Metric Nearness Made Practical

Wenye Li^{1,2}, Fangchen Yu^{1}, Zichen Ma^{1}

^{1} The Chinese University of Hong Kong, Shenzhen
^{2} Shenzhen Research Institute of Big Data
2001 Longxiang Boulevard, Longgang District, Shenzhen, China
wyli@cuhk.edu.cn, fangchenyu@link.cuhk.edu.cn, zichenma1@link.cuhk.edu.cn

Abstract

Given a square matrix with noisy dissimilarity measures between pairs of data samples, the metric nearness model computes the best approximation of the matrix from a set of valid distance metrics. Despite its wide applications in machine learning and data processing tasks, the model faces non-trivial computational requirements in seeking the solution due to the large number of metric constraints associated with the feasible region. Our work designed a practical approach in two stages to tackle the challenge and improve the model's scalability and applicability. The first stage computes a fast yet high-quality approximate solution from a set of isometrically embeddable metrics, further improved by an effective heuristic. The second stage refines the approximate solution with the Halpern-Lions-Wittmann-Bauschke projection algorithm, which converges quickly to the optimal solution. In empirical evaluations, the proposed approach runs at least an order of magnitude faster than the state-of-the-art solutions, with significantly improved scalability, complete conformity to constraints, less memory consumption, and other desirable features in real applications.

Introduction

A distance metric, or distance matrix, measures the pairwise dissimilarity relationship between data samples. It is crucial and lays a foundation for numerous data processing tasks and machine learning models. For example, many supervised and unsupervised learning algorithms firmly reside on distance metrics (Jain, Murty, and Flynn 1999; Hart, Stork, and Duda 2000; Schölkopf et al. 2002; Xing et al. 2002).

Assumptions are often made about the characteristics that a distance matrix needs to satisfy. Non-negativity, symmetry, and a zero diagonal of the matrix entries are commonly considered under various scenarios (Boytsov and Naidan 2013; Schleif and Tino 2015). For metric-based models (Brickell et al. 2008; Gisbrecht and Schleif 2015), triangle inequalities of the pairwise distances are further assumed. However, with erroneous or missing measurements, an observed matrix often loses its conformity to such characteristics. In practice, recovering an actual distance matrix's characteristics can be beneficial, which may lead to an improved estimate of the unknown ground truth (Deutsch 2001).

A series of work (Brickell et al. 2008; Duggal et al. 2013; Boytsov and Naidan 2013; Gilbert and Jain 2017; Fan, Raichek, and Van Buskirk 2018; Gilbert and Sonthalia 2018; Veldt et al. 2018; Ruggles, Veldt, and Gleich 2020) models the scenario of *repairing* a metric under different settings, among which we are interested in a specific *metric nearness* model in this paper. For a set of input dissimilarity values between pairs of data samples, the model aims to output a set of distances that satisfy the metric constraints while keeping the output distances as *near* as possible to the input dissimilarity according to a certain measure of *nearness*.

Conceptually general and straightforward as it is, the metric nearness model encounters significant computational difficulty. The number of constraints associated with a valid metric rapidly grows when the problem size increases. Despite the recent trials devoted to the problem (Brickell et al. 2008; Sonthalia and Gilbert 2020; Pitis et al. 2020), the proposed approaches still faced non-trivial challenges from medium to large-sized applications with over a few thousand samples on a common computing platform.

New optimization strategies have to be sought to ensure scalability. We developed a dedicated approach that is very different from existing ones to the metric nearness problem. The approach first shrinks the scope of distance metrics to isometrically embeddable matrices. Consequently, the number of constraints in the optimization problem is significantly reduced, which admits a fast solution with high quality. Then, starting from the solution and refining it iteratively, the proposed approach is expected to converge faster to the optimal solution to the original problem.

Evaluation results empirically justified our idea. The proposed approach has a significantly faster running speed than the state-of-the-art solutions. It consumes much less memory. The solution found by our approach completely conforms to the metric constraints. As a result, the application scenarios of the metric nearness model are significantly expanded toward a practical tool for large-scale tasks.

A note on notations. Throughout the paper, a capital letter in calligraphic fonts, usually with a subscript, denotes a set, such as $\mathcal{M}_n$. A capital letter in regular fonts denotes a real matrix with each entry represented by a lower-case letter with subscripts, such as X and x_{ij}. By default, the sizes of the matrices are all assumed to be $n \times n$, where n is a given natural number.

The rest of the paper is structured as follows. We first introduce the basic metric nearness model and algorithms. Then we present our algorithm and report the experimental results, followed by concluding remarks.

Related Work

The Metric Nearness Problem

Problem Formulation: Denote by $\mathcal{M}_n$ a subset of $n \times n$ real matrices, each satisfying the metric constraints. Specifically, a matrix $X \in \mathbb{R}^{n\times n}$ meets the metric conditions if and only if it is a symmetric and non-negative matrix with a zero diagonal and all its entries satisfying the triangle inequalities:

$$x_{ii} = 0,\ x_{ij} \geq 0,\ x_{ij} = x_{ji},\ \text{and}\ x_{ij} \leq x_{ik} + x_{kj} \quad (1)$$

for all $1 \leq i, j, k \leq n$. In this paper, we also call such a matrix X a *distance metric*.

For n given samples, let $D^o \in \mathbb{R}^{n\times n}$ be an observed or measured matrix with each entry quantifying a dissimilarity value between two samples. Assume that $D^o = D^g + \Theta$ where D^g denotes the (unknown) ground truth of the pairwise distance matrix between these samples, and Θ denotes the measurement noise. Due to the noise, D^o may not satisfy the metric constraints (i.e., $D^o \notin \mathcal{M}_n$) that the ground truth conforms to.

The metric nearness model (Brickell et al. 2008) seeks a valid metric X that is *nearest* to the observation D^o by:

$$\min_{X\in\mathcal{M}_n} \|X - D^o\|_F^2 \quad (2)$$

where the *nearness* is measured by the squared Frobenius norm of $X - D^o$. It can be seen that the feasible region $\mathcal{M}_n$ is a closed convex cone, and the problem defined in Eq. (2) is a convex optimization problem with a unique minimizer.

