



# Robust Euclidean embedding via EDM optimization

Shenglong Zhou<sup>1</sup> · Naihua Xiu<sup>2</sup> · Hou-Duo Qi<sup>1</sup>

Received: 21 March 2018 / Accepted: 5 July 2019  
© The Author(s) 2019

## Abstract

This paper aims to propose an efficient numerical method for the most challenging problem known as the robust Euclidean embedding (REE) in the family of multi-dimensional scaling (MDS). The problem is notoriously known to be nonsmooth, nonconvex and its objective is non-Lipschitzian. We first explain that the semidefinite programming (SDP) relaxations and Euclidean distance matrix (EDM) approach, popular for other types of problems in the MDS family, failed to provide a viable method for this problem. We then propose a penalized REE (PREE), which can be economically majorized. We show that the majorized problem is convex provided that the penalty parameter is above certain threshold. Moreover, it has a closed-form solution, resulting in an efficient algorithm dubbed as PREEEDM (for Penalized REE via EDM optimization). We prove among others that PREEEDM converges to a stationary point of PREE, which is also an approximate critical point of REE. Finally, the efficiency of PREEEDM is compared with several state-of-the-art methods including SDP and EDM solvers on a large number of test problems from sensor network localization and molecular conformation.

**Keywords** Euclidean embedding · Euclidean distance matrix · Matrix optimization · Majorization and minimization method

**Mathematics Subject Classification** 49M20 · 65B05 · 90C26 · 90C30 · 90C52

---

✉ Hou-Duo Qi  
hdqi@soton.ac.uk  
Shenglong Zhou  
sz3g14@soton.ac.uk  
Naihua Xiu  
nhxiu@bjtu.edu.cn

<sup>1</sup> School of Mathematical Sciences, University of Southampton, Southampton SO17 1BJ, UK

<sup>2</sup> Department of Applied Mathematics, Beijing Jiaotong University, Beijing, China

## 1 Introduction

This paper aims to propose an efficient numerical method for the most challenging problem in the Multi-Dimensional Scaling (MDS) family, which has found many applications in social and engineering sciences [6,10]. The problem is known as the Robust Euclidean Embedding, a term borrowed from [8]. In the following, we first describe the problem and its three variants. We then explain our approach and main contribution. We will postpone the relevant literature review to the next section in order to shorten the introduction.

### 1.1 Problem description

The problem can be described as follows. Suppose we are given some dissimilarity measurements (e.g., noisy distances), collectively denoted as  $\delta_{ij}$ , for some pairs  $(i, j)$  among  $n$  items. The problem is to find a set of  $n$  points  $\mathbf{x}_i \in \mathfrak{R}^r$ ,  $i = 1, \dots, n$  such that

$$d_{ij} := \|\mathbf{x}_i - \mathbf{x}_j\| \approx \delta_{ij} \quad (i, j) \in \mathcal{E}, \quad (1)$$

where  $\|\mathbf{x}\|$  is the Euclidean norm (i.e.,  $\ell_2$  norm) in  $\mathfrak{R}^r$  and  $\mathcal{E}$  is the set of the pairs  $(i, j)$ , whose dissimilarities  $\delta_{ij} > 0$  are known ( $\mathcal{E}$  can be thought of the edge set if we treat  $\delta_{ij}$  as a weighted edge distance between vertex  $i$  and vertex  $j$ , resulting in a weighted graph.) Throughout, we use “:=” or “=:” to mean “define”. The space  $\mathfrak{R}^r$  is called an embedding space and it is most interesting when  $r$  is small (e.g.,  $r = 2, 3$  for data visualization). One may also try to find a set of embedding points such that:

$$D_{ij} := \|\mathbf{x}_i - \mathbf{x}_j\|^2 \approx \delta_{ij}^2 \quad (i, j) \in \mathcal{E}. \quad (2)$$

A great deal of effort has been made to seek the best approximation from (1) or (2). The most robust criterion to quantify the best approximation is the Robust Euclidean Embedding (REE) defined by

$$\min_X f^{(d,1)}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \sum_{i,j=1}^n W_{ij} |d_{ij} - \delta_{ij}|, \quad (3)$$

where  $W_{ij} > 0$  if  $\delta_{ij} > 0$  and  $W_{ij} \geq 0$  otherwise ( $W_{ij}$  can be treated as a weight for the importance of  $\delta_{ij}$ ), and  $X := [\mathbf{x}_1, \dots, \mathbf{x}_n]$  with each  $\mathbf{x}_i$  being a column vector. In [1,8], Problem (3) was referred to as a *robust variant of MDS* and is denoted as  $\text{rMDS}$ . We will reserve  $\text{rMDS}$  for the Robust MDS problem:

$$\min_X f^{(D,1)}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \sum_{i,j=1}^n W_{ij} |D_{ij} - \delta_{ij}^2|. \quad (4)$$

The reference  $\text{rMDS}$  for the problem (4) is more appropriate because it involved the squared distances  $D_{ij}$ , which were used by the classical MDS [22,29,43,49,53]. The

preceding two problems are robust because of the robustness of the  $\ell_1$  norm used to quantify the errors [31, Sect. IV].

When the least squares criterion is used to (1), we have the popular model known as the Kruskal’s stress [30] minimization:

$$\min_X f^{(d,2)}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \sum_{i,j=1}^n W_{ij} \left( d_{ij} - \delta_{ij} \right)^2, \tag{5}$$

Similarly, when the least-squares criterion was applied to (2), we get the so-called squared stress [6]:

$$\min_X f^{(D,2)}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \sum_{i,j=1}^n W_{ij} \left( D_{ij} - \delta_{ij}^2 \right)^2, \tag{6}$$

In many applications such as molecular conformation [21], lower and upper bounds data on the distances can also be collected:

$$L_{ij} \leq D_{ij} \leq U_{ij}, \quad \forall (i, j), \tag{7}$$

where  $0 \leq L_{ij} \leq U_{ij}$ . In applications such as nonlinear dimensionality reduction [47] and sensor network localization [44,54], upper bounds  $U_{ij}$  can be computed by the shortest path distances and  $L_{ij}$  are simply set to be zero.

According to [8, Sect. 5.1], all of those problems are NP-hard. However, some problems are computationally more “difficult” to solve than the others. The most challenging one, which is also the main focus of this paper, is the problem (3) with/without the constraint (7). The difficulty comes from the nonsmooth term of  $\ell_1$  norm and the distance terms  $d_{ij}$  used. All other problems either involve the squared distances  $D_{ij}$  or the squared  $\ell_2$  norm, which make them “easier” to approximate. We will explain the reasons in the literature review part.

In contrast to all other three problems, there lacks efficient methods for the REE problem (3). One of the earliest computational papers that discuss this problem is Heiser [23], which is followed up by [28], where the Huber smoothing function was used to approximate the  $\ell_1$  norm near zero with a majorization technique. It was emphasized in [28] that “the function is not differentiable at its minimum and is hard to majorize, leading to a degeneracy that makes the problem numerically unstable”. Another important method is the PlaceCenter (PC for short) algorithm studied in [1]. We will compare with it in the numerical part. The difficulty in solving (3) is also well illustrated by a sophisticated Semi-definite Programming (SDP) approach in [35, Sect. IV] (see the literature review part). We now describe our approach proposed in this paper.

### 1.2 Our approach and main contributions

Our approach heavily makes use of the concept of Euclidean Distance Matrix (EDM). We need some notation. Let  $\mathcal{S}^n$  denote the space of all  $n \times n$  symmetric matrices,

endowed with the standard inner product. The induced norm is the Frobenius norm, denoted by  $\|A\|$  for  $A \in \mathcal{S}^n$ . The  $(i, j)$ th element of  $A \in \mathcal{S}^n$  is often written as  $A_{ij}$ . Let  $\mathcal{S}_+^n$  be the cone of positive semidefinite matrices in  $\mathcal{S}^n$  and we write  $A \geq 0$  for  $A \in \mathcal{S}_+^n$ . A matrix  $D \in \mathcal{S}^n$  is called an EDM if there exists a set of points  $\mathbf{x}_i \in \mathfrak{R}^r$ ,  $i = 1, 2, \dots, n$  such that the  $(i, j)$ th element of  $D$  is given by  $D_{ij} := \|\mathbf{x}_i - \mathbf{x}_j\|^2$ ,  $i, j = 1, \dots, n$ . The smallest dimension  $r$  is called the embedding dimension of  $D$  and  $r = \text{rank}(JDJ)$ , where  $J := I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$  is known as the centring matrix with  $I$  being the identity matrix in  $\mathcal{S}^n$  and  $\mathbf{1}$  being the vector of all ones in  $\mathfrak{R}^n$ . We use  $\mathcal{D}^n$  to denote the set of all Euclidean distance matrices of size  $n \times n$ .

A very useful characterization for  $D \in \mathcal{D}^n$  [22,49] is

$$\text{diag}(D) = 0 \quad \text{and} \quad -(JDJ) \geq 0. \quad (8)$$

This result shows that  $\mathcal{D}^n$  is a closed and convex cone. Moreover, a set of embedding points are generated by the classical MDS method [22,43,49,53]:

$$[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n] = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \dots, \sqrt{\lambda_r}) [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_r]^T, \quad (9)$$

where the eigenvalues  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r > 0$  and the corresponding eigenvectors  $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_r$  are from the eigen-decomposition:

$$-\frac{1}{2}(JDJ) = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_r] \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_r) [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_r]^T \quad (10)$$

with  $r = \text{rank}(JDJ)$ . Therefore, the REE problem (3) with the constraint (7) can be reformulated in terms of EDM as

$$\begin{aligned} \min_D f(D) &:= \sum_{i,j=1}^n W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| = \|W \circ (\sqrt{D} - \Delta)\|_1 \\ \text{s.t.} \quad D &\in \mathcal{D}^n, \quad \text{rank}(JDJ) \leq r \\ D &\in \mathcal{B} := \{A \mid L \leq D \leq U\}, \end{aligned} \quad (11)$$

where “ $\circ$ ” is the Hadamard product for matrices (i.e.,  $A \circ B = (A_{ij}B_{ij})$ ),  $\sqrt{D}$  is the elementwise square root of  $D$ ,  $\Delta_{ij} := \delta_{ij}$ , and  $\|\cdot\|_1$  is the  $\ell_1$  norm. Once we obtained an optimal solution of (11), we use (9) and (10) to generate the required embedding points.

The reformulation well captures four difficulties in solving the REE problem (3).

- (i) The objective function  $f(D)$  is not convex. The term  $|\sqrt{D_{ij}} - \delta_{ij}|$  is convex when  $\delta_{ij}^2 > D_{ij}$  and concave otherwise.
- (ii) The objective function is nonsmooth. It is not differentiable at certain points due to the  $\ell_1$  norm and the square root operation involved.
- (iii) The objective function is not Lipschitzian. The Lipschitz constant goes to infinity as  $D_{ij}$  goes to zero. The implication is that the subdifferential of the objective function [42, Def. 8.3] may be unbounded. This would create a huge obstacle in establishing any convergence results of iterative algorithms for (11).

- (iv) The rank constraint is not convex and is hard to approximate. This is a common issue for any optimization problem with a rank constraint.

We note that no matter what reformulations one may use for (3), those four difficulties would appear in different forms and won't go away. We also note that all other three problems, when reformulated in terms of EDM, have a convex objective function. This distinctive feature alone makes the problem (11) the most challenging one to solve.

Existing numerical experiments have shown that the MDS embedding (9) and (10) works well as long as  $D$  is close to a true EDM. A typical example is when the data sits on a lower-dimensional manifold [47]. Motivated by this, we are going to generate an approximate EDM instead of a true EDM in our algorithm. It follows from (8) that (also see [32, Theorem A]):

$$D \in \mathcal{D}^n \iff \text{diag}(D) = 0 \quad \text{and} \quad -D \in \mathcal{K}_+^n, \tag{12}$$

where  $\mathcal{K}_+^n$  is known to be the conditionally positive semidefinite cone:

$$\mathcal{K}_+^n := \left\{ A \in \mathcal{S}^n \mid \mathbf{v}^T A \mathbf{v} \geq 0, \forall \mathbf{v} \in \mathbf{1}^\perp \right\}$$

and  $\mathbf{1}^\perp$  is the subspace in  $\mathfrak{N}^n$  orthogonal to  $\mathbf{1}$ . The diagonal constraint in (12) can be integrated to the set  $\mathcal{B}$  with the choice  $L_{ii} = U_{ii} = 0$  for  $i = 1, \dots, n$ . We combine  $\mathcal{K}_+^n$  with the rank constraint into the set  $\mathcal{K}_+^n(r)$ :

$$\mathcal{K}_+^n(r) := \mathcal{K}_+^n \cap \left\{ A \in \mathcal{S}^n \mid \text{rank}(JAJ) \leq r \right\}.$$

We call it the conditionally positive semidefinite cone with the rank- $r$  cut. Consequently, the constraints in (11) become  $-D \in \mathcal{K}_+^n(r)$  and  $D \in \mathcal{B}$ .

Next, we quantify the feasibility of  $-D$  belonging to  $\mathcal{K}_+^n(r)$  as follows. Let  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  be the set of all nearest points in  $\mathcal{K}_+^n(r)$  from a give matrix  $A \in \mathcal{S}^n$ . That is

$$\Pi_{\mathcal{K}_+^n(r)}^B(A) := \text{argmin} \left\{ \|A - Y\| \mid Y \in \mathcal{K}_+^n(r) \right\}. \tag{13}$$

Since  $\mathcal{K}_+^n(r)$  is not convex (unless  $r \geq n - 1$ ), the projection  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  is a set instead of a single point. We let  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  be any element in  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  and define the function

$$g(A) := \frac{1}{2} \|A + \Pi_{\mathcal{K}_+^n(r)}(-A)\|^2. \tag{14}$$

Since  $g(A)$  is just the half of the squared distance from  $(-A)$  to  $\mathcal{K}_+^n(r)$ , it does not depend on which element  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  is being used. It is easy to see that

$$-D \in \mathcal{K}_+^n(r) \quad \text{if and only if} \quad g(D) = 0.$$

Hence, the problem (11) is equivalent to

$$\begin{aligned} \min_D f(D) &= \|W \circ (\sqrt{D} - \Delta)\|_1 \\ \text{s.t. } g(D) &= 0, \quad D \in \mathcal{B}. \end{aligned} \quad (15)$$

This is a classical constrained optimization problem with an equality constraint and a simple box constraint. Therefore, the quadratic penalty method [34, Chp. 17] can be applied to get the following problem:

$$\min_D f_\rho(D) := f(D) + \rho g(D), \quad \text{s.t. } D \in \mathcal{B}, \quad (16)$$

where  $\rho > 0$  is the penalty parameter. We refer to this problem as the penalized REE problem (PREE).

The quadratic penalty method is used often in practice [34, P. 497]. In fact, it is particularly suitable to (11) because it overcomes all four difficulties discussed above. We will need two more important tools to help us efficiently solve the penalty problem (16). One is the majorization technique that has recently become very popular in engineering sciences [46] (also see [6, Chp. 8] for its extensive use in MDS). Suppose we have the current iterate  $D^k$ . We construct a majorization function  $g_m(D, D^k)$  for  $g(D)$  at  $D^k$  such that

$$g_m(D^k, D^k) = g(D^k) \quad \text{and} \quad g_m(D, D^k) \geq g(D) \quad \forall D \in \mathcal{S}^n. \quad (17)$$

The majorization is constructed in such a way that it is easier to solve the majorized problem:

$$D^{k+1} = \operatorname{argmin} \left\{ f_\rho^k(D) := f(D) + \rho g_m(D, D^k), \quad D \in \mathcal{B}. \right\} \quad (18)$$

It can be seen that

$$\begin{aligned} f_\rho(D^{k+1}) &= f(D^{k+1}) + \rho g(D^{k+1}) \\ &\stackrel{(17)}{\leq} f(D^{k+1}) + \rho g_m(D^{k+1}, D^k) = f_\rho^k(D^{k+1}) \\ &\stackrel{(18)}{\leq} f_\rho^k(D^k) = f(D^k) + \rho g_m(D^k, D^k) = f(D^k) + \rho g(D^k) = f_\rho(D^k). \end{aligned}$$

Hence, the algorithm generates a sequence  $\{D^k\}$  that is nonincreasing in  $f_\rho(D)$ . Since  $f_\rho(D)$  is bounded below by 0, the functional sequence  $\{f_\rho(D^k)\}$  converges. However, we are more concerned where the iterate sequence  $\{D^k\}$  converges. The second concern is how the subproblem (18) has to be solved. This brings out the second technique, which is to solve the following one-dimensional problem:

$$\min_{x \in \mathfrak{H}} \left\{ q(x) := (1/2)(x - \omega)^2 + \beta|\sqrt{x} - \delta| \mid a \leq x \leq b \right\}, \quad (19)$$

for given  $\delta > 0$  and  $0 \leq a \leq b$ . We will show that the solution of this problem will lead to a close-form solution of (18).

Since our method is for the Penalized REE by EDM optimization, we call it PREEEDM. The major contribution of this paper is to make the outlined solution procedure water-tight. In particular, we will investigate the relationship between the PREE problem (16) and the original problem (11) in terms of the  $\epsilon$ -optimality (Proposition 1). We will also show that the majorization function  $g_m(\cdot, \cdot)$  can be economically constructed (Sect. 3.2). Moreover, the majorized function  $f_\rho^k(D)$  is guaranteed to be convex provided that the penalty parameter is above certain threshold and the subdifferentials at the generated sequences are bounded (Proposition 4). Furthermore, each majorization subproblem has a closed form solution (Theorem 1). We are also able to prove that any accumulation of the generated sequence by PREEEDM is a stationary point of (16), which is also an approximate stationary point of (11) (Theorem 2). Built upon its solid convergence results and simple implementation, PREEEDM is demonstrated to be comparable to six state-of-the-art software packages in terms of solution quality and outperform them in terms of the computational time for a large number of tested problems from sensor network localizations and molecular conformations.

