Inexact proximal splitting methods for Euclidean distance matrix optimization

Xin Jiang* Chaorui Yao†

March 9, 2024

Abstract

This paper presents an inexact first-order proximal splitting method for matrix optimization problems involving Euclidean distance matrices. In the proposed method, the large number of linear (in)equality constraints in the problems are handled efficiently by exploiting data structure. The presented method also replaces the Euclidean distance matrix projection at each iteration with a low-rank approximation, and the introduced inexactness errors are compensated by an adaptive choice of stepsizes. We test the presented algorithm on non-metric multidimensional scaling and maximum variance unfolding problems. In these problems, the proposed low-rank approximation helps construct a low-dimensional embedding desirable for practical purposes. Numerical results also validate the efficiency and scalability of the presented algorithm.

1 Introduction

This paper presents a first-order proximal splitting method for a class of large-scale optimization problems involving Euclidean distance matrices (EDMs). These problems often include a large set of linear equality constraints, and have EDMs as the optimization variable. A symmetric $n \times n$ matrix X is a *Euclidean distance matrix* (EDM) if its entries can be expressed as squared pairwise Euclidean distances of a set of points: there exist vectors $y_1, \ldots, y_n$ such that

$$X_{ij} = \|y_i - y_j\|^2, \quad i, j = 1, \ldots, n. \tag{1}$$

The set of points $\{y_1, \ldots, y_n\} \subseteq \mathbb{R}^d$ is called a *realization* (or *embedding*) of the EDM X. The set of $n \times n$ EDMs forms a convex cone $\mathbb{D}^n$.

Properties and theories of EDMs have been extensively studied in linear algebra and matrix analysis (see, *e.g.*, [18,35]). Optimization over the EDM cone also has a wide variety of applications in graph theory [4], bioinformatics [6], signal processing [7, 22], machine learning [1, 13, 37, 55, 58], *etc.* In particular, many dimension reduction techniques are developed to build a low-dimensional embedding that satisfies certain application-specific constraints [1, 57]. In these applications, the number of linear (in)equality constraints scales very quickly with the matrix size. Thus, even when the matrix size is tractable, the huge number of linear constraints might prevent the use of general-purpose solvers. Moreover, classical ML approaches often use the embedding as the optimization

*Department of Industrial and Systems Engineering, Lehigh University. Email: xjiang@lehigh.edu.
†Department of Electrical and Computer Engineering, UCLA. Email: chaorui@ucla.edu.

variable and formulate the problem as a semidefinite program (SDP). But it might be more efficient to handle the EDM structure directly. In this paper, we use the EDM matrix as the variable, and reformulate the ML model as an EDM optimization problem. We show that such an EDM reformulation facilitates the use of proximal splitting methods. These algorithms iteratively decompose the EDM optimization problem, and separately handle the EDM conic constraint and other application-specific constraints. At each iteration, the conic constraint requires projection onto the (reduced) EDM cone, while the linear constraints can be handled efficiently by exploiting specific problem structure. Therefore, proximal methods can address the scalability concern mentioned in [1, 57], and are able to solve large-scale EDM optimization problems that cannot be handled by general-purpose solvers.

Moreover, in these problems, a solution corresponding to a low-dimensional embedding is of great interest. Thus, a regularization term is often added to promote low rank [1, 37, 59]. In comparison, for the EDM reformulation, it is tempting to replace the EDM projection at each iteration of the proximal method with a low-rank approximation. The modified algorithm then generates a sequence of low-rank matrices, and retrieves a low-dimensional realization at optimum. On the other hand, however, such modification introduces inexactness errors and in our case, the errors are not necessarily summable. Thus, classical analyses of inexact proximal algorithms cannot provide convergence guarantees, and new techniques and analyses are needed for the modified (inexact) proximal method.

Contributions. In this paper, we study an inexact primal–dual first-order proximal method. In the presented algorithm, the inexactness errors are not necessarily summable (as required in most previous work on inexact optimization methods), and are compensated by an adaptive choice of stepsizes at each iteration. The proposed algorithm is applicable to a wide variety of convex optimization problems (not limited to EDM optimization), and is of independent interest as a first-order proximal method.

As an example, we apply the proposed method to non-metric multidimensional scaling (NMDS) and maximum variance unfolding (MVU), two classical ML models for dimension reduction. The ML models are formulated as EDM optimization problems, and the presented proximal method handles the EDM conic constraint and the linear constraints separately at each iteration. In particular, the method can exploit specific problem structure to handle the large number of linear constraints efficiently. Moreover, motivated by the need of a low-rank EDM solution, our method replaces the EDM projection at each iteration with a low-rank approximation. Although such modification introduces inexactness errors, convergence is still guaranteed by our analysis. Numerical experiments are conducted to verify the efficiency and scalability of the proposed algorithm. In particular, our method solves large-scale EDM problems that cannot be handled by general-purpose solvers, and returns a low-rank EDM solution desirable for ML applications.

Related work. The research on EDMs dates back to the early 1900s [47, 61], and is of great importance for both theoretical and practical purposes; see [18, 22, 35] for recent surveys. Besides the aforementioned applications, EDMs have been extensively used in sensor network localization [3, 7, 21]. Customized algorithms for EDM optimization problems range from interior-point methods [6, 7] to various first-order algorithms [19, 20, 60, 62, 63].

First-order primal–dual proximal methods can solve a general form of nonsmooth, convex optimization problems, and their development has become an active research area; see [14, 16] for recent

surveys. The algorithm studied in this paper extends the well-known primal–dual hybrid gradient (PDHG) method [10, 24, 44] to include line search and inexact evaluation of proximal operators. Line search in PDHG is first proposed in [39], and soon becomes popular due to its flexibility and promising performance. For example, the line search technique is extended for generalized proximal operators [9,31] and for a more general type of problems [32]. Inexact prox-evaluation is also extensively studied for many proximal methods [5, 23, 38, 46, 50], and various definitions and conditions are proposed for inexact proximal operators; see [45] for a brief summary and comparison.

Outline. The rest of the paper is organized as follows. Section 2 summarizes the basic properties of EDMs. In Section 3, we present two machine learning applications and reformulate them into EDM optimization problems. For each example problem, we explain how the main steps in primal–dual proximal methods can be simplified and customized by exploiting problem structure and properties of EDMs. The proposed inexact preconditioned primal–dual hybrid gradient method (inexact PDHG) and its convergence are discussed in Section 4, and Section 5 contains results of numerical experiments.

2 Euclidean distance matrices

In this section, we review some basic properties of Euclidean distance matrices (EDMs). In particular, we introduce two parametrizations (or representations) of EDMs that will help us reformulate the optimization problems in Section 3. For more details and applications of EDMs, we refer interested readers to recent surveys [22, 35] and the book [18].

We denote by $\mathbb{S}^n$ the set of symmetric $n \times n$ matrices, and by $\mathbb{S}^n_+$ the set of symmetric, positive semidefinite (PSD) $n \times n$ matrices. The inequality $X \succeq 0$ ($X \succ 0$) means X is PSD (PD). We use the notation $\langle x, y\rangle = x^T y$ for the standard inner product of vectors x and y, and $\|x\| = \langle x, x\rangle^{1/2}$ for the Euclidean norm of a vector x. For any positive definite matrix P, the quadratic form for vectors x and y is $\langle x, y\rangle_P = \langle x, Py\rangle$, and the associated quadratic norm is $\|x\|_P = \langle x, x\rangle_P^{1/2}$. The **diag** operator takes the diagonal elements of a matrix and forms a column vector. The notation $\mathbb{1}$ indicates the vector of all ones with compatible size.

Gram matrix parametrization. Expanding the square in the definition (1) gives $X_{ij} = y_i^T y_i + y_j^T y_j - 2y_i^T y_j$, and then it follows that in matrix notation, X is an EDM if and only if

$$X = \mathbf{diag}(Y)\mathbb{1}^T + \mathbb{1}\,\mathbf{diag}(Y)^T - 2Y \tag{2}$$

for some PSD matrix Y (*i.e.*, the Gram matrix $Y = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}^T \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}$). Hence, we have the following characterization of EDMs: $X \in \mathbb{D}^n$ if and only if there exists a matrix $Y \in \mathbb{S}^n$ such that

$$Y \succeq 0, \qquad \mathbf{diag}(Y)\mathbb{1}^T + \mathbb{1}\,\mathbf{diag}(Y)^T - 2Y = X. \tag{3}$$

We refer to the matrix Y that satisfies (3) as a *Gram matrix* of X.

Schoenberg's parametrization. An equivalent characterization of EDMs is due to Schoenberg [47, 48]: X is an EDM if and only if $\mathbf{diag}(X) = 0$ and $c^T X c \leq 0$ for all c with $\mathbb{1}^T c = 0$.

The second condition states that X is negative semidefinite on the orthogonal complement of the n-vector $\mathbb{1}$. Schoenberg's conditions can be rewritten in the matrix form as

$$\mathbf{diag}(X) = 0, \qquad V^T X V \preceq 0, \tag{4}$$

where V is any matrix whose columns span the orthogonal complement of $\mathbb{1}$, *i.e.*, $\mathrm{range}(V) = \{c \in \mathbb{R}^n \mid \mathbb{1}^T c = 0\}$. A common choice of such a matrix V is $V = I - (1/n)\mathbb{1}\mathbb{1}^T \in \mathbb{S}^n$. With this choice, it can be shown that $Y = -(1/2)V^T X V$ is a valid Gram matrix of X, and the Gram matrix satisfies $\mathbb{1}^T Y \mathbb{1} = 0$; see, *e.g.*, [56, §11.2].

