral/fileexchange/64855-l1-pca-toolbox. MATLAB Central File Exchange. Retrieved July 14, 2025.

[Markopoulos et al., 2013] Markopoulos, P. P., Karystinos, G. N., and Pados, D. A. (2013). Some options for l1-subspace signal processing. In *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*, pages 1–5. VDE.

[Markopoulos et al., 2014] Markopoulos, P. P., Karystinos, G. N., and Pados, D. A. (2014). Optimal algorithms for l_1-subspace signal processing. *IEEE Transactions on Signal Processing*, 62(19):5046–5058.

[Markopoulos et al., 2017] Markopoulos, P. P., Kundu, S., Chamadia, S., and Pados, D. A. (2017). Efficient l1-norm principal-component analysis via bit flipping. *IEEE Transactions on Signal Processing*, 65(16):4252–4264.

[Mittelmann, 2003] Mittelmann, H. D. (2003). An independent benchmarking of sdp and socp solvers. *Mathematical Programming*, 95(2):407–430.

[Molzahn and Hiskens, 2015] Molzahn, D. K. and Hiskens, I. A. (2015). Mixed sdp/socp moment relaxations of the optimal power flow problem. In *2015 IEEE Eindhoven PowerTech*, pages 1–6. IEEE.

[Mowforth and Shepherd,] Mowforth, P. and Shepherd, B. Statlog (Vehicle Silhouettes).

UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HG6N.

[Nesterov, 1994] Nesterov, Y. (1994). Interior point polynomial methods in convex programming. *Theory and Applications*.

[of Electrical et al., 1985] of Electrical, I., Committee, E. E. C. S. S., and Stevenson, D. (1985). *IEEE standard for binary floating-point arithmetic*. IEEE.

[Pan and Chen, 1999] Pan, V. Y. and Chen, Z. Q. (1999). The complexity of the matrix eigenproblem. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 507–516.

[Pearson, 1901] Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

[Ramond, 2022] Ramond, P. (2022). The abel–ruffini theorem: Complex but not complicated. *The American Mathematical Monthly*, 129(3):231–245.

[Sarlos, 2006] Sarlos, T. (2006). Improved approximation algorithms for large matrices via random projections. In *2006 47th annual IEEE symposium on foundations of computer science (FOCS'06)*, pages 143–152. IEEE.

[Shyamalkumar and Varadarajan, 2007] Shyamalkumar, N. D. and Varadarajan, K. (2007). Efficient subspace approximation algorithms. In *SODA*, volume 7, pages 532–540.

[Siebert, 1987] Siebert, J. P. (1987). Vehicle recognition using rule based methods.

[Smale, 1998] Smale, S. (1998). Mathematical problems for the next century. *The mathematical intelligencer*, 20(2):7–15.

[Sohler and Woodruff, 2018] Sohler, C. and Woodruff, D. P. (2018). Strong coresets for k-median and subspace approximation: Goodbye dimension. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 802–813. IEEE.

[Song et al., 2017] Song, Z., Woodruff, D. P., and Zhong, P. (2017). Low rank approximation with entrywise l1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 688–701.

[Ta and Dellaert, 2014] Ta, D.-N. and Dellaert, F. (2014). Linear-time estimation with tree assumed density filtering and low-rank approximation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4556–4563. IEEE.

[Tardella, 2011] Tardella, F. (2011). The fundamental theorem of linear programming: extensions and applications. *Optimization*, 60(1-2):283–301.

[Tukan et al., 2020] Tukan, M., Maalouf, A., and Feldman, D. (2020). Coresets for near-convex functions. *Advances in Neural Information Processing Systems*, 33:997–1009.

[Varadarajan et al., 2007] Varadarajan, K., Venkatesh, S., Ye, Y., and Zhang, J. (2007). Approximating the radii of point sets. *SIAM Journal on Computing*, 36(6):1764–1776.

[Varadarajan and Xiao, 2012] Varadarajan, K. and Xiao, X. (2012). On the sensitivity of shape fitting problems. In *32nd International Conference on Foundations of Software Technology and Theoretical Computer Science*, page 486.

[Wang, 2015] Wang, H. (2015). On least squares solutions subject to a rank restriction. *Linear and Multilinear Algebra*, 63(2):264–273.

[Wendel and Underwood, 2016] Wendel, A. and Underwood, J. (2016). Self-supervised weed detection in vegetable crops using ground based hyperspectral imaging. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 5128–5135. IEEE.

[Woodruff and Yasuda, 2024] Woodruff, D. P. and Yasuda, T. (2024). Ridge leverage score sampling for *ell_p* subspace approximation. *arXiv preprint arXiv:2407.03262*.

[Zhang et al., 2014] Zhang, D., Zhao, X., Han, J., and Zhao, Y. (2014). A comparative study on PCA and LDA based EMG pattern recognition for anthropomorphic robotic hand. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4850–4855. IEEE.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes. See Section 4 and Section 5.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes. See Theorem 5.2.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes.
2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. Yes.

(b) Complete proofs of all theoretical results. Yes. See Section D and Section E.

(c) Clear explanations of any assumptions. Yes.

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes/No/Not Applicable]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes.

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. Yes.

(b) The license information of the assets, if applicable. Not Applicable.

(c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable.

(d) Information about consent from data providers/curators. Not Applicable.

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. Not Applicable.

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

k-PCA for (Non-Squared) Euclidean Distances: Supplementary Materials

A RELATED WORK

k-rank matrix approximation. We now clarify how (2) is related to the k-subspace median problem. Let $A \in \mathbb{R}^{n\times d}$ be the data matrix whose ith row corresponds to the ith point of P, in some arbitrary order. For a matrix $X \in \mathbb{R}^{d\times k}$, the projection of these points onto the subspace spanned by the columns of X is given by $AX \in \mathbb{R}^{n\times k}$. Mapping these projected coordinates back into the original ambient space yields $AXX^T = A\Pi \in \mathbb{R}^{n\times d}$, whose rows are precisely the projected points in $\mathbb{R}^d$.

Next, let $X_\perp \in \mathbb{R}^{d\times(d-k)}$ be a matrix whose columns form an orthonormal basis for the orthogonal complement of the subspace spanned by the columns of X. Then

$$V := [X \mid X_\perp] \in \mathbb{R}^{d\times d}$$

is an orthogonal matrix, and the associated projectors satisfy

$$XX^T + X_\perp X_\perp^T = \Pi + \Pi_\perp = V^TV = VV^T = I,$$

where Π and $\Pi_\perp$ are the orthogonal projections onto the subspaces spanned by X and $X_\perp$, respectively.

Defining

$$U := AV = [AX \mid AX_\perp] \in \mathbb{R}^{n\times d}$$

leads to the factorization

$$A = AVV^T = A(XX^T + X_\perp X_\perp^T) = A(\Pi + \Pi_\perp) = A\Pi + A\Pi_\perp = UV^T.$$

Thus, A can be decomposed as UV^T. If we now restrict U and V to their first k columns, denoted U_k and V_k in $\mathbb{R}^{d\times k}$, we obtain a rank-k approximation $U_kV_k^T$ to A. The precise quality or optimality of this approximation depends on the particular factorization used.

For a given k-dimensional subspace spanned by the columns of V_k, the point in this subspace that is closest (in Euclidean distance) to any input point $p \in P$ is simply the orthogonal projection of p onto that subspace. In terms of the factorization, this projection corresponds to the row of U_k associated with p.

Consequently, taking $X = V_k$ that minimizes (1) above means that the column space of X coincides with the column space of a rank-k matrix B that optimally approximates A in the sense of the "mixed norm"

$$\min_B \|A - B\|_{2,z} = \min_{U_k,V_k} \|A - U_kV_k\|_{2,z}\,. \tag{9}$$

In particular, the column space of V_k matches the column space of some optimal solution X, and this common column space defines the k-dimensional subspace $S \subseteq \mathbb{R}^d$ that minimizes the sum of the z-th powers of the Euclidean distances from the points to S.