### 1.3 Organization of the paper

In the next section, we give a selective literature review mainly on the Semi-Definite Programming (SDP) and EDM approaches. In Sect. 3, we introduce some necessary background and prove a key technical result (Lemma 1) that is crucial to the convexity of the majorization subproblem. We study the relationship between the penalized REE (16) and the original REE in Sect. 4, where the majorized subproblem is shown to have a closed-form solution. In Sect. 5, we provide a complete set of convergence results for the proposed PREEEDM algorithm. Numerical experiments are included in Sect. 6. The paper concludes in Sect. 7. All proofs except that of Theorem 2 can be found in “Appendix”.

## 2 Literature review

One can find a thorough review on all of the four problems in [17] by France and Carroll, mainly from the perspective of applications. One can also find valuable discussion on some of those problems in [2]. So the starting point of our review is that those problems have their own reasons to be studied and we are more concerned how they can be efficiently solved.

Most of existing algorithms can be put in three groups. The first group consists of alternating coordinates descent methods, whose main variables are  $\mathbf{x}_i$ ,  $i = 1, \dots, n$ . A famous representative in this group is the method of SMACOF for the stress minimization (5) [13, 14]. The key idea is to alternatively minimize the function  $f^{(d,2)}$  with respect to each  $\mathbf{x}_i$ , while keeping other points  $\mathbf{x}_j$  ( $j \neq i$ ) unchanged, and each minimization problem is relatively easier to solve by employing the technique of *majorization*. SMACOF has been widely used and the interested reader can refer to [6]

for more references and to [54] for some critical comments on SMACOF when it is applied to the sensor network localization problem. The second and third group consist respectively the methods of SDP and EDM optimization. We will give a more detailed review on the two groups because of their close relevance to our proposed method in this paper. The main purpose of our review is to show that there lacks efficient numerical methods for the REE problem (3).

## 2.1 On SDP approach

We note that each of the four objective functions either involves the Euclidean distance  $d_{ij}$  or its squared  $D_{ij} = d_{ij}^2$ . A crucial observation is that constraints on them often have SDP relaxations. For example, it is easy to see

$$\begin{aligned} D_{ij} = d_{ij}^2 &= \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\mathbf{x}_i^T \mathbf{x}_j \\ &= Y_{ii} + Y_{jj} - 2Y_{ij}, \end{aligned} \quad (20)$$

where  $Y := X^T X \succeq 0$ . Hence, the squared distance  $d_{ij}^2$  is a linear function of the positive semidefinite matrix  $Y$ . Consequently, the EDM cone  $\mathcal{D}^n$  can be described through linear transformations of positive semidefinite matrices. One can further relax the constraint  $Y = X^T X$  to  $Y \succeq X^T X$ . By the Schur-complement, one has

$$Z := \begin{bmatrix} Y & X^T \\ X & I_r \end{bmatrix} \succeq 0 \text{ has rank } r \iff Y = X^T X. \quad (21)$$

By dropping the rank constraint, the robust MDS problem (4) can be relaxed to a SDP, which was initiated by Biswas and Ye [15].

For the Euclidean distance  $d_{ij}$ , we introduce a new variable  $T_{ij} = d_{ij}$ . One may relax this constraint to  $T_{ij} \leq d_{ij}$ , which has a SDP representation:

$$T_{ij}^2 \leq d_{ij}^2 = D_{ij} \iff \begin{bmatrix} 1 & T_{ij} \\ T_{ij} & D_{ij} \end{bmatrix} \succeq 0. \quad (22)$$

Combination of (20), (21) and (22) leads to a large number of SDP relaxations. Typical examples, for the robust MDS problem (4), are the SDP relaxation method [5] and the edge-based SDP relaxation method [38,50] and [27], which leads to a comprehensive Matlab package `SFSDP`. For the squared stress (6), one may refer to [16,25]. For the stress problem (5), a typical SDP relaxation can be found in [35, Problem (8)]. However, unlike the problems (4), (5) and (6), the REE problem (3) does not have a straightforward SDP relaxation. We use an attempt made in [35] to illustrate this point below.

First, it is noted that problem (3) can be written in terms of EDM:

$$\begin{aligned} \min & \sum_{i,j=1}^n W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| \\ \text{s.t. } & D \in \mathcal{D}^n, \text{ rank}(JDJ) \leq r. \end{aligned}$$



The term  $|\sqrt{D_{ij}} - \delta_{ij}|$  is convex if  $\delta_{ij} > \sqrt{D_{ij}}$  and is concave otherwise. A major obstacle is how to efficiently deal with the concavity in the objective.

Secondly, by dropping the rank constraint and through certain linear approximation to the concave term, a SDP problem is proposed for (3) (see [35, Eq. (20)]):

$$\begin{aligned}
 & \min_{D, T \in \mathcal{S}^n} \langle W, T \rangle \\
 & \text{s.t.} \quad (\delta_{ij} - T_{ij})^2 \leq D_{ij}, \quad (i, j) \in \mathcal{E} \\
 & \quad \quad a_{ij}D_{ij} + b_{ij} \leq T_{ij}, \quad (i, j) \in \mathcal{E} \\
 & \quad \quad D \in \mathcal{D}^n,
 \end{aligned} \tag{23}$$

where the quantities  $a_{ij}$  and  $b_{ij}$  can be computed from  $\delta_{ij}$ . We note that each quadratic constraint in (23) is equivalent to a positive semidefinite constraint on  $\mathcal{S}^2_+$  and  $D \in \mathcal{D}^n$  is a semidefinite constraint on  $\mathcal{S}^n_+$  by (8). Therefore, the total number of the semidefinite constraints is  $|\mathcal{E}| + 1$ . Finally, the optimal solution of (23) is then refined through a second-stage algorithm (see [35, Sect. IV(B)]). Both stages of the algorithmic scheme above would need sophisticated implementation skills and its numerical efficiency and solution quality are yet to be confirmed due to the twice relaxation schemes (dropping the rank constraint and linearization). The lack of efficient algorithms for (3) motivated our research in this paper.

### 2.2 On EDM approach

A distinguishing feature from the SDP approach is that this approach treats EDM  $D$  as the main variable, without having to rely on its SDP representation. This approach works because of the characterization (12) and that the orthogonal projection onto  $\mathcal{K}^n_+$  has a closed-form formula [19,20]. Several methods are based on this formula. The basic model for this approach is the so-called nearest EDM problem:

$$\min_{D \in \mathcal{S}^n} \|D - \Delta^{(2)}\|^2 \quad \text{s.t.} \quad \text{diag}(D) = 0 \quad \text{and} \quad -D \in \mathcal{K}^n_+, \tag{24}$$

which is a convex relaxation of (6) with the special choice  $W_{ij} \equiv 1$ . Here the elements of the matrix  $\Delta^{(2)}$  are given by  $\Delta^{(2)}_{ij} := \delta_{ij}^2$ . The relaxation is obtained by dropping the rank constraint  $\text{rank}(JDJ) \leq r$ . Since the constraints of (24) are the intersection of a subspace and a convex cone, the method of alternation projection was proposed in [19,20] with applications to molecule conformation [21]. Newton’s method for (24) was developed in [39]. Extensions of Newton’s method for the model (24) with more constraints including general weights  $W_{ij}$ , the rank constraint  $\text{rank}(JDJ) \leq r$  or the box constraints (7) can be found in [3,11,40]. A recent application of the model (24) with a regularization term to Statistics is [55], where the problem is solved by an SDP, similar to that proposed by Toh [48].

There are two common features in this class of methods. One is that they require the objective function to be convex, which is true for the problems (4), (5) and (6) when formulated in EDM. The second feature is that the nonconvexity is only caused by the rank constraint. However, as already seen in Sect. 1.2, the REE problem (3) in

terms of EDM has a nonconvex objective coupled with the distance  $d_{ij}$  (not squared distances) being used. This has caused various difficulties in applying the existing EDM-based methods mentioned above to (3). A latest research [56] by the authors has tried to extend the EDM approach to the stress minimization problem (5) along a similar line as outlined in Sect. 1.2. Once again, we emphasize that the key difference between the problem (3) and (5) is about nonconvex objective vs convex objective and non-differentiability vs differentiability. Hence, the problem (3) is significantly more difficult to solve than (5). Nevertheless, we will show that it can be efficiently solved by the proposed EDM optimization.

### 3 Background and technical lemmas

In this part, we introduce the necessary background about subgradient and positive roots of a special depressed cubic equation. In particular, we will prove a technical result about a composite function between the absolute value and the square root functions. This result (Lemma 1) is in the style of Taylor-expansion for differentiable functions.

#### 3.1 Subgradients of functions

An important function appearing in our EDM reformulation (11) of the REE problem (3) is  $\phi_\delta(\cdot) : \mathfrak{R}_+ \mapsto \mathfrak{R}_+$  defined for a given constant  $\delta > 0$  by

$$\phi_\delta(x) := |\sqrt{x} - \delta|, \quad \forall x \geq 0,$$

where  $\mathfrak{R}_+$  is the set of all nonnegative numbers. We will need to compute its subgradient in the sense of Rockafellar and Wets [42].

**Definition 1** [42, Def. 8.3] Consider a function  $f : \mathfrak{R}^n \mapsto \mathfrak{R} \cup \{-\infty, +\infty\}$  and a point  $\bar{\mathbf{x}}$  with  $f(\bar{\mathbf{x}})$  finite. For a vector  $\mathbf{v} \in \mathfrak{R}^n$ , one says that

(a)  $\mathbf{v}$  is a *regular subgradient* of  $f$  at  $\bar{\mathbf{x}}$ , written  $\mathbf{v} \in \widehat{\partial} f(\bar{\mathbf{x}})$ , if

$$f(\mathbf{x}) \geq f(\bar{\mathbf{x}}) + \langle \mathbf{v}, \mathbf{x} - \bar{\mathbf{x}} \rangle + o(\|\bar{\mathbf{x}} - \mathbf{x}\|),$$

where the little ‘ $o$ ’ term is a short-hand notation for the one-sided limit condition:

$$\liminf_{\substack{\mathbf{x} \rightarrow \bar{\mathbf{x}} \\ \mathbf{x} \neq \bar{\mathbf{x}}}} \frac{f(\mathbf{x}) - f(\bar{\mathbf{x}}) - \langle \mathbf{v}, \mathbf{x} - \bar{\mathbf{x}} \rangle}{\|\mathbf{x} - \bar{\mathbf{x}}\|} \geq 0;$$

(b)  $\mathbf{v}$  is a (general) *subgradient* of  $f$  at  $\bar{\mathbf{x}}$ , written  $\mathbf{v} \in \partial f(\bar{\mathbf{x}})$ , if there are sequences  $\mathbf{x}^v \rightarrow \bar{\mathbf{x}}$  with  $f(\mathbf{x}^v) \rightarrow f(\bar{\mathbf{x}})$  and  $\mathbf{v}^v \in \widehat{\partial} f(\mathbf{x}^v)$  with  $\mathbf{v}^v \rightarrow \mathbf{v}$ .

We call  $\partial f(\bar{\mathbf{x}})$  the subdifferential of  $f$  at  $\bar{\mathbf{x}}$ . For a given number  $x \in \Re$ , we define its sign by

$$\text{sign}(x) := \begin{cases} \{1\} & \text{if } x > 0 \\ [-1, 1] & \text{if } x = 0 \\ \{-1\} & \text{if } x < 0. \end{cases}$$

Apparently,  $\phi_\delta(x)$  is continuous for  $x > 0$  and its subdifferential at  $x > 0$  is given by directly applying Def. 1 (note  $\delta > 0$ )

$$\partial\phi_\delta(x) = \frac{\text{sign}(\sqrt{x} - \delta)}{2\sqrt{x}} \quad \text{for } x > 0. \tag{25}$$

We note that the subdifferential of  $\phi_\delta(x)$  at  $x = 0$  is more complicated to describe. Fortunately, we won't need it in our analysis. We state our key lemma below.

**Lemma 1** *Let  $\delta > 0$  be given. It holds*

$$\phi_\delta(y) \geq \phi_\delta(x) + \zeta(y - x) - \frac{(x - y)^2}{8\delta^3}, \quad \forall x > 0, y > 0, \zeta \in \partial\phi_\delta(x).$$

Compared with the definition of classical convex function, the inequality above has an extra negative quadratic term, which is not sufficient for  $\phi_\delta(x)$  to be convex on  $(0, +\infty)$  (actually,  $\phi_\delta(x)$  is quasi-convex.) Fortunately, this property allows us to make the function  $\phi_\delta + \mu\varphi$  convex provided  $\varphi$  is a strongly convex and  $\mu$  is chosen properly. And this is one of the key ideas developed in this paper.

### 3.2 Construction of the majorization function

A major building block in our algorithm is the majorization function  $g_m(D, D^k)$  at a given point  $D^k$  for the function  $g(A)$  defined in (14). We construct it below.

Suppose  $A \in S^n$  has the following eigenvalue-eigenvector decomposition:

$$A = \lambda_1 \mathbf{p}_1 \mathbf{p}_1^T + \lambda_2 \mathbf{p}_2 \mathbf{p}_2^T + \dots + \lambda_n \mathbf{p}_n \mathbf{p}_n^T, \tag{26}$$

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

$$\text{PCA}_r^+(A) := \sum_{i=1}^r \max\{0, \lambda_i\} \mathbf{p}_i \mathbf{p}_i^T. \tag{27}$$

Recall the definition of  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  in (13). We let  $\Pi_{\mathcal{K}_+^n(r)}(A)$  be an element of  $\Pi_{\mathcal{K}_+^n(r)}^B(A)$  and note that the function  $g(A)$  in (14) does not depend on the choice of

$\Pi_{\mathcal{K}_+^n(r)}(A)$ . As seen from the known results below, one particular element  $\Pi_{\mathcal{K}_+^n(r)}(A)$  can be computed through  $\text{PCA}_r^+(A)$ .

**Lemma 2** For a given matrix  $A \in \mathcal{S}^n$  and an integer  $r \leq n$ . The following results hold.

- (i) [40, Eq. (22), Proposition 3.3] One particular  $\Pi_{\mathcal{K}_+^n(r)}(A)$  can be computed through

$$\Pi_{\mathcal{K}_+^n(r)}(A) = \text{PCA}_r^+(JAJ) + (A - JAJ) \quad (28)$$

- (ii) [40, Eq. (26), Proposition 3.3] We have

$$\langle \Pi_{\mathcal{K}_+^n(r)}(A), A - \Pi_{\mathcal{K}_+^n(r)}(A) \rangle = 0. \quad (29)$$

- (iii) [40, Proposition 3.4] The function

$$h(A) := \frac{1}{2} \|\Pi_{\mathcal{K}_+^n(r)}(A)\|^2$$

is well defined and is convex. Moreover,

$$\Pi_{\mathcal{K}_+^n(r)}(A) \in \partial h(A),$$

where  $\partial h(A)$  is the subdifferential of  $h(\cdot)$  at  $A$ .

- (iv) [56, Lemma 2.2] Let  $g(A)$  be defined in (14). We have for any  $A \in \mathcal{S}^n$

$$g(A) = \frac{1}{2} \|A\|^2 - h(-A) \quad \text{and} \quad \|\Pi_{\mathcal{K}_+^n(r)}(A)\| \leq 2\|A\|. \quad (30)$$

Since  $h(\cdot)$  is convex and  $\Pi_{\mathcal{K}_+^n(r)}(A) \in \partial h(A)$  (Lemma 2)(ii), we have

$$h(-D) \geq h(-Z) + \langle \Pi_{\mathcal{K}_+^n(r)}(-Z), -D + Z \rangle \quad \forall D, Z \in \mathcal{S}^n.$$

This, with Lemma 2(iii), implies

$$\begin{aligned} g(D) &= (1/2)\|D\|^2 - h(-D) \\ &\leq (1/2)\|D\|^2 - h(-Z) + \langle \Pi_{\mathcal{K}_+^n(r)}(-Z), D - Z \rangle \\ &= (1/2)\|D + \Pi_{\mathcal{K}_+^n(r)}(-Z)\|^2 + \langle \Pi_{\mathcal{K}_+^n(r)}(-Z), -Z - \Pi_{\mathcal{K}_+^n(r)}(-Z) \rangle \\ &\stackrel{(29)}{=} (1/2)\|D + \Pi_{\mathcal{K}_+^n(r)}(-Z)\|^2 \\ &=: g_m(D, Z). \end{aligned} \quad (31)$$

It is straightforward to check that the function  $g_m(\cdot, \cdot)$  in (31) satisfies the majorization properties (17).

### 3.3 Positive roots of depressed cubic equations

In our algorithm, we will encounter the positive root of a depressed cubic equation [7, Chp. 7], which arises from the optimality condition of the following problem

$$\min_{x \geq 0} s(x) := (x - t)^2 + v\sqrt{x}, \tag{32}$$

where  $v > 0$  and  $t \in \Re$  are given. A positive stationary point  $x$  must satisfy the optimality condition

$$0 = s'(x) = 2(x - t) + \frac{v}{2\sqrt{x}}. \tag{33}$$

Let  $y := \sqrt{x}$ . The optimality condition above becomes

$$4y^3 - 4ty + v = 0.$$

This is in the classical form of the so-called depressed cubic equation [7, Chp. 7]. Its roots (complex or real) and their computational formulae have a long history with fascinating and entertaining stories. A comprehensive revisit of this subject can be found in Xing [51] and a successful application of the depressed cubic equation to the compressed sensing can be found in [36,52]. The following lemma says that, under certain conditions, the Eq. (33) has two distinctive positive roots and its proof is a specialization of [9, Lemma 2.1(iii)] when  $p = 1/2$  therein.

**Lemma 3** [9, Lemma 2.1(iii)] *Consider the problem (32). Let*

$$\bar{x} = (v/8)^{2/3} \quad \text{and} \quad \bar{t} = 3\bar{x}.$$

*When  $t > \bar{t}$ ,  $s(x)$  has two different positive stationary point  $\hat{x}_1$  and  $\hat{x}_2$  satisfying*

$$s'(x) = 0 \quad \text{and} \quad \hat{x}_1 < \bar{x} < \hat{x}_2.$$

## 4 Penalized REE model and its' majorization subproblem

With the preparation above, we are ready to address our penalized REE problem (16) and its majorization subproblem (18). We first address the relationship between (16) and its original problem (11). We then show how the subproblem (18) is solved.