Relationship with Best Approximations: One meaningful application of the metric nearness model can be illustrated from the viewpoint of best approximations. Recall a characterization of best approximations from convex sets (Deutsch 2001).

Lemma 1 *Let $\mathcal{C}$ be a convex subset of the inner product space $\mathcal{H}$. Let $x^0 \in \mathcal{H}$ and $x^0_{\mathcal{C}} \in \mathcal{C}$. Denote by $\mathrm{P}_{\mathcal{C}}(x^0)$ the projection of x^0 onto $\mathcal{C}$. Then $x^0_{\mathcal{C}} = \mathrm{P}_{\mathcal{C}}(x^0)$ if and only if*

$$\left\langle x^0 - x^0_{\mathcal{C}}, x - x^0_{\mathcal{C}} \right\rangle \leq 0, \text{ for all } x \in \mathcal{C}$$

The "=" holds if and only if $x^0 \in \mathcal{C}$.

The characterization leads to the following result:

Fact 1 *Let X^o be an $n \times n$ observed distance matrix, and let $\mathcal{M}_n$ be the set of $n \times n$ matrices that satisfy the metric conditions defined in Eq.* (1). *Let $X^o_{\mathcal{M}_n}$ be the projection of X^o onto $\mathcal{M}_n$ and let $X^g \in \mathcal{M}_n$ be the (unknown) ground truth metric. Then*

$$\left\|X^o_{\mathcal{M}_n} - X^g\right\|_F^2 \leq \|X^o - X^g\|_F^2$$

The "=" holds if and only if $X^o \in \mathcal{M}_n$.

This result presents a valuable property of the metric nearness model. Although the ground truth X^g is unknown, calibrating a noisy observation X^o to meet the metric conditions produces a better estimate of the ground truth in terms of a shorter distance derived from the Frobenius norm (Li 2015).

Existing Approaches

Existing approaches to solving the metric nearness problem defined in Eq. (2) can be divided into three categories.

Off-the-Shelf Solvers: The problem defined in Eq. (2) can be converted to a standard quadratic program (Boyd and Vandenberghe 2004) and processed by mathematical solvers such as CPLEX or MOSEK. Unfortunately, the $O(n^3)$ triangle inequality constraints inherent in the feasible region $\mathcal{M}_n$ significantly limit the solution's scalability. Although the solvers can be combined with active set methods (Boggs and Tolle 1995), which may potentially reduce the number of operational constraints, the intrinsic difficulty remains as the number of constraints grows too fast with n.

Generic Optimization Techniques: Gradient-based approaches and variants (Beck and Teboulle 2009; Nedić 2011; Wang and Bertsekas 2015; Nesterov 2018) work efficiently on a broad range of optimization problems. Meanwhile, all these approaches encounter non-trivial difficulties when being applied to metric-constrained problems. With increased constraints, gradient-based approaches typically need more iterations to converge. Another classical approach, the Lagrangian-type method (Fletcher 2013), which augments the objective by adding a term for each constraint, suffers similarly from a large number of constraints, where computing the gradient becomes intractable.

Dedicated Algorithms: Dedicated algorithms were developed to solve the metric nearness problem more efficiently. The Triangle Fixing (TRF) algorithm (Brickell et al. 2008) iterates through the triangle inequalities based on a primal-dual method, optimally enforces the unsatisfied inequality, and scales better than the generic optimization approaches. A more recent solution, the Project and Forget (PAF) algorithm (Sonthalia and Gilbert 2020), examines each constraint once, then introduces new constraints and removes old constraints. In this way, the method reduces the number of constraints and improves the scalability. Besides, by further assuming that the distance metric is isometrically embeddable, a more scalable solution is possible which utilizes the metric's relationship with positive semidefiniteness (Schoenberg 1938; Li and Yu 2023).

In addition to the optimization techniques, a learning-based approach was investigated. A neural network structure called "DeepNorm" (Pitis et al. 2020) is trained with a noisy matrix (with violations of metric constraints) as input and a distance matrix (without violations) as the output. The learned network can be applied to future input matrices to produce the outputs, aiming to remove the violations of the metric constraints.

Despite the partial success that has been achieved, the dedicated approaches still suffer heavily from the $O(n^3)$ metric constraints, and the improvement seems limited. More scalable solutions are expected to make the metric nearness model practical in applications.

Model

Our work proposes a new solution to the metric nearness problem defined in Eq. (2). The solution is comprised of two stages. The initialization stage seeks a fast approximate solution in $\mathcal{M}_n$ yet with high quality through embedding calibration and heuristic improvement (Li and Yu 2023). The alternating projection stage iteratively refines the approximate solution to the optimal one.

Embedding Calibration

Seeking a solution to Eq. (2) in $\mathcal{M}_n$ involves $O(n^3)$ metric constraints defined in Eq. (1). To handle the computational difficulty from so many constraints, we turn to a subset of $\mathcal{M}_n$, which avoids the explicit specification of so many constraints. Denote the region by $\mathcal{I}_n$, including matrices that are isometrically embeddable in the Euclidean space. We define:

Definition 1 *A matrix $X \in \mathbb{R}^{n\times n}$ is said to be isometrically embeddable in the Euclidean space if there exists a set of points $p_1, \cdots, p_n$ in the space and a distance function ρ defined on pairs of points, having the properties that $\rho(p,p') = \rho(p',p) \geq 0$ and $\rho(p,p) = 0$ for all p, p' in the space, such that*

$$\rho(p_i, p_j) = x_{ij}$$

holds for all $1 \leq i,j \leq n$.

Schoenberg's classical result on isometrical embedding (Schoenberg 1938; Wells and Williams 1975) provides a sufficient and necessary condition for a matrix to be isometrically embeddable.

Theorem 1 *Given $X \in \mathbb{R}^{n\times n}$ with $x_{ii} = 0$ and $x_{ij} \geq 0$ for all $1 \leq i,j \leq n$, $X \in \mathcal{I}_n$ if and only if the quadratic form*

$$\sum_{i,j=1}^{n} \exp\left(-\gamma x_{ij}\right) \xi_i \xi_j \geq 0$$

holds for all choices of real numbers $\xi_1, \cdots, \xi_n$ and $\gamma \to 0^+$ (γ tends to 0 from the right), i.e., the matrix $E = \exp(-\gamma X) = \{\exp(-\gamma x_{ij})\} \succeq 0$.