For the special case $z = 2$, the optimal solution is $B = U_kD_kV_k$, where UDV^T is the SVD of A, and V_k consists of the top k right singular vectors of A. The columns of U_k are the left singular vectors after scaling by a diagonal matrix $D \in \mathbb{R}^{d\times d}$. In this special case, U_k also has mutually orthogonal columns, i.e., $U_k^TU_k = I$. In other words, the optimal k-subspace implies an optimal k-rank approximation to a given matrix that depends on the given norm.

Why k-subspace median? Although the k-subspace mean can be computed efficiently via SVD, and the affine k-subspace mean (k-PCA) is the same after translating the mean of the input points to the origin, it lacks robustness when handling outliers or corrupted input points. In particular, these methods can perform poorly when the

data contain significant outliers or are heavily corrupted, and the coordinates are usually not spared on any basis. A natural approach to handling these issues is to approximate the k-subspace median (kSM), which aims to minimize the sum of (non-squared) Euclidean distances to the input points ($z = 1$). To understand why the kSM is more robust to outliers and noise than k-PCA, it helps to compare the mean with the median of the distance vector to a subspace—essentially a collection of scalar distances in $d = 1$ [Lopuhaa and Rousseeuw, 1991].

Recall that the mean is the value that reduces the total squared differences from the input numbers, whereas the median minimizes the total (non-squared) differences. If a single data point is moved toward infinity, the mean will also approach infinity. In contrast, to achieve a similar result on the median, at least half of the data points must be shifted to infinity. This number is known as a *breakdown point* [Lopuhaa and Rousseeuw, 1991].

Affine subspace (flat). By definition, the k-subspace median always passes through the origin. A natural generalization of the k-subspace median (or, more generally, of any estimator based on an $z \geq 1$ norm) is the *k-flat median*, which is defined as the affine k-dimensional subspace of $\mathbb{R}^d$ that minimizes the sum of distances to the points. In other words, instead of restricting ourselves to linear k-subspaces that must contain the origin, we allow translated k-subspaces—affine k-flats—that are not required to intersect the origin.

In the special case $z = 2$, it is straightforward to verify that the optimal k-flat (the k-flat mean) always passes through the mean of the point set P. Consequently, one can obtain it by first translating the input so that the mean of P moves to the origin, computing the optimal k-subspace (the mean k-subspace) for the translated data, and then translating this subspace back to the original coordinates [Pearson, 1901]. This equivalence underpins the relationship between k-PCA (which works with the k-flat mean) and k-SVD (which operates with the k-subspace mean).

However, when $z \neq 2$, this convenient reduction breaks down: the optimal affine k-flat no longer necessarily aligns with a simple translation of the optimal linear k-subspace through the mean. To address this, Maalouf et al. [Maalouf et al., 2020] proposed a different reduction, showing how the k-flat problem in $\mathbb{R}^d$ can be transformed into a k-subspace problem in $\mathbb{R}^{d+1}$. Due to this reduction and for conceptual clarity, in the remainder of this section—and throughout this work—we focus exclusively on the k-subspace median, and do not explicitly treat the k-flat (or affine) median as a separate object.

Hardness. Clarkson and Woodruff [Clarkson and Woodruff, 2015, Theorem 2] showed that, when k is given as part of the input (as in our setting), it is NP-hard to approximate the k-subspace median of n points in $\mathbb{R}^d$, up to a multiplicative factor of $(1 + 1/d^{O(1)})$. In contrast, for the $\ell_{1,1}$ objective, works by Markopoulos et al. [Markopoulos et al., 2013, Markopoulos et al., 2014] proposed exact algorithms: the first runs in time exponential in n, while the latter achieves an exact solution with running time exponential only in k.

Convex optimization. Our technique involves transforming the non-convex k-subspace median task into a convex optimization problem involving two cones. The first cone is the set of positive semi-definite (PSD) matrices in the vector space of symmetric matrices (see Fig. 1), and the second cone is the second-order cone (SOC) in $\mathbb{R}^{d+1}$. The corresponding dual techniques are called SDP (semi-definite programming) and SOCP (second-order cone programming), and their combination is called SDP-SOCP [Nesterov, 1994, Mittelmann, 2003, Molzahn and Hiskens, 2015].

B RELATED OPTIMIZATION FUNCTIONS

The k-subspace median can be generalized to minimize the sum of Euclidean distances over every distance to the power of $z > 0$ [Shyamalkumar and Varadarajan, 2007, Ding et al., 2006]. For $z \in \{1, 2, \infty\}$, we obtain the k-subspace median (as in this paper), mean, and center, respectively.

The case $z = 2$. The optimal subspace for the common and easiest case $z = 2$ is sometimes called the k-dimensional *subspace mean* or k-SVD/PCA. It minimizes the sum of *squared* Euclidean distances to the input points. The name comes from the fact that (a) the mean of P (center of mass) minimizes the sum of the squared distances to the input points and (b) this subspace minimizes the mean or root mean squared (RMS) of the distances to the n points.

It is frequently asserted that, for the case $z = 2$, the optimal subspace can be computed *exactly* in $O(nd^2)$ time by taking the span of the top k right singular vectors of the $n \times d$ real matrix A, whose rows represent the input

points [Golub and Van Loan, 1996, Pearson, 1901]. Equivalently, these correspond to the top k eigenvectors of A^TA, with any ties resolved arbitrarily. This statement appears in most related academic publications (see bibliography) as well as in sources like Wikipedia. However, computing eigenvalues of A^TA can encode the problem of finding the roots of a polynomial via its companion matrix, and such roots do not admit a closed-form expression in radicals for $d > 4$ by the Abel–Ruffini Theorem [Ramond, 2022]. Therefore, even in the basic case $z = 2$, one can in general only obtain an approximation of the optimal subspace, unless additional assumptions are imposed on the input; see the paragraph on precision for a more detailed discussion.

Modern research [Sarlos, 2006, Har-Peled, 2014, Deshpande and Vempala, 2006, Deshpande et al., 2006, Drineas et al., 2006b, Drineas et al., 2006a] has concentrated on developing algorithms that compute k-mean subspace in sub-quadratic time in d, achieving a $(1+\varepsilon)$-approximation to the optimal solution in expected time $O\left(nd \cdot \text{poly}\left(k, \frac{1}{\varepsilon}\right)\right)$.

Precision. After computing A^TA in $O(nd^2)$ time, obtaining a multiplicative $(1+\varepsilon)$-approximation to the k-subspace mean of P for a given $\varepsilon > 0$ reduces to computing the eigenvalues of A, which requires $O(d^3 + (d\log^2 d)\log\log(1/\varepsilon))$ time [Pan and Chen, 1999]. For instance, by choosing $\varepsilon = 1/n^9$, we can achieve a $(1+1/n^9)$ approximation in $O(nd^2)$ time for sufficiently large n.

This level of accuracy, which depends only polylogarithmically on $1/\varepsilon$, is often referred to as *machine precision*. Assume now that every coordinate of each input point $p \in P$ is stored with only b bits of precision for some integer $b \geq 1$, for example, $b = 23$ for C's float type or $b = 52$ for double [of Electrical et al., 1985]. Equivalently, after scaling the coordinates of P by a sufficiently large integer, all coordinates become integers in $\left[-2^b, 2^b\right]$, and thus $P \subseteq \left[-2^b, \ldots, 2^b\right]^d$.