We denote by $\mathbb{D}_0^n$ the *reduced* EDM cone, *i.e.*, the set of matrices that satisfies the second of Schoenberg's conditions:

$$\mathbb{D}_0^n = \{X \in \mathbb{S}^n \mid V^T X V \preceq 0\}.$$

In other words, $\mathbb{D}^n = \{X \in \mathbb{D}_0^n \mid \mathbf{diag}(X) = 0\}$. Projection onto the reduced EDM cone $\mathbb{D}_0^n$ is much more easier than projection onto $\mathbb{D}^n$, and the computational complexity is dominated by an eigen-decomposition [25, 27, 29, 53]. To see this, consider the optimization problem

$$\begin{array}{ll} \text{minimize} & \|X - A\|_F^2 \\ \text{subject to} & X \in \mathbb{D}_0^n \end{array} \tag{5}$$

with variable $X \in \mathbb{S}^n$ and data $A \in \mathbb{S}^n$. We first define

$$U = \frac{1}{n + \sqrt{n}} \begin{bmatrix} (1+\sqrt{n})\mathbb{1}^T \\ \mathbb{1}\mathbb{1}^T - (n+\sqrt{n})I \end{bmatrix} \in \mathbb{R}^{n \times (n-1)}, \qquad e = \frac{1}{\sqrt{n}}\mathbb{1} \in \mathbb{R}^n, \qquad \widetilde{U} = \begin{bmatrix} U & e \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

By definition $U \in \mathbb{R}^{n \times (n-1)}$ has orthonormal columns that span the orthogonal complement of $\mathbb{1}$, and $\widetilde{U}$ is an $n \times n$ orthogonal matrix. Then the constraint $X \in \mathbb{D}_0^n$ in (5) can be replaced by $U^T X U \preceq 0$, and the optimal solution is the projection of $A \in \mathbb{S}^n$ on $\mathbb{D}_0^n$, given by

$$\Pi_{\mathbb{D}_0^n}(A) = \widetilde{U} \begin{bmatrix} -\Pi_{\mathbb{S}_+^{n-1}}(-U^T A U) & U^T A e \\ e^T A U & e^T A e \end{bmatrix} \widetilde{U}^T,$$

where $-\Pi_{\mathbb{S}_+^{n-1}}(-U^T A U) = \Pi_{\mathbb{S}_-^{n-1}}(U^T A U)$ is the projection of $U^T A U$ on the negative semidefinite cone, obtained by replacing the positive eigenvalues of $U^T A U$ by zero.

Vector notation. In this paper it would be more convenient to represent an matrix variable by a vector variable. For a symmetric $n \times n$ matrix X, we define

$$\mathbf{vec}(X) = \Big(X_{11}, X_{22}, \ldots, X_{nn}, \sqrt{2}X_{21}, \ldots, \sqrt{2}X_{n1}, \sqrt{2}X_{32}, \ldots, \sqrt{2}X_{n2}, \ldots, \sqrt{2}X_{n,n-1}\Big).$$

The function **vec** maps X to a vector of length $p = n(n+1)/2$ that contains the lower triangular entries of X. The first n elements in $\mathbf{vec}(X)$ contains the diagonal elements of X, and the strictly lower triangular entries are scaled by $\sqrt{2}$. The scaling ensures that

$$\mathbf{tr}(XY) = \langle \mathbf{vec}(X), \mathbf{vec}(Y) \rangle, \qquad \|X\|_F = \|\,\mathbf{vec}(X)\|.$$

Using $x = \mathbf{vec}(X)$ and Schoenberg's parametrization (4), we can rewrite the EDM conic constraint $X \in \mathbb{D}^n$ as $x \in \mathbf{vec}(\mathbb{D}_0^n)$ and $Cx = 0$, where $C = \begin{bmatrix} I_n & 0 \end{bmatrix} \in \mathbb{R}^{n \times p}$.

3 Proximal splitting methods for EDM optimization

3.1 Problem formulation

The optimization problems studied in this paper arise from machine learning and will be formulated into the canonical form

$$\text{minimize} \quad f(x) + g(Ax), \tag{6}$$

where f, g are closed convex functions and potentially nonsmooth. This general problem is a useful formulation for many important structure arising from various large-scale applications in machine learning, signal and image processing [11, 16, 26, 33, 42].

In the general problem (6), the structure of the nonsmooth function f (and g) is often exploited via its *(preconditioned) proximal operator*:

$$\mathbf{prox}_f^P(y, a) = \operatorname*{argmin}_x \left(f(x) + \langle a, x \rangle + \tfrac{1}{2}\|x - y\|_P^2 \right),$$

where $P \succ 0$ is positive definite. When $P = I$, it reduces to the classical proximal operator [40], and the preconditioner P is added to better capture the structure of f or to reduce the computational complexity. By definition, $\hat{x} = \mathbf{prox}_f^P(y, a)$ if and only if

$$f(u) \geq f(\hat{x}) + \langle u - \hat{x}, P(y - \hat{x}) + a \rangle, \quad \text{for all } u \in \mathbf{dom}\, f. \tag{7}$$

With the proximal operators of the objectives, the problem (6) can be solved efficiently by splitting (or decomposition) methods. These methods iteratively decompose a large-scale optimization problem into smaller, simpler problems and then solve them separately. A well-known proximal splitting method for (6) is the preconditioned primal–dual hybrid gradient (PrePDHG) method [10, 24, 43]:

$$\bar{x}^{(k+1)} = x^{(k)} + \theta(x^{(k)} - x^{(k-1)}) \tag{8a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^Q(z^{(k)}, -\sigma A\bar{x}^{(k+1)}) \tag{8b}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^P(x^{(k)}, \tau A^T z^{(k+1)}), \tag{8c}$$

where $g^*(z) = \sup_y (\langle y, z \rangle - g(y))$ is the convex conjugate of g. When $\theta = 1$, the stepsizes τ, σ must satisfy $\sqrt{\sigma\tau}\|A\| \leq 1$, where the operator norm of A is defined as $\|A\| = \sup_{u \neq 0} \|Au\|_{Q^{-1}}/\|u\|_P$. Convergence analysis with these conditions can be found in [10, 12, 15]. Other details and discussion on the algorithm are presented in Section 4. In the remainder of this section, we focus on the EDM reformulation of some machine learning problems and how the reformulation can be solved efficiently by (8). In particular, we explain how the basic iterations (8) can be simplified or customized by exploiting problem structure or properties of EDMs.

3.2 Generalized non-metric multidimensional scaling

Non-metric multidimensional scaling (NMDS) is a classical problem in statistics and machine learning [17, 36, 49], and it aims to find a low-dimensional realization such that the Euclidean distances between the realization points satisfy a pre-determined ordering. In this section, we consider a generalization of NMDS which is able to handle incomplete or contradictory ordering [1]. The

generalized NMDS problem can be formulated as the following SDP:

$$\begin{array}{ll} \text{minimize} & \sum\limits_{(i,j,s,t)\in\Omega} w_{ijst} + \lambda\,\mathbf{tr}(Y) \\ \text{subject to} & Y_{ss} - 2Y_{st} + Y_{tt} - Y_{ii} + 2Y_{ij} - Y_{jj} \geq 1 - w_{ijst} \quad \text{for } (i,j,s,t)\in\Omega \\ & w_{ijst} \geq 0 \quad \text{for } (i,j,s,t)\in\Omega \\ & \mathbb{1}^T Y \mathbb{1} = 0, \;\; Y \succeq 0 \end{array} \tag{9}$$

with variables $Y \in \mathbb{S}^n$ and $w \in \mathbb{R}^m$ (where we denote $m = |\Omega|$). The variable Y is the Gram matrix of the realization $\{y_1, \ldots, y_n\} \subseteq \mathbb{R}^d$ (with $n \gg d$). Very often in applications, a desirable dimension d is known in advance, and thus a rank-d solution of (9) yields a realization in $\mathbb{R}^d$. In view of this, the second term in the objective, $\mathbf{tr}\,Y = \sum_i \lambda_i(Y)$, is the sum of eigenvalues and serves as a convex regularization promoting low rank (or a convex relaxation for the low-rank constraint) [1]. Moreover, the first two sets of inequalities in (9) describe the *soft* ordering constraints, and very often in applications, one has $m = O(n^2)$. Note each w_{ijst} is constrained by only two lower bounds, so at the optimum,

$$w_{ijst} = \max\{0, 1 - Y_{ss} + 2Y_{st} - Y_{tt} + Y_{ii} - 2Y_{ij} + Y_{jj}\}, \quad \text{for } (i,j,s,t)\in\Omega.$$

