Euclidean Distance Matrix Optimization for Sensor Network Localization[*]

Joerg Fliege[†] Hou-Duo Qi[‡] and Naihua Xiu[§]

September 24, 2018

Abstract

Sensor Network Localization (SNL) is a general framework that generates a set of embedding points in a low-dimensional space so as to preserve given distance information as much as possible. Typical applications include source localization in two or three dimensional space, molecular conformation in three dimensions, graph embedding and data visualization. There are three main difficulties in solving SNL: (i) low-dimensional embedding that gives rise to non-convexity of the problem, coupled with infinitely many local minima; (ii) a large number of lower and upper bounds for certain distances used to improve the embedding quality; and (iii) non-differentiability of some loss functions used to model SNL. There exist a few promising approaches including co-ordinates minimization and semi-definite programming. This survey mainly focus on a recently established approach: Euclidean Distance Matrix (EDM) Optimization. We will give a short but essential introduction how this approach is theoretically well-developed and demonstrate how EDM optimization nicely handles those difficulties through a few widely used loss functions. We also show how regularization terms can be naturally incorporated into EDM optimization. Numerical examples are used to demonstrate the potential of EDM optimization in tackling large scale problems and effect of regularizations.

AMS subject classifications. 49M45, 90C25, 90C33.

Keywords: Sensor network localization, Euclidean distance matrix, multi-dimensional scaling, stress criterion, conditionally positive semidefinite cone, majorization and minimization.

1 Introduction

The sensor network localization (SNL), rather than being treated as an individual localization problem, will be employed as a general framework to model various types of localization problems under often partially given distance information. The global positioning system (GPS) is one of such problems. The essential information that GPS uses is the distances between the receiver and a few satellites in terms of the amount of times to receive a transmitted signal. The GPS calculates the location of the user by the trilateration method based on the obtained distances. Another example, where GPS does not work, is the indoor facility localization [30]. A

[*]The paper is jointly supported by the Royal Society's International Exchange Scheme IE150919 and the National Natural Science Foundation of China (11728101).

[†]School of Mathematics, University of Southampton, Highfield, Southampton SO17 1BJ, UK. E-mail: J.Fliege@soton.ac.uk.

[‡]School of Mathematics, University of Southampton, Highfield, Southampton SO17 1BJ, UK. E-mail: hdqi@soton.ac.uk.

[§]Department of Applied Mathematics, Beijing Jiaotong University, Beijing, China. E-mail: nhxiu@bjtu.edu.cn.

significantly different localization problem is from the geometric graph embedding [27]. A typical example is to place all nodes of a graph on the surface of the smallest sphere so that the distance of all neighboring nodes are of unit distance. We refer to [26] for many such problems. The SNL framework in fact covers many instances of those problems. In the following, we describe the SNL framework, its four widely used optimization formulations and the computational challenges.

1.1 Sensor network localization

Suppose there are m anchors, denoted as column vectors $\mathbf{x}_i = \mathbf{a}_i \in \mathbb{R}^r$, $i = 1, \ldots, m$, and $(n-m)$ unknown sensors $\mathbf{x}_i \in \mathbb{R}^r$, $i = m+1, \ldots, n$. We are given some (noisy) distances (denoted as δ_{ij}) from sensors to anchors and from sensors to sensors:

$$\left\{ \begin{array}{ll} \delta_{ij} \approx \|\mathbf{x}_j - \mathbf{a}_i\|, & \text{for some } j \in \{m+1, \cdots, n\} \\ \delta_{ij} \approx \|\mathbf{x}_i - \mathbf{x}_j\|, & \text{for some } i, j \in \{m+1, \cdots, n\}, \end{array} \right. \tag{1}$$

where $\|\cdot\|$ is the Euclidean norm. The purpose is to locate those unknown sensors and it is often done by optimizing certain loss functions. The most often used loss function is Kruskal's stress function [25], which is a direct application of the least-squares criterion to (1):

$$\begin{array}{ll} \min & f_1(X) := \sum_{i,j=1}^n w_{ij} \Big(\|\mathbf{x}_i - \mathbf{x}_j\| - \delta_{ij} \Big)^2 \\ \text{s.t.} & \mathbf{x}_i = \mathbf{a}_i, \quad i = 1, \ldots, m, \end{array} \tag{2}$$

where ":=" means "define", $X := [\mathbf{x}_1, \ldots, \mathbf{x}_n]$, and the weights w_{ij} are defined as follows

$$w_{ij} := \left\{ \begin{array}{ll} \text{positive constant} & \text{if } \delta_{ij} \text{ is given} \\ 0 & \text{otherwise.} \end{array} \right.$$

When both $\mathbf{x}_i$ and $\mathbf{x}_j$ are anchors, $\delta_{ij} := \|\mathbf{a}_i - \mathbf{a}_j\|$. We note that $f_1(X)$ is nonconvex and nondifferentiable. If the least-squares criterion is applied to the squared distances, we end up with a differentiable objective known as the squared stress function [8]:

$$\begin{array}{ll} \min & f_2(X) := \sum_{i,j=1}^n w_{ij} \Big(\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \delta_{ij}^2 \Big)^2 \\ \text{s.t.} & \mathbf{x}_i = \mathbf{a}_i, \quad i = 1, \ldots, m. \end{array} \tag{3}$$

It has long been known that the absolute value loss function is more robust than the least squares loss function. Therefore, one can consider the following robust variant of (2) and (3) [9, 20], respectively

$$\begin{array}{ll} \min & f_3(X) := \sum_{i,j=1}^n w_{ij} \Big| \|\mathbf{x}_i - \mathbf{x}_j\| - \delta_{ij} \Big| \\ \text{s.t.} & \mathbf{x}_i = \mathbf{a}_i, \quad i = 1, \ldots, m, \end{array} \tag{4}$$

and

$$\begin{array}{ll} \min & f_2(X) := \sum_{i,j=1}^n w_{ij} \Big| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \delta_{ij}^2 \Big| \\ \text{s.t.} & \mathbf{x}_i = \mathbf{a}_i, \quad i = 1, \ldots, m. \end{array} \tag{5}$$

The question now boils down how to efficiently solve those optimization problems.

1.2 Computational challenges

We explain why those optimization problems are computationally challenging, though some may be more difficult to solve than the others. There are essentially three intrinsic difficulties. The first is the constraint of low-dimensional embedding space $\mathbb{R}^r$. If r is allowed to be no less than $(n-1)$, those problems can be reformulated as convex optimization and hence would become significantly easier to solve. We will see the low-dimensional embedding is equivalent to a rank constraint when the problem is reformulated as Semi-Definite Programming (SDP) or Euclidean Distance Matrix (EDM) optimization, and the rank constraint is notoriously known to be hard to tackle in optimization.

The second challenge is from possibly a large number of lower and upper bounds that some embedding points should obey:

$$\ell_{ij} \le \|\mathbf{x}_i - \mathbf{x}_j\| \le u_{ij}, \tag{6}$$

where $0 \le \ell_{ij} \le u_{ij}$ are given lower and upper bounds and they are known as interval distance constraints in [18]. Such constraints appear in molecular conformation [5] and graph embedding [27], and the number of them could be as high as $O(n^2)$. For example, the number of constraints could be in millions if the number of atoms in molecular conformation is just couple of thousands. Adding those constraints to any of the four optimization problems would make them extremely difficult to solve. The third challenge is the non-dfferentiability of some of the loss functions above. In particular, it is more involved to characterize a stationary point and it is hard to analyse the convergence of corresponding algorithms. This last challenge is also coupled with possibly infinitely many local minima. This can be clearly seen when there are no anchors (i.e., $m = 0$). If $\{\mathbf{x}_i^*\}$ is a local solution, then any of the $\{Qx_i^* + \mathbf{b}\}$ would also be a local solution, where Q is a $r \times r$ rotation matrix and $\mathbf{b} \in \mathbb{R}^r$ is any given (translation) vector.

1.3 Organization

There exist many algorithms each developed for a particular problem of the four above. The purpose of this paper is to introduce a recently developed EDM optimization approach, which is suitable to all of the four problems and always follows a differentiable path regardless of the loss function used. The paper is organized as follows. We give a compact review on relevant research in next section, including the coordinate minimization, widely used SDP approach and more recent EDM optimization. In Sect. 3, we give some essential mathematical background for EDM optimization with particular attention on why EDM optimization has no easy solution by focusing on the EDM cone. In Sect. 4, we review three main algorithms in EDM optimization: method of alternating projections, Newton's method and a penalty method. The former two can be regarded as convex relaxation methods. In Section 5, we study the role of regularization and investigate two types of regularization. We show that each regularization can be easily incorporated into the penalty approach. Numerical results are reported in Sect. 6 and we conclude the paper in Sect. 7.

2 Literature Review

It is not possible to give a comprehensive review here. We only focus on three groups of approaches/algorithms for the reason that they are major representatives and they well reflect the difficulties faced and favorable properties enjoyed by each of the problems in (2)-(5). The first group is the direct approach that solve those problems directly without having to reformulate them to other forms. The second group is the convex relaxation through SDP. The last group is the focus of the paper: EDM optimization.

2.1 The direct approach

By it, we mean any algorithm that solves one of the four problems without reformulating it into another form. That is, $\{\mathbf{x}_i\}$ are its main variables. Let us use the problem (2) to illustrate this approach. One efficient way to solve it is by *majorization and minimization* (MM) scheme. Consider an optimization problem

$$\min \ f(\mathbf{x}), \quad \text{s.t.} \ \ \mathbf{x} \in \mathcal{B},$$

where $f(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}$ is a loss function and $\mathcal{B} \subseteq \mathbb{R}^p$ represents constraints. More often than not, $f(\mathbf{x})$ is difficult to minimize. Suppose we have another function: $f_m(\cdot, \cdot) : \mathbb{R}^p \times \mathbb{R}^p \mapsto \mathbb{R}$ satisfying the following property:

$$f_m(\mathbf{x}, \mathbf{y}) \geq f(\mathbf{x}) \qquad \text{and} \qquad f_m(\mathbf{y}, \mathbf{y}) = f(\mathbf{y}), \quad \forall \ \mathbf{x}, \ \mathbf{y} \in \mathbb{R}^p. \tag{7}$$

In other words, for any fixed point $\mathbf{y} \in \mathbb{R}^p$, the graph of $f_m(\cdot, \mathbf{y})$ is always above that of $f(\cdot)$ and they meet at $\mathbf{x} = \mathbf{y}$. Such function $f_m(\cdot, \cdot)$ is called a majorization of $f(\cdot)$. The main purpose of constructing a majorization is to minimize it instead of minimizing $f(\cdot)$.