### 4.1 $\epsilon$ -optimal solution and critical point

The classical results on penalty methods in [34] on the differentiable case (i.e., all functions involved are differentiable) are not applicable here. Recently, the penalty approach was studied by Gao in her PhD thesis [18] in the context of semidefinite programming, which motivated our investigation below. The main result is that (16)

provides an  $\epsilon$ -optimal solution and  $\epsilon$ -critical point for the original problem when the penalty parameter is above certain threshold.

We first introduce the concept of critical point of (11) [i.e., (15)], which is a first order optimality condition. It is associated with the Lagrange function of (15) defined by

$$\mathcal{L}(D, \beta) := f(D) + \beta g(D) + I_{\mathcal{B}}(D), \quad (34)$$

where  $\beta \in \Re$  is the Lagrangian multiplier and  $I_{\mathcal{B}}(D)$  is the indicator function taking value 0 if  $D \in \mathcal{B}$  and  $+\infty$  otherwise. We say  $\bar{D} \in \mathcal{S}^n$  is a critical point of (15) if it satisfies  $0 \in \partial \mathcal{L}(\bar{D}, \bar{\beta})$  for some  $\bar{\beta} \in \Re$ , namely, there exists  $\bar{\Gamma} \in \partial f(\bar{D})$  and  $\bar{C} \in \text{Conv}(\Pi_{\mathcal{K}_+^n(r)}^{\mathcal{B}}(-\bar{D}))$  such that

$$g(\bar{D}) = 0 \quad \text{and} \quad \langle \bar{\Gamma} + \bar{\beta} \bar{D} + \bar{\beta} \bar{C}, D - \bar{D} \rangle \geq 0, \quad \forall D \in \mathcal{B}$$

where we used the following facts:  $\partial g(D) = D + \text{Conv}(\Pi_{\mathcal{K}_+^n(r)}^{\mathcal{B}}(-D))$  (see [40, Proposition 3.4] and  $\text{Conv}(\Omega)$  denotes the convex hull of a given set  $\Omega$ ), and the sub-differential of the indicator function of a convex set  $\Omega$  is its normal cone, namely,  $\partial I_{\Omega}(x) = N_{\Omega}(x) = \{y : \langle y, z - x \rangle \leq 0, \forall z \in \Omega\}$ . We will see the choice  $\bar{C} = \Pi_{\mathcal{K}_+^n(r)}^{\mathcal{B}}(-\bar{D})$  is enough for our analysis and hence it is used in our definition of approximate critical point.

**Definition 2** (i) ( $\epsilon$ -Optimal solution) Suppose  $D^*$  is an optimal solution of (11). For a given error tolerance  $\epsilon > 0$ , a point  $\hat{D}$  is called an  $\epsilon$ -optimal solution of (11) if it satisfies

$$\hat{D} \in \mathcal{B}, \quad g(\hat{D}) \leq \epsilon \quad \text{and} \quad f(\hat{D}) \leq f(D^*).$$

(ii) ( $\epsilon$ -Critical point) A point  $\bar{D}$  is called an  $\epsilon$ -critical point of (11) if there exist  $\bar{\Gamma} \in \partial f(\bar{D})$  and  $\bar{\beta} \in \Re$  such that

$$g(\bar{D}) \leq \epsilon, \quad \langle \bar{\Gamma} + \bar{\beta} \bar{D} + \bar{\beta} \Pi_{\mathcal{K}_+^n(r)}^{\mathcal{B}}(-\bar{D}), D - \bar{D} \rangle \geq 0, \quad \forall D \in \mathcal{B}. \quad (35)$$

Obviously, if  $\epsilon = 0$ ,  $\epsilon$ -optimal solution  $\hat{D}$  would be an optimal solution of (11). We will show that the optimal solution of (16) is  $\epsilon$ -optimal as well as an  $\epsilon$ -critical point provided that  $\rho$  is large enough. Let  $D_{\rho}^*$  be an optimal solution of the penalized REE (16) and  $D_r$  be any feasible solution of the original problem (11). If the lower bound matrix  $L \equiv 0$ , then we can simply choose  $D_r = 0$ . Define

$$\rho_{\epsilon} := f(D_r)/\epsilon.$$

We have the following result.

**Proposition 1** For any  $\rho \geq \rho_{\epsilon}$ ,  $D_{\rho}^*$  must be an  $\epsilon$ -optimal solution as well as an  $\epsilon$ -critical point of (11). That is, there exists  $\Gamma^* \in \partial f(D_{\rho}^*)$  such that

$$D_\rho^* \in \mathcal{B}, \quad f(D_\rho^*) \leq f(D^*), \quad g(D_\rho^*) \leq \epsilon \tag{36}$$

$$\langle \Gamma^* + \rho D_\rho^* + \rho \Pi_{\mathcal{K}_+^n(r)}(-D_\rho^*), D - D_\rho^* \rangle \geq 0, \quad \forall D \in \mathcal{B}. \tag{37}$$

### 4.2 Solving the subproblem

Having constructed the majorization function in (31), we now focus on how to solve the majorization subproblem (18), which is equivalent to the solution of the following problem. Given the current iterate  $Z \in \mathcal{B}$ , the majorization subproblem aims to compute an improved iterate, denoted by  $Z^+$ , by solving

$$\begin{aligned} Z^+ &= \arg \min_{D \in \mathcal{B}} f(D) + \rho g_m(D, Z) \\ &= \arg \min_{D \in \mathcal{B}} \sum_{i,j=1}^n W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| + \frac{\rho}{2} \|D + \Pi_{\mathcal{K}_+^n(r)}(-Z)\|^2 \\ &= \arg \min_{D \in \mathcal{B}} \sum_{i,j=1}^n W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| + \frac{\rho}{2} \|D - Z_K\|^2, \end{aligned} \tag{38}$$

where the matrix  $Z_K := -\Pi_{\mathcal{K}_+^n(r)}(-Z)$ . This subproblem has a perfect separability property that allows it to be computed elementwise:

$$\begin{aligned} Z_{ij}^+ &= \arg \min_{L_{ij} \leq D_{ij} \leq U_{ij}} \frac{\rho}{2} [D_{ij} - (Z_K)_{ij}]^2 + W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| \\ &= \arg \min_{L_{ij} \leq D_{ij} \leq U_{ij}} \frac{1}{2} [D_{ij} - (Z_K)_{ij}]^2 + \frac{W_{ij}}{\rho} |\sqrt{D_{ij}} - \delta_{ij}|. \end{aligned} \tag{39}$$

For the ease of our description, we denote the subproblem solution process by

$$Z^+ = \text{PREEEDM}_{\mathcal{B}}(Z_K, W/\rho, \Delta). \tag{40}$$

Here, PREEEDM stands for the Penalized REE by EDM optimization. We will show how (40) can be computed.

Let us consider a simplified one-dimensional optimization problem, whose solution will eventually give rise to PREEEDM. Let  $B$  denote the interval  $[a, b]$  in  $\Re$  with  $0 \leq a \leq b$ . For given  $\omega \in \Re, \delta > 0$  and  $\beta > 0$ , we aim to compute

$$\text{dcroot}_{B[\omega, \beta, \delta]} := \arg \min_{a \leq x \leq b} q(x) := \frac{1}{2}(x - \omega)^2 + \beta |\sqrt{x} - \delta|. \tag{41}$$

The acronym `dcroot` stands for the root of depressed cubic equation, which will eventually give rise to the solution formula of (41). It suffices to consider the case that matters to us:

$$\beta > 0, \quad \delta > 0 \quad \text{and} \quad a \leq \delta^2 \leq b.$$

Before solving the above problem, we define some notation for convenience

$$\begin{cases} \gamma_{\omega,\beta} := \frac{[\omega + \sqrt{\omega^2 + 2\beta}]^2}{4}, \\ u := \frac{\beta}{4}, \quad v := \frac{\omega}{3} \quad \text{and} \quad \tau := u^2 - v^3 \\ B_- := [a, \delta^2] \quad \text{and} \quad B_+ := [\delta^2, b]. \end{cases} \quad (42)$$

Obviously,  $q(x)$  has a representation of two pieces:

$$q(x) = \begin{cases} q_-(x) := \frac{1}{2}(x - \omega)^2 - \beta\sqrt{x} + \beta\delta & \text{for } x \in B_- \\ q_+(x) := \frac{1}{2}(x - \omega)^2 + \beta\sqrt{x} - \beta\delta & \text{for } x \in B_+ \end{cases}$$

It is noted that  $q_-(x)$  is convex, but  $q_+(x)$  may not necessarily so. We will show that both pieces have a closed-form formula for their respective minimum.

**Proposition 2** Consider the optimization problem:

$$x_-^* := \operatorname{argmin} q_-(x), \quad \text{s.t. } x \in B_-. \quad (43)$$

Define

$$x_{\omega,\beta}^- = \begin{cases} [(u + \sqrt{\tau})^{1/3} + (u - \sqrt{\tau})^{1/3}]^2, & \tau \geq 0, \\ 4v \cos^2 \left[ \frac{1}{3} \arccos(uv^{-3/2}) \right], & \tau < 0. \end{cases} \quad (44)$$

Then (43) has a unique solution  $x_-^*$  given by

$$x_-^* = \Pi_{B_-}(x_{\omega,\beta}^-) := \min\{\delta^2, \max\{a, x_{\omega,\beta}^-\}\} \quad \text{and} \quad x_-^* \geq \min\{\delta^2, 1, \gamma_{\omega,\beta}\}.$$

Now we characterize the optimal solution of  $q_+(x)$  over  $B_+$ .

**Proposition 3** Assume that  $\beta < 4\delta^3$  and consider the optimization problem:

$$x_+^* := \operatorname{argmin} q_+(x), \quad \text{s.t. } x \in B_+. \quad (45)$$

Define

$$x_{\omega,\beta}^+ := \begin{cases} \delta^2 & \text{if } \tau \geq 0 \\ 4v \cos^2 \left[ \frac{1}{3} \arccos(-uv^{-3/2}) \right] & \text{if } \tau < 0, \end{cases} \quad (46)$$

Then  $q_+(x)$  is strictly convex over the interval  $[\delta^2, \infty)$  and

$$x_+^* = \Pi_{B_+}(x_{\omega,\beta}^+) := \max\{\delta^2, \min\{b, x_{\omega,\beta}^+\}\}.$$



Putting together Propositions 2 and 3 gives rise to the optimal solution of (41). The optimal solution is either  $x_-^*$  or  $x_+^*$ , whichever gives a lower functional value of  $q(x)$ . This is the first result of our major theorem below. We note that both Propositions 2 and 3 make use of the convexity of  $q_-(x)$  and  $q_+(x)$  on the respective interval  $[a, \delta^2]$  and  $[\delta^2, b]$ . In fact, we can establish a stronger result that when the two pieces join together, the resulting function  $q(x)$  is still convex on the whole interval  $[a, b]$ . This result is very important to our convergence analysis in the next section and is the second result of the theorem below. A key tool for the proof is Lemma 1.

**Theorem 1** *Let  $B$  denote the interval  $[a, b]$  with  $0 \leq a \leq \delta^2 \leq b$ . We assume  $0 < \beta < 4\delta^3$ . Then, the following hold.*

(i) *The optimal solution of the problem (41) is given by*

$$\text{dcroot}_B[\omega, \beta, \delta] = \begin{cases} x_-^*, & \text{if } q(x_-^*) \leq q(x_+^*), \\ x_+^*, & \text{if } q(x_-^*) > q(x_+^*). \end{cases}$$

(ii) *The function  $q(x)$  is strictly convex on  $[a, b]$ . Consequently, there exists  $\xi \in \partial q(\text{dcroot}_B[\omega, \beta, \delta])$  such that*

$$\xi(x - \text{dcroot}_B[\omega, \beta, \delta]) \geq 0 \quad \text{for any } x \in B.$$

(iii) *Let  $\gamma_{\omega, \beta}$  be defined in (42), then  $\text{dcroot}_B[\omega, \beta, \delta] \geq \min\{\delta^2, b, 1, \gamma_{\omega, \beta}\}$ . We view  $\text{dcroot}_B[\omega, \beta, \delta]$  as a function of  $\omega$ . Suppose  $C > 0$  is an arbitrarily given constant. Then there exists a constant  $\kappa > 0$  such that*

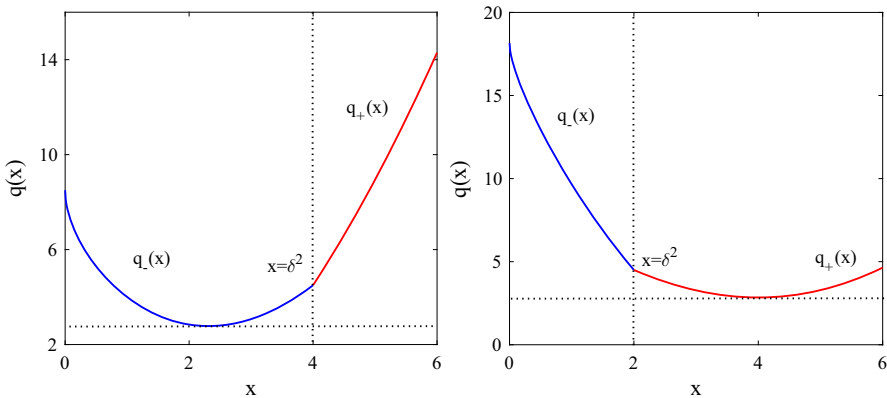
$$\text{dcroot}_B[\omega, \beta, \delta] > \kappa \quad \forall \omega \text{ such that } |\omega| \leq C.$$

**Comment:** The optimal solution  $\text{dcroot}_B[\omega, \beta, \delta]$  is unique, since  $q(x)$  is strictly convex over  $[a, b]$ . However, its location could be within the interval  $[a, \sigma^2]$  or  $[\sigma^2, b]$ , depending on the magnitudes of the parameters ( $\omega, \beta$  and  $\delta$ ) involved. The dependence is illustrated in Fig. 1. We also note that the function  $q(x)$  may not be convex if the condition  $\beta < 4\delta^3$  is violated. □

It now follows from Theorem 1 that the optimal solution  $Z_{ij}^+$  in (39) can be computed by:

$$Z_{ij}^+ = \begin{cases} \text{dcroot}_{[L_{ij}, U_{ij}]}\left[(Z_K)_{ij}, W_{ij}/\rho, \delta_{ij}\right], & W_{ij} > 0 \\ \Pi_{[L_{ij}, U_{ij}]}\left((Z_K)_{ij}\right), & W_{ij} = 0 \end{cases} \tag{47}$$

Consequently,  $Z^+ = \text{PREEEDM}_B(Z_K, W/\rho, \Delta)$  in (40) is well defined and its elements can be computed by (47).



**Fig. 1** Illustration of the convexity of  $q(x) = 0.5(x - \omega)^2 + \beta|\sqrt{x} - \delta|$  over the interval  $[0, 6]$  and  $\beta = 4$ : global minimum happens on  $q_-(x)$  (left) with  $\omega = 1, \delta = 2$  and global minimum happens on  $q_+(x)$  (right) with  $\omega = 5, \delta = \sqrt{2}$

## 5 Algorithm PREEDM and its convergence

With the preparations above, we are ready to state our algorithm. Let  $D^k \in \mathcal{B}$  be the current iterate. We update it by solving the majorization subproblem of the type (38) with  $Z$  replaced by  $D^k$ :

$$D^{k+1} = \arg \min \left\{ f_\rho^k(D) := f(D) + \rho g_m(D, D^k) \right\}, \text{ s.t. } D \in \mathcal{B}, \quad (48)$$

which can be computed by

$$D^{k+1} = \text{REEEDM}_{\mathcal{B}}(-\Pi_{\mathcal{K}_+^n(r)}(-D^k), W/\rho, \Delta). \quad (49)$$

In more detail, we have

$$\begin{aligned} f_\rho^k(D) &= \|W \circ (\sqrt{D} - \Delta)\|_1 + \frac{\rho}{2} \|D + \Pi_{\mathcal{K}_+^n(r)}(-D^k)\|^2 \\ &= \sum_{i,j} \underbrace{\left[ \frac{\rho}{2} (D_{ij} - (Z_K^k)_{ij})^2 + W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| \right]}_{=: f_{ij}^k(D_{ij})} \end{aligned}$$

where  $Z_K^k := -\Pi_{\mathcal{K}_+^n(r)}(-D^k)$ , and the elements of  $D^{k+1}$  are computed as follows:

$$\begin{aligned} D_{ij}^{k+1} &= \operatorname{argmin}_{L_{ij} \leq D_{ij} \leq U_{ij}} \left\{ \frac{1}{2} [D_{ij} - (Z_K^k)_{ij}]^2 + \frac{W_{ij}}{\rho} |\sqrt{D_{ij}} - \delta_{ij}| \right\} \\ &= \begin{cases} \operatorname{dcroot}_{[L_{ij}, U_{ij}]} \left[ (Z_K^k)_{ij}, W_{ij}/\rho, \delta_{ij} \right], & \text{if } W_{ij} > 0 \\ \Pi_{[L_{ij}, U_{ij}]} \left[ (Z_K^k)_{ij} \right], & \text{if } W_{ij} = 0. \end{cases} \quad (50) \end{aligned}$$

Our algorithm PREEEDM is formally stated as follows.