Since the variable Y in (9) is the Gram matrix in an EDM parametrization, the problem is more easily written in terms of the corresponding EDM X with elements $X_{ij} = Y_{ii} - 2Y_{ij} + Y_{jj}$. Using Schoenberg's parametrization, we can write (9) equivalently as

$$\begin{array}{ll} \text{minimize} & \sum\limits_{(i,j,s,t)\in\Omega} \max\{0, 1 + X_{ij} - X_{st}\} - (\lambda/2)\,\mathbf{tr}(V^T X V) \\ \text{subject to} & \mathbf{diag}(X) = 0, \;\; X \in \mathbb{D}_0^n, \end{array}$$

where $V = I - (1/n)\mathbb{1}\mathbb{1}^T$. This problem can further be written in the vector form as

$$\begin{array}{ll} \text{minimize} & \sum\limits_{k=1}^{m} \max\{0, 1 + (Bx)_k\} + \lambda \mathbb{1}^T x \\ \text{subject to} & Cx = 0, \;\; x \in \mathbf{vec}(\mathbb{D}_0^n). \end{array} \tag{10}$$

Here the optimization variable x is a vector of length $p = n(n+1)/2$. The data matrix B has $m = |\Omega|$ rows, and each row of B represents a tuple $(i,j,s,t) \in \Omega$. If the kth row represents (i,j,s,t), then $(Bx)_k = X_{ij} - X_{st}$ for $x = \mathbf{vec}(X)$. The regularization term $\mathbf{tr}\,Y$ in (9) is rewritten in the vector form as

$$\mathbf{tr}\,Y = -\tfrac{1}{2}\,\mathbf{tr}(V^T X V) = -\tfrac{1}{2}\,\mathbf{tr}(V V^T X) = -\tfrac{1}{2}\,\mathbf{tr}((I - \tfrac{1}{n}\mathbb{1}\mathbb{1}^T)X) = \tfrac{1}{2n}\mathbb{1}^T X \mathbb{1} = \tfrac{1}{\sqrt{2}n}\mathbb{1}^T x,$$

if the diagonal elements of X are zero and $x = \mathbf{vec}(X)$. For simplicity, a factor $1/(\sqrt{2}n)$ was absorbed in the parameter λ in (10).

To apply PrePDHG (8), we formulate the problem in the form of (6) with

$$\begin{array}{c} f(x) = \delta_{\mathcal{C}}(x), \qquad g(u,v) = g_1(u) + \lambda \mathbb{1}^T v + \delta_{\mathbf{vec}(\mathbb{D}_0^n)}(v), \\ \mathcal{C} = \{x \in \mathbb{R}^p \mid Cx = 0\}, \qquad g_1(u) = \sum\limits_{k=1}^{m} \max\{0, 1 + u_k\}, \qquad A = \begin{bmatrix} B \\ I \end{bmatrix}. \end{array} \tag{11}$$

The basic PrePDHG iterations (8) are

$$\bar{x}^{(k+1)} = x^{(k)} + \theta(x^{(k)} - x^{(k-1)}) \tag{12a}$$
$$z_1^{(k+1)} = \mathbf{prox}_{\sigma g_1^*}(z_1^{(k)} + \sigma B\bar{x}^{(k+1)}) \tag{12b}$$
$$z_2^{(k+1)} = z_2^{(k)} + \sigma\bar{x}^{(k+1)} - \Pi_{\mathbf{vec}(\mathbb{D}_0^n)}\big(z_2^{(k)} + \sigma\bar{x}^{(k+1)} - (\lambda/\sigma)\mathbb{1}\big) \tag{12c}$$
$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{P}\left(x^{(k)}, \tau(B^T z_1^{(k+1)} + z_2^{(k+1)})\right). \tag{12d}$$

When $P = I$, the x-update reduces to

$$x^{(k+1)} = \Pi_{\mathcal{C}}(x^{(k)} - \tau(B^T z_1^{(k)} + z_2^{(k)}));$$

i.e., setting the first n elements to zero. The z_1-update is separable and has a closed-form formula. In particular, we have

$$\mathbf{prox}_{\sigma\phi^*}(t) = \max\{t+\sigma, 0\} - \max\{t+\sigma-1, 0\} = \begin{cases} 0 & t < -\sigma \\ t+\sigma & -\sigma \le t \le 1-\sigma \\ 1 & t > 1-\sigma, \end{cases}$$

where $\phi(t) = \max\{0, 1+t\}$. The z_2-update involves projection onto $\mathbf{vec}(\mathbb{D}_0^n)$.

Preconditioning. The updates in z_1 and z_2 are simple, and preconditioning does not help to reduce complexity or to improve efficiency. Here we discuss one preconditioner for the x-update: $P = B^TB + \gamma I$. This preconditioner has been widely used in applications arising from image processing and inverse problems; see, *e.g.*, [30,38,43]. The positive constant γ is added to guarantee the positive-definiteness of P and to improve numerical stability. In practice we often simply set $\gamma = 1$. With this choice of preconditioner, the x-update (12d) involves the optimization problem

$$\text{minimize} \quad \delta_{\mathcal{C}}(x) + \tfrac{1}{2}\|Bx - Bx^{(k)}\|^2 + \tfrac{\gamma}{2\tau}\|x - x^{(k)} + \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)})\|^2$$

with variable $x \in \mathbb{R}^p$. The function $\delta_{\mathcal{C}}$ requires the first n entries of x to be zero, while the vector $(Bx)_k = Y_{ij} - Y_{st}$ does not contain $x_{1:n}$. Thus, $x_{1:n}^{(k+1)} = 0$ and $x_{n+1:p}^{(k+1)}$ is the last $n(n-1)/2$ entries of the vector

$$\begin{aligned} \tilde{x} &= \operatorname*{argmin}_x \left(\tfrac{1}{2}\|Bx - Bx^{(k)}\|_2^2 + \tfrac{\gamma}{2\tau}\|x - x^{(k)} + \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)})\|_2^2\right) \\ &= x^{(k)} - \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)}) + \tfrac{\tau}{\sqrt{\gamma}}(B^TB + \gamma I)^{-1}B^TB(B^T z_1^{(k)} + z_2^{(k)}) \qquad (13) \\ &= x^{(k)} - \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)}) + \tfrac{\tau}{\sqrt{\gamma}}B^T(BB^T + \gamma I)^{-1}B(B^T z_1^{(k)} + z_2^{(k)}). \qquad (14) \end{aligned}$$

Proof. We consider the regularized least squares problem

$$\text{minimize} \quad \tfrac{1}{2}\|Bx - Bx^{(k)}\|_2^2 + \tfrac{\gamma}{2\tau}\|x - x^{(k)} + \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)})\|_2^2$$

with variable $x \in \mathbb{R}^p$. With a change of variables $y = x - x^{(k)} + \frac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)})$, the problem becomes

$$\text{minimize} \quad \tfrac{1}{2}\|By - \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)})\|_2^2 + \tfrac{\gamma}{2\tau}\|y\|_2^2$$

with variable $y \in \mathbb{R}^p$. This is a least squares problem with Tokhonov regularization, of which the optimal solution is given by (13). The next equation (14) follows from the identity

$$(B^T B + \gamma I)^{-1} B^T = B^T (BB^T + \gamma I)^{-1}$$

for any matrix B. □

In addition, we note that each row of B has only two nonzero entries, and then we expect that BB^T is sparser than $B^T B$. Hence the update (14) should be more efficient than (13). We also note that the matrix $B^T B + \gamma I$ does not change during the PrePDHG iterations. So we can factorize the matrix once and reuse the factor in every x-update. Suppose $BB^T + \gamma I = PLL^T P^T$ is the sparse Cholesky factorization, with the permutation matrix P and sparse Cholesky factor L. Then $\tilde{x}$-update (14) becomes

$$\tilde{x} = x^{(k)} - \tfrac{\tau}{\sqrt{\gamma}}(B^T z_1^{(k)} + z_2^{(k)}) + \tfrac{\tau}{\sqrt{\gamma}} B^T P L^{-T} L^{-1} P^T B (B^T z_1^{(k)} + z_2^{(k)}).$$

The most expensive steps are forward and back substitutions with a sparse matrix L.

Comparing the x-updates with and without preconditioning, we see that preconditioning slightly increases the computation complexity, and it is compensated by an improvement in the numerical performance. In particular, we observe in numerical experiments that with preconditioning, the total number of PDHG iterations dramatically decreases; see details in Section 5.

Low-rank approximation as inexact prox-operator. It is well known that if an EDM $X \in \mathbb{D}^n$ has a realization in $\mathbb{R}^d$ (with $d < n$), then the rank of X is at most $d+2$ [28, Theorem 5]. Hence, if the NMDS problem (9) has a d-dimensional realization at optimum, then the EDM problem (10) has a solution of rank at most $d+2$. This motivates us to replace the projection onto $\mathbb{D}_0^n$ involved in (12c) with a rank-$(d+2)$ PSD approximation: we only keep the leading $(d+2)$ eigenvalues and remove the rest. In this case, the advantage of a low-rank PSD approximation is threefold. First, this modification decreases the computational complexity in z_2-update because we only need to compute the leading $(d+2)$ eigenvalues and the corresponding eigenvectors. In addition, the low-rank regularization term $\mathbf{tr}\, Y$ in (9) is no longer needed. Thus, we avoid tuning the parameter λ, which is difficult but critical in practice [1]. The third benefit is on practical performance. We observe in experiments that the use of low-rank approximation often helps to generate an embedding that is "visually clearer"; see results in Section 5. This is because the inexact algorithm obtains a low-rank EDM solution while the original iterations (12) might not generate a realization in $\mathbb{R}^d$.