Starting from a given point $\mathbf{x}^k \in \mathbb{R}^p$, compute

$$\mathbf{x}^{k+1} = \arg\min \ f_m(\mathbf{x}, \mathbf{x}^k), \quad \text{s.t.} \ \ \mathbf{x} \in \mathcal{B}. \tag{8}$$

We then have

$$f(\mathbf{x}^{k+1}) \overset{(7)}{\leq} f_m(\mathbf{x}^{k+1}, \mathbf{x}^k) \overset{(8)}{\leq} f_m(\mathbf{x}^k, \mathbf{x}^k) \overset{(7)}{=} f(\mathbf{x}^k).$$

Therefore, the MM scheme (7) and (8) generates a sequence $\{\mathbf{x}^k\}$ with non-increasing function values. Consequently, $\{f(\mathbf{x}^k)\}$ will converge provided that $f(\mathbf{x})$ is bounded from below on $\mathcal{B}$. The whole point of (8) is that the majorization function $f(\mathbf{x}, \mathbf{x}^k)$ should be easier to minimize. The rule of thumb is for (8) to have a close form solution. This is exactly what can be achieved for the stress function (2). It follows from the Cauchy-Schwartz inequality

$$\langle \mathbf{x}_i - \mathbf{x}_j, \ \mathbf{y}_i - \mathbf{y}_j \rangle \leq \|\mathbf{x}_i - \mathbf{x}_j\| \|\mathbf{y}_i - \mathbf{y}_j\|$$

that

$$\phi(\mathbf{x}_i, \mathbf{x}_j) := -\|\mathbf{x}_i - \mathbf{x}_j\| \leq \phi_m(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y}_i, \mathbf{y}_j) := \begin{cases} -\frac{\langle \mathbf{x}_i - \mathbf{x}_j, \ \mathbf{y}_i - \mathbf{y}_j \rangle}{\|\mathbf{y}_i - \mathbf{y}_j\|} & \text{if } \mathbf{y}_i \neq \mathbf{y}_j \\ 0 & \text{otherwise.} \end{cases}$$

Therefore,

$$\begin{aligned} f_1(X) &= \sum_{i,j=1}^{n} w_{ij} \Big(\|\mathbf{x}_i - \mathbf{x}_j\| - \delta_{ij} \Big)^2 \\ &\leq \sum_{i,j=1}^{n} \Big(w_{ij} \|\mathbf{x}_i - \mathbf{x}_j\|^2 + 2 w_{ij} \delta_{ij} \phi_m(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y}_i, \mathbf{y}_j) + \delta_{ij}^2 \\ &=: f_1^m(X, Y). \Big) \end{aligned}$$

We note that the majorization function $f_1^m(X, Y)$ is quadratic in X and is easy to minimize when $\mathcal{B} = \mathbb{R}^p$ (unconstrained). The resulting algorithm is the famous SMACOF (Scaling by MAjorizing a COmplicated Function) algorithm [12]. SMACOF is widely used in the field of Multi-Dimensional Scaling (MDS) [8]. A major weakness of SMACOF is when $\mathcal{B}$ is constrained (e.g., $\mathcal{B}$ contains the lower and upper bounds in (6)) because it would not have a closed form solution any more, severely limiting its applications in many other practical problems. It is worth pointing out that the MM scheme is very popular in the direct approach. We will see it also plays a key role in our EDM optimization.

2.2 SDP convex relaxation

As emphasized before, none of the four problems is convex. The main idea is to construct close convex approximation to those nonconvex problems with a belief that the convex optimization would yield a good approximate solution to its true solution. Let us use (3) to illustrate this approach. We define a couple of notation first. Let $\mathcal{S}^n$ be the space of $n \times n$ symmetric matrices. endowed with the standard trace inner product $\langle \cdot, \cdot \rangle$ and the induced Frobenius norm $\| \cdot \|$. Let $\mathcal{S}^n_+$ be the cone of all positive semidefinite matrices in $\mathcal{S}^n$. Note that

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \|\mathbf{x}_i\|^2 + \mathbf{x}_j\|^2 - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle = Y_{ii} + Y_{jj} - 2Y_{ij},$$

where the Gram matrix Y is defined by

$$Y := X^T X = \Big(\langle \mathbf{x}_i,\ \mathbf{x}_j \rangle \Big)_{i,j=1}^{n}.$$

Therefore, the squared stress function can be represented as

$$f_2(X) = \sum_{i,j=1}^{n} w_{ij} \Big(Y_{ii} + Y_{jj} - 2Y_{ij} - \delta_{ij}^2 \Big),$$

which is a linear function of Y. The constraint on Y is that $Y \in \mathcal{S}^n_+$ and it has the rank r (the embedding dimension). This was initially used by Biswas and Ye [6] to derive a SDP for SNL. By dropping the rank constraint, one obtains a convex SDP relaxation for the problem (3). The approach in [6] has been extensively investigated in many subsequent studies, see e.g., [7, 23] and the references therein. A different line of development, but still employing SDP, is [21, 22] where an efficient proximal point algorithm with a semismooth Newton step is developed. When there are many constraints in (6), those SDP-based methods become more expensive. We would also like to point out that the "plain" distance $\|\mathbf{x}_i - \mathbf{x}_j\|$ does not have a straightforward SDP representation, see [29] for such an attempt. Therefore, this approach gets more involved when it tries to approximate the problem (4). We also refer to the thesis [13] for more applications of SDP to Euclidean distance geometry problems.

2.3 The origin of EDM optimization

(a) Classical Euclidean geometry: From distances to embedding points. We explain why EDM optimization is a natural reformulation of the SNL problems. Suppose we have n points $\mathbf{x}_i \in \mathbb{R}^r$, $i = 1, \ldots, n$. We can easily calculate the pairwise (squared) Euclidean distances

$$D_{ij} = d_{ij}^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2, \quad i, j = 1, \ldots, n. \tag{9}$$

Now suppose we only know the matrix D, can we generate a set of points $\{\mathbf{x}_i\}$ that satisfy (9)? There is a classical way to do it.

Consider the double centering matrix

$$B := -\frac{1}{2} JDJ \quad \text{with} \quad J := I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T, \tag{10}$$

where I is the identity matrix in $\mathcal{S}^n$ and $\mathbf{1}_n$ is the column vector of all ones in $\mathbb{R}^n$. The matrix J is known to be the centering matrix. It is known [35, 39, 37] that B is positive semidefinite and $\operatorname{rank}(B) = r$. Hence, it has the following eigenvalue-eigenvector decomposition:

$$B = [\mathbf{p}_1, \ldots, \mathbf{p}_r] \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_r \end{bmatrix} \begin{bmatrix} \mathbf{p}_1^T \\ \vdots \\ \mathbf{p}_r^T \end{bmatrix}, \tag{11}$$

where $\lambda_1 \geq \cdots \geq \lambda_r > 0$ are the positive eigenvalues of B and $\mathbf{p}_i$, $i = 1, \ldots, r$ are the corresponding orthonormal eigenvectors. The embedding points are then given by

$$X := [\mathbf{x}_1, \ldots, \mathbf{x}_n] = [\sqrt{\lambda_1}\mathbf{p}_1, \ \cdots, \ \sqrt{\lambda_r}\mathbf{p}_r]^T, \tag{12}$$

which must satisfy (9).

If we already know that the first m points are anchors as in SNL, we can always find a mapping such that

$$\mathbf{a}_i = Q\mathbf{x}_i + \mathbf{b}, \qquad i = 1, \ldots, m \tag{13}$$

where Q is a rotation matrix and $\mathbf{b}$ is a shifting vector [2, 34]. We then map the rest of the points $\mathbf{x}_i$, $i = m+1, \ldots, n$ to the coordinates system defined by the anchors. This part is known as the Procrustes analysis, mapping one set of points to another as much accurate as possible.

(b) Classical Multi-dimensional scaling (cMDS). The computational procedure from (10) to (12) is fundamental in generating embedding points that preserve the given Euclidean distances. This is based on the assumption that D is a true EDM and has no missing values. However, in practice, the given distances are often noisy and many may be missing. Let Δ represent such given distances. In particular, if the value of δ_{ij} is missing, we simply set $\delta_{ij} = 0$. We can modify the computational procedure in (10)-(12) to generate a set of embedding points $\mathbf{y}_i$, $i = 1, \ldots, n$ such that

$$\|\mathbf{y}_i - \mathbf{y}_j\| \approx \delta_{ij} \qquad \text{if } \ \delta_{ij} > 0. \tag{14}$$

This is done as follows. Let $\overline{\Delta} := (\delta_{ij}^2)$ (the squared dissimilarity matrix). Compute

$$\overline{B} := -\frac{1}{2}J\overline{\Delta}J \tag{15}$$

and decompose it as

$$\overline{B} = [\bar{\mathbf{p}}_1, \ldots, \bar{\mathbf{p}}_n] \begin{bmatrix} \bar{\lambda}_1 & & \\ & \ddots & \\ & & \bar{\lambda}_n \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_1 \\ \vdots \\ \bar{\mathbf{p}}_n \end{bmatrix}, \tag{16}$$

where $\bar{\lambda}_1 \geq \cdots \geq \bar{\lambda}_n$ are the real eigenvalues of $\overline{B}$ (because it is symmetric). Suppose that $\overline{B}$ has s positive eigenvalues. The embedding points $\{\mathbf{y}_i\}$ can be generated by

$$[\mathbf{y}_1, \ldots, \mathbf{y}_n] = [\sqrt{\bar{\lambda}_1}\bar{\mathbf{p}}_1, \ \cdots, \sqrt{\bar{\lambda}_s}\bar{\mathbf{p}}_s]^T. \tag{17}$$

The corresponding Euclidean distance matrix is

$$D^{\texttt{mds}} := \Big(\|\mathbf{y}_i - \mathbf{y}_j\|^2\Big)_{i,j=1}^n.$$

The computational procedure from (15) to (17) is known to be the classical multidimensional scaling (cMDS), originated from the work of Schoenberg [35] and Young and Householder [39], popularized by Torgerson [37] and Gower [19].

A natural question is how good cMDS is. This has been partially answered by Sibson [36] for the case $\delta_{ij} = d_{ij} + \epsilon_{ij}$ with ϵ_{ij} being a small perturbation of the true Euclidean distances d_{ij}. We will not review this analysis and its subsequent development. Instead, we cast cMDS as an optimization problem, which will reveal its sub-optimality. To this purpose, we formally state the definition of the (squared) Euclidean distance matrix.

Definition 2.1 *A matrix $D \in \mathcal{S}^n$ is called a (squared) Euclidean distance matrix (EDM) if there exist a set of points $\mathbf{x}_i \in \mathbb{R}^p$, $i = 1, \ldots, n$ with p being any positive integer, such that*

$$D_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2, \quad i, j = 1, \ldots, n.$$

The smallest such p is called the embedding dimension of D, denoted as r. It is known that $r = \mathrm{rank}(JDJ)$. We let $\mathcal{D}^n$ denote the set of all $n \times n$ EDM D.