The result establishes the relationship between isometrically embeddable matrices and positive semi-definite matrices. Based on the theoretical discovery, we can seek a fast approximate solution to the metric nearness problem by confining the solution within $\mathcal{I}_n$ instead of $\mathcal{M}_n$:

$$\min_{X\in\mathcal{I}_n} \frac{1}{2}\sum_{i,j=1}^{n}\left(x_{ij} - d_{ij}^o\right)^2 \tag{3}$$

Denote the optimal solution to Eq. (3) by X^c. Up to now, obtaining X^c still seems not straightforward. To make it clear, note that for a sufficiently small and positive value γ [1], $\exp(-\gamma x_{ij}) = 1 - \gamma x_{ij} + O(\gamma^2 x_{ij}^2)$ and $\exp(-\gamma d_{ij}^o) = 1 - \gamma d_{ij}^o + O(\gamma^2 (d_{ij}^o)^2)$. Then we have $\left(x_{ij} - d_{ij}^o\right)^2 \approx \frac{1}{\gamma^2}\left[\exp(-\gamma x_{ij}) - \exp(-\gamma d_{ij}^o)\right]^2$. Thus, it is expected that the optimal solution to Eq. (3), X^c, approximately solves:

$$\min_{X\in\mathcal{I}_n} \frac{1}{2\gamma^2}\sum_{i,j=1}^{n}\left[\exp(-\gamma x_{ij}) - \exp(-\gamma d_{ij}^o)\right]^2 \tag{4}$$

[1]In this paper, we set $\gamma = 0.02/\max\left\{d_{ij}^o\right\}$.

Denote $E = \{e_{ij}\} = \{\exp(-\gamma x_{ij})\}$. Based on Theorem 1, we reach a new optimization problem for Eq. (4):

$$\min_{E\in\mathcal{E}} \frac{1}{2\gamma^2}\sum_{i<j}\left[e_{ij} - \exp(-\gamma d_{ij}^o)\right]^2 \tag{5}$$

where

$$\mathcal{E} = \left\{X \in \mathbb{R}^{n\times n} | X \succeq 0, x_{ii} = 1, 0 \leq x_{ij} \leq 1, \forall i,j\right\}.$$

Denote the solution to Eq. (5) by E^*. Then we have $X^c \approx -\frac{1}{\gamma}\log(E^*)$.

Now the problem changes to solving E^* in Eq. (5), which is a well-studied problem in the optimization community. The feasible region $\mathcal{E}$ is the intersection of a convex cone of positive semi-definite matrices

$$\mathcal{S} = \left\{X \in \mathbb{R}^{n\times n} | X \succeq 0\right\}$$

and the set of matrices with box constraints

$$\mathcal{T} = \left\{X \in \mathbb{R}^{n\times n} | x_{ii} = 1, 0 \leq x_{ij} \leq 1, \forall i,j\right\}.$$

In literature, projecting a given matrix $E^o = \exp(-\gamma D^o) = \left\{\exp(-\gamma d_{ij}^o)\right\}$ in $\mathbb{R}^{n\times n}$ onto $\mathcal{S}\cap\mathcal{T}$ can be solved by Newton's method (Qi and Sun 2006), or by alternating projections such as Dykstra's algorithm (Boyle and Dykstra 1986; Higham 2002; Li 2015).

We chose Dykstra's projection. Starting from E^0, the algorithm generates a sequence of iterates $\left\{E_\mathcal{S}^t, E_\mathcal{T}^t\right\}$ and increments $\left\{I_\mathcal{S}^t, I_\mathcal{T}^t\right\}$, for $t = 1, 2, \cdots$, by:

$$E_\mathcal{S}^t = \mathrm{P}_\mathcal{S}\left(E_\mathcal{T}^{t-1} - I_\mathcal{S}^{t-1}\right) \tag{6}$$
$$I_\mathcal{S}^t = E_\mathcal{S}^t - \left(E_\mathcal{T}^{t-1} - I_\mathcal{S}^{t-1}\right) \tag{7}$$
$$E_\mathcal{T}^t = \mathrm{P}_\mathcal{T}\left(E_\mathcal{S}^t - I_\mathcal{T}^{t-1}\right) \tag{8}$$
$$I_\mathcal{T}^t = E_\mathcal{T}^t - \left(E_\mathcal{S}^t - I_\mathcal{T}^{t-1}\right) \tag{9}$$

where $E_\mathcal{T}^0 = E^0$, $I_\mathcal{S}^0 = \mathbf{0}$, $I_\mathcal{T}^0 = \mathbf{0}$, $\mathbf{0}$ is an all-zero matrix, and P denotes the projection operation onto a specified closed convex subset. The sequences $\left\{E_\mathcal{S}^t\right\}$ and $\{E_\mathcal{T}^t\}$ converge to the optimal solution as $t \to \infty$.

For the projection onto $\mathcal{S}$ and $\mathcal{T}$, we have:

Fact 2 *Let $U\Sigma U^\top$ be the eigen-decomposition of $X \in \mathbb{R}^{n\times n}$ with $\Sigma = \mathrm{diag}\left(\lambda_1, \cdots, \lambda_n\right)$. The projection of X onto $\mathcal{S}$ is given by: $X_\mathcal{S} = \mathrm{P}_\mathcal{S}(X) = U\Sigma' U^\top$ where $\Sigma' = \mathrm{diag}\left(\lambda_1', \cdots, \lambda_n'\right)$ and each $\lambda_i' = \max\left\{\lambda_i, 0\right\}$.*

Fact 3 *The projection of $X \in \mathbb{R}^{n\times n}$ onto $\mathcal{T}$ is given by: $X_\mathcal{T} = P_\mathcal{T}(X) = \left\{(x_\mathcal{T})_{ij}\right\}_{i,j=1}^{n}$ where $(x_\mathcal{T})_{ij} = \mathrm{med}\left\{0, x_{ij}, 1\right\}$, i.e., the median of the three numbers.*

Obtaining the accurate E^* is actually unnecessary in our problem. We just run Dykstra's projection for a few iterations, which has been empirically verified to be efficient in obtaining a good approximation to the projection onto the intersection of convex sets (Censor 2006).