On the other hand, however, the low-rank approximation introduces inexactness error in z_2-updates, and thus PrePDHG with constant stepsizes might not converge. To resolve this, backtracking line search is needed to compensate inexact prox-evaluation and to guarantee convergence; see Section 4.

3.3 Maximum variance unfolding

Besides (generalized) NMDS, maximum variance unfolding (MVU) [57, 58] is another dimension reduction technique widely used in computer vision and machine learning; see [18, 51] for extensions and more examples. MVU is used to analyze high-dimensional data that lie on (or near) a low-dimensional manifold. For example, given a set of images, MVU computes a low-dimensional

embedding of each image with the property that the Euclidean distances between nearby images are preserved. It is shown that MVU admits the following SDP relaxation [57]

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(Y) \\ \text{subject to} & Y_{ii} - 2Y_{ij} + Y_{jj} = b_{ij}, \quad (i,j) \in \Omega \\ & \mathbb{1}^T Y \mathbb{1} = 0, \;\; Y \succeq 0, \end{array} \tag{15}$$

where the variable is the Gram matrix $Y \in \mathbb{S}^n$. Again an embedding dimension d is often known in advance, and a low-rank solution Y is desirable for practical purpose.

We can rewrite (15) in an EDM variable X or in the vector form with variable $x = \mathbf{vec}(X)$ as

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{tr}(V^T X V) \\ \text{subject to} & X_{ij} = b_{ij}, \;\; (i,j) \in \Omega \\ & X \in \mathbb{D}^n \end{array} \qquad \begin{array}{ll} \text{minimize} & -\mathbb{1}^T x \\ \text{subject to} & Bx = b \\ & Cx = 0, \;\; x \in \mathbf{vec}(\mathbb{D}_0^n). \end{array} \tag{16}$$

The matrix B has size $m \times p$ (recall $m = |\Omega|$, $p = n(n+1)/2$), and each row of B has only one nonzero entry.

To apply PrePDHG (8), we further reformulate the vectorized problem in (16) into the form (6) with $f(x) = \delta_{\mathbf{vec}(\mathbb{D}_0^n)}(x) - \mathbb{1}^T x$, $A = I$, and $g = \delta_{\mathcal{H}}$ is the indicator function of the set $\mathcal{H} = \{x \in \mathbb{R}^p \mid Bx = b,\ Cx = 0\}$. Then PrePDHG (8) takes the iterations

$$\bar{x}^{(k+1)} = x^{(k)} + \theta(x^{(k)} - x^{(k-1)}) \tag{17a}$$

$$z^{(k+1)} = z^{(k)} + \sigma\bar{x}^{(k+1)} - \Pi_{\mathcal{H}}\big(z^{(k)} + \sigma\bar{x}^{(k+1)}\big) \tag{17b}$$

$$x^{(k+1)} = \Pi_{\mathbf{vec}(\mathbb{D}_0^n)}\big(x^{(k)} - \tau z^{(k+1)} + \tfrac{\tau}{2}\mathbb{1}\big). \tag{17c}$$

For this splitting strategy, both updates in x and in z are simple, and preconditioning does not help to reduce complexity nor to improve efficiency. But the low-rank approximation technique discussed in Section 3.2 is still applicable to the x-update (17c).

4 Inexact preconditioned PDHG with line search

We now propose an inexact preconditioned primal–dual hybrid gradient method with line search (short: inexact PDHG). The presented algorithm allows for inexact evaluation of proximal operators, and at each iteration, the generated errors are compensated by a backtracking line search process. Thus, the typical assumption on summable errors is no longer needed.

4.1 Optimality conditions and duality theory

This section summarizes some facts from convex duality theory, which will be used in algorithm design and convergence analysis.

The dual of problem (6) is

$$\text{maximize} \quad -f^*(-A^T z) - g^*(z). \tag{18}$$

The primal–dual optimality conditions for (6) and (18) are

$$0 \in \partial f(x) + A^T z, \qquad 0 \in \partial g^*(z) - Ax, \tag{19}$$

where ∂f and ∂g^* are the subdifferentials of f and g^*. Throughout the paper we assume the optimality conditions (19) are solvable.

The convex–concave *Lagrangian* of (6) is defined as

$$\mathcal{L}(x,z) = f(x) + \langle z, Ax\rangle - g^*(z).$$

We use the convention that $\mathcal{L}(x,z) = +\infty$ if $x \notin \mathbf{dom}\, f$ and $\mathcal{L}(x,z) = -\infty$ if $x \in \mathbf{dom}\, f$ and $z \notin \mathbf{dom}\, g^*$. The solution $x^\star$, $z^\star$ of the optimality conditions (19) forms a saddle point of $\mathcal{L}$:

$$\inf_x \sup_z \mathcal{L}(x,z) = \sup_z \mathcal{L}(x^\star, z) = \inf_x \mathcal{L}(x, z^\star) = \sup_z \inf_x \mathcal{L}(x,z).$$

In particular, $\mathcal{L}(x^\star, z^\star)$ is the optimal value of (6) and (18).

4.2 Algorithm description

Inexact proximal operators. For many convex functions, the computation of its proximal operator is expensive and often involves solving a convex subproblem. This motivates the use of an inexact proximal operator in proximal splitting methods; see, *e.g.*, [5,23,38,45,46,50]. One classical definition of an inexact proximal operator is

$$\hat{x} \approx_\epsilon \mathbf{prox}_f^P(y,a) \quad \Longleftrightarrow \quad f(u) \geq f(\hat{x}) + \langle u - \hat{x}, P(y-\hat{x}) + a\rangle - \epsilon \ \text{ for all } u \in \mathbf{dom}\, f, \tag{20}$$

where $\epsilon > 0$ is a small tolerance for inexactness [23, 38, 45]. For simple convex functions, such an error can be easily computed. For example, when $f = \delta_{\mathcal{S}}$ is the indicator function of a unit 2-norm ball $\mathcal{S} = \{x \mid \|x\|_2 \leq 1\}$, the error ϵ satisfies

$$\epsilon \geq \sup_{u \in \mathcal{S}} \langle u - \hat{x}, P(y-\hat{x}) + a\rangle = \|P(y-\hat{x}) + a\|_2 - \langle \hat{x}, P(y-\hat{x}) + a\rangle.$$

To guarantee the convergence of inexact proximal methods, additional conditions are needed to bound the error ϵ. For instance, the following bound is used in [38]: $\epsilon \leq (\eta/2)\|\hat{x} - y\|_P^2$.

Algorithm description. Now we present the proposed algorithm (inexact PDHG). Select starting points $x^{(-1)} = x^{(0)} \in \mathbf{dom}\, f$ and $z^{(0)} \in \mathbf{dom}\, g^*$. For $k = 0, 1, \ldots$, repeat the following steps:

$$\bar{x}^{(k+1)} = x^{(k)} + \theta_k (x^{(k)} - x^{(k-1)}) \tag{21a}$$

$$z^{(k+1)} \approx_{\epsilon_k} \mathbf{prox}^Q_{\sigma_k g^*}(z^{(k)}, -A\bar{x}^{(k+1)}) \tag{21b}$$

$$x^{(k+1)} = \mathbf{prox}^P_{\tau_k f}(x^{(k)}, \tau A^T z^{(k)}), \tag{21c}$$

where the error ϵ_k in z-update must satisfy

$$\epsilon_k \leq \tfrac{\eta}{2\sigma_k}\|z^{(k+1)} - z^{(k)}\|_Q^2. \tag{22}$$

The parameters τ_k, σ_k, θ_k are determined by a backtracking line search. We select the initial parameters $\bar{\theta}_{-1} = 1$, $\tau_{-1}, \sigma_{-1} > 0$, and tolerance $0 < \eta < \delta \leq 1$. To start line search at iteration k, we choose $\bar{\theta}_k \in [1, \sqrt{1 + \theta_{k-1}}]$. For $i = 0, 1, 2, \ldots$, we set $\theta_k = 2^{-i}\bar{\theta}_k$, $\tau_k = \theta_k \tau_{k-1}$, $\sigma_k = \theta_k \tau_{k-1}$, and compute $\bar{x}^{(k+1)}$, $z^{(k+1)}$, $x^{(k+1)}$ using (21). If

$$\langle z^{(k+1)} - z^{(k)}, A(\bar{x}^{(k+1)} - x^{(k+1)})\rangle \leq \tfrac{1}{2\tau_k}\|\bar{x}^{(k+1)} - x^{(k)}\|_P^2 + \tfrac{\delta - \eta}{2\sigma_k}\|z^{(k+1)} - z^{(k)}\|_Q^2, \tag{23}$$

we accept the computed iterates $\bar{x}^{(k+1)}$, $x^{(k+1)}$, $z^{(k+1)}$ and stepsizes τ_k, σ_k, and terminate the line search. If (23) does not hold, we increment i and continue the line search. In practice δ is chosen as one, and $\delta < 1$ is only needed for some convergence analysis (see Theorem 1). The constant η is used to control the error bound on ϵ_k so that the inexactness error can be compensated by a smaller stepsize chosen in (23). The convergence analysis can be easily extended to handle the case where η varies in different iterations (as η_k), but for simplicity we only consider the constant case.