---

**Algorithm 1** PREEEDM 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:**  $D^{k+1} = \text{PREEEDM}_{\mathcal{B}}(-\Pi_{\mathcal{K}_+^n(\rho)}(-D^k), W/\rho, \Delta)$  by (50).
  - 3: **Convergence check:** Set  $k := k + 1$  and go to Step 2 until convergence.
- 

A major obstacle in analysing the convergence for the penalized EDM model (16) is the non-differentiability of the objective function. We need the following two reasonable assumptions:

**Assumption 1** The constrained box  $\mathcal{B}$  is bounded.

**Assumption 2** For  $\Delta$  and  $U$ , we require  $W_{ij} = 0$  if  $\delta_{ij} = 0$  and  $U_{ij} \geq \delta_{ij}^2 \geq L_{ij}$  if  $\delta_{ij} > 0$

Assumption 1 can be easily satisfied (e.g., setting the upper bound to be  $n^2 \max\{\delta_{ij}^2\}$ ). Assumption 2 means that if  $\delta_{ij} = 0$  (e.g., value missing), the corresponding weight  $W_{ij}$  should be 0. This is a common practice in applications. If  $\delta_{ij} > 0$ , then we require  $\delta_{ij}^2$  to be between  $L_{ij}$  and  $U_{ij}$ . We further define a quantity that bounds our penalty parameter  $\rho$  from below:

$$\rho_o := \rho_o(W, \Delta) := \max_{(i,j):W_{ij}>0} \frac{W_{ij}}{4\delta_{ij}^3} \tag{51}$$

Our first result in this section is about the boundedness of the subdifferential of  $f(\cdot)$  along the generated sequence  $\{D^k\}$ .

**Proposition 4** *Suppose Assumptions 1 and 2 hold. Let  $\rho > \rho_o$  and  $\{D^k\}$  be the sequence generated by Algorithm 1. Then the following hold.*

- (i) *There exists a constant  $c_1 > 0$  such that*

$$D_{ij}^k \geq c_1 \quad \text{for all } (i, j) \text{ such that } W_{ij} > 0 \text{ and } k = 1, 2, \dots$$

- (ii) *Let  $\partial f(D)$  denote the subdifferential of  $f(D) = \|W \circ (\sqrt{D} - \Delta)\|_1$ . Then there exists a constant  $c_2 > 0$  such that*

$$\|\Gamma\| \leq c_2 \quad \forall \Gamma \in \partial f(D^k), \quad k = 1, 2, \dots$$

- (iii) *The function  $f_\rho^k(D)$  is convex for all  $k = 1, 2, \dots$ . Moreover, there exists  $\Gamma^{k+1} \in \partial f(D^{k+1})$  such that the first-order optimality condition for (49) is*

$$\left\langle \Gamma^{k+1} + \rho D^{k+1} + \rho \Pi_{\mathcal{K}_+^n(\rho)}(-D^k), D - D^{k+1} \right\rangle \geq 0, \quad \forall D \in \mathcal{B}. \tag{52}$$

Theorem 1(i) ensures that  $D_{ij}^k > 0$  for all  $k = 1, \dots, \dots$ . Hence, we can apply Lemma 1 to each function  $\phi_{\delta_{ij}}(\cdot)$  with  $x = D_{ij}^{k+1}$  and  $y = D_{ij}^k$ . This yields for any  $\zeta_{ij}^{k+1} \in \partial\phi_{\delta_{ij}}(D_{ij}^{k+1})$

$$\phi_{\delta_{ij}}(D_{ij}^{k+1}) - \phi_{\delta_{ij}}(D_{ij}^k) \leq \zeta_{ij}^{k+1}(D_{ij}^{k+1} - D_{ij}^k) + \frac{1}{2} \frac{(D_{ij}^{k+1} - D_{ij}^k)^2}{4\delta_{ij}^3},$$

Multiplying  $W_{ij}$  on both sides and adding those inequalities over  $(i, j)$ , we get

$$f(D^{k+1}) - f(D^k) \leq \langle \Gamma^{k+1}, D^{k+1} - D^k \rangle + \frac{\rho_0}{2} \|D^{k+1} - D^k\|^2, \quad (53)$$

where  $\Gamma_{ij}^{k+1} := W_{ij}\zeta_{ij}^{k+1}$ . We note that the inequality (53) holds for any  $\Gamma^{k+1} \in \partial f(D^{k+1})$ .

**Theorem 2** Let  $\rho > \rho_0$  and  $\{D^k\}$  be the sequence generated by Algorithm 1. Suppose Assumptions 1 and 2 hold.

(i) We have

$$f_\rho(D^{k+1}) - f_\rho(D^k) \leq -\frac{\rho - \rho_0}{2} \|D^{k+1} - D^k\|^2 \text{ for any } k = 0, 1, \dots, \dots$$

Consequently,  $\|D^{k+1} - D^k\| \rightarrow 0$ .

(ii) Let  $\widehat{D}$  be an accumulation point of  $\{D^k\}$ . Then there is  $\widehat{\Gamma} \in \partial f(\widehat{D})$  such that

$$\langle \widehat{\Gamma} + \rho\widehat{D} + \rho\Pi_{\mathcal{K}_+^n(r)}(-\widehat{D}), D - \widehat{D} \rangle \geq 0 \text{ for any } D \in \mathcal{B}. \quad (54)$$

That is,  $\widehat{D}$  is a critical point of the problem (16). Moreover, for a given  $\epsilon > 0$ , if  $D^0 \in \mathcal{K}_+^n(r) \cap \mathcal{B}$  and

$$\rho \geq \rho_\epsilon := \max\{\rho_0, f(D^0)/\epsilon\},$$

then  $\widehat{D}$  is an  $\epsilon$ -critical point of the original problem (11).

(iii) If  $\widehat{D}$  is an isolated accumulation point of the sequence  $\{D^k\}$ , then the whole sequence  $\{D^k\}$  converges to  $\widehat{D}$ .

**Proof** (i) We are going to use the following facts that are stated on  $D^{k+1}$  and  $D^k$ . The first fact is the identity:

$$\|D^{k+1}\|^2 - \|D^k\|^2 = 2\langle D^{k+1} - D^k, D^{k+1} \rangle - \|D^{k+1} - D^k\|^2. \quad (55)$$

The second fact is due to the convexity of  $h(D)$  (see Lemma 2(ii)):

$$h(-D^{k+1}) - h(-D^k) \geq \langle \Pi_{\mathcal{K}_+^n(r)}(-D^k), -D^{k+1} + D^k \rangle. \quad (56)$$

The last fact is that there exists  $\Gamma^{k+1} \in \partial f(D^{k+1})$  such that (52). Those facts yield the following chain of inequalities:

$$\begin{aligned}
 & f_\rho(D^{k+1}) - f_\rho(D^k) \\
 &= f(D^{k+1}) - f(D^k) + \rho g(D^{k+1}) - \rho g(D^k) \\
 &\stackrel{(53)}{\leq} \langle \Gamma^{k+1}, D^{k+1} - D^k \rangle + \frac{\rho_o}{2} \|D^{k+1} - D^k\|^2 + \rho g(D^{k+1}) - \rho g(D^k) \\
 &\stackrel{(30)}{=} \langle \Gamma^{k+1}, D^{k+1} - D^k \rangle + \frac{\rho_o}{2} \|D^{k+1} - D^k\|^2 \\
 &\quad + (\rho/2)(\|D^{k+1}\|^2 - \|D^k\|^2) - \rho[h(-D^{k+1}) - h(-D^k)] \\
 &\stackrel{(55)}{=} \langle \Gamma^{k+1} + \rho D^{k+1}, D^{k+1} - D^k \rangle \\
 &\quad - \frac{\rho - \rho_o}{2} \|D^{k+1} - D^k\|^2 - \rho[h(-D^{k+1}) - h(-D^k)] \\
 &\stackrel{(56)}{\leq} \langle \Gamma^{k+1} + \rho D^{k+1} + \rho \Pi_{\mathcal{K}_+^n(r)}(-D^k), D^{k+1} - D^k \rangle - \frac{\rho - \rho_o}{2} \|D^{k+1} - D^k\|^2 \\
 &\stackrel{(52)}{\leq} -\frac{\rho - \rho_o}{2} \|D^{k+1} - D^k\|^2.
 \end{aligned}$$

This proves that the sequence  $\{F_\rho(D^k)\}$  is non-increasing and it is also bounded below by 0. Taking the limits on both sides yields  $\|D^{k+1} - D^k\| \rightarrow 0$ .

(ii) Suppose  $\widehat{D}$  is the limit of a subsequence  $\{D^{k_\ell}\}$ ,  $\ell = 1, \dots$ . Since we have established in (i) that  $(D^{k_\ell+1} - D^{k_\ell}) \rightarrow 0$ , the sequence  $\{D^{k_\ell+1}\}$  also converges to  $\widehat{D}$ . Furthermore, there exist a sequence of  $\Gamma^{k_\ell+1} \in \partial f(D^{k_\ell+1})$  such that (52) holds. Proposition 4(ii) ensures that there exists a constant  $c_2 > 0$  such that  $\|\Gamma^{k_\ell+1}\| \leq c_2$  for all  $k_\ell$ . Hence, there exists a subsequence of  $\{k_\ell\}$  (we still denote the subsequence by  $\{k_\ell\}$  for simplicity) such that  $\Gamma^{k_\ell+1}$  converges to some  $\widehat{\Gamma} \in \partial f(\widehat{D})$ . Now taking the limits on both sides of (52) on  $\{k_\ell\}$ , we reach the desired inequality (54). We now prove  $\widehat{D}$  is an  $\epsilon$ -critical point of (11). Since we already have (54), we only need to show  $g(\widehat{D}) \leq \epsilon$ . It follows from  $D^0 \in \mathcal{K}_+^n(r) \cap \mathcal{B}$  that

$$\begin{aligned}
 f(D^0) &= f(D^0) + \rho g(D^0) && \text{(because } g(D^0) = 0\text{)} \\
 &\stackrel{(18)}{\geq} f(D^1) + \rho g_m(D^1, D^0) && \text{(because } D^0 \in \mathcal{B}\text{)} \\
 &\stackrel{(31)}{\geq} f(D^1) + \rho g(D^1) = f_\rho(D^1) \\
 &\geq \dots \geq f_\rho(D^k) = f(D^k) + \rho g(D^k). && \text{(because of (i))}
 \end{aligned}$$

Taking the limit on the right-hand side yields

$$f(D^0) \geq f(\widehat{D}) + \rho g(\widehat{D}) \geq \rho g(\widehat{D}),$$

where we used  $f(\widehat{D}) \geq 0$ . Therefore, thanks to  $\rho > \rho_\epsilon$ , it has

$$g(\widehat{D}) \leq f(D^0)/\rho \leq f(D^0)/\rho_\epsilon = \epsilon.$$

(iii) We note that we have proved in (i) that  $(D^{k+1} - D^k) \rightarrow 0$ . The convergence of the whole sequence to  $\widehat{D}$  follows from [26, Proposition 7].  $\square$

## 6 Numerical experiments

In this part, we will conduct extensive numerical experiments of our algorithm PREEDM by using MATLAB (R2014a) on a desktop of 8GB of memory and Inter(R) Core(TM) i5-4570 3.2Ghz CPU, against 6 leading solvers on the problems of sensor network localizations (SNL) in  $\mathbb{R}^2$  ( $r = 2$ ) and Molecular Conformation (MC) in  $\mathbb{R}^3$  ( $r = 3$ ). This section is split into the following parts. The MATLAB package is available at DOI: <https://doi.org/10.5281/zenodo.3343047>. Our implementation of PREEDM was described in Sect. 6.1. We will give a brief explanation how the six benchmark methods were selected in Sect. 6.2. Descriptions of how the test data of SNL and MC were collected and generated, and extensive numerical comparisons are reported in Sect. 6.3.

### 6.1 Implementation

The PREEDM Algorithm 1 is easy to implement. We first address the issue of its stopping criterion that is to be used in Step 3 of Algorithm 1. We monitor two quantities. One is on how close of the current iterate  $D^k$  is to be Euclidean (belonging to  $\mathcal{K}_+^n(r)$ ). This can be computed by using (28) as follows.

$$\begin{aligned} \kappa_{\text{prog}_k} &:= \frac{2g(D^k)}{\|JD^kJ\|^2} = \frac{\|\text{PCA}_r^+(-JD^kJ) + (JD^kJ)\|^2}{\|JD^kJ\|^2} \\ &= 1 - \frac{\sum_{i=1}^r [\lambda_i^2 - (\lambda_i - \max\{\lambda_i, 0\})^2]}{\lambda_1^2 + \dots + \lambda_n^2} \\ &\leq 1, \end{aligned}$$

where  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$  are the eigenvalues of  $(-JD^kJ)$ . The smaller  $\kappa_{\text{prog}_k}$  is, the closer  $D^k$  is to  $\mathcal{K}_+^n(r)$ . The benefit of using  $\kappa_{\text{prog}}$  over  $g(D)$  is that the former is independent of any scaling of  $D$ .

The other quantity is to measure the progress in the functional values  $f_\rho(\cdot)$  by the current iterate  $D^k$ . In theory (see Theorem 2), we should require  $\rho > \rho_0$ , which is defined as (51) and is potentially large. As with the most penalty methods [34, Chp. 17], starting with a very large penalty parameter may degrade the performance of the method (e.g., causing ill-conditioning). We adopt a dynamic updating rule for  $\rho$ . In particular, we choose  $\rho_0 = \frac{\kappa \max_{i,j} \delta_{ij}}{n^{3/2}}$  and update it as

$$\rho_{k+1} = \begin{cases} 1.25\rho_k, & \text{if } \kappa_{\text{prog}_k} > \text{Ktol}, \text{Fprog}_k \leq 0.2\text{Ftol}, \\ 0.75\rho_k, & \text{if } \text{Fprog}_k > \text{Ftol}, \kappa_{\text{prog}_k} \leq 0.2\text{Ktol}, \\ \rho_k, & \text{otherwise,} \end{cases}$$

where

$$F_{\text{prog}_k} := \frac{f_{\rho_{k-1}}(D^{k-1}) - f_{\rho_{k-1}}(D^k)}{1 + \rho_{k-1} + f_{\rho_{k-1}}(D^{k-1})}, \tag{57}$$

and  $F_{\text{tol}} = \ln(\kappa) \times 10^{-4}$  and  $K_{\text{tol}} = 10^{-2}$  with  $\kappa$  being the number of non-zero elements of  $\Delta$ . We terminate PREEEDM when

$$F_{\text{prog}_k} \leq F_{\text{tol}} \quad \text{and} \quad K_{\text{prog}_k} \leq K_{\text{tol}},$$

Since our computation of each iteration is dominated by  $\Pi_{\mathcal{K}_+^n(r)}(-D)$  in the construction of the majorization function  $g_m(\cdot, \cdot)$  in (31), the computational complexity is about  $O(rn^2)$  (we used MATLAB’s built-in function `eigs.m` to compute  $\text{PCA}_r^+(A)$  in (27)). For the problem data input,  $\Delta$ ,  $L$  and  $U$  will be described below. For the initial point, we follow the popular choice used in [44,47]  $\sqrt{D^0} := \widehat{\Delta}$ , where  $\widehat{\Delta}$  is the matrix obtained by the shortest path distances among  $\Delta$ . If  $\Delta$  has no missing values, then  $\widehat{\Delta} = \Delta$ .

### 6.2 Benchmark methods

We select six representative state-of-the-art methods for comparison. They are ADMMSNL [37], ARAP [54], EVEDM (short for EepVecEDM) [12], PC [1], PPAS (short for PPA Semismooth) [24] and SFSDP [27]. Those methods have been shown to be capable of returning satisfactory localization/embedding in many applications. We will compare our method PREEEDM with ADMMSNL, ARAP, EVEDM, PC and SFSDP for Sensor Network Localization (SNL,  $r = 2$ ) problems and with EVEDM, PC, PPAS and SFSDP for Molecular Conformation (MC,  $r = 3$ ) problems since the current implementations of ADMMSNL, ARAP do not support the embedding for  $r \geq 3$ .

We note that ADMMSNL is motivated by [45] and aims to enhance the package `diskRelax` of [45] for the SNL problems ( $r = 2$ ). Both methods are based on the stress minimization (5). As we mentioned before, SMACOF [13,14] has been a very popular method for (5). However, we will not compare it with other methods here since its performance demonstrated in [54,56] was not very satisfactory (e.g., when comparing with ARAP) for either SNL or MC problems. To our best knowledge, PC is the only viable method, whose code is also publicly available for the model (3). We select SFSDP and PPAS because of their high reputation in the field of SDP and quadratic SDP in returning quality localizations and conformations. We note that SFSDP is for the model (4) and the methods PPAS and EVEDM are proposed for the model (6). It is worth mentioning that the MADMM package in [29] is capable of solving the Robust MDS (4) as well as other nonsmooth optimization problems. However, MADMM does not contain the implementation of its listed example: Robust MDS. So we were not able to compare it with ours here. We also implemented the subgradient method of Cayton and Dasgupta [8] for their robust Euclidean embedding. Numerical experiments showed that its performance was similar to PC on our tested

problems. It works well when a large number of the dissimilarities in  $\Delta$  are available and it often performs poorly otherwise. Hence, we omitted it from our reported results.

In our tests, we used all of their default parameters except one or two in order to achieve the best results. In particular, for PC, we terminate it when  $|f(D^{k-1}) - f(D^k)| < 10^{-4} \times f(D^k)$  and set its initial point to be the embedding by cMDS on  $\Delta$ . For SFSDP which is a high-level MATLAB implementation of the SDP approach initiated in [50], we set `pars.SDPsolver = "sedumi"` because it returns the best overall performance, and `pars.objSW = 1` when  $m > r + 1$  and  $= 3$  when  $m = 0$ . We also note that the parameter `pars.minDegree` controls the degree of a graph and thus enhances the strength of the SDP relaxation. Numerical experiments have shown that the larger it is, the more accurate solutions might be generated by SFSDP. However, the computational time shoots up dramatically when it increases even for small  $n$ . Our extensive experiments suggest that its default value (`pars.minDegree = r + 2`) is a balanced choice between solution quality and time of computation for large  $n$ . Hence we choose to use its default setting in our test. For ARAP, in order to speed up the termination, we let `tol = 0.05` and `IterNum = 20` to compute its local neighbour patches. Numerical performance demonstrated that ARAP could yield satisfactory embedding, but would take very long time for some examples with large  $n$ .