Heuristic Improvement

With E^* and an approximation of X^c by $-\frac{1}{\gamma}\log(E^*)$, we further improve X^c with a heuristic. The basic idea is the following. For each element x_{ij}^c of X^c, we move it nearer to d_{ij}^o as much as possible while keeping the improved X^c within the feasible region $\mathcal{M}_n$.

The heuristic works as follows. Suppose x_{ij}^c is larger than d_{ij}^o. In order to reduce the objective $\|X - D^o\|_F^2$, we need to reduce x_{ij}^c in the right side of d_{ij}^o while maintaining the triangle inequalities involving x_{ij}^c. The smallest possible value of x_{ij}^c to keep the triangle inequalities is determined by the maximum value of all $\left|d_{ki}^o - d_{kj}^o\right|$ $(k \neq i, j)$. On the other hand, if x_{ij}^c is smaller than d_{ij}^o, then we need to increase the value of x_{ij}^c in the left side of d_{ij}^o. This can be done by setting it to the minimum value of $d_{ik}^o + d_{kj}^o$ $(k \neq i, j)$.

One by one, we refine the element of X^c and denote the improved metric by X^0. It is clear that, for $X^c \in \mathcal{M}_n$, the heuristic improvement guarantees that $X^0 \in \mathcal{M}_n$ and $\left\|X^0 - D^o\right\|_F^2 \leq \|X^c - D^o\|_F^2$.

The HLWB Projection Algorithm

For $1 \leq i \neq j \neq k \leq n$, let

$$\mathcal{C}_{ijk} = \left\{X \in \mathbb{R}^{n\times n} | x_{ij} \leq x_{ik} + x_{kj}\right\}.$$

Each $\mathcal{C}_{ijk}$ denotes a half space defined on the entries of $X \in \mathbb{R}^{n\times n}$. The region defined by the triangle inequalities in Eq. (1) is the intersection of all $\mathcal{C}_{ijk}$'s. For simplicity of discussion, here we ignore the constraints of symmetry, nonnegativity, and a zero diagonal on the matrix entries, which are not crucial to the development of our optimization procedure but can be easily included back into the algorithm.

For the projection onto the intersection of multiple closed convex sets, we have (Halpern 1967; Lions 1977; Wittmann 1992; Bauschke 1996):

Theorem 2 *Let $\mathcal{C}_1, \cdots, \mathcal{C}_m$ be a family of closed convex subsets of the Euclidean space such that $\mathcal{M}_n = \mathcal{C}_1 \cap \cdots \cap \mathcal{C}_m$ is not empty. Let $D^o \in \mathbb{R}^{n\times n}$ and $X^0 \in \mathbb{R}^{n\times n}$. Set*

$$\begin{cases} Y^{t+1} = \dfrac{1}{t+2} D^o + \dfrac{t+1}{t+2} X^t \\ X^{t+1} = \mathrm{P}_{\mathcal{C}_1} \cdots \mathrm{P}_{\mathcal{C}_m}(Y^t) \end{cases} \tag{10}$$

where $t = 0, 1, \cdots$. Then $X^t \to \mathrm{P}_{\mathcal{M}_n}(D^o)$ and $Y^t \to \mathrm{P}_{\mathcal{M}_n}(D^o)$ as $t \to \infty$.

Theorem 2 provides an alternating procedure, called the Halpern-Lions-Wittmann-Bauschke (HLWB) algorithm, to sequentially project a given point D^o in the Euclidean space onto the intersection of multiple closed convex subsets. The algorithm starts from a point X^0 in the space, which can be the same as or different from the given D^o. Project the point successively onto each subset. Then move the projection a bit along the direction to D^o. Adopting an appropriate step size (e.g., $\frac{1}{t+2}$) and repeating the projection and movement, the procedure converges to the unique projection of D^o onto the intersection of the subsets.

Each convex set $\mathcal{C}_{ijk}$ denotes a half-space specified by $x_{ij} - x_{ik} - x_{kj} \leq 0$ and the projection onto the half-space is a well-known result in literature (Bregman et al. 2003).

Fact 4 *For any $X \in \mathbb{R}^{n\times n}$, its projection onto $\mathcal{C}_{ijk}$ is given by a matrix $X' \in \mathbb{R}^{n\times n}$ where $x'_{ij} = x_{ij} - \delta_{ijk}$, $x'_{ik} = x_{ik} + \delta_{ijk}$, $x'_{jk} = x_{jk} + \delta_{ijk}$ with $\delta_{ijk} = \max\left\{\frac{x_{ij}-x_{ik}-x_{jk}}{3}, 0\right\}$ and all the other elements of X' remain the same as X. Furthermore, if $X \in \mathcal{M}_n$, then $X' \in \mathcal{M}_n$.*

The sequential HLWB algorithm based on Theorem 2 projects a given point onto various half spaces $\mathcal{C}_{ijk}$ one by one. In addition, a parallel update scheme is feasible, which projects the point onto the half spaces concurrently in each iteration and then averages the projection results with a theoretical guarantee of convergence and optimality (Bauschke and Combettes 2011). The scheme leads to a computing procedure suitable for parallel computing platforms, often at the cost of more iterations to converge. Further details can be found in (Censor 2006).

Algorithm Summary and Complexity Analysis

The proposed approach is summarized in Algorithm 1. Here is an analysis of the algorithm's complexity. Suppose the input matrix is of size $n \times n$. In the initialization stage (steps 1-2), decomposing the matrix D^o (see Fact 2) typically has a complexity of $O(n^3)$ (Pan and Chen 1999), so does the heuristic improvement. In each iteration of the alternating projection stage (steps 3-14), $O(n^3)$ triples of matrix elements will be checked on the conformity to the triangle inequalities. Only those triples that violate the constraints will be updated, the number of which is usually far less than $O(n^3)$.

The proposed algorithm must store and update the distance matrix in both stages, with a memory complexity of $O(n^2)$. The requirement is far less than those of the TRF and PAF algorithms, which are both $O(n^3)$ in the worst case. In our evaluation, the results clearly demonstrated the benefit of the reduced memory requirement.