The error bound condition (22) is different from the conditions used in most inexact proximal methods. In particular, it does not require the errors are summable (*i.e.*, $\sum_k \epsilon_k < \infty$). Instead, the error is required to be bounded at each iteration, and convergence is guaranteed via an adaptive choice of stepsizes. The condition (22) is similar to the one used in [38], but inexact PDHG has substantial difference from the algorithm proposed in [38]. In [38], the inexactness error is due to an early-stopped subproblem solver while for problems in Section 3, an inexact prox-evaluation is motivated by the need for a low-rank solution. In addition, the paper [38] only studies the case $P = I$ and $Q = AA^T$, and their analysis needs additional assumptions on the problem (6) (*e.g.*, f is strongly convex). In comparison, the convergence results in this paper hold for general choices of preconditioners P, Q, and for general convex problems (6).

Application to EDM optimization. When inexact PDHG is applied to the problem (10), $g = \delta_{\mathbf{vec}(\mathbb{D}_0^n)}$, $Q = I$, and the error ϵ_k in z-update is due to the low-rank approximation. (Similar argument can also be made for MVU (16).) This error is actually computable, and the computational complexity is dominated by an eigen-decomposition. In practice, however, it is not worth the effort. In the implementation, we fix $\eta \in (0, 1)$ in advance, and check the inexactness condition (22) at each iteration. We only take an inexact update if the condition is satisfied; otherwise we perform an *exact* z-update at this iteration. In all the numerical experiments in Section 5, we observe that the inexactness condition (22) can always be satisfied with a pre-chosen η, and thus inexact PDHG generates a sequence of low-rank iterates converging to an EDM solution.

4.3 Convergence analysis

We now present the convergence results for inexact PDHG. The analysis follows the ideas in [12, 31, 32, 39]. We start with a lemma that guarantees the line search exits at each iteration.

Lemma 1. *The stepsizes τ_k, σ_k generated by* inexact PDHG *are bounded below by positive constants:*

$$\tau_k \geq \tau_{\min} = \min\left\{\tau_{-1}, \frac{\sqrt{\delta - \eta}}{2\sqrt{\beta}\|A\|}\right\}, \qquad \sigma_k \geq \beta\tau_{\min}, \quad \text{for all } k = 1, 2, \ldots, \tag{24}$$

where $\beta = \sigma_{-1}/\tau_{-1}$.

Proof. First note that the exit condition (23) certainly holds if

$$\sqrt{\sigma_k\tau_k}\|A\| \leq \sqrt{\delta - \eta}.$$

This can be shown by the definition of the operator norm:

$$\begin{aligned}&\langle z^{(k+1)} - z^{(k)}, A(\bar{x}^{(k+1)} - x^{(k+1)})\rangle \\ &\leq \|A\|\|z^{(k+1)} - z^{(k)}\|_Q\|\bar{x}^{(k+1)} - x^{(k+1)}\|_P\end{aligned}$$

$$\leq \frac{\sqrt{\sigma_k\tau_k}\|A\|}{\delta-\eta}\left(\frac{\|\bar{x}^{(k+1)}-x^{(k+1)}\|_P^2}{\tau_k}\frac{(\delta-\eta)\|z^{(k+1)}-z^{(k)}\|_Q^2}{\sigma_k}\right)^{1/2}$$
$$\leq \frac{1}{2\tau_k}\|\bar{x}^{(k+1)}-x^{(k+1)}\|_P^2+\frac{(\delta-\eta)}{2\sigma_k}\|z^{(k+1)}-z^{(k)}\|_Q^2.$$

Based on this observation, we use induction to show the lower bounds (24). Suppose $\tau_{k-1}\geq\tau_{\min}$ and $\sigma_{k-1}\geq\sigma_{\min}$, which holds at $k=0$. The first value of θ_k tested in the line search is $\theta_k=\bar{\theta}_k\geq 1$. If this value is accepted, then

$$\tau_k=\bar{\theta}_k\tau_{k-1}\geq\tau_{k-1}\geq\tau_{\min},\qquad \sigma_k=\bar{\theta}_k\sigma_{k-1}\geq\sigma_{k-1}\geq\sigma_{\min}.$$

If $\theta=\bar{\theta}_k$ is rejected, one or more backtracking steps are taken. Denote by $\tilde{\theta}_k$ the last rejected value. Then $\tilde{\theta}_k\sqrt{\sigma_{k-1}\tau_{k-1}}\|A\|>\sqrt{\delta-\eta}$, and the accepeted θ_k satisfies

$$\theta_k=\frac{\tilde{\theta}_k}{2}>\frac{\sqrt{\delta-\eta}}{2\sqrt{\sigma_k\tau_k}\|A\|}=\frac{\sqrt{\delta-\eta}}{2\tau_k\sqrt{\beta}\|A\|}.$$

Therefore,

$$\tau_k=\theta_k\tau_{k-1}>\frac{\sqrt{\delta-\eta}}{2\sqrt{\beta}\|A\|}\geq\tau_{\min},\qquad \sigma_k=\beta\tau_k\geq\beta\tau_{\min}.$$ □

Theorem 1. *If $\eta<1$, the iterates $(x^{(k)},z^{(k)})$ generated by* inexact PDHG *converge to primal and dual optimal solutions $(x^\star,z^\star)$.*

Proof. The optimality condition for the dual update (21b) implies that

$$g^*(z^{(k+1)})-g^*(z)\leq\frac{1}{\sigma_k}\langle z-z^{(k+1)},z^{(k+1)}-z^{(k)}\rangle_Q-\langle z-z^{(k+1)},A\bar{x}^{(k+1)}\rangle+\epsilon_k \tag{25}$$

for all $z\in\mathbf{dom}\,g^*$. Similarly, the optimality condition for the primal update (21c) implies

$$f(x^{(k+1)})-f(x)\leq\frac{1}{\tau_k}\langle x^{(k+1)}-x^{(k)},x-x^{(k+1)}\rangle_P+\langle z^{(k+1)},A(x-x^{(k+1)})\rangle \tag{26}$$

for all $x\in\mathbf{dom}\,f$. At the previous iteration we have for all $x\in\mathbf{dom}\,f$,

$$\begin{aligned}f(x^{(k)})-f(x)&\leq\frac{1}{\tau_{k-1}}\langle x^{(k)}-x^{(k-1)},x-x^{(k)}\rangle_P+\langle z^{(k)},A(x-x^{(k)})\rangle\\&=\frac{\theta_k}{\tau_k}\langle x^{(k)}-x^{(k-1)},x-x^{(k)}\rangle_P+\langle z^{(k)},A(x-x^{(k)})\rangle.\end{aligned}$$

We evaluate this inequality at $x:=x^{(k+1)}$ and add it to the inequality at $x:=x^{(k-1)}$ multiplied by θ_k:

$$\begin{aligned}&(1+\theta_k)f(x^{(k)})-\theta_k f(x^{(k-1)})-f(x^{(k+1)})\\&\leq\frac{\theta_k}{\tau_k}\langle x^{(k)}-x^{(k-1)},x^{(k+1)}-\bar{x}^{(k+1)}\rangle_P+\langle z^{(k)},A(x^{(k+1)}-\bar{x}^{(k+1)})\rangle\\&=\frac{1}{\tau_k}\langle\bar{x}^{(k+1)}-x^{(k)},x^{(k+1)}-\bar{x}^{(k+1)}\rangle+\langle z^{(k)},A(x^{(k+1)}-\bar{x}^{(k+1)})\rangle\end{aligned}$$

$$= \frac{1}{2\tau_k}\big(\|x^{(k+1)} - x^{(k)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k+1)}\|_P^2\big) + \langle z^{(k)}, A(x^{(k+1)} - \bar{x}^{(k+1)})\rangle. \tag{27}$$

In the second step we use $\bar{x}^{(k+1)} = x^{(k)} + \theta_k(x^{(k)} - x^{(k-1)})$. Adding (25) with $z := z^\star$, (26) with $x := x^\star$, and (27) gives

$$\begin{aligned}
&(1+\theta_k)f(x^{(k)}) - \theta_k f(x^{(k-1)}) - f(x^\star) + g^*(z^{(k+1)}) - g^*(z^\star) \\
&\leq \frac{1}{2\tau_k}\big(\|x^\star - x^{(k)}\|_P^2 - \|x^\star - x^{(k+1)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k+1)}\|_P^2\big) \\
&\quad + \frac{1}{2\sigma_k}\big(\|z^\star - z^{(k)}\|_Q^2 - \|z^\star - z^{(k+1)}\|_Q^2 - \|z^{(k+1)} - z^{(k)}\|_Q^2\big) + \epsilon_k \\
&\quad - \langle z^\star, A(\bar{x}^{(k+1)} - x^\star)\rangle + \langle z^{(k+1)} - z^\star, Ax^\star\rangle + \langle z^{(k+1)} - z^{(k)}, A(\bar{x}^{(k+1)} - x^{(k+1)})\rangle.
\end{aligned} \tag{28}$$

We define

$$F(x) = f(x) - f(x^\star) - \langle z^\star, A(x - x^\star)\rangle, \quad G(z) = g^*(z) - g^*(z^\star) - \langle z - z^\star, Ax^\star\rangle.$$