### 6.3 Numerical comparison

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

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

where  $\mathbf{x}_i$ 's are the true positions of the sensors or atoms in our test problems and  $\hat{\mathbf{x}}_i$ 's are their corresponding estimates. The  $\hat{\mathbf{x}}_i$ 's were obtained by applying the classical MDS (cMDS) method to the final output of the distance matrix, followed by aligning them to the existing anchors through the well-known Procrustes procedure (see [54], [6, Chp. 20] or [41, Proposition 4.1] for more details). Furthermore, upon obtaining  $\hat{\mathbf{x}}_i$ 's, a heuristic gradient method can be applied to improve their accuracy and it is called the refinement step in [5]. We report `rRMSD` to highlight its contribution. As we will see, all tested methods benefit from this step, but with varying degrees.

The quality of the general performance of each method can be better appreciated through visualizing their key indicators: `RMSD`, `rRMSD`, `rTime` (time for the refinement step) and the `CPU Time` (in s) which is the total time including `rTime`. Hereafter, for all examples, we test 20 randomly generated instances for each case  $(n, m, R, \text{nf})$  in SNL or each case  $(n, R, \text{nf})$  in MC, and record the average results.

#### 6.3.1 Comparison on SNL

SNL has been widely used to test the viability of many existing methods for the stress minimization. In such a problem, we typically have  $m$  anchors (e.g., sensors with



known locations) and the rest sensors need to be located. We will test two types of SNL problems. One has a regular topological layout (Examples 1 and 2 below). The other has an irregular layout (Example 3).

**Example 1** (Square Network with 4 fixed anchors) This example is widely tested since its detailed study in [5]. In the square region  $[-0.5, 0.5]^2$ , 4 anchors  $\mathbf{x}_1 = \mathbf{a}_1, \dots, \mathbf{x}_4 = \mathbf{a}_4$  ( $m = 4$ ) are placed at  $(\pm 0.2, \pm 0.2)$ . The generation of the rest  $(n - m)$  sensors  $(\mathbf{x}_{m+1}, \dots, \mathbf{x}_n)$  follows the uniform distribution over the square region. The noisy  $\Delta$  is usually generated as follows.

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

where  $R$  is known as the radio range,  $\epsilon_{ij}$ 's are independent standard normal random variables, and  $\text{nf}$  is the noise factor (e.g.,  $\text{nf} = 0.1$  was used and it corresponds to 10% noise level). In literature (e.g., [5]), this type of perturbation in  $\delta_{ij}$  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.

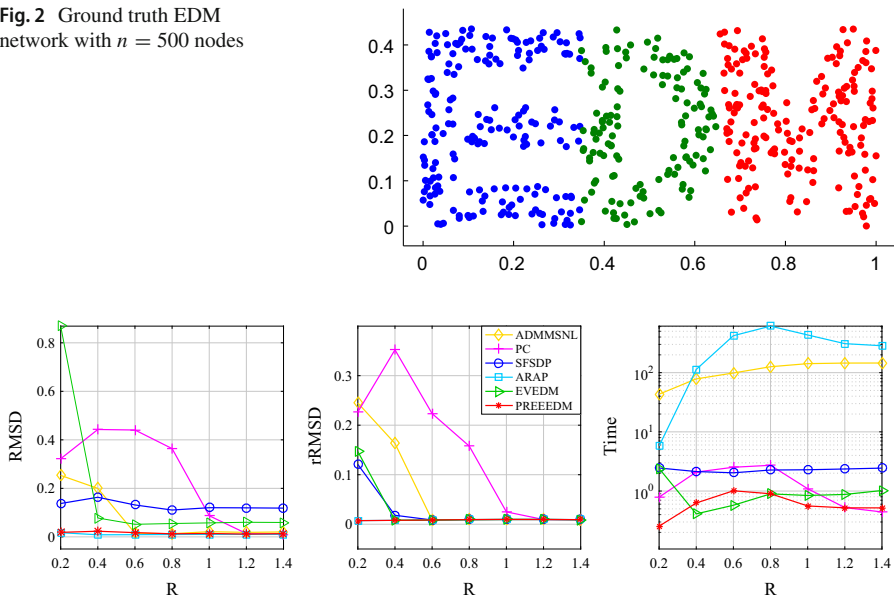
$(i, j)$	$W_{ij}$	$\Delta_{ij}$	$L_{ij}$	$U_{ij}$
$i = j$	0	0	0	0
$i, j \leq m$	0	0	$\ \mathbf{a}_i - \mathbf{a}_j\ ^2$	$\ \mathbf{a}_i - \mathbf{a}_j\ ^2$
$(i, j) \in \mathcal{N}$	1	$\delta_{ij}$	0	$R^2$
otherwise	0	0	$R^2$	$M^2$

**Example 2** (Square Network with  $m$  random anchors) This example also tested in [5] is similar to Example 1 but with randomly generated anchors. The generation of  $n$  points follows the uniform distribution over the square region  $[-0.5, 0.5]^2$ . Then the first  $m$  points are chosen to be anchors and the last  $(n - m)$  points to be sensors. The rest of the data generation is same as in Example 1.

**Example 3** (EDM word network) This problem has a non-regular topology and was first used in [3] to challenge existing methods. In this example,  $n$  points are randomly generated in a region whose shape is similar to the letters “E”, “D” and “M”. The ground truth network is depicted in Fig. 2. We choose the first  $m$  points to be the anchors. The rest of the data generation is same as in Example 1.

(a) *Effect of the radio range  $R$*  It is easy to see that the radio range  $R$  decides the amount of missing dissimilarities among all elements of  $\Delta$ . The smaller  $R$  is, the

**Fig. 2** Ground truth EDM network with  $n = 500$  nodes



**Fig. 3** Average results for Example 1 with  $n = 200$ ,  $m = 4$ ,  $n_f = 0.1$

fewer numbers of  $\delta_{ij}$  are available, leading to more challenging problems. Therefore, we first demonstrate the performance of each method to the radio range  $R$ . For Example 1, we fix  $n = 200$ ,  $m = 4$ ,  $n_f = 0.1$ , and alter the radio range  $R$  among  $\{0.2, 0.4, \dots, 1.4\}$ . The average results were demonstrated in Fig. 3. It can be seen that ARAP and PREEEDM were joint winners in terms of both RMSD and  $r$ RMSD. However, the time used by ARAP was the longest. When  $R$  became bigger than 0.6, ADMMSNL, SFSDP and EVEDM produced similar  $r$ RMSD as ARAP and PREEEDM, while the time consumed by ADMMSNL was significantly larger than that by SFSDP, EVEDM and PREEEDM. By contrast, PC only worked well when  $R \geq 1$ .

Next we test a number of instances with larger size  $n \in \{300, 500, 1000, 2000\}$ . For Example 1, the average results were recorded in Table 1. When  $R = \sqrt{2}$  under which no dissimilarities were missing because Example 1 was generated in a unit region, PC, ARAP and PREEEDM produced the better RMSD (almost in the order of  $10^{-3}$ ). But with the refinement step, all methods led to similar  $r$ RMSD. This meant SFSDP and EVEDM benefited a lot from the refinement step. For the computational speed, PREEEDM outperformed others, followed by PC, EVEDM and SFSDP. By contrast, ARAP consumed too much time even for  $n = 500$ . When  $R = 0.2$ , the picture was significantly different since there were large amounts of unavailable dissimilarities in  $\Delta$ . Basically, ADMMSNL, PC and SFSDP failed to localize even with the refinement due to undesirable RMSD and  $r$ RMSD (both in the order of  $10^{-1}$ ). Clearly, ARAP and PREEEDM produced the best RMSD and  $r$ RMSD, and EVEDM got comparable  $r$ RMSD but inaccurate RMSD. In terms of the computational speed, EVEDM and PREEEDM were very fast, consuming about 30s to solve problems with  $n = 2000$  nodes. By contrast, ARAP still was the slowest, followed by ADMMSNL and PC.

**Table 1** Comparison of six methods for Example 1 with  $m = 4, n_f = 0.1$

$n$			ADMMSNL	PC	SFSDP	ARAP	EVEDM	PREEDM
$R = \sqrt{2}$	300	RMSD	2.07e-2	8.31e-3	1.21e-1	1.01e-2	5.95e-2	1.11e-2
		rRMSD	7.82e-3	7.86e-3	7.89e-3	7.96e-3	7.93e-3	7.80e-3
		rTime	3.63	0.66	3.87	0.94	3.35	1.06
		Time	348.13	1.36	6.79	503.86	3.84	1.36
	500	RMSD	-	6.11e-3	1.19e-1	7.51e-3	5.87e-2	8.46e-3
		rRMSD	-	5.94e-3	5.96e-3	6.04e-3	6.70e-3	6.11e-3
		rTime	-	1.37	14.79	3.26	13.35	3.92
		Time	-	3.83	20.22	2479.8	14.44	4.41
	1000	RMSD	-	4.46e-3	1.25e-1	-	5.81e-2	6.59e-3
		rRMSD	-	4.15e-3	7.34e-3	-	6.53e-3	4.59e-3
		rTime	-	3.51	83.96	-	68.06	9.75
		Time	-	23.05	103.29	-	71.52	10.85
	2000	RMSD	-	3.30e-3	1.20e-1	-	5.92e-2	4.57e-3
		rRMSD	-	3.10e-3	7.82e-3	-	1.24e-2	3.37e-3
		rTime	-	12.74	282.88	-	258.97	13.04
		Time	-	143.41	398.87	-	271.91	18.49
$R = 0.2$	300	RMSD	3.48e-1	4.42e-1	1.93e-1	4.02e-2	6.81e+1	1.88e-2
		rRMSD	3.33e-1	3.12e-1	1.73e-1	6.83e-3	1.72e-1	6.84e-3
		rTime	0.50	0.44	0.41	0.36	0.48	0.36
		Time	84.19	2.37	3.45	24.11	0.56	0.47
	500	RMSD	3.53e-1	4.30e-1	2.02e-1	1.95e-2	1.52e-1	1.77e-2
		rRMSD	3.35e-1	3.11e-1	1.80e-1	5.57e-3	5.59e-2	5.51e-3
		rTime	1.11	1.15	1.06	0.80	1.11	0.92
		Time	156.76	5.50	6.90	161.04	1.30	1.23
	1000	RMSD	3.62e-1	4.54e-1	1.79e-1	9.96e-3	7.21e-2	1.46e-2
		rRMSD	3.44e-1	3.16e-1	1.28e-1	3.57e-3	4.06e-3	3.83e-3
		rTime	5.58	5.58	5.25	1.69	5.16	3.76
		Time	450.03	24.82	19.90	2833.5	6.00	5.86
	2000	RMSD	3.71e-1	4.35e-1	1.80e-1	-	5.92e-2	1.37e-2
		rRMSD	3.51e-1	3.63e-1	8.29e-2	-	3.53e-3	3.29e-3
		rTime	40.40	40.65	37.94	-	24.72	4.58
		Time	1255.1	171.01	77.03	-	32.31	17.51

Now we test those methods for the irregular network in Example 3. The average results were recorded in Table 2. We note that this example was generated in the region  $[0, 1] \times [0, 0.5]$  as presented in Fig. 2. It implies that no dissimilarities in  $\Delta$  were missing when  $R = \sqrt{1.25}$  while a large number of dissimilarities in  $\Delta$  were missing when  $R = 0.1$ . When  $R = \sqrt{1.25}$ , it can be clearly seen that SFSDP and EVEDM failed to localize before the refinement step due to their large RMSD (in the order of  $10^{-1}$ ), whilst the rest four methods succeeded. However, they all achieved a

**Table 2** Comparisons of six methods for Example 3 with  $m = 10$ ,  $n_f = 0.1$ 

		$n$	ADMMSNL	PC	SFSDP	ARAP	EVEDM	PREEEDM	
$R = \sqrt{1.25}$	300	RMSD	4.02e-2	5.33e-3	1.45e-1	1.27e-2	1.62e-1	9.26e-3	
		rRMSD	5.12e-3	5.14e-3	5.11e-3	5.12e-3	5.09e-3	5.15e-3	
		rTime	3.28	0.66	3.71	1.69	3.94	1.44	
		Time	346.98	2.00	6.74	553.87	4.42	1.87	
	500	RMSD	–	4.09e-3	1.07e-1	8.50e-3	1.63e-1	7.15e-3	
		rRMSD	–	4.03e-3	4.04e-3	4.05e-3	1.02e-1	4.15e-3	
		rTime	–	2.68	17.28	7.07	17.39	3.12	
		Time	–	7.24	23.44	2556.3	18.89	5.13	
	1000	RMSD	–	3.07e-3	1.12e-1	–	1.28e-1	5.05e-3	
		rRMSD	–	2.98e-3	3.50e-3	–	4.15e-3	3.15e-3	
		rTime	–	10.35	119.79	–	122.12	15.73	
		Time	–	43.69	140.66	–	125.46	20.11	
	2000	RMSD	–	2.36e-3	1.15e-1	–	1.03e-1	3.75e-3	
		rRMSD	–	2.28e-3	7.34e-3	–	7.78e-3	2.26e-3	
		rTime	–	13.43	537.70	–	489.30	10.59	
		Time	–	238.31	659.71	–	500.72	20.25	
	$R = 0.1$	300	RMSD	1.81e-1	3.77e-1	8.64e-2	8.19e-2	4.06e-1	3.97e-2
			rRMSD	1.43e-1	1.24e-1	6.69e-2	5.38e-2	1.17e-1	8.21e-3
			rTime	0.27	0.22	0.21	0.21	0.22	0.21
			Time	76.57	1.21	3.24	7.24	3.41	0.32
500		RMSD	9.73e-2	3.30e-1	5.08e-2	5.77e-2	2.16e-1	3.63e-2	
		rRMSD	7.82e-2	1.15e-1	3.48e-2	3.08e-2	9.78e-2	3.63e-3	
		rTime	0.67	0.63	0.60	0.58	0.61	0.50	
		Time	148.06	3.63	6.41	50.81	2.07	1.85	
1000		RMSD	2.26e-1	3.29e-1	4.80e-2	8.75e-2	2.22e-1	5.01e-2	
		rRMSD	1.01e-1	1.21e-1	9.15e-3	4.55e-2	1.02e-1	2.95e-3	
		rTime	2.74	2.66	2.67	2.58	2.61	2.60	
		Time	353.07	18.01	17.10	842.43	3.22	4.24	
2000		RMSD	1.66e-1	3.29e-1	8.21e-2	–	1.02e-1	5.73e-2	
		rRMSD	1.22e-1	1.53e-1	7.10e-2	–	3.64e-2	4.97e-3	
		rTime	23.22	23.30	23.06	–	23.12	17.99	
		Time	887.30	108.81	62.65	–	26.12	29.89	

similar  $r$ RMSD after the refinement except for EVEDM under the case  $n = 500$ . Still, PREEEDM ran the fastest and ARAP came the last, (5.13s vs. 2556.3s when  $n = 500$ ). Their performances for the case  $R = 0.1$  are quite contrasting. PREEEDM generated the most accurate RMSD and  $r$ RMSD (in the order of  $10^{-3}$ ) whilst the results of the rest methods were only in the order of  $10^{-2}$ . Obviously, ADMMSNL, PC and EVEDM failed to localize. Compared with the other methods, EVEDM and PREEEDM were joint

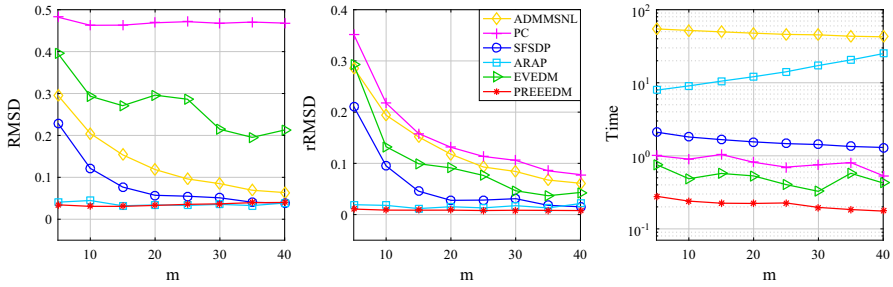


Fig. 4 Average results for Example 2 with  $n = 200$ ,  $R = 0.2$ ,  $nf = 0.1$

winner in terms of the computational speed, only using 30s when  $n = 2000$  (a larger scale network). But we should mention that EVEDM failed to localize.

(b) *Effect of the number of anchors  $m$*  As one would expect, more anchors would lead to more information available, and hence lead to easier localization. In this part, we demonstrate the degree of the effect of the varying anchors' numbers on the 6 methods. For Example 2, we fix  $n = 200$ ,  $R = 0.2$ ,  $nf = 0.1$  with choosing  $m$  from  $\{5, 10, \dots, 40\}$ . As illustrated in Fig. 4, ARAP and PREEEDM were again joint winners in terms of RMSD and rRMSD. And rRMSD produced by the rest methods declined rapidly as more anchors being used. Moreover, PREEEDM was the fastest, followed by EVEDM, PC and SFSDP, whilst ADMMSNL and ARAP were quite slow.

For Example 3 with fixed  $n = 500$ ,  $R = 0.1$ ,  $nf = 0.1$ , we test it under  $m \in \{10, 30, 50\}$ . As depicted in Fig. 5, ARAP and PREEEDM were always capable of capturing the shape of letters 'E', 'D' and 'M' that was similar to Fig. 2. By contrast, SFSDP and EVEDM derived desirable outline of three letters only when  $m = 50$ , and the localization quality of both ADMMSNL and PC improved along with the increasing  $m$  but still with a deformed shape of letter 'M'.

Finally we test a number of instances of Example 2 with choosing  $n \in \{300, 500, 1000, 2000\}$  and  $m \in \{10, 50\}$ . The average results were recorded in Table 3. When  $m = 10$ , ADMMSNL and PC produced undesirable RMSD and rRMSD (both in the order of  $10^{-1}$ ). SFSDP benefited greatly from the refinement because it generated relatively inaccurate RMSD. By contrast the rest three methods enjoyed the successful localization except for EVEDM under the case  $n = 300$ . With regard to the computational speed, EVEDM and PREEEDM were the fastest, followed by SFSDP, PC, ADMMSNL and ARAP. When  $m = 50$ , more information was known, the results were better than before, especially for the methods ADMMSNL and PC. But PC still heavily relied on the refinement step to get the satisfactory localization. The rest five methods produced a satisfactory localization with varying degree of accuracy. It is encouraging to see that PREEEDM produced the most accurate rRMSD for all cases. The comparison of the computational speed is similar to the case of  $m = 10$ . We repeated the test for Example 3 and the average results were recorded in Table 4, where we observed a similar performance of the six methods as for Example 2. We omit the details.