We also define $\|A\|_J := \|JAJ\|$ for $A \in \mathcal{S}^n$. Then $\|A\|_J$ is a semi-norm and it is not a true norm because $\|A\|_J = 0$ does not imply $A = 0$. What makes this semi-norm interesting is that $D^{\mathtt{mds}}$ is the optimal solution of the following problem:

$$D^{\mathtt{mds}} = \arg\min \|D - \overline{\Delta}\|_J, \quad \text{s.t.} \quad D \in \mathcal{D}^n.$$

That is, cMDS computes $D^{\mathtt{mds}}$ that is nearest to $\overline{\Delta}$ under the semi-norm $\|\cdot\|_J$ [28]. A more natural measurement of the nearness is under the Frobenius norm, giving rise to the following optimization problem:

$$D^{\mathtt{edm}} = \arg\min \|D - \overline{\Delta}\| \qquad \text{s.t.} \quad D \in \mathcal{D}^n. \tag{18}$$

This is the basic model of EDM optimization. It purely focus on D as its main variable and leaves the aligning the generated embedding points to existing anchors to the Procrustes analysis.

(c) Early research on EDM optimization. The earliest research that had a serious attempt to solve (18) includes Gaffke and Mathar [15] and Glunt et. al. [16]. Both papers treated D as the main variable. [17] has also successfully applied the method of alternating projections in [16] to molecular conformation. The methods in [15] and [16] are of first-order methods. A second-order method (i.e., Newton's method) was later developed by Qi [31] and is proved to enjoy a quadratic convergence rate. Alfakih et. al. [1] took a different route that relies on SDP solvers. It maps $\mathcal{D}^n$ to $\mathcal{S}^{n-1}_+$ through a linear transformation. This approach has been further investigated by Wolkowicz's group in a series of papers, see [24, 14] and the references therein.

Compared to the large number of research papers on convex relaxations on SNL, the EDM optimization has not been well explored, mainly because people believe that SDP is the best vehicle to solve such problems given that the off-shelf SDP solvers are well developed. However, it is widely known that SDP has serious computational bottleneck when the size of the problem is beyond a thousand unless the problem has some special structure, and it only works for some loss functions. We will see in our numerical test, the EDM optimization can solve general SNL problems of size $n = 3000$ in a few minutes (under 3 mintutes) on a standard laptop. Moreover, EDM optimization has a great capability of tackling more complicated problems with all four loss functions. We start by reformulating the four problems in Introduction as EDM optimization below.

2.4 SNL as EDM optimization

Let us recall the data information that we are going to use. We have a dissimilarity matrix $\Delta = (\delta_{ij})$, which contains the given distance measurements among anchors and sensors, and the corresponding weight matrix $W = (w_{ij})$, which indicates the quality of the given δ_{ij}. We require $w_{ij} > 0$ when $\delta_{ij} > 0$ and $w_{ij} = 0$ when $\delta_{ij} = 0$ (missing or not given).

Our main variable is $D \in \mathcal{D}^n$. If there are m anchors $\mathbf{x}_i = \mathbf{a}_i$, $i = 1, \ldots, m$, we may enforce

$$D_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|^2, \quad i, j = 1, \ldots, m. \tag{19}$$

This condition may be fed into the lower and upper bound constraints in (6) by setting

$$\ell_{ij} = u_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|, \quad i, j = 1, \ldots, m.$$

In general, we define

$$\mathcal{B} := \left\{ D \in \mathcal{S}^n \;\middle|\; L_{ij} \le D_{ij} \le U_{ij}, \;\; i, j = 1, \ldots, n \right\},$$

where L_{ij} and U_{ij} are given lower and upper bounds for D_{ij}. For example, $L_{ii} = U_{ii} = 0$ for all $i = 1, \ldots, n$ because $D \in \mathcal{D}^n$ must have zero diagonal. Furthermore, in the case we have m anchors, we may set $L_{ij} = U_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|^2$ for $i, j = 1, \ldots, m$. Then any embedding points of D satisfying (19) can be mapped to the anchors $\mathbf{a}_i$ through the linear mapping (13). Therefore, the anchor information is completely encoded to the box constraint $\mathcal{B}$. In this way, the problem (2) can be reformulated as EDM optimization:

$$\begin{array}{ll} \min & f^{(d,2)}(D) := \frac{1}{2}\|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 \\ \text{s.t.} & D \in \mathcal{D}^n, \quad D \in \mathcal{B}, \quad \text{and} \;\; \operatorname{rank}(JDJ) \le r, \end{array} \tag{20}$$

where $\sqrt{W}$ is the elementwise square root (hence $\sqrt{D_{ij}} = d_{ij}$), and $\circ$ is the elementwise multiplication known as the Hadamard product. It is worth pointing out that the objective function is convex in D because

$$f^{(d,2)}(D) = \frac{1}{2}\langle W,\, D\rangle - \langle W \circ \Delta,\, \sqrt{D}\rangle + \frac{1}{2}\langle W,\, \overline{\Delta}\rangle$$

Note that the first term above is linear in D, the second term is negative of a square root term and hence is convex, and the third term is constant. Since the rank of J is $(n-1)$, therefore, $\operatorname{rank}(JDJ) \le n-1$. If the embedding dimension r is allowed to be no less than $(n-1)$, then the problem (20) is convex in D because $\mathcal{D}^n$ is a convex cone, which will be explained later.

Similarly, the other three problems can also be put into EDM optimization. We list them below for easy reference.

$$\begin{array}{ll} \min & f^{(D,2)}(D) := \frac{1}{2}\|\sqrt{W} \circ (D - \Delta)\|^2 \\ \text{s.t.} & D \in \mathcal{D}^n, \quad D \in \mathcal{B}, \quad \text{and} \;\; \operatorname{rank}(JDJ) \le r, \end{array} \tag{21}$$

$$\begin{array}{ll} \min & f^{(d,1)}(D) := \frac{1}{2}\|\sqrt{W} \circ (\sqrt{D} - \Delta)\|_1 \\ \text{s.t.} & D \in \mathcal{D}^n, \quad D \in \mathcal{B}, \quad \text{and} \;\; \operatorname{rank}(JDJ) \le r, \end{array} \tag{22}$$

where $\|A\|_1 := \sum_{i,j=1}^n |A_{ij}|$ is the ℓ_1 norm in $\mathcal{S}^n$. The last EDM optimization corresponding to (5) is

$$\begin{array}{ll} \min & f^{(D,1)}(D) := \frac{1}{2}\|\sqrt{W} \circ (D - \Delta)\|_1 \\ \text{s.t.} & D \in \mathcal{D}^n, \quad D \in \mathcal{B}, \quad \text{and} \;\; \operatorname{rank}(JDJ) \le r. \end{array} \tag{23}$$

In the above notation of defining the objective functions, we used D to mean that the "squared" distance is used in the objective and d meaning the "plain" distance is used. For example, $f^{(D,2)}$ means that the squared distance is used with the least squares, while $f^{(d,1)}$ is that the "plain" distance is used with the least absolute-values (i.e., ℓ_1 norm). The remaining task is to show how those EDM optimization can be efficiently solved even when n is large. Obviously, any efficient algorithm would heavily depend on how efficiently we can handle the essential constraints in EDM optimization: the rank constraint and the cone constraint on $\mathcal{D}^n$. We explain below that they can be dealt with through an efficient computational manner.

3 EDM Optimization: Mathematical Background

Let us set up the minimal requirements for what consist of EDM optimization. It usually takes the following form:

$$\min\ f(D), \qquad \text{s.t.}\ \ D \in \mathcal{D}^n,\ \ D \in \mathcal{B},\ \ \text{rank}(JDJ) \le r, \tag{24}$$

where $f : \mathcal{S}^n \mapsto \mathbb{R}$. The three constraints are essential in the sense that most of practical problems would require their embedding points to satisfy them. Of course, other variables and constraints are permitted to add to the basic model (24). For example, the spherical constraints studied in [4] may be considered. But such extra features are not our concern here.

The two most difficult parts in solving EDM optimization are the constraint on $\mathcal{D}^n$ (conic constraint) and the rank constraint on $\text{rank}(JDJ) \le r$. In this part, we will review their mathematical properties, essential for our computation.

3.1 EDM cone $\mathcal{D}^n$

The first and also an important issue that we should address is the membership of $\mathcal{D}^n$ because we must need to know if a candidate matrix belongs to it or not in order to solve the EDM optimization efficiently. One answer to this question is to see if we can calculate the orthogonal projection $\Pi_{\mathcal{D}^n}(A)$ defined by

$$\Pi_{\mathcal{D}^n}(A) := \arg\min\ \|A - D\| \qquad \text{s.t.} \quad D \in \mathcal{D}^n. \tag{25}$$

We note that $\Pi_{\mathcal{D}^n}(A)$ is well defined because $\mathcal{D}^n$ is a close convex cone, which is a direct consequence of its characterization below due to Schoenberg [35]:

$$D \in \mathcal{D}^n \quad \Longleftrightarrow \quad \text{diag}(D) = 0 \ \text{ and } \ -D \in \mathcal{K}^n_+, \tag{26}$$

where $\mathcal{K}^n_+$ is the conditionally positive semidefinite cone defined as follows.

We recall that the semidefinite cone $\mathcal{S}^n_+$ can be defined as

$$\mathcal{S}^n_+ = \left\{ A \in \mathcal{S}^n \ \middle|\ \mathbf{v}^T A \mathbf{v} \ge 0 \quad \forall\ \mathbf{v} \in \mathbb{R}^n \right\}.$$

If we restrict $\mathbf{v}$ to the subspace $\mathbf{1}_n^\perp := \left\{ \mathbf{v} \in \mathbb{R}^n \mid \mathbf{v}^T \mathbf{1} = 0 \right\}$, then we have $\mathcal{K}^n_+$:

$$\mathcal{K}^n_+ := \left\{ A \in \mathcal{S}^n \ \middle|\ \mathbf{v}^T A \mathbf{v} \ge 0 \quad \forall\ \mathbf{v} \in \mathbf{1}_n^\perp \right\}.$$

Since the centering matrix J is the projection matrix onto $\mathbf{1}_n^\perp$, it is equivalent to say

$$\mathcal{K}^n_+ = \left\{ A \in \mathcal{S}^n \ \middle|\ JAJ \in \mathcal{S}^n_+ \right\}. \tag{27}$$

It is easy to see that $\mathcal{K}^n_+$ is closed and convex, so is $\mathcal{D}^n$ by (26). It is worth noting that $\mathcal{S}^n_+ \subset \mathcal{K}^n_+$ and $\mathcal{K}^n_+$ is bigger than $\mathcal{S}^n_+$. Moreover, $\mathcal{K}^n_+$ is not self-dual in the sense that

$$\left(\mathcal{K}^n_+\right)^* \subset \mathcal{K}^n_+ \qquad \text{and} \qquad \left(\mathcal{K}^n_+\right)^* \ne \mathcal{K}^n_+,$$

where for a cone $\mathcal{C}$ in $\mathcal{S}^n$, its dual cone $\mathcal{C}^*$ is defined by

$$\mathcal{C}^* := \left\{ Y \in \mathcal{S}^n \mid \langle Y,\ X \rangle \ge 0 \ \ \forall\ X \in \mathcal{C} \right\}.$$

We note that $\mathcal{S}^n_+$ is self-dual. Hence EDM optimization is significantly different from SDP.