Under this assumption, the entries of the eigenvectors of A^TA, as well as its eigenvalues, also have bounded precision $O(b)$; see, e.g., [Feldman et al., 2020]. Consequently, setting $\varepsilon \in 1/2^{\Omega(b)}$ yields an exact solution to the k-subspace mean in time $O\left(nd^2 + d^2b^{O(1)}\right)$. In particular, when each coordinate of P has fixed finite precision $b \in O(1)$, the running time simplifies to $O(nd^2)$. This realistic assumption in practical settings often explains why this subtlety is ignored in much of the literature.

Another approach is to define a *weakly polynomial time* algorithm [Grötschel et al., 2012] whose execution time is polynomially dependent on the input size in bits. In the context of the previous paragraph, if every coordinate of every input point can be represented via b bits, then the size of the input in bits is $N \in O(ndb)$, and it can be stated that the running time $O\left(nd^2 + d^2b^{O(1)}\right) = N^{O(1)}$ above is polynomial in the size of the input, without mentioning b. This situation is often common in convex optimization. Specifically, a strongly polynomial-time algorithm (which is independent of b) for linear programming has been cited as one of the 18 greatest unsolved problems of the 21st century [Smale, 1998].

The case $z = \infty$. This variant of the problem is referred to as the *k-subspace center* problem, by analogy with the one-dimensional setting $d = 1$, similar to the mean and median formulations. Here, the objective is to find a subspace that minimizes the maximum distance from the input points to the subspace. Consequently, the method is highly susceptible to the influence of outliers—arguably even more so than PCA. Varadarajan et al. [Varadarajan et al., 2007] developed a polynomial-time randomized algorithm that achieves an $O\left(\sqrt{\log n}\right)$-approximation guarantee for this problem.

The case $z \in (2, \infty)$. Deshpande et al. [Deshpande et al., 2011] substitute the non-convex rank constraint with its convex relaxation in the form of a trace constraint. Nonetheless, their theoretical results do not extend to the k-subspace median problem; they are restricted to the less robust setting where $p \geq 2$. In addition, the algorithm they propose is randomized rather than deterministic.

C DATA REDUCTION

Many data reduction methods, such as coresets and sketches, do not directly approximate variants of the k-subspace median problem. Instead, they reduce the original input set P of n points to a much smaller collection of weighted m points, with $m \ll n$ (a "coreset"), or to linear combinations of these points (a "sketch"). Yet, only a handful of algorithms actually approximate the k-subspace median itself, even when operating on the reduced set of m points. Moreover, the contemporary coreset constructions we know of mainly address non-mixed norms

$\ell_z := \ell_{z,z}$; see, for instance, [Dasgupta et al., 2009] and Section C.

Consider a weighted point set (P, w), where $P \subseteq \mathbb{R}^d$, $|P| = n$, and $w \in \mathbb{R}^n_{++}$, together with a desired approximation parameter $\varepsilon \in (0, 1)$. A weighted subset (C, w_C) is called a (strong) ε-*coreset* for (P, w) if $C \subseteq P$ and, for every k-dimensional subspace, the total distance from P to this subspace is approximated—up to a multiplicative factor of $1 + \varepsilon$—by the total weighted distance from C to the same subspace.

Many have suggested efficient randomized algorithms to compute such a coreset to the k-subspace median; among them are Tukan et al. [Tukan et al., 2020], Braverman et al. [Braverman et al., 2016], and others [Feldman et al., 2010, Feldman et al., 2020, Feng et al., 2021, Sohler and Woodruff, 2018]. For $k = 0$ (point-median), coresets were suggested in [Cohen-Addad et al., 2021].

Randomized coreset constructions typically run in near-linear time in the input size nd and polynomial time in $\frac{k}{\varepsilon}$. By contrast, deterministic constructions, as shown by Sohler and Woodruff [Sohler and Woodruff, 2018], require time exponential in k. Note that a coreset is a data-reduction mechanism rather than a stand-alone algorithm for the kSM problem studied here; instead, it serves to speed up such algorithms so that their running time becomes linear or near-linear in the input size.

Sketches are closely related to coresets, but rather than selecting a weighted subset of the data, each point in a sketch may be expressed as a linear combination of the original input points. As a result, a sketch can be represented as multiplying the $n \times d$ input matrix A by a "fat" sketching matrix $S \in \mathbb{R}^{m \times n}$, producing a reduced matrix $SA \in \mathbb{R}^{m \times d}$. Constructions of such sketches for operator norms, Schatten norms, and subspace embeddings were given in [Li and Woodruff, 2016], which also provides many further references.

With this method, we can employ the strong coreset of size $\tilde{O}\left(k\varepsilon^{-4}\right)$ constructed in time $\tilde{O}\left(\operatorname{nnz}(P) + d^w\right)$ by Woodruff and Yasuda [Woodruff and Yasuda, 2024], and then solve the Frobenius-norm version of the problem while preserving all costs up to $\tilde{O}\left(\sqrt{k\varepsilon^{-4}}\right)$. This yields a randomized $\tilde{O}\left((1+\varepsilon)\sqrt{k\varepsilon^{-4}}\right)$-approximation for the k-subspace median.

D PROOF OF CORRECTNESS

The goal of this section is to prove the main technical result in Theorem D.2. That is, Algorithm 1 indeed returns a $\sqrt{d}$ approximation to the ksm of the input set P, and its running time is polynomial in the input. The proof is based on the intuition of Section 11.

Consider the value of ζ that is computed during the execution of Line 9. The next proposition states that ζ minimizes the linear program that is defined in (10). That is, taking the largest $d-k$ entries of q, is an "binary rounding" that yields the largest value $\lambda^T q$ over any combination $\lambda \in [0,1]^d$ whose sum is $\|\lambda\|_1 = d-k$. The proof is a straightforward application of the "Fundamental theorem of linear programming" [Tardella, 2011] and can be considered as a variant of the Eckart–Young–Mirsky [Wang, 2015] and the Courant–Fischer [Ikebe et al., 1987] theorems for the case of $z = 2$, as in k-SVD. A main difference is that these theorems take the largest k values from the same ordered list of singular values, regardless of k. In our case, every value of k yields a different set of candidate entries in λ. This is probably related to the monotonicity property of the k-subspace mean, which is contained in the $(k+1)$-subspace mean, unlike the k-subspace median case. Without loss of generality, the proposition below assumes that the entries of q are sorted. Otherwise, we sort them and rearrange them back after computing λ.

As a result, $E = V^T \operatorname{diag}(\zeta)\, V$ is not only a $d-k$ projection matrix that lies on the boundary of the gray ellipsoid of Fig. 1, but also a minimizer of $\lambda^T q$ above. This fact will be used in our main proof to bound the approximation factor.

Proposition D.1. *Let $q \in [0,\infty)^d$ be a vector that satisfies $q_1 \leq \cdots \leq q_d$. Let $k \in [d-1]$ be an integer, and $\zeta = (1, 1, \cdots, 0, 0) \in \{0,1\}^d$ be a binary vector that satisfies $\zeta_i = 1$ if and only if $i \in [d-k]$. Then*

$$\zeta \in \arg\min_{\lambda} \lambda^T q, \tag{10}$$

where the minimum is over every $\lambda \in [0,1]^d$ whose sum of entries is $\sum_{i=1}^d \lambda_i = d-k$.

Proof. Minimizing the linear function $\lambda^T q$ over the polytope

$$\left\{ \lambda \in [0,1]^d \mid \sum_{i=1}^{d} \lambda_i = d - k \right\}$$

is a linear program. The proposition then follows from the fundamental theorem of linear programming, see e.g. [Tardella, 2011]. □

Using Proposition D.1, we can prove that the output of Algorithm 1 is a $\sqrt{d}$ approximation as follows.