Algorithm 1: The Proposed HLWB Algorithm

```
Input: $D^o, maxiter, \epsilon$
Output: $X^t$
1:  $X^c \leftarrow$ EmbeddingCalibration $(D^o)$
2:  $X^0 \leftarrow$ HeuristicImprovement $(X^c)$
3:  for $t \leftarrow 1, \cdots, maxiter$ do
4:      $X^t \leftarrow \frac{1}{t+2} \times D^o + \frac{t+1}{t+2} \times X^{t-1}$;
5:      for $i \leftarrow 1, \cdots, n$ do
6:          for $j \leftarrow 1, \cdots, n$ do
7:              for $k \leftarrow 1, \cdots, n$ do
8:                  $\delta \leftarrow \frac{X_{ij}^t - X_{ik}^t - X_{kj}^t}{3}$;
9:                  if $\delta > 0$ then                ▷ violation found
10:                     $X_{ij}^t \leftarrow X_{ij}^t - \delta$;
11:                     $X_{ik}^t \leftarrow X_{ik}^t + \delta$;
12:                     $X_{kj}^t \leftarrow X_{kj}^t + \delta$;
13:     if $\left\|X^t - X^{t-1}\right\|_F < \epsilon$ then
14:         break;
```

Algorithm	CPLEX	MOSEK	TRF	PAF	DeepNorm	HLWB
Largest Size (n)	< 300	< 300	$< 2,000$	$\sim 3,000$	$< 1,500$	$> 10,000$

Table 1: The size of the largest problem that each algorithm solved within 12 hours in the experiments.

Evaluation

The proposed algorithm was compared with existing approaches, including the Triangle Fixing (TRF) algorithm (Brickell et al. 2008) [2], the Project and Forget (PAF) algorithm (Sonthalia and Gilbert 2020) [3], and the DeepNorm algorithm (Pitis et al. 2020) [4]. As a baseline, two generic optimization solvers CPLEX [5] and MOSEK [6] were also tested. Most experiments were executed on a conventional server with a single CPU (intel Xeon 8180) enabled, except the DeepNorm algorithm that ran on a deep learning platform.

Artificial and real datasets were used in the evaluation. Similar to the work of (Sonthalia and Gilbert 2020), two sets of complete graphs with $n = 100/500/1,000/1,500$ nodes were artificially generated. On one set of graphs (ref. Graph-t1), the weight of each edge was set to $w \sim \mathrm{U}(0,1)$, a uniform distribution between 0 and 1. On the other set (ref. Graph-t2), each weight was set to $\lceil 1000 \times u \times v^2 \rceil$ with $u \sim \mathrm{U}(0,1)$ and $v \sim \mathrm{N}(0,1)$, a normal distribution. Then the edge weights were used as the measurement matrix D^o.

Besides, the real MNIST dataset (LeCun et al. 1998)[7], which consists of grayscale images of hand-written digits, was used in the experiment. The pairwise distances of the images were calculated as the ground-truth metric D^g. The measurement matrix D^o was formed by adding random noise to the elements of D^g, with each $d^o_{ij} = \max\left\{0, d^g_{ij} + \zeta \times \mathrm{mean}(D^g) \times \mathrm{N}(0,1)\right\}$ where $\zeta = 0.5$ and 0.8 respectively and $\mathrm{mean}(D^g)$ denotes the mean value of all entries of D^g.

Scalability

The algorithms exhibited very different scalability in practice. As shown in Table 1, the generic solvers, CPLEX and MOSEK, ran out of memory when the sample size (n) reached 300. The TRF algorithm encountered the same issue with $n = 2,000$. The PAF algorithm did not converge with $n = 3,000$ in 12 hours, and the DeepNorm algorithm did not finish the training with $n = 1,500$ samples within the same time limit. Comparatively, the proposed HLWB algorithm converged within the same period with $n = 10,000$, which scaled significantly better than the other approaches.

As the generic solvers' scalability on the metric nearness problem is too limited, we did not consider them in the following discussion. On the other hand, the DeepNorm algorithm resides on supervised training to obtain a neural network for repairing the metric and has to be evaluated under a very different experimental setting from the other approaches, so we didn't include its results in the sequel without affecting the presentation of optimization qualities and comparisons.

[2]https://optml.mit.edu/work/soft/metricn.html.

[3]https://github.com/rsonthal/ProjectAndForget.

[4]https://github.com/spitis/deepnorms.

[5]https://www.ibm.com/academic/topic/data-science.

[6]https://www.mosek.com/downloads/.

[7]http://yann.lecun.com/exdb/mnist/.

NMSE and CSR

We first compared the optimization quality of the TRF, PAF, and HLWB algorithms on the datasets with sizes n varying from 100 to $1,500$ under which all three algorithms converged. Since all the algorithms update the metric entries iteration by iteration to improve the *nearness* and increase the conformity to the constraints, naturally, we present the algorithms' performances against the number of iterations.

Nearness was measured by the normalized mean squared error (NMSE), which is defined by $\frac{\|X^* - D^o\|_F^2}{\|D^o\|_F^2}$ where X^* is the projection of D^o obtained by different algorithms. The results were given in Fig. 1, where all algorithms converged to similar NMSE values in less than 50 iterations, with optimization objective differences less than 10^{-4}.

The constraints satisfaction ratio (CSR) was used to quantify the conformity to triangle inequalities, which is defined as the number of triangle inequalities that are satisfied divided by the total number of triangle inequalities. TRF and PAF algorithms did not achieve complete conformity (CSR $= 1.0$) in most experiments, falling into the range of 0.970 to 0.999, which resulted in many constraint violations. Even if we increased the number of iterations to 500, the CSR did not improve significantly [8]. Comparatively, the proposed HLWB algorithm consistently achieved CSR $= 1.0$ from all experiments' first iteration.