What does $\mathcal{D}^n$ look like? Let us use $\mathcal{D}^3$ as an example to understand its (computational) complexity.

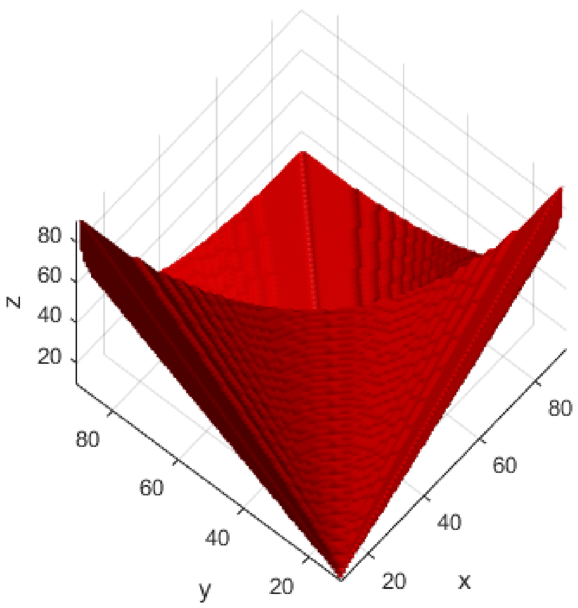


Figure 1: Shape of $\mathcal{D}^3$

Example 3.1 *(Projection onto $\mathcal{D}^3$) It follows from (26) that*

$$\mathcal{D}^3 = \left\{ D = \begin{bmatrix} 0 & x & y \\ x & 0 & z \\ y & z & 0 \end{bmatrix} \;\middle|\; \begin{array}{l} 2(x+y) - z \geq 0 \\ 2(x+z) - y \geq 0 \\ 2(y+z) - x \geq 0 \end{array} \quad \textit{and} \quad x^2 + y^2 + z^2 \leq \frac{1}{2}(x+y+z)^2 \right\}.$$

For a given matrix $A \in \mathcal{S}^3$ specified by (a, b, c). Its projection to $\mathcal{D}^3$ is

$$\Pi_{\mathcal{D}^3}(A) = \arg\min \left\{ \|(a,b,c) - (x,y,z)\| \;\middle|\; \begin{array}{l} 2(x+y) - z \geq 0 \\ 2(x+z) - y \geq 0 \\ 2(y+z) - x \geq 0 \end{array} \quad \textit{and} \quad \begin{array}{l} \|(x,y,z)\| \leq \frac{1}{2}t \\ t = x + y + z \end{array} \right\}.$$

The constraint $\|(x,y,z)\| \leq 0.5t$ is a constraint of second-order cone. Hence $\Pi_{\mathcal{D}^3}(A)$ is a quadratic second order cone programming and hence in general would need an iterative algorithm for its solution. The shape of $\mathcal{D}^3$ can be seen in Fig. 3.1.

Although $\Pi_{\mathcal{D}^3}$ is difficult to compute, it is surprisingly easy to compute the projection onto $\mathcal{K}_+^n$ [15]:

$$\Pi_{\mathcal{K}_+^n}(A) = A - \Pi_{\mathcal{S}_+^n}(JAJ). \tag{28}$$

This projection formula is fundamental to the alternating projection methods in [15, 16] and the Newton method in [31].

Another attempt to answer the question of the membership of $\mathcal{D}^n$ is by the *triangle inequality test.* For a given pre-distance matrix D satisfying the condition $\text{diag}(D) = 0$ and $D_{ij} \geq 0$. We test all the triangle inequalities:

$$\sqrt{D_{ij}} + \sqrt{D_{jk}} \geq \sqrt{D_{ik}} \qquad \text{for all triple } (i, j, k).$$

The following example shows that the triangle inequality test is not enough to verify $D \in \mathcal{D}^n$:

$$D = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 4 & 4 \\ 1 & 4 & 0 & 4 \\ 1 & 4 & 4 & 0 \end{bmatrix}.$$

For this example, all triangle inequalities hold. However, it is not Euclidean because the matrix $(-JDJ)$ has a negative eigenvalue (-0.5), which implies $(-D) \notin \mathcal{K}^n_+$ and hence not in $\mathcal{D}^n$ due to (26).

3.2 Two representations of the rank constraint

The second issue we should address is about the rank constraint. We describe two representations that will lead to two different types of algorithms for EDM optimization.

(a) Representation by eigenvalue functions. Recall that the fact $(-D) \in \mathcal{K}^n_+$ means $(-JDJ) \in \mathcal{S}^n_+$. This implies

$$\mathrm{rank}(JDJ) \le r \qquad \Longleftrightarrow \qquad \sum_{i,j=1}^{n} \lambda_i = \sum_{i,j=1}^{r} \lambda_i, \tag{29}$$

where $\lambda_1 \ge \ldots \ge \lambda_n \ge 0$ are the eigenvalues of $(-JDJ)$ in non-increasing order. Define the eigenvalue function:

$$p(D) := \sum_{i,j=1}^{n} \lambda_i - \sum_{i,j=1}^{r} \lambda_i = \mathrm{trace}(-JDJ) - \sum_{i,j=1}^{r} \lambda_i \ge 0, \quad \forall \ -D \in \mathcal{K}^n_+.$$

It is known that the sum of largest eigenvalues is a convex function. Therefore, $p(D)$ is a difference of two convex functions and it is so called DC function. The characterization in (26) and the equivalence in (29) translate to

$$\{D \in \mathcal{D}^n \text{ and } \mathrm{rank}(JDJ) \le r\} \quad \Longleftrightarrow \quad \left\{\mathrm{diag}(D) = 0,\ -D \in \mathcal{K}^n_+,\ \text{and } p(D) = 0\ \right\}. \tag{30}$$

(b) Representation by distance functions. The essential constraints in EDM optimization can be grouped as follows.

$$\{D \in \mathcal{D}^n \ \text{ and } \ \mathrm{rank}(JDJ) \le r\} \quad \overset{(26)}{\Longleftrightarrow} \quad \left\{\mathrm{diag}(D) = 0 \ \text{ and } \ -D \in \mathcal{K}^n_+(r)\right\}$$

where $\mathcal{K}^n_+(r)$ is the rank-r cut of $\mathcal{K}^n_+$:

$$\mathcal{K}^n_+(r) := \mathcal{K}^n_+ \cap \{A \in \mathcal{S}^n \mid \mathrm{rank}(JAJ) \le r\}.$$

An important point to make here is that it is quite computational inexpensive to check the membership of $\mathcal{K}^n_+(r)$ as we see below.

Let us define the Euclidean distance from a given point $A \in \mathcal{S}^n$ to $\mathcal{K}^n_+(r)$ by

$$\mathrm{dist}(A,\ \mathcal{K}^n_+(r)) := \min\left\{\|A - Y\| \mid Y \in \mathcal{K}^n_+(r)\right\}$$

and define the function $g(\cdot) : \mathcal{S}^n \mapsto \mathbb{R}$ to be the squared distance function from $(-A)$ to $\mathcal{K}^n_+(r)$:

$$g(A) := \frac{1}{2}\mathrm{dist}^2(-A,\ \mathcal{K}^n_+(r)). \tag{31}$$

It can be seen that

$$-D \in \mathcal{K}_+^n(r) \qquad \Longleftrightarrow \qquad g(D) = 0. \tag{32}$$

We further define $h(\cdot) : \mathcal{S}^n \mapsto \mathrm{I\!R}$ by

$$h(A) := \frac{1}{2}\|A\|^2 - g(-A). \tag{33}$$

It is proved in [33, Prop. 3.4] that $h(\cdot)$ is convex and

$$h(A) = \frac{1}{2}\|\Pi_{\mathcal{K}_+^n(r)}(A)\|^2 \qquad \text{and} \qquad \Pi_{\mathcal{K}_+^n(r)}(A) \in \partial h(A), \tag{34}$$

where $\partial h(A)$ is the subdifferential of $h(\cdot)$ at A and $\Pi_{\mathcal{K}_+^n(r)}(A)$ is an nearest point in $\mathcal{K}_+^n(r)$ from A:

$$\Pi_{\mathcal{K}_+^n(r)}(A) \in \arg\min \ \|A - Y\|, \quad \text{s.t. } \ Y \in \mathcal{K}_+^n(r).$$

We note that there might be many nearest points since $\mathcal{K}_+^n(r)$ is not convex. The characterizations in (26) and (32) translate to

$$\{D \in \mathcal{D}^n \text{ and } \operatorname{rank}(JDJ) \le r\} \quad \Longleftrightarrow \quad \{\operatorname{diag}(D) = 0 \ \text{ and } \ g(D) = 0 \ \}. \tag{35}$$

(c) Efficient computation. Both the representations in (30) and (35) reply on how to compute the function $p(D)$ and $g(D)$. We show that they can be both computed efficiently.

Suppose $A \in \mathcal{S}^n$ has the following spectral decomposition:

$$A = \lambda_1 \mathbf{p}_1\mathbf{p}_1^T + \lambda_2 \mathbf{p}_2\mathbf{p}_2^T + \cdots + \lambda_n \mathbf{p}_n\mathbf{p}_n^T,$$

where $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_n$ are the eigenvalues of A in non-increasing order, and $\mathbf{p}_i$, $i = 1, \ldots, n$ are the corresponding orthonormal eigenvectors. We define a PCA-style matrix truncated at r:

$$\mathrm{PCA}_r^+(A) := \sum_{i=1}^{r} \max\{0, \lambda_i\}\mathbf{p}_i\mathbf{p}_i^T.$$

It is proved in [33, Eq. (22), Prop. 3.3] that one particular $\Pi_{\mathcal{K}_+^n(r)}(A)$ can be computed through

$$\Pi_{\mathcal{K}_+^n(r)}(A) = \mathrm{PCA}_r^+(JAJ) + (A - JAJ).$$

Therefore, both $p(D)$ and $g(D)$ can be computed after we obtained $\mathrm{PCA}_r^+(-JDJ)$, which can be obtained by computing just first r largest eigenvalues of $(-JDJ)$ and their corresponding eigenvectors (e.g., via `eigs.m` in Matlab). Please refer to [40, Sect. II] for more detail.