(c) *Effect of the noise factor  $nf$*  To see the dependence of the performance of each method on the noise factor, we first test Example 3 with fixing  $n = 200$ ,  $m = 10$ ,  $R =$

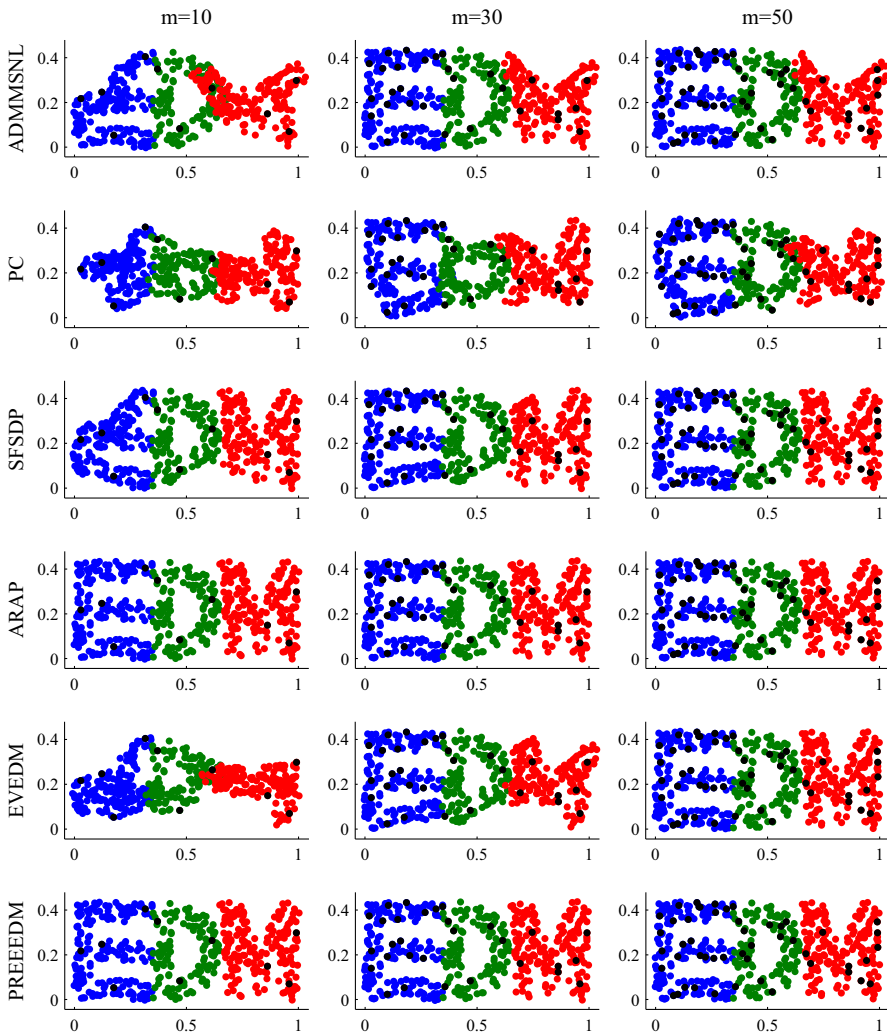


Fig. 5 Localization for Example 3 with  $n = 500$ ,  $R = 0.1$ ,  $nf = 0.1$

0.3 and varying the noise factor  $nf \in \{0.1, 0.2, \dots, 0.7\}$ . As shown in Fig. 6, in terms of RMSD it can be seen that ARAP got the smallest ones, whilst EVEDM and PC obtained the worst ones. The line of ADMMSNL dropped down from  $0.1 \leq nf \leq 0.3$  and then ascended. By contrast the line of PREEEDM reached the peak at  $nf = 0.3$  but declined afterwards and gradually approached to RMSD of ARAP. However, after the refinement step, ARAP, SFSDP and PREEEDM all derived a similar  $r$ RMSD while the other three methods produced undesirable ones. Apparently, EVEDM was indeed the fastest (yet with the worst  $r$ RMSD), followed by PC, SFSDP and PREEEDM. Again, ARAP and ADMMSNL were quite slow.

**Table 3** Comparisons of six methods for Example 2 with  $R = 0.2$ ,  $\text{nf} = 0.1$

$n$			ADMMSNL	PC	SFSDP	ARAP	EVEDM	PREEEDM
$m = 10$	300	RMSD	2.56e-1	4.59e-1	1.34e-1	2.60e-2	2.72e-1	3.99e-2
		rRMSD	2.49e-1	2.43e-1	7.19e-2	6.71e-3	1.44e-1	6.69e-3
		rTime	0.40	0.43	0.36	0.26	0.39	0.28
		Time	81.62	2.02	3.18	24.92	0.47	0.40
	500	RMSD	1.86e-1	4.41e-1	9.70e-2	2.42e-2	8.62e-2	3.29e-2
		rRMSD	1.82e-1	2.07e-1	4.99e-2	5.07e-3	5.05e-3	5.07e-3
		rTime	0.81	1.30	0.93	0.69	0.84	0.64
		Time	163.55	4.70	6.67	170.82	1.04	1.02
	1000	RMSD	1.82e-1	4.39e-1	9.93e-2	2.71e-2	6.88e-2	3.95e-2
		rRMSD	1.60e-1	1.96e-1	2.92e-2	3.21e-3	3.20e-3	3.63e-3
		rTime	4.79	5.53	4.38	3.90	4.66	3.71
		Time	441.08	24.70	18.64	2861.9	5.47	5.88
2000	RMSD	2.17e-1	4.39e-1	1.30e-1	-	6.08e-2	5.03e-2	
	rRMSD	1.87e-1	2.54e-1	6.88e-2	-	2.64e-3	2.82e-3	
	rTime	39.22	39.32	36.29	-	33.85	14.43	
	Time	1251.07	170.55	75.29	-	37.33	28.95	
$m = 50$	300	RMSD	3.19e-2	4.49e-1	3.09e-2	5.30e-2	1.09e-1	5.07e-2
		rRMSD	3.10e-2	4.39e-2	1.13e-2	1.26e-2	1.84e-2	5.78e-3
		rTime	0.12	0.20	0.09	0.09	0.11	0.09
		Time	74.71	1.44	2.41	48.83	0.22	0.25
	500	RMSD	2.80e-2	4.60e-1	3.54e-2	4.39e-2	5.10e-2	6.09e-2
		rRMSD	2.68e-2	4.93e-2	6.77e-3	4.42e-3	5.61e-3	4.42e-3
		rTime	0.24	0.50	0.21	0.21	0.19	0.19
		Time	144.93	4.25	4.67	232.14	0.46	0.72
	1000	RMSD	1.91e-2	4.57e-1	3.21e-2	2.27e-2	5.06e-2	5.99e-2
		rRMSD	1.27e-2	3.75e-2	4.76e-3	2.94e-3	2.94e-3	2.94e-3
		rTime	1.05	2.52	1.10	1.05	1.01	1.12
		Time	406.88	20.29	12.48	3150.6	1.86	4.02
2000	RMSD	2.17e-2	4.47e-1	3.63e-2	-	5.16e-2	4.72e-2	
	rRMSD	6.13e-3	2.78e-2	3.52e-3	-	2.06e-3	2.06e-3	
	rTime	11.89	25.95	10.43	-	8.80	7.71	
	Time	1171.22	156.45	40.45	-	11.15	22.53	

Next, we test Example 2 with a moderate size (for the visualization purpose in Fig. 7)  $n = 200$ ,  $m = 4$  and  $R = 0.3$  and varying  $\text{nf} \in \{0.1, 0.3, 0.5\}$ . The actual embedding by each method was shown in Fig. 7, where the four anchors were plotted in green square and  $\hat{\mathbf{x}}_i$  in pink points were jointed to its ground truth location (blue circle). It can be clearly seen that ARAP and PREEEDM were quite robust to the noise factor since their localization matched the ground truth well. EVEDM failed to locate

**Table 4** Comparisons of six methods for Example 3 with  $R = 0.1$ ,  $nf = 0.1$ 

$n$			ADMMSNL	PC	SFSDP	ARAP	EVEDM	PREEEDM
$m = 10$	300	RMSD	1.80e-1	3.77e-1	8.86e-2	7.97e-2	3.88e-1	4.05e-2
		rRMSD	1.48e-1	1.24e-1	6.24e-2	4.51e-2	1.19e-1	6.25e-3
		rTime	0.28	0.22	0.21	0.22	0.23	0.21
		Time	76.83	1.12	3.00	7.22	5.92	0.41
	500	RMSD	9.71e-2	3.30e-1	4.97e-2	5.97e-2	2.10e-1	3.81e-2
		rRMSD	8.07e-2	9.98e-2	3.21e-2	3.38e-2	1.04e-1	3.91e-3
		rTime	0.68	0.59	0.60	0.59	0.59	0.47
		Time	142.20	3.37	6.35	48.98	2.10	0.98
	1000	RMSD	2.30e-1	3.29e-1	4.98e-2	8.86e-2	2.24e-1	4.93e-2
		rRMSD	1.02e-1	1.18e-1	2.33e-2	4.53e-2	1.07e-1	2.37e-3
		rTime	2.92	2.84	2.82	2.80	2.85	2.84
		Time	354.77	18.59	17.43	838.48	3.85	4.55
2000	RMSD	1.66e-1	3.29e-1	7.96e-2	-	1.03e-1	5.72e-2	
	rRMSD	1.22e-1	1.52e-1	6.92e-2	-	4.25e-2	4.89e-3	
	rTime	23.24	23.17	23.08	-	23.05	13.07	
	Time	882.40	98.20	66.58	-	26.21	24.17	
$m = 50$	300	RMSD	2.24e-2	3.34e-1	1.72e-2	4.04e-2	2.22e-1	3.35e-2
		rRMSD	2.13e-2	2.44e-2	8.36e-3	1.19e-2	2.11e-2	4.34e-3
		rTime	0.22	0.21	0.11	0.13	0.22	0.14
		Time	69.59	0.55	2.30	26.37	0.30	0.29
	500	RMSD	2.53e-2	3.41e-1	2.50e-2	4.95e-2	6.14e-2	4.18e-2
		rRMSD	2.46e-2	3.67e-2	6.64e-3	4.89e-3	2.97e-3	2.96e-3
		rTime	0.34	0.58	0.42	0.38	0.43	0.37
		Time	130.58	2.96	5.07	83.65	0.60	0.79
	1000	RMSD	1.97e-2	3.30e-1	1.94e-2	4.82e-2	5.60e-2	4.90e-2
		rRMSD	1.89e-2	3.07e-2	1.95e-3	3.50e-3	1.96e-3	1.96e-3
		rTime	1.02	2.78	1.04	1.49	1.36	1.23
		Time	314.30	14.31	12.87	947.29	1.97	3.48
2000	RMSD	4.64e-3	3.28e-1	2.10e-2	-	6.30e-2	5.72e-2	
	rRMSD	1.32e-3	3.01e-2	1.32e-3	-	1.32e-3	1.32e-3	
	rTime	14.59	23.23	13.03	-	13.87	8.28	
	Time	811.23	99.02	44.12	-	15.99	19.45	

when  $nf = 0.5$ . By contrast, SFSDP generated worse results when  $nf$  got bigger, and ADMMSNL and PC failed to localize for all cases.

Finally, we test Example 1 with larger sizes  $n \in \{300, 500, 1000, 2000\}$  and fixed  $m = 4$ ,  $R = 0.3$ . The average results were recorded in Table 5. When  $nf = 0.1$ , ADMMSNL and PC failed to render accurate embedding. Compared with ARAP, EVEDM and PREEEDM, SFSDP generated larger RMSD and rRMSD. Again, EVEDM and PREEEDM ran faster than ARAP. When  $nf = 0.7$ , the results were different. ARAP



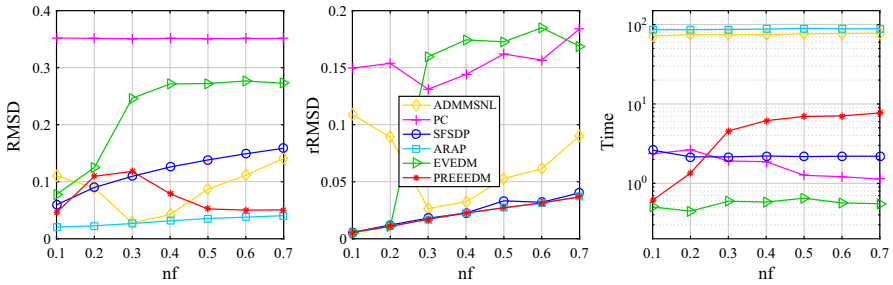


Fig. 6 Average results for Example 3 with  $n = 200, m = 10, R = 0.3$

and PREEEDM were still able to produce high-quality RMSD and rRMSD. However, the former took extremely long time (16617 vs. 83 s). By contrast, ADMMSNL and PC again failed to reconstruct the network. Furthermore, EVEDM got large RMSD but comparable rRMSD when  $n \leq 1000$ , but it failed when  $n = 2000$ .

### 6.3.2 Comparison on MC

MC has long been an important application of EDM optimization [2,21,33]. We will test two types of MCs respectively from an artificial data set and a real data set in Protein Data Bank (PDB) [4]. For the former, we adopt the rule of generating data from [2,33]. For the latter, we used the real data of 12 molecules derived from 12 structures of proteins from PDB. They are 1GM2, 304D, 1PBM, 2MSJ, 1AU6, 1LFB, 104D, 1PHT, 1POA, 1AX8, 1RGS, 2CLJ. They provide a good set of test problems in terms of the size  $n$ , which ranges from a few hundreds to a few thousands (the smallest  $n = 166$  for 1GM and the largest  $n = 4189$  for 2CLJ). The distance information was obtained in a realistic way as done in [24].

**Example 4** (Artificial data) As described in [2,33], the artificial molecule has  $n = s^3$  atoms  $(\mathbf{x}_1, \dots, \mathbf{x}_n)$  located in the three-dimensional lattice

$$\{(i_1, i_2, i_3) : i_1, i_2, i_3 = 0, 1, \dots, s - 1\}$$

for some integer  $s \geq 1$ , i.e.,  $\mathbf{x}_i = (i_1, i_2, i_3)^T$ . We define  $\mathcal{N}_x$  for the index set on which  $\delta_{ij}$  are available as:

$$\mathcal{N}_x := \{(i, j) : |p(\mathbf{x}_i) - p(\mathbf{x}_j)| \leq R\} \tag{58}$$

where  $p(\mathbf{x}_i) := 1 + (1, s, s^2)^T \mathbf{x}_i = 1 + i_1 + si_2 + s^2i_3$  and  $R$  is a given constant (e.g.,  $R = s^2$ ). The corresponding dissimilarity matrix  $\Delta$ , weight matrix  $W$  and the lower and upper bound matrices  $L$  and  $U$  are given as in the table below. Here the generation of  $\delta_{ij}$  is the same as Example 1.