Number of Updates and Running Time

We recorded the actual running time of all algorithms iteration by iteration. On the MNIST dataset with $n = 1,500$ and $\zeta = 0.8$, the TRF algorithm took around 34 seconds per iteration and around $1,700$ seconds for 50 iterations, including the algorithm's initialization time. The PAF algorithm took mostly around 18 seconds per iteration, except for the first iteration, which took nearly 500 seconds, causing around $1,600$ seconds in total for 50 iterations. Comparatively, the HLWB algorithm took around 0.96 seconds per iteration and around 50 seconds for 50 iterations, including the initialization time. The proposed algorithm brought about more than an order of magnitude improvement over the other two algorithms to converge.

In addition to the running time, we recorded the number of updates on the distance matrix entries iteration by iteration.

[8]On minor problems with $n = 100/500$, all constraint violations diminished after running TRF and PAF for a vast number $(10,000)$ of iterations.

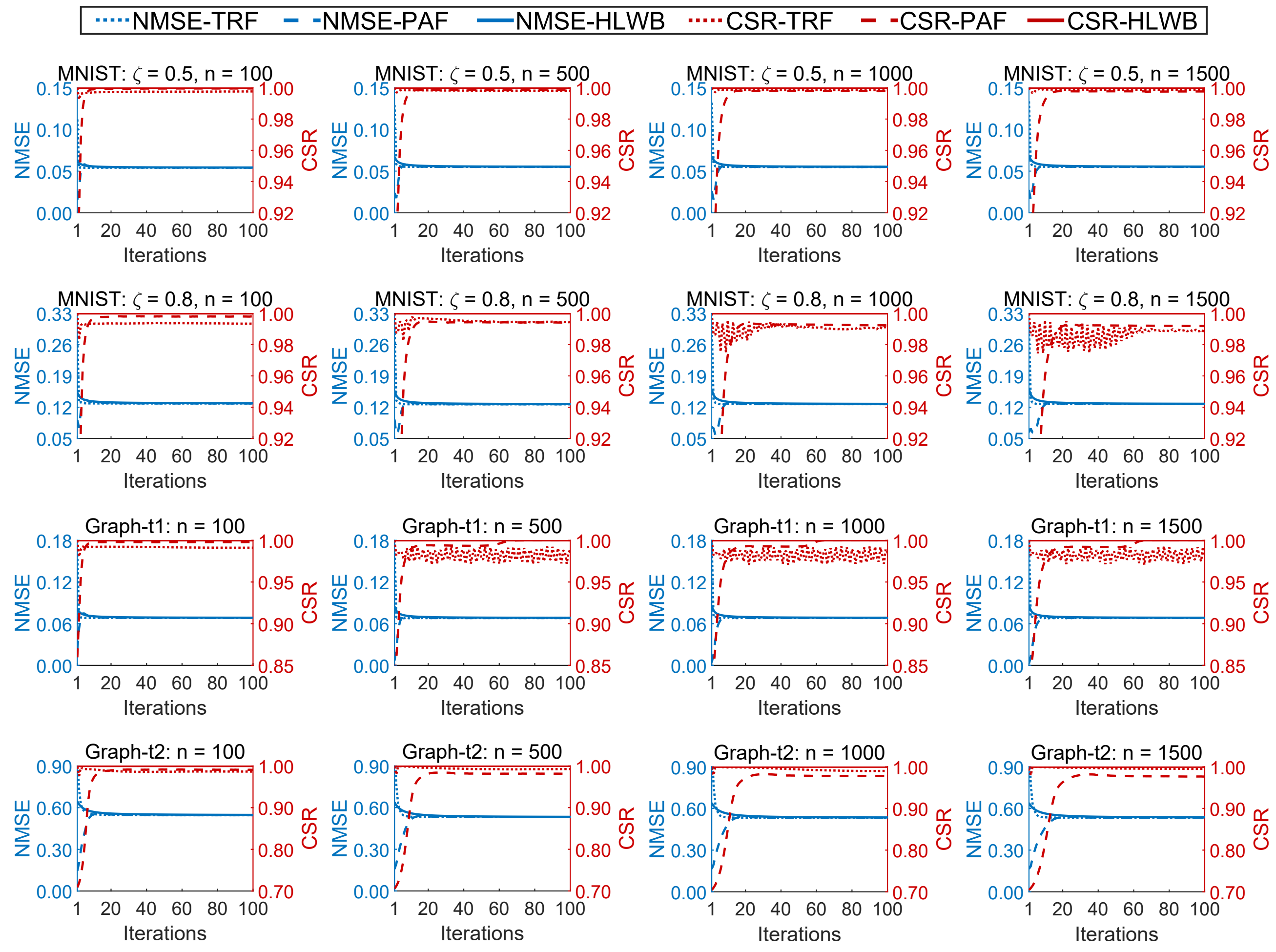

Figure 1: Comparison of NMSE and CSR versus Iterations. All algorithms reported similar NMSE values when converged. The HLWB algorithm exhibited complete conformity to the metric constraints.

On the MNIST dataset under the same setting, the TRF algorithm updated 2.53×10^{11} times in 50 iterations. Comparatively, the HLWB algorithm updated 1.77×10^{9} times, and the PAF algorithm updated 3.0×10^{8} times only. Although the PAF algorithm carried out fewer updates, it needed to perform expensive operations other than the simple matrix updates, such as calculating graph shortest paths, which are significantly more time-consuming and make the PAF algorithm much slower than the HLWB algorithm.

Memory Consumption

After evaluating the optimization quality and running speed, we evaluate the memory requirements to execute the algorithms. Fig. 3 compares the empirical memory consumption by different algorithms on the MNIST dataset with $n = 1,500$ samples. The TRF and the PAF algorithms needed more than 3GB memory, while the HLWB algorithm only consumed less than 0.1GB memory. With $n = 10,000$ samples, TRF reported memory overflow, and PAF needed around 30GB memory. Comparatively, HLWB only consumed around 3GB memory.

The empirical results justified the theoretical advantage of the proposed algorithm in memory consumption. The memory complexity of $O(n^2)$ by the HLWB algorithm significantly improves the memory complexity of $O(n^3)$ by the TRF algorithm and the PAF algorithm.