4 EDM Optimization: Algorithms

We recall that the EDM optimization is essentially nonconvex mainly due to the rank constraint and possible non-convex loss function. We still review some important convex relaxation methods because of two reasons: (i) They are globally convergent; and (ii) they may act as a subproblem solver in nonconvex approaches. However, our main focus will be on a recently proposed nonconvex approach for its easy implementation and low-computational cost.

4.1 Convex relaxation methods

If we drop the rank constraint from the least squares models (20) and (21), we will get a convex problem. Let us use (21) as an example. We start with its basic form (by dropping the rank constraint and letting all weights to be 1 in (21)):

$$\min \frac{1}{2}\|D - \overline{\Delta}\|^2, \qquad \text{s.t.} \quad D \in \mathcal{S}^n_h \ \text{ and } \ -D \in \mathcal{K}^n_+, \tag{36}$$

where $\mathcal{S}^n_h$ is the hollow subspace

$$\mathcal{S}^n_h := \{A \in \mathcal{S}^n \mid \operatorname{diag}(A) = 0\}.$$

(a) Method of Alternating Projections (MAP). It is easy to see that (36) is a projection problem onto $\mathcal{S}^n_h \cap \mathcal{K}^n_-$, the intersection of the subspace $\mathcal{S}^n_h$ and the convex cone $\mathcal{K}^n_- := -\mathcal{K}^n_+$. The MAP was developed in [15] and [16]: Start with an initial point $D^0 \in \mathcal{S}^n$, compute

$$D^{k+1} = \Pi_{\mathcal{K}^n_-}\Big(\Pi_{\mathcal{S}^n_h}(D^k)\Big), \qquad k = 0, 1, 2 \ldots, . \tag{37}$$

The two projections are easy to compute. In fact,

$$\Pi_{\mathcal{S}^n_h}(A) = A_0, \qquad \forall \ A \in \mathcal{S}^n,$$

where A_0 is A except its diagonal is being replaced by 0, and by (28)

$$\Pi_{\mathcal{K}^n_-}(A) = -\Pi_{\mathcal{K}^n_+}(-A) = A + \Pi_{\mathcal{S}^n_+}(-JAJ).$$

MAP can be extended to deal with the weighted problem:

$$\min \ f(D) = \frac{1}{2}\|\sqrt{W} \circ (D - \overline{\Delta})\|^2, \quad \text{s.t.} \quad D \in \mathcal{S}^n_h \cap \mathcal{K}^n_-. \tag{38}$$

Let w_i be the largest element in the i row of $\sqrt{W}$:

$$w_i := \max\left\{\sqrt{W_{ij}} \mid j = 1, \ldots, n\right\}$$

and $\mathbf{w} := (w_1, \ldots, w_n)^T$. Let $\operatorname{Diag}(\mathbf{w})$ be the diagonal matrix formed by $\mathbf{w}$. A majorization function of $f(D)$ is

$$f_m(D, D^k) := f(D^k) + \nabla f(D^k), \ D - D^k\rangle + \frac{1}{2}\|\operatorname{Diag}(\sqrt{\mathbf{w}})(D - \overline{\Delta})\operatorname{Diag}(\sqrt{\mathbf{w}}))\|^2,$$

where D^k is the current iterate. Hence, the MM (majorization and minimization) scheme can be applied to $f_m(D, D^k)$ to get

$$D^{k+1} = \arg\min \ f_m(D, D^k), \quad \text{s.t.} \quad D \in \mathcal{S}^n_h \cap \mathcal{K}^n_-,$$

which is known as the diagonally weighted projection problem and there exists a computational formula for the weighted projection onto $\mathcal{K}^n_-$ (see [31, Sect. 4.2]).

MAP is guaranteed to converge. However, it is incapable of dealing with the rank constraint. It has to reply on other mechanism to get the embedding dimension right, for example, by enforcing many box constraints (6), see [17] for such a mechanism.

(b) Newton's method. MAP is a first-order method and hence may take many iterations to terminate. Newton's method was developed in [31] and it is designed for the Lagrange dual problem of (36):

$$\min_{\mathbf{y}\in\mathbb{R}^n} \ \theta(\mathbf{y}) := \frac{1}{2}\|\Pi_{\mathcal{K}^n_+}(\overline{\Delta} + \mathrm{Diag}(\mathbf{y}))\|^2.$$

Being a Lagrange dual function, $\theta(\mathbf{y})$ is convex and is differentiable in this case. The optimality condition is

$$F(\mathbf{y}) := \nabla\theta(\mathbf{y}) = \mathrm{diag}\Big(\Pi_{\mathcal{K}^n_+}(\overline{\Delta} + \mathrm{Diag}(\mathbf{y}))\Big) = 0.$$

This is a system of nonlinear equations. Newton's method hence takes the following form:

$$\mathbf{y}^{k+1} = \mathbf{y}^k - V_k^{-1}F(\mathbf{y}^k), \quad k = 0, 1, \ldots,$$

where V_k is a generalized Jacobian of $F(\cdot)$ at $\mathbf{y}^k$. Suppose $\{\mathbf{y}^k\}$ converges to $\mathbf{y}^*$ satisfying $F(\mathbf{y}^*) = 0$ ($\mathbf{y}^*$ must be optimal for the dual problem), then the corresponding optimal EDM is

$$D^* = -\Pi_{\mathcal{K}^n_+}(\overline{\Delta} + \mathrm{Diag}(\mathbf{y}^*)).$$

The Newton method is proved to converge quadratically and can be further generalized to deal with the weighted problem (38). We refer to [31] for detailed analysis of this method, which is further used to solve the subproblems for the rank constrained problem (21) in [33].

4.2 Penalty methods

If we take into account of the rank representations in (30) and (35), the penalty method comes naturally as a promising approach to EDM optimization. We describe them separately, with a main focus on penalizing (35).

(a) Penalizing the eigen-function. Take the problem (21) for example. By using (30), it is equivalent to

$$\begin{array}{ll} \min & f^{(D,2)}(D) = \|\sqrt{W} \circ (D - \overline{\Delta})\|^2 \\ \text{s.t.} & D \in \mathcal{B}, \ -D \in \mathcal{K}^n_+, \ \ p(D) = 0, \end{array}$$

where the diagonal constraint $\mathrm{diag}(D) = 0$ is submerged to the box constraint $\mathcal{B}$. We emphasize that all the constraints except $p(D) = 0$ are convex. The penalty problem is

$$\begin{array}{ll} \min & f_\rho^{(D,2)}(D) = \|\sqrt{W} \circ (D - \overline{\Delta})\|^2 + \rho p(D) \\ \text{s.t.} & D \in \mathcal{B}, \ -D \in \mathcal{K}^n_+, \end{array}$$

where $\rho > 0$ is a penalty parameter. We note that $p(D) \geq 0$ whenever $-D \in \mathcal{K}^n_+$. Therefore, it is a penalty function over the feasible region. This penalty problem was thoroughly investigated in [33], where the Newton method reviewed above was used to solve its subproblems. The corresponding analysis is quite involved and it is not possible here to give it a complete review. Instead, we focus on the other penalty method due to (35) for its easy implementation and low computational cost each step.

(b) Penalizing the distance function. To illustrate its power, we use the nondifferentiable problem (20) as an example. We note that this approach can also be used to other EDM optimization problems. By (35), the problem (20) is equivalent to

$$\begin{array}{ll} \min & f_\rho^{(d,2)}(D) = \|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 \\ \text{s.t.} & D \in \mathcal{B}, \ \ g(D) = 0. \end{array}$$

The penalty problem is

$$\min \ F(D) := \|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 + \rho g(D), \qquad \text{s.t.} \quad D \in \mathcal{B}. \tag{39}$$

The function $F(\cdot)$ is still difficult to minimize. We will construct a majorization that is easy to minimize.

Suppose D^k is the current iterate. We recall from (33) that

$$\begin{aligned} g(D) &= \frac{1}{2}\|D\|^2 - h(-D) \\ &\le \frac{1}{2}\|D\|^2 - \underbrace{\Big\{ h(-D^k) + \langle \Pi_{\mathcal{K}^n_+(r)}(-D^k),\ -D + D^k\rangle \Big\}}_{\text{by convexity of } h(\cdot) \text{ and (34)}} \\ &= \frac{1}{2}\|D\|^2 - h(-D^k) + \langle \Pi_{\mathcal{K}^n_+(r)}(-D^k),\ D - D^k\rangle \\ &=: g_m(D, D^k). \end{aligned}$$

That is, $g_m(D, D^k)$ is a majorization of $g(D)$ (by verifying the relationship in (7)). Consequently, a majorization for $F(D)$ is

$$F_m(D, D^k) = \|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 + \rho g_m(D, D^k).$$

By applying the MM scheme (8), we arrive at the following algorithm:

$$D^{k+1} = \arg\min \ F_m(D, D^k), \qquad \text{s.t.} \qquad D \in \mathcal{B}. \tag{40}$$

We see below that the majorization problem has a very nice structure.

$$\begin{aligned} D^{k+1} &= \arg\min_{D\in\mathcal{B}} \ \|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 + \rho/2\|D\|^2 + \langle \rho\Pi_{\mathcal{K}^n_+(r)}(-D^k),\ D\rangle \\ &= \arg\min_{D\in\mathcal{B}} \ \frac{\rho}{2}\|D\|^2 + \langle W + \rho\Pi_{\mathcal{K}^n_+(r)}(-D^k),\ D\rangle - 2\langle W \circ \Delta,\ \sqrt{D}\rangle \\ &= \arg\min_{D\in\mathcal{B}} \ \frac{1}{2}\|D - D^k_\rho\|^2 - \frac{2}{\rho}\langle W \circ \Delta,\ \sqrt{D}\rangle, \end{aligned}$$

where

$$D^k_\rho := -W/\rho - \Pi_{\mathcal{K}^n_+(r)}(-D^k).$$

Hence, D^{k+1} can be computed through the following $n(n-1)/2$ one-dimensional optimization problems:

$$D^{k+1}_{ij} = \arg\min \ \frac{1}{2}\Big(D_{ij} - (D^k_\rho)_{ij}\Big)^2 - 2(w_{ij}\delta_{ij}/\rho)\sqrt{D_{ij}}, \qquad \text{s.t.} \qquad L_{ij} \le D_{ij} \le U_{ij}. \tag{41}$$

This is equivalent to solving the following one-dimensional problem. For given $\omega \in \mathbb{R}$, $\alpha > 0$ and an interval $B := [a, b]$ with $0 \le a \le b$, find the optimal solution $\mathcal{S}_B(\omega, \alpha)$ of the problem:

$$\mathcal{S}_B(\omega, \alpha) := \arg\min \frac{1}{2}(x - \omega)^2 - 2\alpha\sqrt{x}, \qquad \text{s.t.} \quad x \in B = [a, b], \tag{42}$$

Define

$$u := \frac{\alpha}{2}, \quad v := \frac{\omega}{3}, \quad \text{and} \quad \tau := u^2 - v^3.$$

Let

$$\mathcal{S}(\omega, \alpha) := \begin{cases} \left[(u + \sqrt{\tau})^{1/3} + (u - \sqrt{\tau})^{1/3}\right]^2 & \text{if } \tau \ge 0 \\ 4v\cos^2(\phi/3) & \text{if } \tau < 0, \end{cases}$$

where the angle ϕ is defined by $\cos(\phi) = u/v^{3/2}$.