Theorem D.2 (Correctness of the Algorithm 1)**.** *Let $k \in [d-1]$ be an integer and $P = (p_i)_{i=1}^n \in (\mathbb{R}^d)^n$. Let $S \subseteq \mathbb{R}^{d \times d}$ be the output of a call to kSM-Approx(P, k); see Algorithm 1. Then S is a $\sqrt{d}$-approximation to the k-subspace median of P. That is,*

$$\sum_{i=1}^{n} \operatorname{dist}(p_i, S) \leq \sqrt{d} \cdot \operatorname{ksm}(P, k).$$

Proof. Consider the values of X^*, y^*, V, D, $\left\{v^{(i)}\right\}_{i=1}^n$, q, ζ, and E during the execution of Line 11 of Algorithm 1. Since ζ consists of $d-k$ ones and k zeros, E is a symmetric matrix as

$$E^2 = \left(V^T \operatorname{diag}(\zeta) V\right)^2 = V^T \operatorname{diag}(\zeta)^2 V = V^T \operatorname{diag}(\zeta) V = E,$$

and have rank k, as

$$\operatorname{rank}(E) = \operatorname{rank}\left(V^T \operatorname{diag}(\zeta) V\right) = \operatorname{rank}(\operatorname{diag}(\zeta)) = d - k.$$

Hence, E is a projection matrix that corresponds to a k-subspace in $\mathbb{R}^d$. It is left to prove that this feasible solution to Problem (5) indeed approximates the k-subspace median.

Indeed, by (7) $x^T X^* x \in [0,1]$ for every unit vector $x \in \mathbb{R}^d$, and thus its eigenvalues are in $[0,1]$. That is, $D \in [0,1]^{d \times d}$. We also have

$$\|\operatorname{diag}(D)\|_1 = \operatorname{trace}(D) = \operatorname{trace}\left(V^T D V\right) = \operatorname{trace}(X^*) = d - k,$$

where the last equality holds by (7). We can thus substitute $\lambda := \operatorname{diag}(D)$ in Proposition D.1 to obtain

$$\zeta^T q \leq \operatorname{diag}(D)^T q. \tag{11}$$

For every $\lambda \in \mathbb{R}^d_+$,

$$\begin{aligned}
\lambda^T q &= \sum_{j=1}^{d} \lambda_j \sum_{i=1}^{n} \left| v_j^{(i)} \right| && (12)\\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} \left| \lambda_j v_j^{(i)} \right| && (13)\\
&= \sum_{i=1}^{n} \left\| \operatorname{diag}(\lambda)\, v^{(i)} \right\|_1 && (14)\\
&= \sum_{i=1}^{n} \left\| \operatorname{diag}(\lambda) V p_i \right\|_1 && (15)
\end{aligned}$$

where (12) and (15) are by the definition of $\left\{v^{(i)}\right\}_{i=1}^n$ and q respectively, (13) holds because $\lambda \in \mathbb{R}^d_+$, and (14) holds by the definition of $\|\cdot\|_1$ and because $\operatorname{diag}(\lambda)$ is diagonal. Furthermore,

$$\begin{aligned}
\sum_{i=1}^{n} \left\| \operatorname{diag}(\zeta) V p_i \right\|_1 &= \zeta^T q && (16)\\
&\leq \operatorname{diag}(D)^T q && (17)\\
&= \sum_{i=1}^{n} \left\| D V p_i \right\|_1 . && (18)
\end{aligned}$$

where (16) and (18) hold by substituting, respectively, $\lambda := \zeta$ and $\lambda := \text{diag}(D)$ in (15), and (17) holds by (11). Therefore,

$$\sum_{i=1}^{n} \|\text{diag}(\zeta) V p_i\|_2 \leq \sum_{i=1}^{n} \|\text{diag}(\zeta) V p_i\|_1 \tag{19}$$

$$\leq \sum_{i=1}^{n} \|DV p_i\|_1 \tag{20}$$

$$\leq \sqrt{d} \sum_{i=1}^{n} \|DV p_i\|_2 , \tag{21}$$

where (20) holds by (18), and (19) and (21) holds since $\frac{1}{\sqrt{d}} \|y\|_1 \leq \|y\|_2 \leq \|y\|_1$ for every $y \in \mathbb{R}^d$.

Let $X \in \mathbb{S}_d^+$ be a projection matrix that corresponds to the k-subspace median of P. That is,

$$\text{ksm}(P, k) = \sum_{i=1}^{n} \|X p_i\|_2 .$$

By its definition in Section 4, a $(d-k)$ projection matrix in $\mathbb{R}^d$ has eigenvalues 0 and 1 with multiplicity k and $d-k$, respectively. Therefore, its sum of eigenvalues is $\text{trace}(X) = d - k$. Letting $y \in \mathbb{R}^n$ be the vector whose ith coordinate is $y_i := \|X p_i\|_2$ for every $i \in [n]$, yields a feasible solution (X, y) for Problem (7). Hence,

$$\text{relaxksm}(P, k) \leq \mathbb{1}_n^T y = \sum_{i=1}^{n} \|X p_i\|_2 = \text{ksm}(P, k) . \tag{22}$$

It follows that

$$\sum_{i=1}^{n} \text{dist}(p_i, S) = \sum_{i=1}^{n} \|p_i - (I - E) p_i\|_2 = \sum_{i=1}^{n} \|E p_i\|_2 \tag{23}$$

$$= \sum_{i=1}^{n} \left\|V^T \text{diag}(\zeta) V p_i\right\|_2 \tag{24}$$

$$= \sum_{i=1}^{n} \|\text{diag}(\zeta) V p_i\|_2 \tag{25}$$

$$\leq \sqrt{d} \sum_{i=1}^{n} \|DV p_i\|_2 \tag{26}$$

$$= \sqrt{d} \sum_{i=1}^{n} \left\|V^T DV p_i\right\|_2 = \sqrt{d} \sum_{i=1}^{n} \|X^* p_i\|_2 \tag{27}$$

$$= \sqrt{d} \cdot \text{relaxksm}(P, k) \tag{28}$$

$$\leq \sqrt{d} \cdot \text{ksm}(P, k) , \tag{29}$$

where (23) follows from the definition of S in Line 10, (24) is by the definition of E in Line 9, (25) and (27) hold because V is an orthogonal matrix, (26) is by (21), (28) holds because $X^* = V^T DV$ is an optimal solution of Problem (7) by Line 1, and (29) is by (22). Therefore, S is a $\sqrt{d}$ approximation to the kSM of P. □

E RUNNING TIME

Theorem D.2 establishes the correctness of our main algorithm, but does not address its running time. In this section, we derive an upper bound on the running time of Algorithm 1, showing that it is $(ndk)^{O(1)}$, i.e., polynomial in all parameters; see Theorem E.4 for details. As discussed in the introductory sections, this running time can be further reduced to near-linear in the input size—up to a small constant factor—by running our algorithm on suitably reduced (small) core sets.

A key tool in our analysis is the *central path method* [Nesterov, 1994], which we introduced in detail in Section F. In this section, we apply that framework to address the particular structure and requirements of our kSM problem.

As with other eigenvalue-based methods, such as PCA and linear programming, the running time of our algorithm in the theoretical RAM model depends on the bit complexity of the numbers involved, that is, it scales logarithmically with the magnitude of the input coordinates. This dependence is inherent and cannot be removed, due to the Abel–Ruffini theorem, which asserts that the roots of polynomials of degree greater than four cannot, in general, be expressed in closed form using radicals.

In real-world computing, machines and programs operate with a limited word size (32 or 64 bits), and eigenvalues are calculated extremely quickly in hardware using optimized libraries such as Intel MKL, ARM Compute, or NVIDIA CUDA. This likely explains why most academic literature—including Wikipedia—tends to overlook this limitation and simply claims that, for instance, the SVD or eigen-decomposition of an n-by-n matrix can be computed exactly in $O(n^3)$ time.