Conclusion

The metric nearness problem is fundamental and commonly found in data processing tasks. In practice, heavy computational requirements and memory consumption significantly limit its application scenarios. Our work developed a novel approach to tackle the challenge. An isometrically embeddable metric (Schoenberg 1938; Hayden and Wells 1988), which lies in a subset of all feasible metrics, is first obtained efficiently based on existing procedures (Qi and Yuan 2014; Li and Yu 2023). The HLWB algorithm is then applied to refine the metric to the optimum with a theoretical guarantee. Empirically, the proposed approach exhibited significantly improved running speed, complete conformity to metric constraints, and less memory consumption.

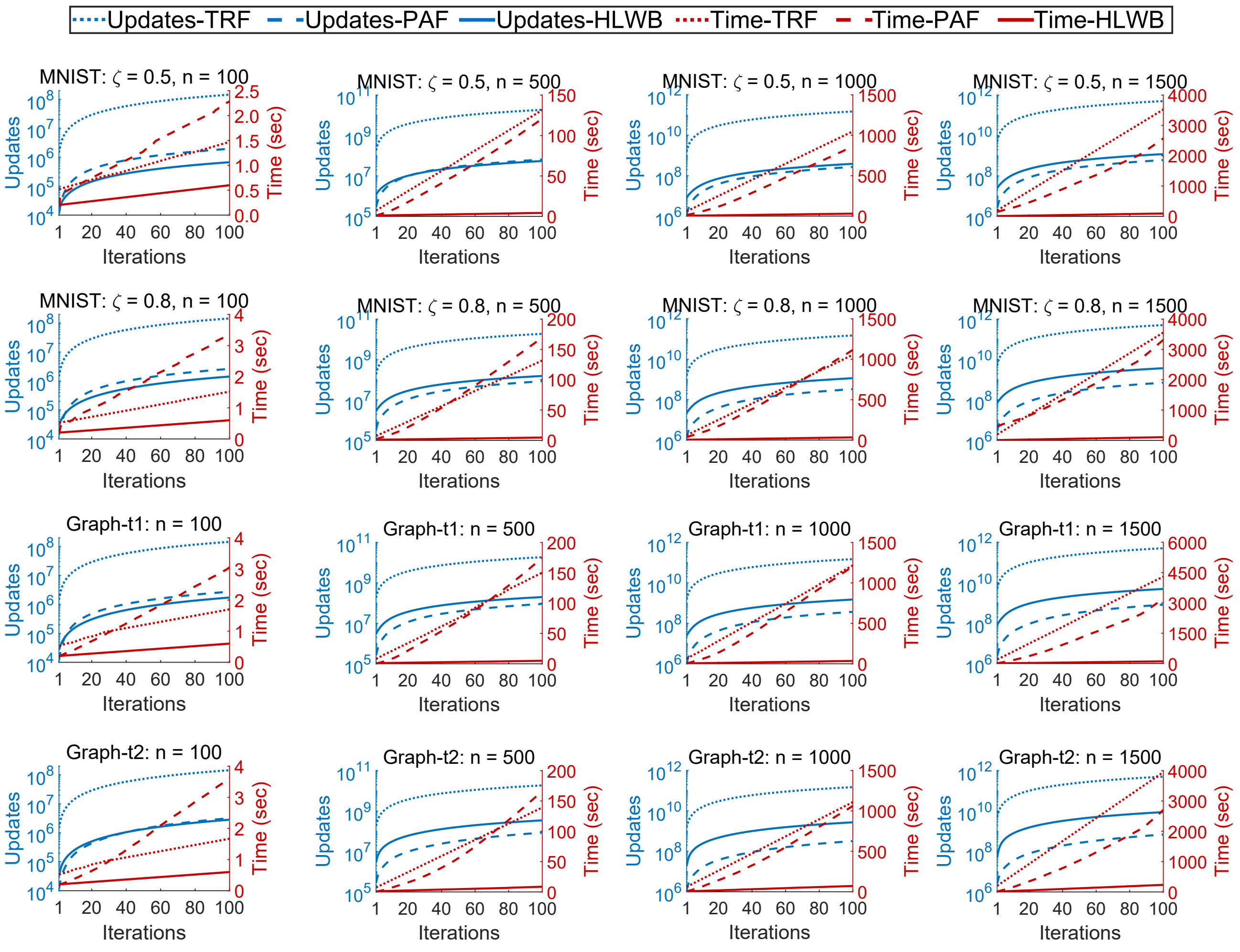
Figure 2: Comparison of Updates and Running Time versus Iterations on the MNIST Dataset. The PAF algorithm and the HLWB algorithm made much fewer updates on matrix entries. The HLWB algorithm ran an order of magnitude faster.

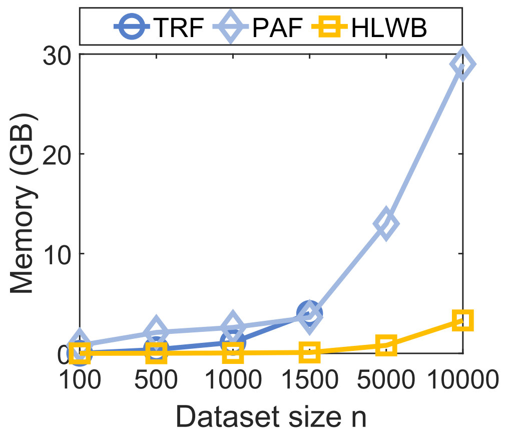
Figure 3: Comparison of Memory Usage vs. Problem Size on the MNIST Dataset. Note: On $n = 5,000/10,000$, TRF reported memory overflow, and PAF didn't converge in 12 hours. The captured usage during the time limit likely underestimated PAF's memory requirements.

Despite the progress, much work can be done and deserves our attention. One line is on a parallel implementation of the proposed approach to ensure even better scalability, which will surely benefit from the development of parallel matrix decomposition methods (Berry et al. 2005) and the development of parallel projection procedures (Bauschke and Combettes 2011; Li 2020). Another meaningful line is on a comparative study of existing distance repairing approaches for metric and non-metric models (Boytsov and Naidan 2013; Schleif and Tino 2015). Both directions can be of considerable value and interest to researchers and practitioners.

Acknowledgements

The work is supported by Guangdong Basic and Applied Basic Research Foundation (2021A1515011825), Shenzhen Science and Technology Program (CUHKSZWDZC0004), and Shenzhen Research Institute of Big Data Scholarship Program.