The following results are proved in [40, Prop. 3.4, Prop. 3.5]:

Proposition 4.1 *Suppose we are given $\omega \in \mathbb{R}$, $\alpha > 0$ and $B = [a, b]$ with $0 \le a \le b$. Then we have*

(i) the solution of (42) is given by

$$\mathcal{S}_B(\omega, \alpha) = \min\Big\{ b,\ \max\{a,\ \mathcal{S}(\omega, \alpha)\}\Big\}.$$

(ii) Suppose $b > 0$ and we have two positive constants $\omega_{\max} > 0$ and $\alpha_0 > 0$. Then there exists $\gamma > 0$ such that

$$\mathcal{S}_B(\omega, \alpha) \ge \gamma$$

for any ω and α satisfying

$$|\omega| \le \omega_{\max} \quad \text{and} \qquad \alpha \ge \alpha_0,$$

Therefore, D^{k+1} in (41) can be computed as follows

$$D^{k+1}_{ij} = \left\{ \begin{array}{ll} \min\{U_{ij},\ \max\{(D^k_\rho)_{ij},\ L_{ij}\} & \text{if } \delta_{ij} = 0 \\ \mathcal{S}_{\mathcal{B}_{ij}}((D^k_\rho)_{ij},\ w_{ij}\delta_{ij}/\rho) & \text{if } \delta_{ij} > 0, \end{array} \right. \tag{43}$$

where $\mathcal{B}_{ij} := [L_{ij}, U_{ij}]$. For simplicity, we denote the update formula in (43) by

$$D^{k+1} = \mathcal{T}(D^k), \quad k = 0, 1, \ldots, . \tag{44}$$

The algorithm just outlined is called `SQREDM` (Square-Root Model via EDM for (20)) in [40]. We summarize it below.

Algorithm 1 SQREDM Method

```
1: Input data: Dissimilarity matrix $\Delta$, weight matrix $W$, penalty parameter $\rho > 0$, lower-bound matrix $L$, upper-bound matrix $U$ and the initial $D^0$. Set $k := 0$.
2: Update: Compute $D^{k+1}$ by (44) via (43).
3: Convergence check: Set $k := k + 1$ and go to Step 2 until convergence.
```

We make a few remarks on the algorithm `SQREDM`.

(R1) (Differentiable path) Without loss of generality, we may assume the box constraint $\mathcal{B}$ is bounded. Then the generated sequence $\{D^k\}$ is also bounded because $D^k \in \mathcal{B}$. Moreover

$$\|D^k_\rho\| = \|W/\rho + \Pi_{\mathcal{K}^n_+(r)}(-D^k)\| \le \frac{1}{\rho}\|W\| + \|\Pi_{\mathcal{K}^n_+(r)}(-D^k)\| \le \frac{1}{\rho}\|W\| + \|D^k\|,$$

where the last inequality used the fact that $0 \in \mathcal{K}^n_+(r)$. The boundedness of $\{D^k\}$ implies the boundedness of $\{D^k_\rho\}$. It follows from Prop. 4.1(ii) that there exists $\gamma > 0$ such that $D^k_{ij} \ge \gamma$ whenever $\delta_{ij} > 0$. If $\delta_{ij} = 0$, then $w_{ij} = 0$ and the term D_{ij} does not enter the objective function $F(D)$. Consequently, $F(D)$ is differentiable on the generated iterates $\{D^k\}$ although it may not be differentiable at other places. Since D^k_{ij} is bounded away from 0 by a constant $\gamma > 0$ (when $\delta_{ij} > 0$), $F(D)$ is also differentiable at any of the accumulation points of $\{D^k\}$. The property of being differentiable along the generated path has greatly simplified the convergence analysis in [40].

(R2) (Computational complexity) The major computation in `SQREDM` is on $\Pi_{\mathcal{K}^n_+(r)}(-D^k)$, which has been dealt with in Sect. 3(c). The overall complexity is $O(Nrn^2)$, where $O(rn^2)$ is from computing $\Pi_{\mathcal{K}^n_+(r)}(-D^k)$ and the formula in (43), and N is the number of updates (number of iterations).

(R3) (Generalization) The convergence of `SQREDM` has been established in [40]. However, its framework can be generalized to other EDM optimization problems in principle, though needing technical adaptation, see [41] for its generalization to the robust model (22).

5 Regularization

Regularization in SNL seems to have been motivated by a similar strategy in dimension reduction in manifold learning [38], where data sitting on a manifold in a high dimensional space are mapped to a low dimensional Euclidean space. Flattening a manifold to a low dimensional space often causes points to be gathered around the geometric center of those points. The idea was to pull apart the embedding points via enforcing some regularization terms. This idea of regularization has not been well addressed in EDM optimization. We show below it can be done without causing too much extra computational burden as least to the algorithm `SQREDM`.

Suppose we have n points that we would like to push them apart. We introduce two ways to achieve this purpose. One is to maximize total Euclidean distances among points

$$\mathcal{R}_1(D) := \sum_{i,j=1}^{n} \|\mathbf{x}_i - \mathbf{x}_j\| = \langle \mathbf{1}_n\mathbf{1}_n^T,\ \sqrt{D}\rangle$$

and the other is to maximize the total squared Euclidean distance suggested by [7]

$$\mathcal{R}_2(D) := \sum_{i,j=1}^{n} \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \langle \mathbf{1}_n\mathbf{1}_n^T,\ D\rangle,$$

where $\mathbf{x}_i,\ i = 1,\ldots,n$ are the embedding points of the EDM D. In particular, $\mathcal{R}_2$ has an interesting interpretation. Suppose the embedding points $\{\mathbf{x}_i\}$ are centred, i.e., $\sum_{i=1}^n \mathbf{x}_i = 0$ (this is so when they are embedding points form D by MDS). Then

$$\mathcal{R}_2(D) = \sum_{i,j=1}^{n} \Big(\|\mathbf{x}_i\|^2 - 2\langle \mathbf{x}_i,\ \mathbf{x}_j\rangle + \|\mathbf{x}_j\|^2 \Big) = 2n \sum_{i=1}^{n} \|\mathbf{x}_i\|^2,$$

which is called the variance among the embedding points and maximizing $\mathcal{R}_2$ is known as maximum variance unfolding in [38].

Adding each term to the penalized problem (39) yields

$$\min\ F^{(i)}(D) := \|\sqrt{W} \circ (\sqrt{D} - \Delta)\|^2 + \rho g(D) - \beta \mathcal{R}_i(D), \qquad \text{s.t.} \quad D \in \mathcal{B}, \tag{45}$$

where $i \in \{1, 2\}$ and $\beta > 0$ is the regularization parameter. Again, we can use MM scheme to solve this regularized problem. A natural majorization function is

$$F^{(i)}_m(D, D^k) = F_m(D, D^k) + \beta \mathcal{R}_i(D).$$

We solve

$$D^{k+1} = \arg\min\ F^{(i)}_m(D, D^k) \qquad \text{s.t.} \qquad D \in \mathcal{B}.$$

When $i = 1$, it is easy to see

$$\begin{aligned} D^{k+1} &= \arg\min_{D\in\mathcal{B}} \ F_m^{(1)}(D, D^k) \\ &= \arg\min_{D\in\mathcal{B}} \ \frac{1}{2}\|D - D_\rho^k\|^2 - 2\langle W\circ\Delta/\rho + \beta/(2\rho)\mathbf{1}_n\mathbf{1}_n^T, \ \sqrt{D}\rangle. \end{aligned}$$

Consequently,

$$D_{ij}^{k+1} = \mathcal{S}_{\mathcal{B}_{ij}}((D_\rho^k)_{ij}, \ w_{ij}\delta_{ij}/\rho + \beta/(2\rho)).$$

We denote this update by

$$D^{k+1} = \mathcal{T}_1(D^k), \quad k = 0, 1, \ldots, \tag{46}$$

and refer to it as `SQREDMR1`.

```
Algorithm 2 SQREDMR1 Method
1: Input data: Dissimilarity matrix $\Delta$, weight matrix $W$, penalty parameter $\rho > 0$, lower-
   bound matrix $L$, upper-bound matrix $U$ and the initial $D^0$. Set $k := 0$.
2: Update: Compute $D^{k+1}$ by (46).
3: Convergence check: Set $k := k + 1$ and go to Step 2 until convergence.
```

When $i = 2$, it is easy to see

$$\begin{aligned} D^{k+1} &= \arg\min_{D\in\mathcal{B}} \ F_m^{(2)}(D, D^k) \\ &= \arg\min_{D\in\mathcal{B}} \ \frac{1}{2}\|D - (D_\rho^k + (\beta/\rho)\mathbf{1}_n\mathbf{1}_n^T)\|^2 - 2\langle W\circ\Delta/\rho, \ \sqrt{D}\rangle. \end{aligned}$$

Consequently,

$$D_{ij}^{k+1} = \begin{cases} \min\{U_{ij}, \ \max\{(D_\rho^k)_{ij}, \ L_{ij}\} & \text{if } \delta_{ij} = 0 \\ \mathcal{S}_{\mathcal{B}_{ij}}((D_\rho^k)_{ij} + \beta/\rho, \ w_{ij}\delta_{ij}/\rho) & \text{if } \delta_{ij} > 0, \end{cases}$$

We denote this update by

$$D^{k+1} = \mathcal{T}_2(D^k), \quad k = 0, 1, \ldots, . \tag{47}$$

and refer to it as `SQREDMR2`.

```
Algorithm 3 SQREDMR2 Method
1: Input data: Dissimilarity matrix $\Delta$, weight matrix $W$, penalty parameter $\rho > 0$, lower-
   bound matrix $L$, upper-bound matrix $U$ and the initial $D^0$. Set $k := 0$.
2: Update: Compute $D^{k+1}$ by (47).
3: Convergence check: Set $k := k + 1$ and go to Step 2 until convergence.
```

Comparing (46) and (47) to the original algorithm (44), the regularization term does not create any extra computational difficulties. In contrast, it would be difficult to include the regularization term $\mathcal{R}_1$ in the SDP model in [7] without causing significant computational burden.

6 Numerical Examples

The purpose of this section does not intend to given full assessment of the EDM optimization on a widely ranged SNL problems, but to illustrate on the role of the regularizations $\mathcal{R}_1(D)$ and $\mathcal{R}_2(D)$. The main message is that they do improve the localization quality as long as the regularization parameter is small (e.g., less than 1 in our tested cases), though it remains unclear how to choose the best value.