Another approach in theoretical computer science is to specify that the running time is, for example, polynomial in the input size n, thereby abstracting away the bit-length of each coordinate. Such an algorithm is said to run in weakly (as opposed to strongly) polynomial time. Other works adopt the Word RAM model, which assumes that each number (word) fits into $\Delta = \log n$ bits, or they analyze complexity in the classical Turing machine model, or in terms of the number of arithmetic operations.

We chose to formulate exact theorems without suppressing any factors, which is why the theorem explicitly includes the logarithmic dependence on Δ under the RAM model.

The kSM Problem (7) gets as input a set $P = (p_1, \cdots, p_n)$ of n points in $\mathbb{R}^d$ and an integer $k \in [d-1]$. We reduce it to an input instance $G := (f_0, F, A, b)$ for a self-concordance conic optimization problem, which is defined in Definition F.4. The variables to minimize in the kSM Problem (7) are $X \in \mathbb{R}^{d\times d}$ and $y \in \mathbb{R}^n$. Although these domains are not subsets of $\mathbb{R}^n$ as in this definition, they are isomorphic to an Euclidean space in a sense that enables us to use the central path in Theorem F.9. This straightforward generalization of convex optimization from Euclidean space to Cartesian products of metric spaces is well known and is explained in textbooks such as [Boyd et al., 2004, Ben-Tal and Nemirovski, 2001, Beck, 2017].

Lemma E.1. *Let $P := (p_i)_{i=1}^n$ be a set of $n \geq 1$ points in $\mathbb{R}^d$. There is a conic instance $G(P) := (f_0, F, A, b)$ for the optimization problem (7) whose t-relaxation, for every $t > 0$, is the function $G_t : \mathcal{D}(G) \to \mathbb{R}$ that maps every $(X, y) \in \mathcal{D}(G)$ to*

$$G_t(X, y) := t\mathbb{1}_n^T y - \sum_{i=1}^{n} \ln\left(y^2{}_i - \|Xp_i\|_2^2\right) - \ln|X| - \ln|I_d - X|\,. \tag{30}$$

Moreover, $\deg(F) = 2(n + d)$.

Proof. Let (P, k) denote the input of Problem (7). That is, $k \in [d-1]$ is an integer and $P := \{p_1, \cdots, p_n\}$ is a set of n points in $\mathbb{R}^d$. The function f_0 and the set $F := \{f_i, K_i, \psi_i\}_{i=1}^{n+2}$ of $m := n + 2$ logarithmic barriers are defined as follows:

- $f_0 : \left(\mathbb{R}^{d\times d} \times \mathbb{R}^n\right) \to \mathbb{R}$ is defined by $f_0\,(X, y) := \mathbb{1}^T y$.
- $f_1 : \left(\mathbb{R}^{d\times d} \times \mathbb{R}^n\right) \to \mathbb{R}^{d\times d}$ is defined by $f_1\,(X, y) := -X$.
- $f_2 : \left(\mathbb{R}^{d\times d} \times \mathbb{R}^n\right) \to \mathbb{R}^{d\times d}$ is defined by $f_2\,(X, y) := X - I_d$.
- $f_{i+2} : \left(\mathbb{R}^{d\times d} \times \mathbb{R}^n\right) \to \left(\mathbb{R}^d \times \mathbb{R}\right)$ is defined by $f_{i+2}\,(X, y) := -\,(Xp_i, y_i)$, for every $i \in [n]$.
- $K_1 := K_2 := \mathbb{S}_d^+$ is the positive semi-definite cone (PSD) in $\mathbb{R}^{d\times d}$; see (3).
- $K_{i+2} := \mathbb{Q}_d^+$, for every $i \in [n]$, is the second order cone (SOC) in $\mathbb{R}^d \times \mathbb{R}$; see (4).
- $\psi_1 := \psi_2 := \psi_{\mathbb{S}^{++}}$ as the generalized logarithm $\psi_{\mathbb{S}^{++}} : \mathbb{S}_d^{++} \to \mathbb{R}$ that maps every positive definite matrix $X \in \mathbb{S}_d^{++}$ to $\psi_{\mathbb{S}^{++}}(X) := \ln|X|$, the determinant of its natural logarithm.

- $\psi_i + 2 := \psi_{\mathbb{Q}^{++}}$, for every $i \in [n]$, as the generalized logarithm $\psi_{\mathbb{Q}^{++}} : \mathbb{Q}_d^{++} \to \mathbb{R}$ that maps every (v, t) that satisfies $\|v\|_2 < t$ to $\ln(t^2 - \|v\|_2^2) \in \mathbb{R}$.

The constraint $-f_1(X, y) \in \mathbf{int}\,(K_1)$ implies $X = -f_1(X, y) \in \mathbf{int}\,(K_1) = \mathbb{S}_d^{++}$, i.e., $X \succ 0$. Similarly, the constraint $-f_2(X, y) \in \mathbf{int}\,(K_2)$ implies $I_d - X = -f_2(X, y) \in \mathbf{int}\,(K_1) = \mathbb{S}_d^{++}$, i.e., $X \prec I_d$. Combining these two constraints yields

$$-f_1(X, y) \times (-f_2(X, y)) \in \mathbf{int}\,(K_1) \times \mathbf{int}\,(K_2) \Leftrightarrow 0_{d\times d} \prec X \prec I_d. \tag{31}$$

The constraint $-f_{i+2}(X, y) \in \mathbf{int}\,(K_i)$ implies $(Xp_i, y_i) = -f_{i+2}(X, y) \in \mathbf{int}\,(K_{i+2}) = \mathbb{Q}_d^{++}$, i.e.,

$$-f_{i+2}(X, y) \in \mathbf{int}\,(K_i) \Leftrightarrow \|Xp_i\| < y_i, \tag{32}$$

for every $i \in [n]$.

To obtain a conic instance G for Problem (7), it remains to incorporate the constraints $X^T = X$ and $\mathrm{trace}\,(X) = d - k$. These are linear constraints and correspond to the matrix A and vector b in the Euclidean space, or, in our notation, to the variables $(X, y) \in \mathbb{R}^{d\times d} \times \mathbb{R}^n$. The first two linear equality constraints on (X, y) in (7) are given by $\mathrm{trace}\,(X) = d - k$ and $X - X^T = 0$, which are affine linear matrix equalities. Specifically, set $b_1 = d - k$, $b_2 = 0$, and define linear maps $A_1, A_2 : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}^{d\times d}$ by

$$A_1(X, y) := \sum_{i=1}^{d} X_{i,i}, \quad A_2(X, y) := X - \sum_{i,j\in[d]} X_{j,i} E^{i,j},$$

where $X_{i,j}$ denotes the (i, j)-entry of X (row i, column j), and $E^{i,j} \in \{0, 1\}^{d\times d}$ is zero everywhere except at $E_{i,j} = 1$. The conditions $A_i(X, y) = b_i$ for $i \in \{1, 2\}$ then recover the remaining constraints of Problem (7), namely

$$A(X, y) = b \Leftrightarrow \mathrm{trace}\,(X) = d - k \text{ and } X - X^T = 0. \tag{33}$$

Combining (31), (32) and (33), and substituting $x := (X, y)$ in Definition F.3 yields that the resulting domain of this conic optimization problem is $\mathcal{D}(G)$ defined as

$$\begin{aligned} &\bigcap_{i\in[n+2]} \left\{(X, y) \in \mathbb{R}^{d\times d} \times \mathbb{R}^n \mid -f_i(X, y) \in \mathbf{int}\,(K_i), A_1(X, y) = b_1, A_2(X, y) = b_2\right\} \\ = &\bigcap_{i\in[n+2]} \left\{(X, y) \in \mathbb{R}^{d\times d} \times \mathbb{R}^n \mid \|Xp_i\| < y_i, 0_{d\times d} \prec X \prec I_d, \mathrm{trace}\,(X) = d - k, X = X^T\right\}. \end{aligned} \tag{34}$$

Let

$$(X, y) \in \arg\min_{(X,y)\in\mathcal{D}(G)} f_0(X, y) = \arg\min_{(X,y)\in\mathcal{D}(G)} 1^T y.$$

By (34), the condition $(X, y) \in \mathcal{D}(G)$ entails that $X = X^T$, $\mathrm{trace}\,(X) = d - k$, $0_{d\times d} \prec X \prec I_d$, and $\|Xp_i\| < y_i$ for all $i \in [n]$. Consequently, any such minimizer (X, y) satisfies all feasibility requirements of Problem (7) and also minimizes the same linear objective $1^T y$ over its feasible region. Hence, (X, y) is indeed an optimal solution to Problem (7), which establishes the first assertion of Lemma E.1.