**Example 5** (Real PDB data) Each molecule comprises  $n$  atoms  $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  in  $\mathfrak{R}^3$  and its distance information is collected as follows. If the Euclidean distance between two of the atoms is less than  $R$ , the distance is chosen; otherwise no distance information

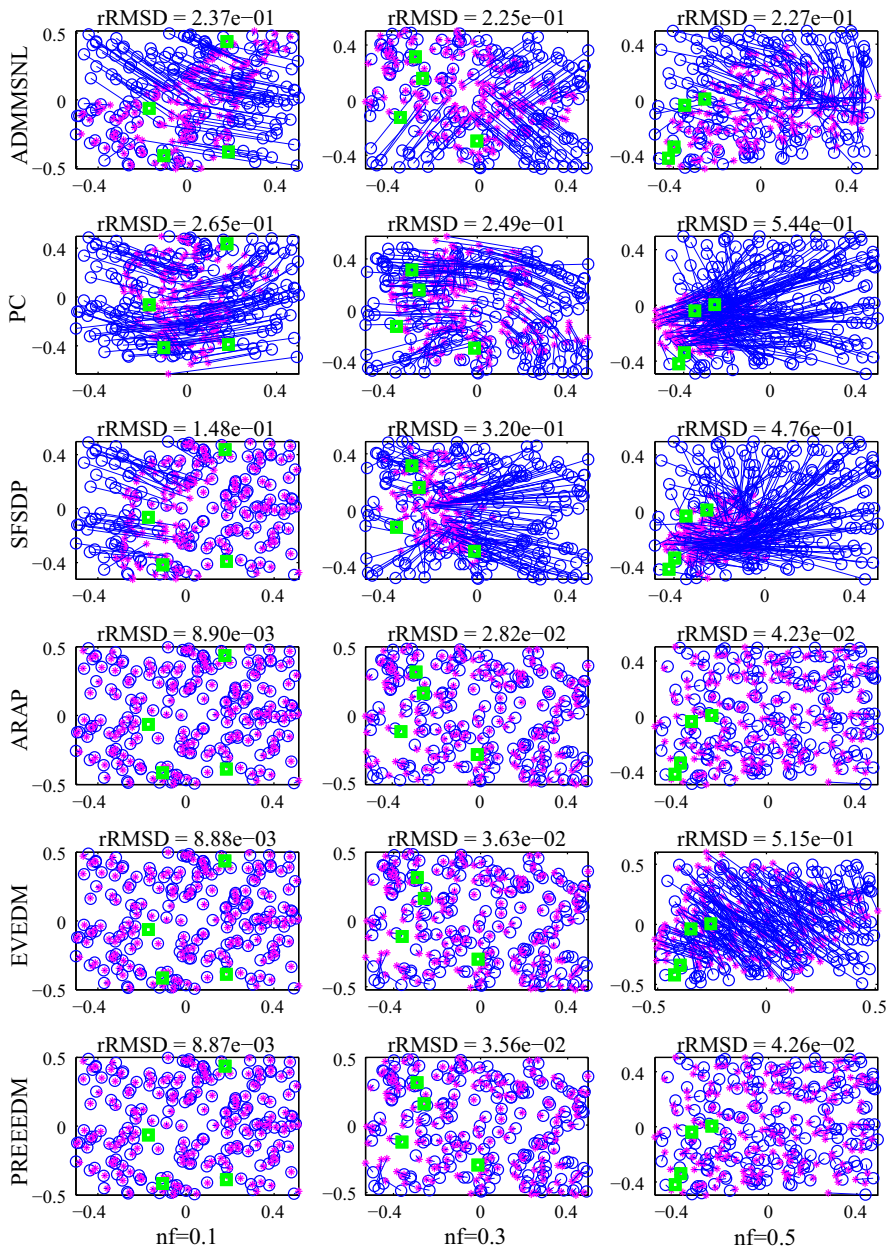


Fig. 7 Localization for Example 2 with  $n = 200$ ,  $m = 4$ ,  $R = 0.3$

about this pair is known. For example,  $R = 6\text{\AA}$  ( $1\text{\AA} = 10^{-8}\text{cm}$ ) is nearly the maximal distance that the nuclear magnetic resonance (NMR) experiment can measure between two atoms. For realistic molecular conformation problems, not all the distances below

$(i, j)$	$W_{ij}$	$\Delta_{ij}$	$L_{ij}$	$U_{ij}$
$i = j$	0	0	0	0
$(i, j) \in \mathcal{N}_x$	1	$\delta_{ij}$	1	$\max_{(i,j) \in \mathcal{N}_x} \ \mathbf{x}_i - \mathbf{x}_j\ ^2$
otherwise	0	0	1	$3(s-1)^2$

$R$  are known from NMR experiments, so one may obtain  $c\%$  (e.g.,  $c = 50\%$ ) of all the distances below  $R$ . Denote  $\mathcal{N}_x$  the set formed by indices of those measured distances. Moreover, the distances in  $\mathcal{N}_x$  can not be exactly measured. Instead, only lower bounds  $\ell_{ij}$  and upper bounds  $u_{ij}$  are provided, that is for  $(i, j) \in \mathcal{N}_x$ ,

$$\ell_{ij} = \max \{1, (1 - |\epsilon_{ij}|)\|\mathbf{x}_i - \mathbf{x}_j\|\}, \quad u_{ij} = (1 + |\epsilon_{ij}|)\|\mathbf{x}_i - \mathbf{x}_j\|.$$

where  $\epsilon_{ij}, \varepsilon_{ij} \sim N(0, \text{nf}^2 \times \pi/2)$  are independent normal random variables. In our test, we set the noise factor  $\text{nf} = 0.1$  and the parameters  $W, \Delta, L, U \in \mathcal{S}^n$  are given as in the table below, where  $M > 0$  is the upper bound (e.g.,  $M := n \max_{ij} \Delta_{ij}$ ).

$(i, j)$	$W_{ij}$	$\Delta_{ij}$	$L_{ij}$	$U_{ij}$
$i = j$	0	0	0	0
$(i, j) \in \mathcal{N}_x$	1	$(\ell_{ij} + u_{ij})/2$	$\ell_{ij}^2$	$u_{ij}^2$
otherwise	0	0	0	$M^2$

As we mentioned before, the current implementations of ADMMSNL, ARAP do not support the embedding for  $r \geq 3$  and thus are removed in the following comparison, where the method PPAS will be added. The main reason for adding PPAS is that it is particularly suitable and credible for the MC problems [24,25].

(d) *Test on Example 4.* To see the performance of each method on this problem, we first test it with fixing  $s = 6$  ( $n = 6^3$ ),  $\text{nf} = 0.1$  but varying  $R \in \{36, 38, \dots, 48\}$ . We note that the percentage of available dissimilarities increased from 32.47 to 39.87% with  $R$  increasing from 36 to 48, making the problem become ‘easier’ for conformation. The Average results were recorded in Fig. 8. Clearly, PREEEDM and PPAS outperformed the other three methods in terms of RMSD and  $r$ RMSD. The former generated the best RMSD when  $R \geq 42$  while the latter got the best RMSD when  $R \leq 42$ , but they both obtained similar  $r$ RMSD. As for the computational speed, PREEEDM ran much faster than PPAS. By contrast, the other three methods failed to produce accurate embeddings due to the worse RMSD and  $r$ RMSD. Notice that the refinement would not always make the final results better. For instance,  $r$ RMSD yielded by SFSDP was bigger than RMSD for each  $s$ .

We then test the example with fixing  $s = 6$  ( $n = 6^3$ ),  $R = s^2$  and varying  $\text{nf} \in \{0.1, 0.2, \dots, 0.5\}$ . As illustrated in Fig. 9, in terms of RMSD and  $r$ RMSD, it can be clearly seen that PREEEDM and PPAS were the joint winners. In particular, our method rendered the best RMSD when  $\text{nf} \geq 0.2$  and also ran much faster than PPAS.

**Table 5** Comparisons of six methods for Example 1 with  $m = 4$ ,  $R = 0.3$ 

$n$			ADMMNL	PC	SFSDP	ARAP	EVEDM	PREEDM
nf = 0.1	300	RMSD	3.16e-1	4.46e-1	1.74e-1	1.03e-2	6.58e-2	1.64e-2
		rRMSD	2.84e-1	3.10e-1	9.63e-2	6.62e-3	6.55e-3	6.57e-3
		rTime	0.75	0.71	0.62	0.31	0.43	0.34
		Time	101.07	3.09	4.39	117.33	0.55	0.57
	500	RMSD	2.96e-1	4.02e-1	1.59e-1	6.73e-3	5.25e-2	1.25e-2
		rRMSD	2.14e-1	2.81e-1	6.05e-2	4.59e-3	4.64e-3	4.73e-3
		rTime	1.68	1.74	1.50	0.50	1.03	0.81
		Time	182.09	9.10	6.16	769.39	1.32	1.48
	1000	RMSD	3.47e-1	4.77e-1	1.83e-1	5.35e-3	5.57e-2	1.13e-2
		rRMSD	2.71e-1	2.52e-1	5.52e-2	3.63e-3	3.65e-3	3.49e-3
		rTime	14.89	15.11	12.00	1.97	10.32	5.22
		Time	601.92	56.65	24.49	15686.4	11.63	10.03
2000	RMSD	-	4.47e-1	1.81e-1	-	5.53e-2	1.16e-2	
	rRMSD	-	4.25e-1	2.21e-2	-	3.32e-3	3.12e-3	
	rTime	-	82.17	82.35	-	45.12	5.85	
	Time	-	470.32	122.45	-	49.18	34.68	
nf = 0.7	300	RMSD	2.80e-1	4.36e-1	3.27e-1	6.70e-2	2.08e-1	5.04e-2
		rRMSD	2.31e-1	3.60e-1	2.47e-1	5.48e-2	6.10e-2	4.92e-2
		rTime	0.75	0.83	0.83	0.29	0.47	0.38
		Time	107.48	1.74	83.73	123.18	0.59	7.49
	500	RMSD	2.64e-1	4.53e-1	-	4.24e-2	1.76e-1	3.73e-2
		rRMSD	1.94e-1	3.59e-1	-	3.52e-2	3.47e-2	3.23e-2
		rTime	1.66	1.88	-	0.47	0.87	0.67
		Time	177.24	5.13	-	844.74	1.31	20.15
	1000	RMSD	2.21e-1	4.52e-1	-	2.84e-2	1.45e-1	2.79e-2
		rRMSD	9.69e-2	3.26e-1	-	2.47e-2	2.93e-2	2.40e-2
		rTime	9.83	15.69	-	1.41	7.78	2.54
		Time	599.30	41.55	-	16617.1	9.16	83.64
2000	RMSD	-	4.51e-1	-	-	2.26e-1	2.13e-2	
	rRMSD	-	3.35e-1	-	-	1.23e-1	1.52e-2	
	rTime	-	92.45	-	-	58.25	3.79	
	Time	-	274.90	-	-	62.52	303.43	

Obviously, the other three methods again failed to obtain desirable RMSD and rRMSD irrelevant of the time they used.

Finally, for larger size problems with  $n = s^3$  and  $s \in \{7, 8, \dots, 13\}$ , the average results were presented in Fig. 10, where we omitted the results by PPAAS for  $s > 10$  because it took too much time to terminate. It is worth mentioning that the percentage of the available dissimilarities over all elements of  $\Delta$  decreases from 26.78% to 14.83% when  $s$  increasing from 7 to 13, making the problems more and more challenging.

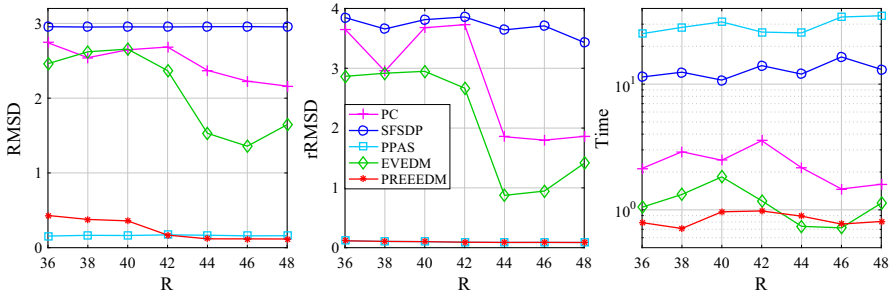


Fig. 8 Average results for Example 4 with  $s = 6$ ,  $nf = 0.1$

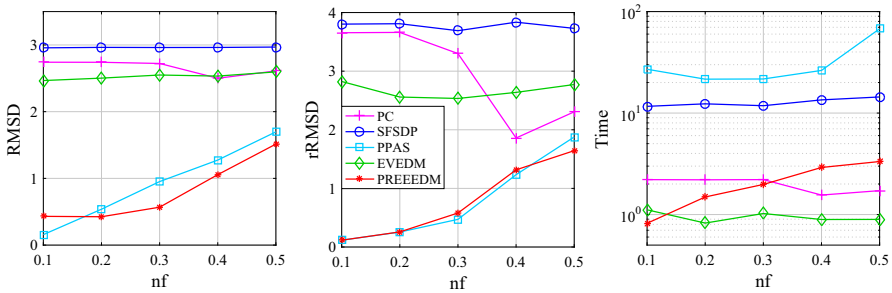


Fig. 9 Average results for Example 4 with  $s = 6$ ,  $R = s^2$

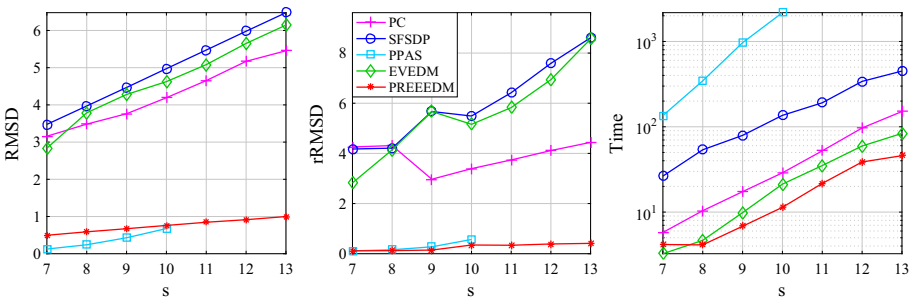
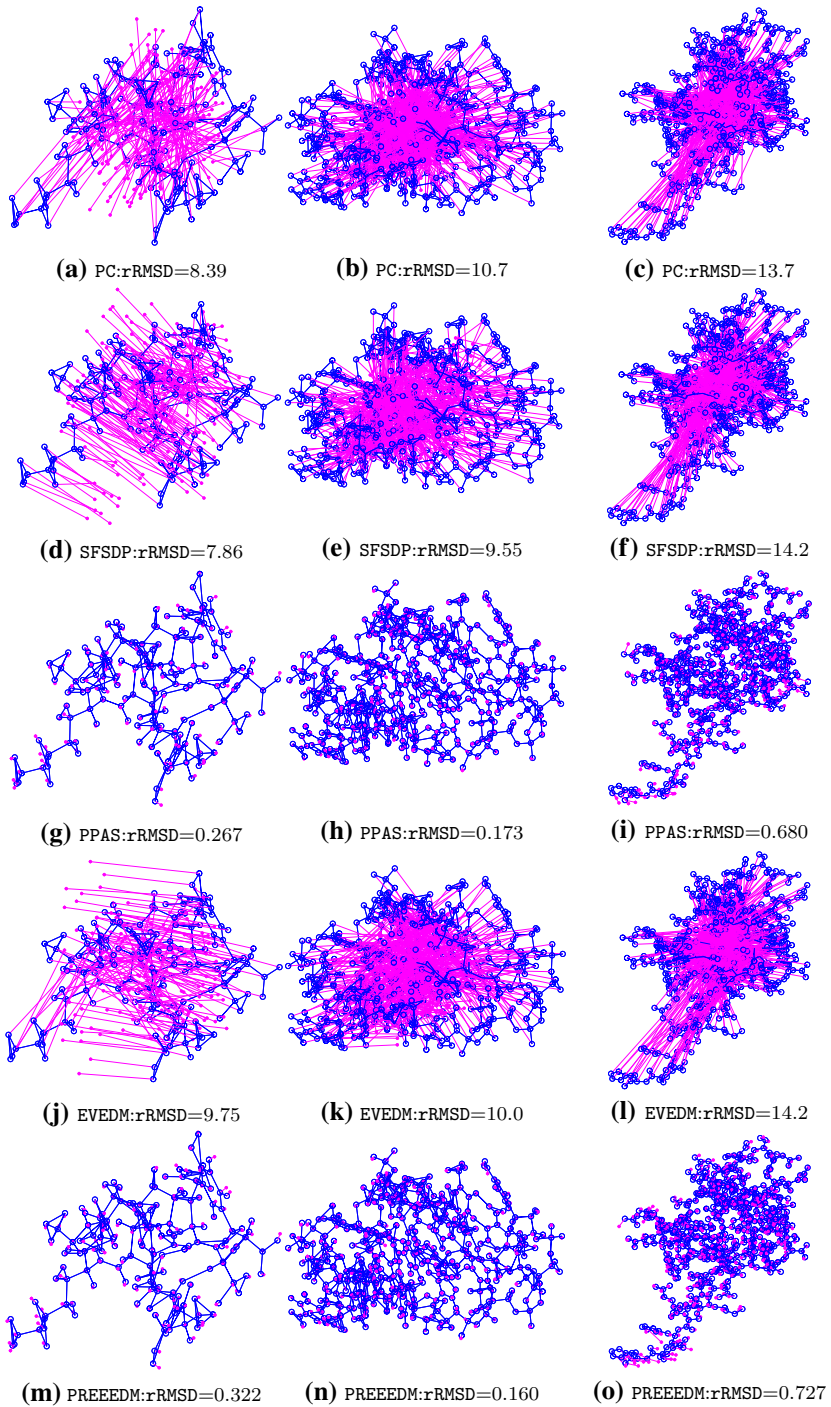


Fig. 10 Average results for Example 4 with  $n = s^3$ ,  $R = s^2$ ,  $nf = 0.1$

Clearly PC, SFSDP and EVEDM failed to locate all atoms in  $\mathfrak{R}^3$ . PPAS rendered the most accurate RMSD when  $s \leq 10$  whilst PREEEDM achieved the most accurate RMSD when  $s > 10$  and the most accurate  $rRMSD$  for all cases. Equally important for PREEEDM is that it spent less than 50s for all tested cases, while PPAS took much more time to terminate (e.g., consuming over 2000s when  $s \geq 10$ ).

(e) *Test on Example 5* For the 12 collected real data, we fixed  $R = 6$ ,  $c = 50\%$  and  $nf = 0.1$ . The generated embeddings by the five methods for the three molecules 1GM2, 1AU6 and 1LFB were shown in Fig. 11, where the true and estimated positions of the atoms were plotted by blue circles and pink stars respectively. Each pink star was linked to its corresponding blue circle by a pink line. For these three data, PREEEDM



**Fig. 11** Molecular conformation by PC, SFSDP, PPAS, EVEDM and PREEDM. Left 1GM2 ( $n = 166$ ); middle: 1AU6 ( $n = 506$ ); right 1LFB ( $n = 641$ )

and PPAS almost conformed the shape of the original data. Clearly, the other three methods failed to conform. The complete numerical results for the 12 problems were reported in Table 6. It can be clearly seen that PREEEDM and PPAS performed significantly better in terms of the RMSD and  $r$ RMSD than the other methods. What is more impressive is that PREEEDM only used a small fraction of the time by PPAS, which in general took relatively long time to terminate. For example, PREEEDM only used 22.64 s for 2CLJ, which is a very large data set with  $n = 4189$ . In contrast, we had to omit the result of PPAS for this instance (as well as to omit for other tested instances, and the missed results were indicated as “–” in Table 6) because it took too long to terminate.

#### 6.4 Robustness of PREEEDM

The excellent performance of PREEEDM reported above was actually due to its robustness to noise. Previous examples all had Gaussian noise. We now demonstrate below that PREEEDM works much better than the other methods when the noise is from a heavy-tailed distribution, for instance,  $t$ -distribution with a small degree of freedom. We also take this opportunity to test SQREDM solver of our own [56], which also made use of penalty, majorization and minimization techniques, yet for the least squares problem (5). We will see that PREEEDM outperforms SQREDM for both types of noise (Gaussian and  $t$  distributions).

To shorten the presentation, we restrict our numerical tests on two representative examples: Example 1 with  $n = 100$ ,  $R = 0.3$  and Example 4 with  $s = 5$ ,  $R = s^2$ . For each example, we generate 20 instances under two types of noise from standard normal distribution and Student- $t$  distribution with the degree of freedom being 1. We alter  $n_f$  from  $\{0.1, 0.2, \dots, 0.9\}$  and from  $\{0.01, 0.02, \dots, 0.09\}$  for the Gaussian and the Student- $t$  noises respectively. Average RMSD were recorded in Fig. 12. We have the following observations.