Both functions are nonnegative for all $x \in \mathbf{dom}\, f$ and $z \in \mathbf{dom}\, g^*$, since $(x^\star, z^\star)$ satisfies the optimality conditions (19). In this notation the inequality (28) can be written as

$$\begin{aligned}
&\tau_k\big((1+\theta_k)F(x^{(k)}) - \theta_k F(x^{(k-1)}) + G(z^{(k+1)})\big) \\
&\leq \frac{1}{2}\big(\|x^\star - x^{(k)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k)}\|_P^2 - \|x^\star - x^{(k+1)}\|_P^2 - \|\bar{x}^{(k+1)} - x^{(k+1)}\|_P^2\big) \\
&\quad + \frac{1}{2\beta}\big(\|z^\star - z^{(k)}\|_Q^2 - \|z^\star - z^{(k+1)}\|_Q^2 - \|z^{(k+1)} - z^{(k)}\|_Q^2\big) \\
&\quad + \tau_k\langle z^{(k+1)} - z^{(k)}, A(\bar{x}^{(k+1)} - x^{(k+1)})\rangle + \epsilon_k.
\end{aligned} \tag{29}$$

The last two terms on the right-hand side can be bounded using the exit condition in the line search (23) and the error bound (22):

$$\tau_k\langle z^{(k+1)} - z^{(k)}, A(\bar{x}^{(k+1)} - x^{(k+1)})\rangle + \epsilon_k \leq \frac{1}{2}\|\bar{x}^{(k+1)} - x^{(k+1)}\|_P^2 + \frac{\delta}{2\beta}\|z^{(k+1)} - z^{(k)}\|_Q^2. \tag{30}$$

Substituting this upper bound into (29) yields

$$\begin{aligned}
&\tau_k(1+\theta_k)F(x^{(k)}) + \frac{1}{2}\|x^\star - x^{(k+1)}\|_P^2 + \frac{1}{2\beta}\|z^\star - z^{(k+1)}\|_Q^2 \\
&\leq \tau_{k-1}(1+\theta_{k-1})F(x^{(k-1)}) + \frac{1}{2}\|x^\star - x^{(k)}\|_P^2 + \frac{1}{2\beta}\|z^\star - z^{(k)}\|_Q^2 \\
&\quad - \frac{1}{2}\|\bar{x}^{(k+1)} - x^{(k+1)}\|_P^2 - \frac{1-\delta}{2\beta}\|z^{(k+1)} - z^{(k)}\|_Q^2 \qquad (31) \\
&\leq \tau_{-1}F(x^{(-1)}) + \frac{1}{2}\|x^\star - x^{(0)}\|_P^2 + \frac{1}{2\beta}\|z^\star - z^{(0)}\|_Q^2 \\
&\quad - \frac{1}{2}\sum_{j=0}^{k}\Big(\|\bar{x}^{(j+1)} - x^{(j+1)}\|_P^2 + \frac{1-\delta}{\beta}\|z^{(j+1)} - z^{(j)}\|_Q^2\Big). \qquad (32)
\end{aligned}$$

This shows that the sequence $x^{(k)}$ and $z^{(k)}$ are bounded and that

$$\|x^{(k)} - \overline{x}^{(k)}\|_P \to 0, \qquad \|z^{(k+1)} - z^{(k)}\|_Q \to 0. \tag{33}$$

The second limit implies $z^{(k+1)} - z^{(k)} \to 0$. It also follows that $(x^{(k)} - x^{(k-1)})/\tau_{k-1}$ goes to zero because

$$\frac{1}{\tau_{k-1}}\|x^{(k)} - x^{(k-1)}\|_P = \frac{\theta_k}{\tau_k}\|x^{(k)} - x^{(k-1)}\|_P = \frac{1}{\tau_k}\|x^{(k+1)} - \overline{x}^{(k+1)}\|_P \leq \frac{1}{\tau_{\min}}\|x^{(k+1)} - \overline{x}^{(k+1)}\|_P$$

and the right-hand side goes to zero. Let $(\hat{x}, \hat{z})$ be a limit point of a converging subsquence $(x^{(k_i)}, z^{(k_i)})$. From (33) it follows that $\overline{x}^{(k_i)}$ also converges to $\hat{x}$, and $z^{(k_i+1)}$ converges to $\hat{z}$.

We then show the uniqueness of $\hat{x}$, $\hat{z}$ by contradiction. Assume there exists another subsequence $(x^{(k_j)}, z^{(k_j)})$ that converges to $(\tilde{x}, \tilde{z})$. Then it follows that

$$\|\hat{x} - \tilde{x}\|_P = \lim_{k\to\infty} \|x^{(k)} - \tilde{x}\|_P = \lim_{k\to\infty} \|x^{(k)} - \hat{x}\|_P = 0.$$

Hence, $\tilde{x} = \hat{x}$, and $\tilde{z} = \hat{z}$ can be shown by a similar argument.

The optimality condition for x-update at iteration $k-1$ is

$$\frac{1}{\tau_{k-1}} P(x^{(k-1)} - x^{(k)}) - A^T z^{(k)} \in \partial(x^{(k)}).$$

The left-hand side converges to $-A^T\hat{z}$ and $x^{(k)}$ converges to $\hat{x}$. It then follows from maximal monotonicity of ∂f that $-A^T\hat{z} \in \partial f(\hat{x})$. Similarly, the optimality condition for z-update is

$$\frac{1}{\sigma_{k-1}} Q(z^{(k-1)} - z^{(k)}) + A\overline{x}^{(k)} \in \partial g^*(z^{(k)}).$$

The left-hand side converges to $A\hat{x}$, and from maximal monotonicity of ∂g^* it follows that $A\hat{x} \in \partial g^*(\hat{z})$. Therefore, we conclude that $(\hat{x}, \hat{z})$ are optimal:

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{z} \end{bmatrix} + \begin{bmatrix} \partial f(\hat{x}) \\ \partial g^*(\hat{z}) \end{bmatrix}. \qquad \square$$

We define the averaged iterates for $k \geq 1$:

$$x_{\text{avg}}^{(k)} = \frac{\tau_1\theta_1 x^{(0)} + \sum_{i=1}^k \tau_i \overline{x}^{(i+1)}}{\tau_1\theta_1 + \omega_k}, \qquad z_{\text{avg}}^{(k)} = \frac{\sum_{i=1}^k \tau_i z^{(i+1)}}{\omega_k},$$

where $\omega_k = \sum_{i=1}^k \tau_i$. The convergence of $\left(x_{\text{avg}}^{(k)}, z_{\text{avg}}^{(k)}\right)$ is given in the following theorem.

Theorem 2. *The averaged iterates* $\left(x_{\text{avg}}^{(k)}, z_{\text{avg}}^{(k)}\right)$ *computed from* inexact PDHG *satisfy*

$$\mathcal{L}(x_{\text{avg}}^{(k)}, z^\star) - \mathcal{L}(x^\star, z_{\text{avg}}^{(k)}) \leq \frac{1}{\omega_k}\Big(\frac{1}{2}\|x^\star - x^{(1)}\|_P^2 + \frac{1}{2\beta}\|z^\star - z^{(1)}\|_Q^2 + \tau_1\theta_1 R_0\Big),$$

where $\omega_k = \sum_{i=1}^k \tau_i$, $\beta = \sigma_{-1}/\tau_{-1}$, *and* $R_0 = f(x^{(0)}) - f(x^\star) - \langle z^\star, A(x^{(0)} - x^\star)\rangle \geq 0$.

Proof. Combining (29) and (30) gives

$$\begin{aligned}&\tau_i\big((1+\theta_i)F(x^{(i)})-\theta_iF(x^{(i-1)})+G(z^{(i+1)})\big)\\&\le\frac{1}{2}\big(\|x^{(i)}-x^\star\|_P^2-\|x^{(i+1)}-x^\star\|_P^2\big)+\frac{1}{2\beta}\big(\|z^{(i)}-z^\star\|_Q^2-\|z^{(i+1)}-z^\star\|_Q^2\big).\end{aligned}$$

Summing up from $i=1$ to k gives

$$\begin{aligned}&\sum_{i=1}^k\tau_i\big((1+\theta_i)F(x^{(i)})-\theta_iF(x^{(i-1)})+G(z^{(i+1)})\big)\\&\le\frac{1}{2}\big(\|x^{(1)}-x^\star\|_P^2-\|x^{(k+1)}-x^\star\|_P^2\big)+\frac{1}{2\beta}\big(\|z^{(1)}-z^\star\|_Q^2-\|z^{(k+1)}-z^\star\|_Q^2\big).\end{aligned}\tag{34}$$

The left-hand side can be written as

$$\begin{aligned}&\sum_{i=1}^k\tau_i\big((1+\theta_i)F(x^{(i)})-\theta_iF(x^{(i-1)})+G(z^{(i+1)})\big)\\&=\tau_k(1+\theta_k)F(x^{(k)})+\sum_{i=2}^k\big((1+\theta_{i-1})\tau_{i-1}-\theta_i\tau_i\big)F(x^{(i-1)})-\theta_1\tau_1F(x^{(0)})+\sum_{i=1}^k\tau_kG(z^{(i+1)}).\end{aligned}$$