References

Bauschke, H. H. 1996. The approximation of fixed points of compositions of nonexpansive mappings in Hilbert space. *Journal of Mathematical Analysis and Applications*, 202(1): 150–159.

Bauschke, H. H.; and Combettes, P. L. 2011. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, volume 408. Springer.

Beck, A.; and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1): 183–202.

Berry, M. W.; Mezher, D.; Philippe, B.; and Sameh, A. 2005. Parallel algorithms for the singular value decomposition. In *Handbook of Parallel Computing and Statistics*, 133–180. Chapman and Hall/CRC.

Boggs, P. T.; and Tolle, J. W. 1995. Sequential quadratic programming. *Acta Numerica*, 4: 1–51.

Boyd, S.; and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Boyle, J. P.; and Dykstra, R. L. 1986. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, 28–47. Springer.

Boytsov, L.; and Naidan, B. 2013. Learning to prune in metric and non-metric spaces. *Advances in Neural Information Processing Systems*, 26.

Bregman, L. M.; Censor, Y.; Reich, S.; and Zepkowitz-Malachi, Y. 2003. Finding the projection of a point onto the intersection of convex sets via projections onto half-spaces. *Journal of Approximation Theory*, 124(2): 194–218.

Brickell, J.; Dhillon, I. S.; Sra, S.; and Tropp, J. A. 2008. The metric nearness problem. *SIAM Journal on Matrix Analysis and Applications*, 30(1): 375–396.

Censor, Y. 2006. Computational acceleration of projection algorithms for the linear best approximation problem. *Linear Algebra and its Applications*, 416(1): 111–123.

Deutsch, F. 2001. *Best Approximation in Inner Product Spaces*, volume 7. Springer.

Duggal, G.; Patro, R.; Sefer, E.; Wang, H.; Filippova, D.; Khuller, S.; and Kingsford, C. 2013. Resolving spatial inconsistencies in chromosome conformation measurements. *Algorithms for Molecular Biology*, 8(1): 1–10.

Fan, C.; Raichek, B.; and Van Buskirk, G. 2018. Metric violation distance: Hardness and approximation. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 196–209. SIAM.

Fletcher, R. 2013. *Practical Methods of Optimization*. John Wiley & Sons.

Gilbert, A. C.; and Jain, L. 2017. If it ain't broke, don't fix it: Sparse metric repair. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 612–619. IEEE.

Gilbert, A. C.; and Sonthalia, R. 2018. Unsupervised metric learning in presence of missing data. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 313–321. IEEE.

Gisbrecht, A.; and Schleif, F.-M. 2015. Metric and non-metric proximity transformations at linear costs. *Neurocomputing*, 167: 643–657.

Halpern, B. 1967. Fixed points of nonexpanding maps. *Bulletin of the American Mathematical Society*, 73(6): 957–961.

Hart, P. E.; Stork, D. G.; and Duda, R. O. 2000. *Pattern Classification*. John Wiley & Sons.

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

Higham, N. J. 2002. Computing the nearest correlation matrix—a problem from finance. *IMA Journal of Numerical Analysis*, 22(3): 329–343.

Jain, A. K.; Murty, M. N.; and Flynn, P. J. 1999. Data clustering: a review. *ACM Computing Surveys*, 31(3): 264–323.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, W. 2015. Estimating Jaccard index with missing observations: a matrix calibration approach. *Advances in Neural Information Processing Systems*, 28.

Li, W. 2020. Scalable Calibration of Affinity Matrices from Incomplete Observations. In *Proceedings of The 12th Asian Conference on Machine Learning*, 753–768. PMLR.

Li, W.; and Yu, F. 2023. Calibrating Distance Metrics Under Uncertainty. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2022*. Springer.

Lions, P.-L. 1977. Approximation de points fixes de contractions. *CR Acad. Sci. Paris Serie, AB*, 284: 1357–1359.

Nedić, A. 2011. Random algorithms for convex minimization problems. *Mathematical Programming*, 129(2): 225–253.

Nesterov, Y. 2018. *Lectures on Convex Optimization*, volume 137. Springer.

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*, 507–516.

Pitis, S.; Chan, H.; Jamali, K.; and Ba, J. 2020. An Inductive Bias for Distances: Neural Nets that Respect the Triangle Inequality. In *8th International Conference on Learning Representations*.

Qi, H.; and Sun, D. 2006. A quadratically convergent Newton method for computing the nearest correlation matrix. *SIAM Journal on Matrix Analysis and Applications*, 28(2): 360–385.

Qi, H.-D.; and Yuan, X. 2014. Computing the nearest Euclidean distance matrix with low embedding dimensions. *Mathematical Programming*, 147(1): 351–389.

Ruggles, C.; Veldt, N.; and Gleich, D. F. 2020. A parallel projection method for metric constrained optimization. In *2020 Proceedings of the SIAM Workshop on Combinatorial Scientific Computing*, 43–53. SIAM.

Schleif, F.-M.; and Tino, P. 2015. Indefinite proximity learning: A review. *Neural Computation*, 27(10): 2039–2096.

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

Schölkopf, B.; Smola, A. J.; Bach, F.; et al. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.

Sonthalia, R.; and Gilbert, A. C. 2020. Project and forget: Solving large-scale metric constrained problems. *arXiv preprint arXiv:2005.03853*.

Veldt, N.; Gleich, D.; Wirth, A.; and Saunderson, J. 2018. A projection method for metric-constrained optimization. *arXiv preprint arXiv:1806.01678*.

Wang, M.; and Bertsekas, D. P. 2015. Incremental constraint projection methods for variational inequalities. *Mathematical Programming*, 150(2): 321–363.

Wells, J. H.; and Williams, L. R. 1975. *Embeddings and Extensions in Analysis*, volume 84. Springer.

Wittmann, R. 1992. Approximation of fixed points of nonexpansive mappings. *Archiv der Mathematik*, 58(5): 486–491.

Xing, E.; Jordan, M.; Russell, S. J.; and Ng, A. 2002. Distance metric learning with application to clustering with side-information. *Advances in Neural Information Processing Systems*, 15.