6.1 Implementation and test problems

The detailed implementation including stopping criterion for `SQREDM` can be found in [40] and we will not repeat it here. We describe our test problems below. There are two cases depending on the positions of the anchors: (i) Outer position and (ii) Inner position. The case (ii) was also tested in [40].

Example 6.1 *This example is widely tested since its detailed study in [7]. In the square region* $[-0.5, 0.5]^2$*,* 4 *anchors* $\mathbf{x}_1 = \mathbf{a}_1, \cdots, \mathbf{x}_4 = \mathbf{a}_4$ *(*$m = 4$*) are placed at*

$$\begin{array}{ll} \textit{Case (i) (Outer position):} & (\pm 0.45, \pm 0.45) \\ \textit{Case (ii) (Inner position):} & (\pm 0.20, \pm 0.20) \end{array}$$

The generation of the rest $(n-m)$ *sensors (*$\mathbf{x}_{m+1}, \cdots, \mathbf{x}_n$*) follows the uniform distribution over the square region. The noisy* Δ *is usually generated as follows.*

$$\begin{array}{rcl} \delta_{ij} & := & \|\mathbf{x}_i - \mathbf{x}_j\| \times |1 + \epsilon_{ij} \times \texttt{nf}|, \ \forall \ (i,j) \in \mathcal{N} \\ \mathcal{N} & := & \mathcal{N}_x \cup \mathcal{N}_a \\ \mathcal{N}_x & := & \{(i,j) \mid \|\mathbf{x}_i - \mathbf{x}_j\| \le R, \ i > j > m\} \\ \mathcal{N}_a & := & \{(i,j) \mid \|\mathbf{x}_i - \mathbf{a}_j\| \le R, \ i > m, \ 1 \le j \le m\}, \end{array}$$

where R *is known as the radio range,* ϵ_{ij}*'s are independent standard normal random variables, and* $\texttt{nf}$ *is the noise factor (e.g.,* $\texttt{nf} = 0.2$ *was used in the tests and it corresponds to* 20% *noise level). In literature (e.g., [7]), this type of perturbation in* δ_{ij} *is known to be multiplicative and follows the unit-ball rule in defining* $\mathcal{N}_x$ *and* $\mathcal{N}_a$ *(see [3, Sect. 3.1] for more detail). The corresponding weight matrix* W *and the lower and upper bound matrices* L *and* U *are given as in the table below. Here,* M *is a large positive quantity. For example,* $M := n \max_{ij} \Delta_{ij}$ *is the upper bound of the longest shortest path if the network is viewed as a graph.*

(i,j)	W_{ij}	Δ_{ij}	L_{ij}	U_{ij}
$i = j$	0	0	0	0
$i, j \le m$	0	0	$\|\mathbf{a}_i - \mathbf{a}_j\|^2$	$\|\mathbf{a}_i - \mathbf{a}_j\|^2$
$(i,j) \in \mathcal{N}$	1	δ_{ij}	0	R^2
otherwise	0	0	R^2	M^2

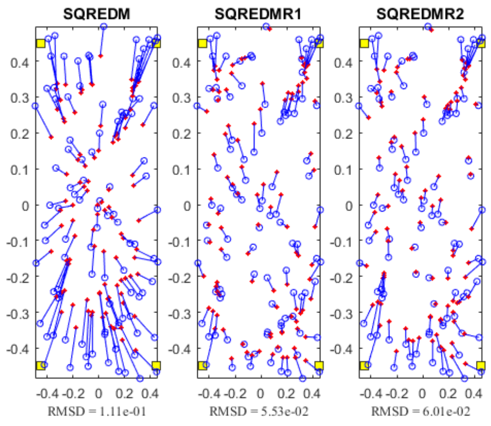
To compare the embedding quality, we use a widely used measure `RMSD` (Root of the Mean Squared Deviation) defined by

$$\texttt{RMSD} := \left[\frac{1}{n-m} \sum_{i=m+1}^{n} \|\widehat{\mathbf{x}}_i - \mathbf{x}_i\|^2 \right]^{1/2},$$

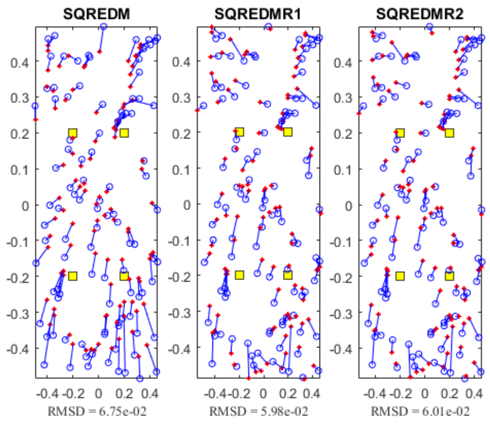
where $\mathbf{x}_i$'s are the true positions of the sensors in our test problems and $\widehat{\mathbf{x}}_i$'s are their corresponding estimates. The $\widehat{\mathbf{x}}_i$'s were obtained by applying `cMDS` to the final output of the distance matrix, followed by aligning them to the existing anchors through the well-known Procrustes procedure (see [8, Chp. 20] or [34, Prop. 4.1]. All tests were run in Matlab 2017b.

6.2 Numerical experiments

First, we would like to assess the level of improvement by the two regularizations. We set $n = 104$ (100 sensors), $\texttt{nf} = 0.2$ and $R = 0.3$, and set the random seed to be $\texttt{rng}(0)$. We also choose $\beta = 0.8$ for both SQREDMR1 and SQREDMR2. Fig. 2 contains the recovered sensors for both cases. Fig. 2a is for Case (i) of outer anchor positions and Fig. 2b is for Case (ii) of inner anchor positions. The Root of Mean Squared Deviation (RMSD) has nearly 50% decrease for the first case while RMSD is only reduced marginally for the second case. This is reasonable because the choice of β is more suitable to Case (i) than to Case (ii).



(a)



(b)

Figure 2: Improving embedding quality by regularization: Fig. 2a is on Case (i) and shows about 50% reduction in RMSD for both regularizations. In contrast, Fig. 2b (Case (ii)) shows a small improvement for both regularizations.

Second, we would like to test how sensitive `SQREDM` is to both regularization terms. As already seen in Fig. 2, the two terms behave quite similarly at $\beta = 0.8$. Surprisingly, it is so over a range of choices. This is clearly shown in Fig. 3 (Case (i)), where RMSD vs β is plotted. SQREDMR1 and SQREDMR2 follow each other quite closely and they produced better RMSD than `SQREDM` for all $\beta \in (0, 1.2)$. Since we are testing the same problem with different choice of β, the plot (in yellow) for `SQREDM` is a straight line. This behaviour is also observed for Case (ii) in Fig. 4, where we see a sudden jump for SQREDMR1 at $\beta \approx 1.66$ and for SQREDMR2 at $\beta \approx 2.45$. It seems that both terms are less sensitive when $\beta \le 1$ than the region $\beta \ge 1$. So our suggestion is to choose β between 0.5 and 1. However, it would be difficult to know what is the best choice of β because it is problem-dependent.

Finally, we test how fast `SQREDM` is when n is getting big. We test problems of size ranging from $n = 200$ to $n = 3000$ and report in Fig. 5 the average results over 20 runs from each randomly generated instance (set `rng('shuffle')`). As seen from Fig. 5a, `SQREDM` used a similar amount of time for both cases (Case (i) and Case (ii)) and are pretty fast. For example $n = 2000$, it only used about 40 seconds to terminate on a Dell laptop with CPU 2.50Ghz and 8GB RAM and it climbed to about 160 seconds when n is increased to 3000. The quality of the localization is shown in Fig. 5b, where we also reported the refined RMSD (rRMSD), which is obtained by using the heuristic gradient method of [7] (called the Refinement Step therein). We see that the quality after the refinement step is very satisfactory, demonstrating that the final embedding provided by `SQREDM` is a good starting point for the refinement step.

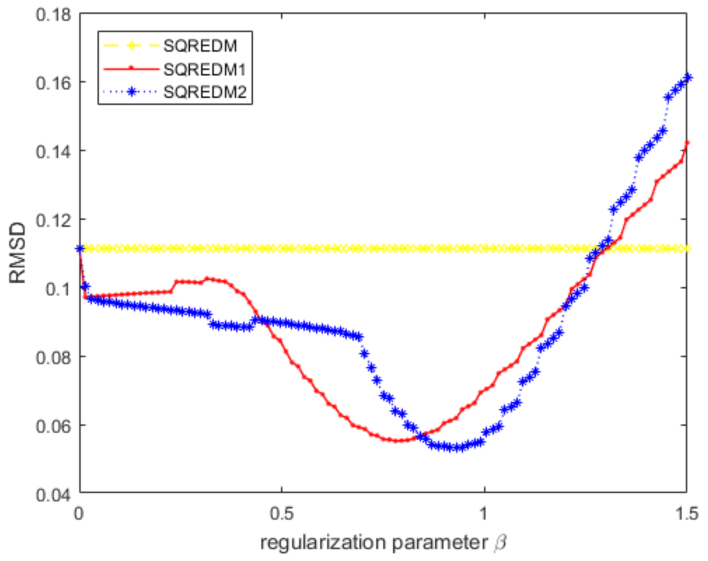
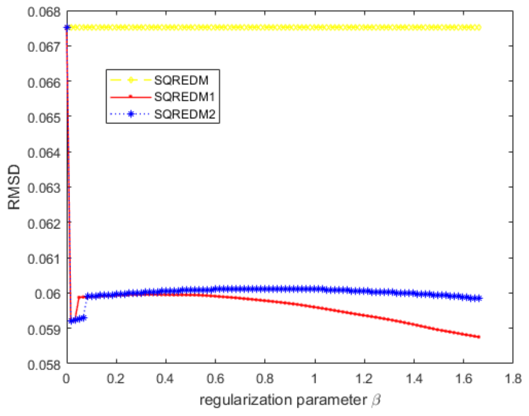
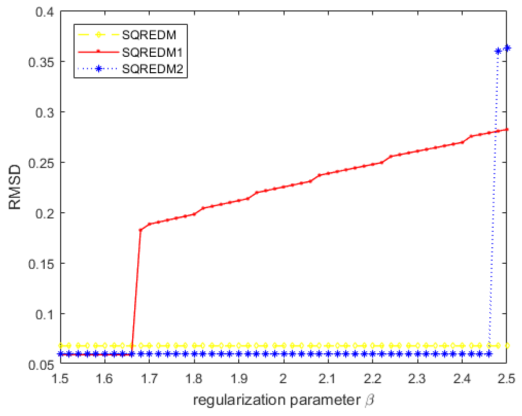


Figure 3: Sensitivity of regularization parameter β on Case (i). Both `SQREDMR1` and `SQREDMR2` behave quite similarly and both improved the embedding quality over a big range of β from 0 to about 1.3.