By convexity of F it follows that

$$\begin{aligned}&\tau_k(1+\theta_k)F(x^{(k)})+\sum_{i=2}^k\big((1+\theta_{i-1})\tau_{i-1}-\theta_i\tau_i\big)F(x^{(i-1)})\\&\ge(\tau_1\theta_1+\omega_k)F\Big(\frac{\tau_1(1+\theta_1)x^{(1)}+\sum_{i=2}^k\tau_i\bar{x}^{(i+1)}}{\tau_1\theta_1+\omega_k}\Big)\\&=(\tau_1\theta_1+\omega_k)F\Big(\frac{\tau_1\theta_1x^{(0)}+\sum_{i=1}^k\tau_i\bar{x}^{(i+1)}}{\tau_1\theta_1+\omega_k}\Big)\\&\ge\omega_kF(x_{\text{avg}}^{(k)}),\end{aligned}$$

where $\omega_k=\sum_{i=1}^k\tau_i$. Similarly,

$$\sum_{i=1}^k\tau_iG(z^{(i+1)})\ge\omega_kG\Big(\frac{\sum_{i=1}^k\tau_iz^{(i+1)}}{\omega_k}\Big)=\omega_kG(z_{\text{avg}}^{(k)}).$$

Hence, (34) becomes

$$\begin{aligned}\mathcal{L}(x_{\text{avg}}^{(k)},z)-\mathcal{L}(x,z_{\text{avg}}^{(k)})&=F(x_{\text{avg}}^{(k)})+G(z_{\text{avg}}^{(k)})\\&\le\frac{1}{\omega_k}\Big(\frac{1}{2}\|x^{(1)}-x^\star\|_P^2+\frac{1}{2\beta}\|z^{(1)}-z^\star\|_Q^2+\tau_1\theta_1F(x^{(0)})\Big).\end{aligned}$$

We have the same $O(1/k)$ ergodic rate of convergence as in [10,12], though with the ergodic sequence $(x_{\text{avg}}^{(k)},z_{\text{avg}}^{(k)})$ defined in a different way. □

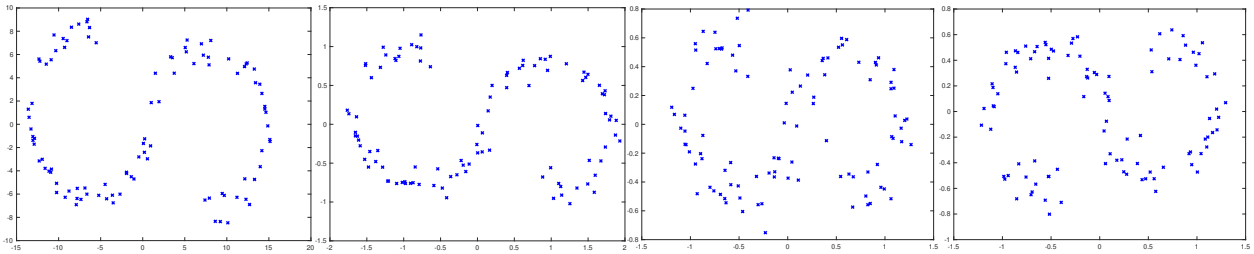

Figure 1: Plot of Scurve realizations generated by inexact PDHG, inexact PDHG without preconditioning, exact PrePDHG ((21) with $\epsilon_k = 0$), and SCS, respectively.

5 Numerical experiments

In this section, we evaluate the performance of inexact PDHG applied to NMDS (10) and MVU (16). The numerical results verify that the presented algorithm can efficiently solve large-scale EDM optimization problems that cannot be handled by general-purpose solvers (*e.g.*, SeDuMi [52], SDPT3 [54]). In addition, it is observed that inexact PDHG generates virtually better realizations with inexact (low-rank) updates than without.

The experiments are carried out in MATLAB 2021b on a desktop with an Intel Core i5 2.4GHz CPU and 16GB RAM. We compare inexact PDHG with the SCS solver [41], which is built on the well-known alternating direction method of multipliers (ADMM) [8] and is able to solve large-scale quadratic cone programs. For fair comparison, SCS is also used to solve the problems (10) and (16).

5.1 Numerical results for NMDS

To test the scalability of inexact PDHG, we test NMDS on five datasets: Scurve, Optdigits [2], Cifar-10, MNIST, and EMNIST. Scurve is a small dataset suggested in [34], and is used here mainly for visualization of the realizations. The last three datasets are commonly used in machine learning applications. In our experiments we only take part of each dataset, and ensure that each label class has the same number of data points. (Recall that the most expensive step in inexact PDHG is the eigen-decomposition, and thus for inexact PDHG, the scale of solvable problems is limited by the size of matrices for which an eigen-decomposition can be performed on a desktop.) The construction of the data matrix B follows the instruction in [1]. For all the experiments, we set $d = 2$, the control parameter for inexactness as $\eta = 0.5$, and the parameter in the preconditioner as $\gamma = 1$.

Table 1 reports the number of iterations and CPU time for and all variants of inexact PDHG and SCS. Our method (inexact PDHG) outperforms SCS in terms of both the number of iterations and runtime. It is also shown that preconditioning significantly reduces the number of iterations, and thus the computation time. Moreover, the use of inexact updates does not affect the number of iterations much. Recall that only a small portion of eigenvalues is needed in inexact updates, and thus for larger datasets, inexact PDHG converges faster than its exact variant. We also note that SCS fails for EMNIST dataset ($n = 8000$) while inexact PDHG is able to solve it efficiently. This could be because SCS needs to generate a large number of auxiliary variables to represent the EDM conic constraint, and thus the solver runs out of memory.

For the illustrative dataset Scurve, we also plot the constructed realizations from inexact PDHG, PrePDHG and SCS in Fig. 1. We see that algorithms with inexact (low-rank) updates generate a clearer and less noisy "S" curve when compared with algorithms with exact prox-evaluations.

data	n	m	algo	precond.	inexact	number of iterations	CPU time (in sec.)
Scurve	100	4,949	inexact PDHG	Y	Y	1,293	9.70
			inexact PDHG	Y	N	1,038	9.02
			inexact PDHG	N	Y	21,357	88.10
			SCS	-	-	1,627	15.25
Optdigits	1264	798,215	inexact PDHG	Y	Y	1,362	1.84×10^2
			inexact PDHG	Y	N	1,265	1.62×10^2
			inexact PDHG	N	Y	26,548	6.48×10^4
			SCS	-	-	1,738	3.34×10^3
Cifar-10	3000	4,498,499	inexact PDHG	Y	Y	1,026	9.43×10^3
			inexact PDHG	Y	N	974	9.40×10^3
			inexact PDHG	N	Y	18,625	2.82×10^5
			SCS	-	-	1,328	1.64×10^4
MNIST	5000	12,497,499	inexact PDHG	Y	Y	1,563	5.45×10^4
			inexact PDHG	Y	N	1,494	6.03×10^4
			inexact PDHG	N	Y	30,054	1.62×10^6
			SCS	-	-	1,998	9.28×10^4
EMNIST	8000	31,995,999	inexact PDHG	Y	Y	1,154	2.27×10^5
			inexact PDHG	Y	N	1,026	3.64×10^5
			inexact PDHG	N	Y	20,245	7.26×10^6
			SCS	-	-	-	OOM

Table 1: Results for NMDS. Four algorithms (three variants of inexact PDHG and SCS) are tested on five datasets. n indicates matrix size and m indicates the number of linear constraints in NMDS. The "precond." and "inexact" columns indicate whether preconditioning and inexact updates are used, respectively. OOM indicates out of memory.

5.2 Numerical results for MVU

MVU is tested on four datasets: Optdigits, Cifar-10, MNIST, and EMNIST. (Scurve is designed for NMDS only.) Again we only take part of these datasets, and the construction of the data matrix B follows the instruction in [57, 58]. We set $d = 2$ and $\eta = 0.5$.

Table 2 reports the number of iterations and CPU time for two variants of inexact PDHG and SCS. The observation is similar to the results for NMDS. Both algorithms take about one or two thousand iterations, which is typical for first-order proximal methods. The computation time of inexact PDHG is faster than that of SCS, because the inexact updates only need to compute four leading eigenvalues and our customization handles the linear constraints efficiently.

6 Conclusions

We present a primal–dual proximal splitting algorithm (inexact PDHG) that allows for inexact prox-evaluations, and apply it to Euclidean distance matrix (EDM) optimization problems. The inexact prox-evaluation is motivated by the existence of a low-rank EDM solution, and backtracking line search is used to compensate the inexactness errors and to guarantee convergence. We showcase two machine learning models as application and reformulate them into EDM optimization problems. Data structure is exploited to further improve the efficiency and scalability of inexact PDHG (for

data	n	m	algo	inexact	number of iterations	CPU time (in sec.)
Optdigits	1264	78,368	inexact PDHG	Y	1,028	1.53×10^2
			inexact PDHG	N	943	1.38×10^2
			SCS	-	1,682	2.84×10^3
Cifar-10	3000	447,000	inexact PDHG	Y	984	8.92×10^3
			inexact PDHG	N	872	7.26×10^3
			SCS	-	1,403	1.23×10^4
MNIST	5000	1,245,000	inexact PDHG	Y	1,438	4.98×10^4
			inexact PDHG	N	1,247	4.22×10^4
			SCS	-	1,813	7.43×10^4
EMNIST	8000	3,192,000	inexact PDHG	Y	1,046	1.89×10^5
			inexact PDHG	N	976	1.43×10^5
			SCS	-	1,572	3.71×10^5

Table 2: Results for MVU. Three algorithms (two variants of inexact PDHG and SCS) are tested on four datasets. n indicates matrix size and m indicates the number of linear constraints in MVU. The "inexact" column indicates whether inexact updates are used.