The barrier function $\psi_{\mathbb{S}^+}$ has degree d, while $\psi_{\mathbb{Q}^+}$ has degree 2. These facts are established, for instance, in [Boyd et al., 2004], which also shows that every $(f, K, \psi) \in F$ defines a valid logarithmic barrier. Consequently, the total degree associated with F is

$$\deg(F) = 2d + 2n.$$

This completes the proof of the second assertion in Lemma E.1. □

Let $t_0 > 0$, and G_{t_0} be a t_0-relaxation of G. Hence,

$$\begin{aligned} (X_0, y_0) &\in \arg\inf_{(X,y)\in\mathcal{D}(G)} G_{t_0}(X, y) \\ &= \arg\inf_{(X,y)\in\mathcal{D}(G)} t_0 f_0(x) - \sum_{i=1}^{n+2} \psi_i(-f_i(X, y)) \\ &= \arg\inf_{(X,y)\in\mathcal{D}(G)} t_0 1^T y - \ln|X| - \ln|I_d - X| - \sum_{i=1}^{n} \ln(y_i^2 - \|Xp_i\|_2^2) \end{aligned}$$

see Definition F.3.

Proposition E.2. *Consider $\varepsilon \in (0,1)$, and let $x^*_{t_0} = (X_{t_0}, y_{t_0})$ satisfy $\lambda\left(G_{t_0}, x^*_{t_0}\right) \leq 0.01$. There exists an algorithm that computes an additive ε-approximation to the minimum of* $\operatorname{relaxksm}\left(k, \{p_i\}_{i=1}^n, w\right)$*, referred to as Problem (7), in time*

$$\begin{aligned} &O\left(\left(n+d^2\right)^\omega \log_\mu\left(\frac{\sum_{i=1}^{n+2}\theta_i + 1}{\varepsilon t_0}\right)\left(\sum_{i=1}^{n+2}\theta_i \cdot (\mu - \log\mu)\right)\right) \\ =&O\left(\left(n+d^2\right)^\omega (n+d)\ln\frac{n+2d}{\varepsilon t_0}\right). \end{aligned} \tag{35}$$

Proof. Recall that the collections $\{f_i\}_{i=0}^{n+2}$ and $\{\psi_i\}_{i=1}^{n+2}$ are families of self-concordant functions. Since each f_i for $i = 1, \ldots, n+2$ is linear, the composed functions $\{\psi_i(-f_i)\}_{i=1}^{n+2}$ also form a self-concordant family. Invoking Theorem F.9 with the specific choices $\mu = 2$, $t := t_0$, and $x^*_t := x^*_{t_0}$, we obtain that there exists an explicit algorithm that, in time given by (35), produces an additive ε-approximation to the optimal value of Problem (7). □

We introduce an integer Δ such that $\ln\Delta$ represents the word size, namely, the number of bits allocated to store entries in our finite-precision setting. Consequently, we assume that the absolute value of every non-zero entry of our input variables lies in the range $\left[\frac{1}{\Delta}, \Delta\right]$.

The next lemma yields an initial point with a provable quality guarantee; by plugging the objective function of G_{t_0} into this result, we derive an upper bound on the distance between this point and the constrained minimizer of G_{t_0}. Consequently, invoking Lemma F.8, we see that the running time needed to compute $(X_{t_0}, y_{t_0}) \in \mathcal{D}(G)$ satisfying $\lambda\left(G_{t_0}, (X_{t_0}, y_{t_0})\right) \leq 0.01$ is finite and can be explicitly bounded.

Lemma E.3 (Initial approximation)**.** *Let $P := (p_i)_{i=1}^n$ be an n-tuple of $n \geq 1$ points in $\mathbb{R}^d$, and let $G(P) = (f_0, F, A, b)$ be its conic instance as defined in Lemma E.1. Let*

$$t_0 := \frac{2n}{\sum_{i=1}^n \sqrt{\|X_0 p_i\|_2^2 + e}}. \tag{36}$$

A pair $(X_{t_0}, y_{t_0}) \in \mathcal{D}(G)$ that satisfies

$$\lambda\left(G_{t_0}, (X_{t_0}, y_{t_0})\right) \leq 0.01 \tag{37}$$

can be computed in $O\left(\left(n+d^2\right)^\omega (n + d\log d)\right)$ time.

Proof. Let $X_0 := \frac{d-k}{d} I_d$ be a matrix in $\mathbb{R}^{d\times d}$. Let $z_0 := (y_{0_1}, \ldots, y_{0_n}) \in \mathbb{R}^n$ be a vector whose ith coordinate is

$$y_{0i} := \sqrt{\|X_0 p_i\|_2^2 + e}, \tag{38}$$

for every $i \in [n]$. Clearly $(X_0, y_0) \in \mathcal{D}(G)$.

Let

$$(X^*, y^*) := (X^*_{t_0}, y^*_{t_0}) \in \arg\inf_{(X,y)\in\mathcal{D}(G)} G_{t_0}(X, y), \tag{39}$$

and

$$\rho := G_{t_0}(X_0, y_0) - G_{t_0}(X^*, y^*).$$

Substituting $t := t_0$ and $z_0 := (X_0, y_0)$ in Lemma F.8 yields that the desired pair $z_{t_0} := (X_{t_0}, y_{t_0})$ that satisfies

$$\lambda\left(G_{t_0}, (X_{t_0}, y_{t_0})\right) \leq 0.01,$$

as in (37), can be computed in time

$$O\left(\rho \cdot \left(n+d^2\right)^\omega\right). \tag{40}$$

It is left to bound

$$\begin{aligned}\rho &= G_{t_0}(X_0, y_0) - G_{t_0}(X^*, y^*) \\ &= t_0 1^T y_0 - \sum_{i=1}^{n} \ln\left(y_{0i}^2 - \|X_0 p_i\|_2^2\right) - \ln|X_0| - \ln|I_d - X_0| \qquad (41) \\ &\quad - \left(t_0 1^T y^* - \sum_{i=1}^{n} \ln\left(y_i^{*2} - \|X^* p_i\|_2^2\right) - \ln|X^*| - \ln|I_d - X^*|\right), \qquad (42)\end{aligned}$$

where (41) and (42) holds by substituting $(X, y) := (X_0, y_0)$ and $(X, y) := (X^*, y^*)$, respectively, in (30), together with $t := t_0$. We now bound each expression as follows.