(a)

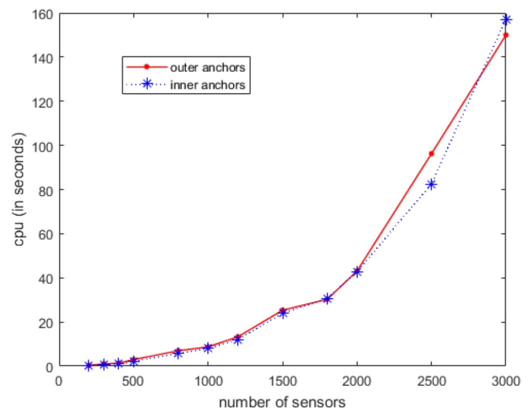


(b)

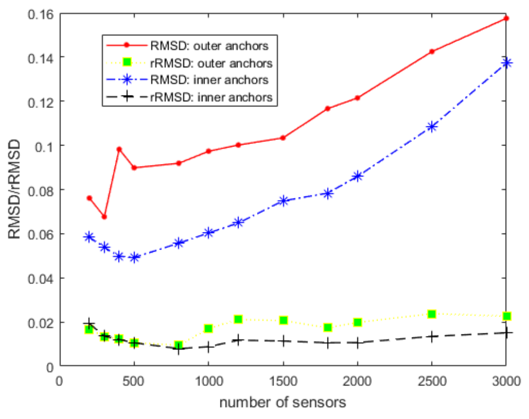
Figure 4: Sensitivity of regularization parameter β on Case (ii). Both `SQREDMR1` and `SQREDMR2` behave quite similarly and both improved the embedding quality over the range of β from 0 to about 1.6 (see Fig. 4a). Beyond this range, `SQREDMR2` continues its steady behaviour, while `SQREDMR1` soon had a big jump to a point that led to bigger RMSD than that by `SQREDM` (see Fig. 4b).

7 Conclusion

Sensor Network Localization has been extensively studied and has a capacity of modelling various practical problems. In this paper, we reviewed some major computational approaches such as the coordinates minimization, SDP and EDM optimization, with the latter two being based on matrix optimization. Therefore, SDP and EDM approaches are both centralized, while the coordinates minimization can be of distributed. We note that the majorization-minimization



(a)



(b)

Figure 5: Speed and quality of `SQREDM` for both Case (i) and Case (ii). The amount of time consumed by `SQREDM` is similar for both cases (see Fig. 5a).. However, the quality for Case (ii) is constantly better than for Case(i) (see Fig. 5b), where we also plotted the refined RMSD.

technique has played an important role in both coordinates minimization and the EDM optimization.

In particular, we discussed the origin of EDM optimization, reviewed in detail some of its major algorithms including the convex relaxation and the latest penalty method `SQREDM`. We also addressed the issue of regularization, that has not been studied before in EDM optimization. We considered two regularization terms, one is on the 'plain" distances and the other is on the "squared" distances. We showed that both terms can be naturally incorporated into `SQREDM`, resulting in `SQREDMR1` and `SQREDMR2`, respectively. Their numerical performance was demonstrated through a SNL test problem. The conclusion is that both regularizations lead to some improvement in embedding quality, but the level of improvement varies with the problem type. The potential of regularization in EDM optimization is worth future investigation with different loss functions and other test problems.

References

[1] A.Y. Alfakih, A. Khandani, and H. Wolkowicz, *Solving Euclidean distance matrix completion problems via semidefinite programming*, Comput. Optim. Appl., 12 (1999), pp. 13–30.

[2] K.S. Arun, T.S. Huang and S.D. Blostein, *Least-squares fitting of two 3-D point sets*, IEEE Trans. Pattern Anal. Machine Intell., 9 (1987), 698–700.

[3] S. Bai and H.-D. Qi, *Tackling the flip ambiguity in wireless sensor network localization and beyond*, Digital Signal Process., 55 (2016), pp. 85-97.

[4] S. Bai, H.-D. Qi, and N. Xiu, *Constrained best Euclidean distance embedding on a sphere: a matrix optimization approach*, SIAM J. Optim., 25 (2015), pp. 439–467.

[5] H.M. Berman, J. Westbrook, Z. Feng, G. Gillilan, T.N. Bhat, H. Weissig, I.N. Shindyalov and P.E. Bourne, "The protein data bank", Nucleic Acids Res. 28, pp. 235242, 2000.

[6] P. Biswas and Y. Ye, *Semidefinite programming for ad hoc wireless sensor network localization*, in *Proceedings of the 3rd IPSN*, Berkeley, CA, pp. 46-54, 2004.

[7] P. Biswas, T.-C. Liang, K.-C. Toh, T.-C. Wang and Y. Ye, *Semidefinite programming approaches for sensor network localization with noisy distance measurements*, IEEE Trans. Auto. Sci. Eng., 3, pp. 360-371, 2006.

[8] I. Borg and P.J.F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, 2nd Ed., Springer Series in Statistics, Springer, 2005.

[9] L. Cayton and S. Dasgupta *Robust Euclidean embedding*, Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA 2006, pp. 169–176.

[10] T.F. Cox and M.A.A. Cox, *Multidimensional Scaling*, 2nd Ed, Chapman and Hall/CRC, 2001.

[11] C. Ding and H.-D. Qi, *Convex optimization learning of faithful Euclidean distance representations in nonlinear dimensionality reduction*, Math. Program., 164 (2017), pp. 341-381.

[12] J. de Leeuw, *Applications of Convex Analysis to Multidimensional Scaling*, In J Barra, F Brodeau, G Romier, B van Cutsem (eds.), *Recent Developments in Statistics*, pp. 133–145. North Holland Publishing Company, Amsterdam, The Netherlands, 1977.

[13] J. Dattorro, *Convex Optimization and Euclidean Distance Geometry*, Meboo Publsihing, USA, 2005.

[14] D. Drusvyatskiy, N. Krislock, Y.-L. Voronin and H. Wolkowicz, *Noisy Euclidean distance realization: Robust facial reduction and the Pareto frontier*, SIAM J. Optim., 27(2017), 2301–2331.

[15] N. Gaffke and R. Mathar, *A cyclic projection algorithm via duality*, Metrika, 36 (1989), pp. 29-54.

[16] W. Glunt, T.L. Hayden, S. Hong and J. Wells, *An alternating projection algorithm for computing the nearest Euclidean distance matrix*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 589-600.

[17] W. Glunt, T.L. Hayden and R. Raydan, *Molecular conformations from distance matrices*, J. Comput. Chemistry, 14 (1993), pp. 114-120.

[18] D.S. Goncalves, A. Mucherino, C. Lavor. and L. Liberti, *Recent advances on the interval distance geometry problem*, J Glob Optim., 69 (2017), 525-545.

[19] J.C. Gower 1966, *Some distance properties of latent root and vector methods used in multivariate analysis*, Biometrika 53 (1966), pp. 325–338.

[20] W.J. Heiser, *Multidimensional scaling with least absolute residuals*, in Proceedings of the First Conference of the International Federation of Classification Societies (IFCS), Aachen, Germany, June 1987, pp. 455-462.

[21] K.F. Jiang, D.F Sun and K.C. Toh, *Solving nuclear norm regularized and semidefinite matrix least squares problems with linear equality constraints*, Discrete Geometry and Optimization, Springer International Publishing, pp. 133-162, 2013.

[22] K.F. Jiang, D.F. Sun and K.-C. Toh, *A partial proximal point algorithm for nuclear norm regularized matrix least squares problems*, Math. Programming Comput., 6 (2014), 281–325.

[23] S. Kim, M. Kojima, H. Waki and M. Yamashita, *Algorithm 920: SFSDP: A sparse version of full semidefinite programming relaxation for sensor network localization problems, ACM Trans. Math. Softw.*, 38(4), pp. 27:1-27:19, 2012.

[24] N. Krislock and H. Wolkowicz, *Euclidean distance matrices and applications*, In *Handbook of Semidefinite, Cone and Polynomial Optimization*, M. Anjos and J. Lasserre (Editors), 2010.

[25] J.B. Kruskal, *Nonmetric multidimensional scaling: a numerical method*, Psychometrika, 29 (1964), pp. 115-129.

[26] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino, *Euclidean distance gemoetry and applications*, SIAM Review 56 (2014), pp. 3–69.

[27] L. Lovász, *Lecture Notes on Semidefinite Programs and combinatorial optimization*, 1992.

[28] K.V. Mardia, *Some properties of classical mulitidimensional scaling*, Comm. Statist. A – Theory Methods, A7:1233-1243, 1978.

[29] P. Oğuz-Ekim, J.P. Gomes, J. Xavier and P. Oliveira, *Robust localization of nodes and time-recursive tracking in sensor networks using noisy range measurements*, IEEE Trans. Signal Process., 59 (2011), pp. 3930-3942.

[30] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'Dea, *Relative location estimation in wireless sensor networks*, IEEE Tran. Signal Processing, 51, 21372148, 2003.

[31] H.-D. Qi, *A semismooth Newton method for the nearest Euclidean distance matrix problem*, SIAM J. Matrix Anal. Appl. 34 (2013), pp. 67-93.

[32] H.-D. Qi, *Conditional quadratic semidefinite programming: examples and methods*, J. Oper. Res. Society of China, 2 (2014) 143-170.

[33] H.-D. Qi and X.M. Yuan, *Computing the nearest Euclidean distance matrix with low embedding dimensions*, Math. Prog., 147 (2014), pp. 351-389.

[34] H.-D. Qi, N.H. Xiu, and X.M. Yuan, *A Lagrangian dual approach to the single source localization problem*, IEEE Trans. Signal Process., 61, pp. 3815-3826, 2013.

[35] I.J. Schoenberg, *Remarks to Maurice Frechet's article Sur la definition axiomatque d'une classe d'espaces vectoriels distancies applicbles vectoriellement sur l'espace de Hilbet*, Ann. Math., 36 (1935), pp. 724-732.

[36] R. Sibson, *Studies in the robustness of multidimensional scaling: perturbational analysis of classical scaling*, J. Royal Statistical Society, B, 41 (1979), 217–219.

[37] W.S. Torgerson, *Multidimensional scaling: I. Theory and method*, Psychometrika, 17 (1952), pp. 401-419.

[38] K.Q. Weinberger and S.K. Saul, *An introduction to nonlinear dimensionality reduction by maximum variance unfolding*, American Association for Artificial Intelligence, 2006, 1683–1686.

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

[40] S. Zhou, N.H. Xiu and H.-D. Qi, *A fast matrix majorization-projection method for penalized stress minimization with box constraints*, IEEE Trans. Signal Process., 66, pp. 4331-4346, 2018.

[41] S. Zhou, N.H. Xiu and H.-D. Qi, *Robust Euclidean embedding via EDM optimization*, Technical Report, March, 2018.