EDM optimization), and numerical experiments are conducted to verify its superiority. Future work might be needed to address the scalability of eigen-decomposition used in the algorithm.

References

[1] S. Agarwal, J. Wills, L. Cayton, G. Lanckriet, D. Kriegman, and S. Belongie. Generalized non-metric multidimensional scaling. In M. Meila and X. Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 11–18. PMLR, 2007.

[2] C. C. Aggarwal and S. Sathe. Theoretical foundations and algorithms for outlier ensembles. *ACM SIGKDD Explorations Newsletter*, 17(4):24–47, 2015.

[3] A. Y. Alfakih, P. Charron, V. Piccialli, and H. Wolkowicz. Euclidean distance matrices, semidefinite programming and sensor network localization. *Portugaliae Mathematica*, 68(1):53–102, 2011.

[4] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. Solving Euclidean distance matrix completion problems via semidefinite programming. *Computational Optimization and Applications*, 12(1):13–30, 1999.

[5] H. H. Bauschke and P. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer Publishing Company, Incorporated, 1st edition, 2011.

[6] P. Biswas, K.-C. Toh, and Y. Ye. A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. *SIAM Journal on Scientific Computing*, 30(3):1251–1277, 2008.

[7] P. Biswas and Y. Ye. Semidefinite programming for Ad-Hoc wireless sensor network localization. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pages 46–54, 2004.

[8] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[9] A. Chambolle and J. P. Contreras. Accelerated Bregman primal–dual methods applied to optimal transport and Wasserstein barycenter problems. *arXiv e-prints*, arXiv:2203.00802, 2022.

[10] A. Chambolle and T. Pock. A first-order primal–dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[11] A. Chambolle and T. Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[12] A. Chambolle and T. Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, 159(1):253–287, 2016.

[13] Z. Cheng, J. Zhang, A. Agrawal, and S. P. Boyd. Joint graph learning and model fitting in Laplacian regularized stratified models. *arXiv e-prints*, arXiv:2305.02573, 2023.

[14] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Springer Optimization and Its Applications, pages 185–212. Springer New York, 2011.

[15] L. Condat. A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications*, 158(2):460–479, 2013.

[16] L. Condat, D. Kitahara, A. Contreras, and A. Hirabayashi. Proximal splitting algorithms for convex optimization: A tour of recent advances, with new twists. *SIAM Review*, 65(2):375–435, 2023.

[17] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman & Hall, 2000.

[18] J. Dattorro. *Convex Optimization & Euclidean Distance Geometry*. Meboo Publishing, 2005.

[19] C. Ding and H.-D. Qi. Convex Euclidean distance embedding for collaborative position localization with NLOS mitigation. *Computational Optimization and Applications*, 66(1):187–218, 2017.

[20] C. Ding and H.-D. Qi. Convex optimization learning of faithful Euclidean distance representations in nonlinear dimensionality reduction. *Mathematical Programming*, 164(1-2):341–381, 2017.

[21] Y. Ding, N. Krislock, J. Qian, and H. Wolkowicz. Sensor network localization, Euclidean distance matrix completions, and graph realization. *Optimization and Engineering*, 11(1):45–66, 2010.

[22] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli. Euclidean distance matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine*, 32(6):12–30, 2015.

[23] J. Eckstein. Approximate iterations in Bregman-function-based proximal algorithms. *Mathematical Programming*, 83(1):113–123, 1998.

[24] E. Esser, X. Zhang, and T. F. Chan. A general framework for a class of first order primal–dual algorithms for convex optimization in imaging science. *SIAM Journal on Imaging Sciences*, 3(4):1015–1046, 2010.

[25] N. Gaffke and R. Mathar. A cyclic projection algorithm via duality. *Metrika*, 36(1):29–54, 1989.

[26] R. Glowinski, S. J. Osher, and W. Yin. *Splitting Methods in Communication, Imaging, Science, and Engineering*. Scientific Computation. Springer, 2017.

[27] W. Glunt, T. L. Hayden, S. Hong, and J. Wells. An alternating projection algorithm for computing the nearest Euclidean distance matrix. *SIAM Journal on Matrix Analysis and Applications*, 11(4):589–600, 1990.

[28] J. C. Gower. Properties of Euclidean and non-Euclidean distance matrices. *Linear Algebra and its Applications*, 67:81–97, 1985.

[29] T. L. Hayden and J. Wells. Approximation by matrices positive semidefinite on a subspace. *Linear Algebra and Its Applications*, 109:115–130, 1988.

[30] M. Jacobs, F. Leger, W. Li, and S. Osher. Solving large-scale optimization problems with a convergence rate independent of grid size. *SIAM Journal on Numerical Analysis*, 57(3):1100–1123, 2019.

[31] X. Jiang and L. Vandenberghe. Bregman primal–dual first-order method and applications to sparse semidefinite programming. *Computational Optimization and Applications*, 81(1):127–159, 2022.

[32] X. Jiang and L. Vandenberghe. Bregman three-operator splitting methods. *Journal of Optimization Theory and Applications*, 196(3):936–972, 2023.

[33] N. Komodakis and J. Pesquet. Playing with duality: an overview of recent primal–dual approaches for solving large-scale optimization problems. *IEEE Signal Processing Magazine*, 32(6):31–54, 2015.

[34] I. Kovan. Multidimensional scaling (MDS) for dimensionality reduction and data visualization. available at https://rb.gy/kmhvc, 2021.

[35] N. Krislock and H. Wolkowicz. Euclidean distance matrices and applications. In M. F. Anjos and J. B. Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, pages 879–914. Springer, 2012.

[36] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[37] B. Kulis, A. C. Surendran, and J. C. Platt. Fast low-rank semidefinite programming for embedding and clustering. In M. Meila and X. Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 235–242. PMLR, 2007.

[38] Y. Liu, Y. Xu, and W. Yin. Acceleration of primal–dual methods by preconditioning and simple subproblem procedures. *Journal of Scientific Computing*, 86(2):21, 2021.

[39] Y. Malitsky and T. Pock. A first-order primal–dual algorithm with linesearch. *SIAM Journal on Optimization*, 28(1):411–432, 2018.

[40] J. Moreau. Proximité et dualité dans un espace Hilbertien. *Bulletin de la Société Mathématique de France*, 93:273–299, 1965.

[41] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. SCS: Splitting conic solver, version 3.2.3. https://github.com/cvxgrp/scs, 2022.

[42] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014.

[43] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal–dual algorithms in convex optimization. In *International Conference on Computer Vision*, pages 1762–1769, 2011.

[44] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. An algorithm for minimizing the Mumford–Shah functional. In *International Conference on Computer Vision*, pages 1133–1140, 2009.

[45] J. Rasch and A. Chambolle. Inexact first-order primal–dual algorithms. *Computational Optimization and Applications*, 76(2):381–430, 2020.

[46] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, 1976.

[47] I. J. Schoenberg. Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert". *Annals of Mathematics*, 36(3):724–732, 1935.

[48] I. J. Schoenberg. Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–536, 1938.

[49] R. N. Shepard. The analysis of proximities: multidimensional scaling with an unknown distance function. i. *Psychometrika*, 27(2):125–140, 1962.

[50] M. V. Solodov and B. F. Svaiter. A unified framework for some inexact proximal point algorithms. *Numerical Functional Analysis and Optimization*, 22(7-8):1013–1035, 2001.

[51] L. Song, A. Smola, K. Borgwardt, and A. Gretton. Colored maximum variance unfolding. In *Advances in Neural Information Processing Systems 20*, 2007.

[52] J. F. Sturm. Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1–4):625–653, 1999.

[53] Y. Sun and L. Vandenberghe. Decomposition methods for sparse matrix nearness problems. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1691–1717, 2015.

[54] K. C. Toh, R. H. Tütüncü, and M. J. Todd. *SDPT3 version 3.02. A MATLAB software for semidefinite-quadratic-linear programming*, 2002. available at www.math.nus.edu.sg/~mattohkc/sdpt3.html.

[55] J. Tuck, S. Barratt, and S. P. Boyd. A distributed method for fitting Laplacian regularized stratified models. *Journal of Machine Learning Research*, 22(60):1–37, 2021.

[56] L. Vandenberghe and M. S. Andersen. Chordal graphs and semidefinite optimization. *Foundations and Trends in Optimization*, 1(4):241–433, 2015.

[57] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, 2004.

[58] K. Q. Weinberger and L. K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *National Conference on Artificial Intelligence*, pages 1683–1686, 2006.

[59] D. M. Witten and R. Tibshirani. Supervised multidimensional scaling for visualization, classification, and bipartite ranking. *Computational Statistics & Data Analysis*, 55(1):789–801, 2011.

[60] D. Yang. *Further developments of multidimensional scaling via Euclidean distance matrix optimization*. PhD thesis, University of Southampton, 2022.

[61] G. Young and A. S. Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3(1):19–22, 1938.

[62] S. Zhou, N. Xiu, and H.-D. Qi. A fast matrix majorization–projection method for penalized stress minimization with box constraints. *IEEE Transactions on Signal Processing*, 66(16):4331–4346, 2018.

[63] S. Zhou, N. Xiu, and H.-D. Qi. Robust Euclidean embedding via EDM optimization. *Mathematical Programming Computation*, 12(3):337–387, 2020.