By (38),

$$\sum_{i=1}^{n} \ln\left(y_{0i}^2 - \|X_0 p_i\|_2^2\right) = \sum_{i=1}^{n} \ln e = n. \qquad (43)$$

We have $|X_0| = \left(\frac{d-k}{d}\right)^d$ and $|I_d - X_0| = \left(\frac{k}{d}\right)^d$, and thus

$$\ln|X_0| + \ln|I_d - X_0| = -d \ln \frac{d^2}{k\,(d-k)} \geq -d \ln(d^2) = -2d \ln d. \qquad (44)$$

By the monotonicity of ln, and since $\ln\left(x^2\right) \leq x$ for every $x > 0$, respectively, we have

$$\sum_{i=1}^{n} \ln\left(y_i^{*2} - \|X^* p_i\|_2^2\right) \leq \sum_{i=1}^{n} \ln\left(y_i^{*2}\right) \leq 1^T y^* \qquad (45)$$

Since $X^* \in \mathcal{D}(G)$, it satisfies $0 \preceq X^* \preceq I$, and thus its eigenvalues $\{\sigma_i\}_{i=1}^d$ are in $[0, 1]$, and thus their multiplication is $|X^*| \leq 1$. Similarly, the eigenvalues $\{1 - \sigma_i\}_{i=1}^d$ of $I_d - X^*$ are in $[0, 1]$. Hence,

$$\ln|X^*| + \ln|I_d - X^*| \leq \ln 1 + \ln 1 = 0. \qquad (46)$$

Plugging (43), (44), (44), (45) and (46) in (42) and (42) yields

$$\rho \leq t_0 1^T y_0 - n + \left(2d \ln d + 1^T y^*\right). \qquad (47)$$

Finally, we bound the last term $1^T y^*$ by

$$\begin{aligned}1^T y^* (t_0 - 1) &= t_0 1^T y^* - 1^T y^* \\ &\leq t_0 1^T y^* - \sum_{i=1}^{n} \ln\left(y_i^{*2} - \|X^* p_i\|_2^2\right) \qquad (48) \\ &\leq t_0 1^T y^* - \sum_{i=1}^{n} \ln\left(y_i^{*2} - \|X^* p_i\|_2^2\right) - \ln|X^*| - \ln|I_d - X^*| \qquad (49) \\ &\leq t_0 1^T y_0 - \sum_{i=1}^{n} \ln\left(y_{0i}^2 - \|X_0 p_i\|_2^2\right) - \ln|X_0| - \ln|I_d - X_0| \qquad (50) \\ &\leq t_0 1^T y_0 - n + 2d \ln d, \qquad (51)\end{aligned}$$

where (48) is established using (45), (49) holds by (46), (50) is by the optimality of (X^*, y^*) in (39), and the last inequality holds by (43) and (44).

Dividing (51) by $t_0 - 1$, and substituting in (47) yields the final bound on (41),

$$\begin{aligned}\rho &\leq t_0 1^T y_0 - n + \left(2d \ln d + 1^T y^*\right) \\ &\leq t_0 1^T y_0 - n + \left(2d \ln d + \frac{t_0 1^T y_0 - n + 2d \ln d}{t_0 - 1}\right) = n + 2d \ln d + \frac{n + 2d \ln d}{t_0 - 1} \\ &\leq 2n + 4d \ln d,\end{aligned}$$

where the last inequality holds since

$$t_0 = \frac{2n}{1_n^T y_0} \geq \frac{2n}{n\sqrt{e}} = \frac{2}{\sqrt{e}} > 1.$$

Substituting (52) in (40) yields that the pair (X_{t_0}, y_{t_0}) can be computed in

$$O\left(\left(n + d^2\right)^{\omega} (n + d \ln d)\right).$$

□

By Proposition E.2 and Lemma E.3, we can bound the total time complexity of our suggested kSM algorithm.

Theorem E.4 (running time)**.** *Let $k \in [d-1]$ and $\Delta \geq 1$ be integers, and $P := (p_i)_{i=1}^n \subseteq \mathbb{R}^d_\Delta$. Let $S \subseteq \mathbb{R}^d$ be the output of a call to k*SM-APPROX *(P, k); see Algorithm 1. Then the projection matrix of S can be computed in time*

$$O(n + d^2)^{\omega}(n + d) \ln\left(\Delta \cdot (n + 2d)\right). \tag{52}$$

Proof. We notice that by (36) and the definition of Δ we obtain that

$$t_0 \geq \frac{2n}{n\sqrt{d \cdot \Delta}} = \Omega\left(\frac{1}{\sqrt{d \cdot \Delta}}\right). \tag{53}$$

By substituting (53) in Proposition E.2, we obtain the given (X_{t_0}, y_{t_0}) calculated as in Lemma E.3, we obtain the the total running time for obtaining an additive ε approximation to Problem (7) is

$$\begin{aligned} & O\left(\left(n + d^2\right)^{\omega}\left((n + d) \ln \frac{\Delta \cdot (n + 2d)}{\varepsilon d} + n + d \ln d\right)\right) \\ =& O\left(\left(n + d^2\right)^{\omega} (n + d) \ln \frac{\Delta \cdot (n + 2d)}{\varepsilon}\right). \end{aligned}$$

□

F CENTRAL PATH METHOD

In this section, we review key concepts from convex optimization, focusing on the *Central Path Method*, which will serve as a core component of our algorithm in Section 5.

Our approach reformulates the non-convex k-subspace median problem as a convex optimization problem defined over two cones. The first is the cone of positive semi-definite (PSD) matrices within the space of symmetric matrices in $\mathbb{R}^{d \times d}$ (see Fig. 1); the second is the second-order cone (SOC) in $\mathbb{R}^{d+1}$. The associated dual optimization frameworks are known as SDP (semi-definite programming) and SOCP (second-order cone programming), and their joint use is referred to as SDP-SOCP [Nesterov, 1994, Mittelmann, 2003, Molzahn and Hiskens, 2015].

We study the central path method for solving a convex optimization problem defined on $\mathbb{R}^n$ for some integer $n \geq 1$. The problem includes $d \geq 1$ linear equality constraints and $m \geq 1$ conic constraints. Our goal is to compute an additive ε-approximation of the optimal value, for a prescribed accuracy level $\varepsilon \in (0, 1)$. The ambient space $\mathbb{R}^n$ can be substituted by any vector space that is linearly isomorphic to it. Typical instances are finite Cartesian products of vector spaces, as well as spaces of real matrices of dimensions $a \times b$ with integers $a, b \geq 1$.

Definition F.1 (proper cone [Boyd et al., 2004])**.** *The interior of a set $C \subseteq \mathbb{R}^n$ is denoted by*

$$\mathbf{int}\,(C) := \{x \in C | \exists \varepsilon > 0 \colon \{y \in \mathbb{R}^n | \|x - y\|_2 \leq \varepsilon\} \subseteq C\}.$$

A set $K \subseteq \mathbb{R}^n$ is called a proper cone *if it satisfies Properties (i)-(iii) as follows.*

(i) For every $x, y \in K$ and $\theta_1, \theta_2 \in \mathbb{R}_+$ we have $\theta_1 x + \theta_2 y \in K$.

(ii) If $x \in K$ and $-x \in K$ then $x = 0$.

(iii) K is closed, and $\mathbf{int}(K) \neq \emptyset$.

Definition F.2 (logarithm barrier [Boyd et al., 2004])**.** *Let $k \geq 1$ be an integer, $K \subseteq \mathbb{R}^k$ be a proper cone and $\theta > 0$. A function $\psi : \mathbf{int}(K) \to \mathbb{R}$ is a* generalized logarithm *of degree $\theta \geq 0$ if it satisfies the following properties:*

(i) ψ is concave, closed, twice continuously differentiable.

(ii) Every $y \in \mathbf{int}(K)$ and $s > 0$ satisfy $\psi(sy) = \psi(y) + \theta \ln s$.

(iii) Every $x \in \mathbb{R}^k$ satisfies $x^T \nabla^2 \psi(y)\, x < 0$, where $\nabla^2 \psi(y)$ denotes the Hessian matrix of $\psi(y)$.

Let $f : \mathbb{R}^n \to \mathbb{R}^k$ be a function that satisfies, for every $x, y \in \mathbb{R}^n$ and $\beta \in [0,1]$,

$$\beta - f(x) - (1-\beta) f(y) + f(\beta x + (1-\beta) y) \in K.$$

The 3-tuple (f, K, ψ) is a logarithmic barrier *in $\mathbb{R}^n$.*

Definition F.3 (t-relaxation)**.** *Let $f_0 : \mathbb{R}^n \to \mathbb{R}$, and $F = \{f_i, K_i, \psi_i\}_{i=1}^m$ be a set of $m \geq 1$ logarithmic barriers in $\mathbb{R}^n$. The* degree *of F is $\deg(F) := \sum_{i=1}^m \theta_i$, where θ_i is the degree of f_i, for $i \in [m]$. For a given matrix $A \in \mathbb{R}^{d \times n}$ and a vector $b \in \mathbb{R}^n$ we define $G := (f_0, F, A, b)$ as the* conic instance *for the optimization problem*

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f_0(x) \\ s.t. \quad & Ax = b \\ & -f_i(x) \in K_i \quad \forall i \in [m]. \end{aligned} \tag{54}$$

*For $t \geq 0$, we introduce the t-*relaxation *of G as the function $G_t : \mathcal{D}(G) \to \mathbb{R}$ that assigns to each $x \in \mathcal{D}(G)$, where*

$$\mathcal{D}(G) := \bigcap_{i \in [m]} \{x \in \mathbb{R}^n \mid -f_i(x) \in \mathbf{int}(K_i) \text{ and } Ax = b\}$$

the value

$$G_t(x) := t f_0(x) - \sum_{i=1}^m \psi_i\big(-f_i(x)\big).$$

Definition F.4 (self-concordant)**.** *Let $H \subseteq \mathbb{R}$. A three-times continuously differentiable function $f : H \to \mathbb{R}$ is called* self-concordant *if, for every $x \in H$,*

$$|f'''(x)| \leq 2\, f''(x)^{\frac{2}{3}}.$$

More generally, let $n \geq 1$ be an integer and $H \subseteq \mathbb{R}^n$. A function $h : H \to \mathbb{R}$ is self-concordant if, for every $x \in H$ and $v \in \mathbb{R}^n$, the univariate function $g_{x,v}$ defined on $\{t \in \mathbb{R} | x + vt \in H\}$ by $g_{x,v}(t) = h(x + vt)$ is self-concordant.

$$\psi(-f) : \{x \in \mathbb{R}^n \mid -f(x) \in \mathbf{int}(K)\} \to \mathbb{R}$$

is self-concordant, and if, in addition, f_0 itself is a self-concordant function.

In what follows, $\psi_i(-f_i) : \{x \in \mathbb{R}^n \mid -f(x) \in \mathbf{int}(K_i)\} \to \mathbb{R}$ denotes the composition $\psi_i \circ (-f_i)$ of the functions ψ_i and $-f_i$, which assigns to each $x \in \mathbb{R}^n$ with $-f(x) \in \mathbf{int}(K_i)$ the value $\psi_i(-f_i(x))$.

Definition F.5 (Newton's Decrement [Nesterov, 1994])**.** *Let $G = (f_0, F, A, b)$ be a self-concordance conic instance and $t > 0$. The* Newton's Decrement *of the t-relaxation G_t at $x \in \mathcal{D}(G)$ is*

$$\lambda(G_t, x) = \sqrt{(\nabla G_t(x))^T \left(\nabla^2 G_t(x)\right)^{-1} (\nabla G_t(x))}.$$

Proposition F.6 ([Nesterov, 1994])**.** *Let $G = (f_0, F, A, b)$ be a self-concordance conic instance, $t > 0$ and $y \in \mathcal{D}(G)$. If $\lambda(G_t, y) \leq 0.01$ then*

$$\inf_{x \in \mathcal{D}(G)} G_t(x) \geq G_t(y) + \lambda^2(G_t, y).$$

Lemma F.7 (Reduction to t-relaxation [Nesterov, 1994]). *Let $G = (f_0, F, A, b)$ be a self-concordance conic instance, $t > 0$ and $x_t \in \mathcal{D}(G)$. If $\lambda(G_t, x_t) \leq 0.01$ then*

$$f_0(x_t) \leq \inf_{x \in \mathcal{D}(G)} f_0(x) + O\left(\frac{\deg(F)+1}{t}\right).$$

We denote the computational complexity of inverting an $n \times n$ matrix over $\mathbb{R}$ by $O(n^\omega)$.

Lemma F.8 (Inner Newton iterations [Nesterov, 1994]). *Let $G := (f_0, F, A, b)$ be a self-concordant conic instance, let $t > 0$, and let $z_0 \in \mathcal{D}(G)$. There exists an algorithm that takes (G, t, z_0) as input and returns $z_t \in \mathcal{D}(G)$ such that $\lambda(G_t, z_t) \leq 0.01$. For every $\mu > 1$, its running time is*

$$n^\omega \deg(F) \cdot O(\mu - 1 - \log \mu),$$

if $\lambda\left(G_{\frac{t}{\mu}}, z_0\right) \leq 0.01$, and

$$n^\omega \cdot O\left(G_t(z_0) - \inf_{x \in \mathcal{D}(G)} G_t(x)\right).$$

otherwise.

Algorithm 2: Central-Path$(G, \mu, t_0, x_0, \varepsilon)$; see Theorem F.9.

Input : A self concordance conic instance $G = (f_0, F, A, b)$ in $\mathbb{R}^n$, $\mu > 1$, an integer $t_0 > 0$, $x_0 \in \mathcal{D}G$ as in Theorem F.9, and $\varepsilon \in (0, 1)$.

Output: $x_t \in \mathcal{D}(G)$.

Let

$$s \in \Theta\left(\log\left(\frac{\deg(F)+1}{\varepsilon t_0}\right) / \log \mu\right)$$

be an integer whose exact value can be determined from the proof of Lemma F.7.

for $j := 1$ *to* s **do**

  Compute $x_j \in \mathcal{D}(G)$ that satisfies $\lambda\left(G_{\mu^j t_0}, x_j\right) \leq 0.01$.

  `// Possibly by substituting` $z_0 := x_{j-1}$ `in Lemma F.8.`

return x_s

By Lemma F.7 and Lemma F.8, Central-Path $(G, \mu, t_0, \varepsilon)$ (Algorithm 2) produces an additive ε-approximation of $\inf_{x \in \mathcal{D}(G)} f_0(x)$ for self-concordant conic instances.

Theorem F.9 (Central path method [Nesterov, 1994]). *Let $G = (f_0, F, A, b)$ be a self-concordant conic instance, and fix parameters $\mu > 1$, $t_0 > 0$, and $\varepsilon \in (0, 1)$. Assume $x_0 \in \mathcal{D}(G)$ satisfies $\lambda\left(G_{t_0}, x_0\right) \leq 0.01$, and let x^* denote the result of running* Central-Path$\left(G, \mu, t_0, x_0, \varepsilon\right)$*; see Algorithm 2. Then $x^* \in \mathcal{D}(G)$ and*

$$f_0(x^*) \leq \inf_{x \in \mathcal{D}(G)} f_0(x) + \varepsilon.$$

Moreover, x^* can be computed in time

$$n^\omega \cdot \deg(F) \cdot O\left(\log\left(\frac{\deg(F)+1}{\varepsilon t_0}\right)\right).$$