- (i) PREEEDM is competitive under Gaussian noise. For Example 1, Fig. 12a showed ARAP yielded the best RMSD followed by PREEEDM and SQREDM. For Example 4, Fig. 12c showed that PREEEDM rendered the smallest RMSD for most cases, followed by PPAS and SQREDM (note that the current implementation of ARAP is only for  $r = 2$  and hence is not applicable to this example). In particular, when the  $n_f$  is over 50%, PPAS and PREEEDM closely follow each other. The behaviour of those methods under Gaussian noise is expected as the least-squares formulation is equivalent to the maximum-likelihood criterion. On the one hand, least squares favour large distances. On the other hand, under Gaussian (thin-tailed), the number of large distance errors is relatively small and hence would not cause significant distortion in locating the unknown sensors.
- (ii) PREEEDM performs the best under heavy-tailed noise (from Student  $t_1$  distribution). For Example 1, both PREEEDM and SQREDM behaved much better than the other methods, see Fig. 12b. For Example 4, PREEEDM stood out as the best method when  $n_f$  is bigger than 0.02 and is much better than SQREDM, see Fig. 12d. The test data now has more numbers of large distance errors than under the Gaussian distribution and the absolute value criterion alleviates the tendency

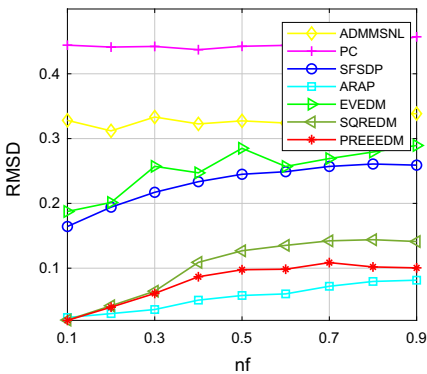
**Table 6** Comparisons of the five methods for Example 5

		PC	SFSDP	PPAS	EVEDM	PREEDM
1GM2 $n = 166$	RMSD	6.60e+0	6.65e+0	4.07e-1	6.51e+0	9.09e-1
	rRMSD	7.07e+0	6.92e+0	2.65e-1	7.41e+0	3.51e-1
	rTime	0.17	0.18	0.22	0.18	0.16
	Time	0.98	4.84	15.43	0.98	0.27
304D $n = 237$	RMSD	1.03e+1	1.03e+1	2.89e+0	1.02e+1	3.61e+0
	rRMSD	1.07e+1	1.08e+1	1.43e+0	1.08e+1	2.50e+0
	rTime	0.16	0.16	0.55	0.16	0.15
	Time	1.07	7.76	36.44	1.36	0.23
1PBM $n = 388$	RMSD	8.45e+0	8.47e+0	5.29e-1	8.35e+0	1.23e+0
	rRMSD	9.13e+0	8.91e+0	2.01e-1	9.28e+0	2.11e-1
	rTime	0.51	0.49	0.53	0.49	0.32
	Time	2.84	28.64	112.82	1.45	0.54
2MSJ $n = 480$	RMSD	1.06e+1	1.06e+1	5.40e-1	1.05e+1	9.15e-1
	rRMSD	1.12e+1	1.11e+1	2.99e-1	1.10e+1	3.34e-1
	rTime	0.40	0.39	0.54	0.39	0.32
	Time	2.32	118.60	196.12	1.47	0.59
1AU6 $n = 506$	RMSD	9.30e+0	9.31e+0	4.02e-1	9.20e+0	6.74e-1
	rRMSD	9.99e+0	9.83e+0	1.68e-1	9.69e+0	1.63e-1
	rTime	0.70	0.68	0.30	0.69	0.35
	Time	4.12	47.68	262.28	1.47	0.70
1LFB $n = 641$	RMSD	1.34e+1	1.34e+1	1.56e+0	1.33e+1	1.57e+0
	rRMSD	1.39e+1	1.35e+1	5.41e-1	1.37e+1	7.38e-1
	rTime	0.49	0.49	1.63	0.48	0.37
	Time	2.93	132.96	956.44	1.64	0.79
104D $n = 766$	RMSD	1.23e+1	1.23e+1	4.30e+0	1.22e+1	3.27e+0
	rRMSD	1.27e+1	1.27e+1	2.02e+0	1.26e+1	1.26e+0
	rTime	0.89	0.86	3.40	0.87	0.61
	Time	5.04	72.16	2024.51	1.47	1.40
1PHT $n = 814$	RMSD	1.23e+1	1.23e+1	1.70e+0	1.23e+1	1.58e+0
	rRMSD	1.29e+1	1.26e+1	9.16e-1	1.26e+1	9.85e-1
	rTime	0.74	0.74	2.57	0.74	0.48
	Time	4.86	411.14	4726.96	1.71	1.25
1POA $n = 914$	RMSD	1.42e+1	1.42e+1	1.39e+0	1.41e+1	1.48e+0
	rRMSD	1.45e+1	1.46e+1	3.27e-1	1.46e+1	4.51e-1
	rTime	0.58	0.55	1.34	0.55	0.52
	Time	5.03	587.14	1623.43	1.99	1.45
1AX8 $n = 1003$	RMSD	1.43e+1	1.43e+1	-	1.43e+1	1.23e+0
	rRMSD	1.47e+1	1.45e+1	-	1.44e+1	5.01e-1
	rTime	0.62	0.58	-	0.59	0.34
	Time	5.78	1404.53	-	1.54	1.49

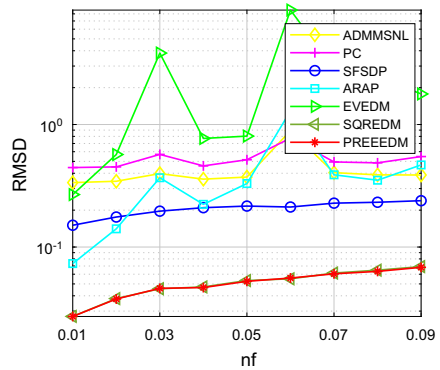


**Table 6** continued

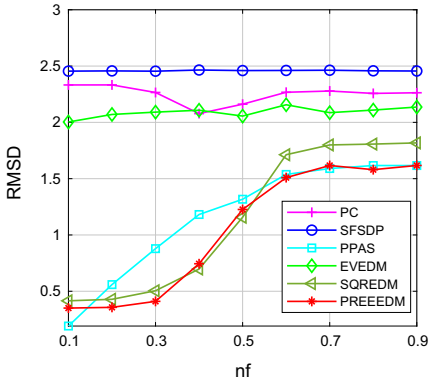
		PC	SFSDP	PPAS	EVEDM	PREEEDM
1RGS	RMSD	2.02e+1	–	–	2.02e+1	1.99e+0
$n = 2015$	rRMSD	2.05e+1	–	–	2.06e+1	6.76e−1
	rTime	1.33	–	–	1.25	0.94
	Time	16.08	–	–	3.69	5.71
	RMSD	2.27e+1	–	–	2.27e+1	1.54e+0
$n = 4189$	rRMSD	2.30e+1	–	–	2.29e+1	6.50e−1
	rTime	4.46	–	–	3.82	2.35
	Time	43.10	–	–	378.35	22.64



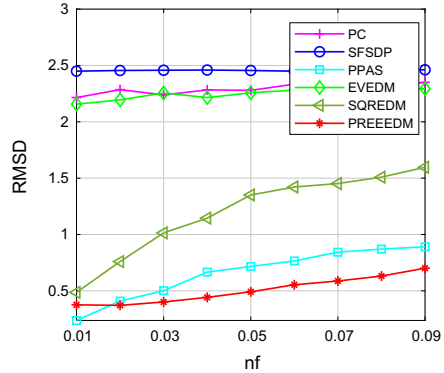
**(a)** Example 1 with standard normal noise



**(b)** Example 1 with Student  $t$  noise



**(c)** Example 4 with standard normal noise



**(d)** Example 4 with Student  $t$  noise

**Fig. 12** Robustness of PREEEDM

of favouring large distances. Therefore, PREEEDM yielded the best performance in such situations.

We conclude that PREEEDM based on the model (3) is robust to the noise in terms of these two examples.

## 7 Conclusion

The purpose of this paper is to develop an efficient method for one of the most challenging distance embedding problems in a low-dimensional space, which have been widely studied under the framework of multi-dimensional scaling. The problem employs  $\ell_1$  norm to quantify the embedding errors. Hence, the resulting model (3) appears to be robust to outliers and is referred to as the robust Euclidean embedding (REE) model.

To the best knowledge of the authors, the only viable method, whose matlab code is also publicly available for REE is the PlaceCenter (PC) algorithm proposed in [1]. Our extensive numerical results on the SNL and MC test problems convincingly demonstrated that the proposed PREEEDM method outperform PC in terms of both the embedding quality and the CPU time. Moreover, PREEEDM is also comparable to several state-of-the-art methods for other embedding models in terms of the embedding quality, but is far more efficient in terms of the CPU time. The advantage becomes even more superior as the size of the problem gets bigger.

The novelty of the proposed PREEEDM lies with its creative use of the Euclidean distance matrix and a computationally efficient majorization technique to derive its subproblem, which has a closed-form solution closely related to the positive root of the classical depressed cubic equation. Furthermore, a great deal of effort has been devoted to its convergence analysis, which well justifies the numerical performance of PREEEDM. We feel that PREEEDM will become a very competitive embedding method in the field of SNL and MC and expect its wide use in other visualization problems.

**Acknowledgements** The authors are grateful to the three anonymous referees for their constructive comments, which have helped to improve the quality of the paper. We would also like to thank Mr Nicola Piovsan and Prof Nathan Krislock respectively for sharing with us their excellent packages `ADMMSNL` and `EepVecEDM`. In particular, we thank Prof Kim-Chuan Toh for his encouragement over the past years in algorithmic implementation, acquiring PDB data and for sharing with us his well-structured package `PPAS`. This work is partly supported by the National Science Foundation of China (11728101, 71611130218).

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## 8 Appendix

### 8.1 Proof of Lemma 1

**Proof** We prove it by considering three cases. Case 1:  $0 < x < \delta^2$ ; Case 2:  $x > \delta^2$  and Case 3:  $x = \delta^2$ . For simplicity, we use  $\phi(x)$  for  $\phi_\delta(x)$  in our proof. Let  $\zeta := \eta/(2\sqrt{x})$ , then  $\zeta \in \partial\phi(x)$  is equivalent to  $\eta \in \text{sign}(\sqrt{x} - \delta)$ .

**Case 1:**  $0 < x < \delta^2$ . For this case,  $\text{sign}(\sqrt{x} - \delta) = \{-1\}$  and  $\eta = -1$ . We note that  $\phi(x) = \delta - \sqrt{x}$  is convex and differentiable at  $0 < x < \delta^2$ . Thus,

$$\phi(y) \geq \phi(x) - \frac{y-x}{2\sqrt{x}} \quad \text{for any } 0 < y < \delta^2.$$

For  $y \geq \delta^2$ , we have the following chain of inequalities

$$\begin{aligned} \phi(x) - \frac{y-x}{2\sqrt{x}} &\leq \delta - \sqrt{x} - \frac{\delta^2-x}{2\sqrt{x}} = \delta - \left[ \frac{\sqrt{x}}{2} + \frac{\delta^2}{2\sqrt{x}} \right] \\ &\leq \delta - 2\sqrt{\frac{\sqrt{x}}{2} \frac{\delta^2}{2\sqrt{x}}} = \delta - \delta = 0 \\ &\leq \sqrt{y} - \delta = \phi(y), \end{aligned}$$

Hence, we proved the conclusion for this case.

**Case 2:**  $x > \delta^2$ . For this case,  $\text{sign}(\sqrt{x} - \delta) = \{1\}$  and  $\eta = 1$ . By defining  $\Phi(\theta, \mu) := \theta(\theta^2 - \mu^2)^2 - 4\delta^3(\theta + \mu)^2 + 16\theta\delta^4$  with  $\theta > \delta$  and  $0 < \mu < \delta$ , we have

$$\frac{\partial \Phi(\theta, \mu)}{\partial \mu} = 2(\theta + \mu)(2\theta\mu(\mu - \theta) - 4\delta^3) \leq 0,$$

which indicates  $\Phi(\theta, \mu)$  is non-increasing with respect  $\mu$  and thus

$$\begin{aligned} \Phi(\theta, \mu) &\geq \Phi(\theta, \delta) = \theta(\theta^2 - \delta^2)^2 - 4\delta^3(\theta + \delta)^2 + 16\delta^4\theta \\ &= (\theta + \delta)^2(\theta(\theta - \delta)^2 - 4\delta^3) + 16\delta^4\theta \\ &\geq (\delta + \delta)^2(\delta(\delta - \delta)^2 - 4\delta^3) + 16\delta^5 \\ &= 0. \end{aligned} \tag{59}$$

For  $0 < y < \delta^2$ , we have

$$\begin{aligned} \phi(x) - \phi(y) &= \sqrt{x} + \sqrt{y} - 2\delta = \frac{x-y}{2\sqrt{x}} + \frac{(\sqrt{x} + \sqrt{y})^2}{2\sqrt{x}} - 2\delta \\ &= \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3} - \left[ \frac{(x-y)^2}{8\delta^3} - \frac{(\sqrt{x} + \sqrt{y})^2}{2\sqrt{x}} + 2\delta \right] \\ &= \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3} - \frac{\Phi(\sqrt{x}, \sqrt{y})}{8\delta^3\sqrt{x}} \\ &\stackrel{(59)}{\leq} \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3} \end{aligned}$$

For  $y \geq \delta^2$ , we have the following chain of inequalities

$$\begin{aligned} \phi(x) - \phi(y) &= \sqrt{x} - \sqrt{y} = \frac{x-y}{2\sqrt{x}} + \frac{(\sqrt{x} - \sqrt{y})^2}{2\sqrt{x}} \\ &= \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{2\sqrt{x}(\sqrt{x} + \sqrt{y})^2} \end{aligned}$$

$$\begin{aligned}
&\leq \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{2\delta(\delta+\delta)^2} \\
&= \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3}.
\end{aligned} \tag{60}$$

Hence, we proved the claim for this case.

**Case 3:**  $x = \delta^2$ . For this case,  $\text{sign}(\sqrt{x} - \delta) = [-1, 1]$  and  $-1 \leq \eta \leq 1$ . For  $0 < y < \delta^2$ , we have

$$\begin{aligned}
\phi(x) - \phi(y) &= \delta - \sqrt{x} - (\delta - \sqrt{y}) \\
&= \sqrt{y} - \sqrt{x} = \frac{y-x}{\sqrt{y} + \sqrt{x}} \leq -\frac{x-y}{2\sqrt{x}} \leq \frac{\eta(x-y)}{2\sqrt{x}}.
\end{aligned}$$

where the first and last inequalities hold due to  $y < \delta^2 = x$  and  $|\eta| \leq 1$ . For  $y \geq \delta^2$ , similar to obtaining (60), we have

$$\phi(x) - \phi(y) = \sqrt{x} - \sqrt{y} \leq \frac{x-y}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3} \leq \frac{\eta(x-y)}{2\sqrt{x}} + \frac{(x-y)^2}{8\delta^3},$$

where the last inequality is due to  $|\eta| \leq 1$  and  $x - y \leq 0$

For all three cases, we proved our claim and hence accomplish our proof.  $\square$

## 8.2 Proof of Proposition 1

**Proof** Since  $D_\rho^*$  is an optimal solution of (16), we have  $D_\rho^* \in \mathcal{B}$ . For any feasible solution  $D$  to (11) (i.e.,  $g(D) = 0$ ,  $D \in \mathcal{B}$  in (15)), it holds the following chain of inequalities.

$$\begin{aligned}
f(D) &= f(D) + \rho g(D) && \text{(because } g(D) = 0\text{)} \\
&= f_\rho(D) \\
&\geq f_\rho(D_\rho^*) && \text{(because } D_\rho^* \text{ minimizes (16))} \\
&= f(D_\rho^*) + \rho g(D_\rho^*) \\
&\geq \max\{f(D_\rho^*), \rho g(D_\rho^*)\}, && \text{(because } \rho, f, g \geq 0\text{)}
\end{aligned}$$

which together with the feasibility of  $D_r$  to (11) yields

$$g(D_\rho^*) \leq f(D_r)/\rho \leq f(D_r)/\rho\epsilon = \epsilon$$

and the feasibility of  $D^*$  to (11) derives

$$f(D^*) \geq f(D_\rho^*).$$

It is well known from [42, Theorem 6.12] that a necessary condition for  $D_\rho^*$  being (locally) optimal of (16) is  $0 \in \partial(f(D_\rho^*) + \rho g(D_\rho^*) + I_B(D_\rho^*))$ , which in turn implies (37). This together with  $g(D_\rho^*) \leq \epsilon$  completes our proof.  $\square$

### 8.3 Proof of Proposition 2

**Proof** For notational simplicity, denote  $z := x_{\omega, \beta}^-$ . Let us consider

$$\min q_-(x), \quad \text{s.t. } x \geq 0. \tag{61}$$

By noticing that the second derivative  $q''_-(x) = 1 + (\beta/4)x^{-2/3} > 1$  for all  $x > 0$ ,  $q_-(x)$  is strongly convex over  $(0, \infty)$ . It has been proved in [56, Proposition 3.1] that  $z > 0$  is the optimal solution of (61). Since  $q_-(x)$  is a univariate convex function, its optimal solution over  $B_-$  is the projection of  $z$  onto  $B_-$ , i.e.,  $x_-^* = \Pi_{B_-}(x_{\omega, \beta}^-)$ .

Note that  $z$  is the optimal solution of (61) and  $z > 0$ . We must have  $q'_-(z) = z - \omega - \beta/(2\sqrt{z}) = 0$ . If  $z \leq 1$  then  $\sqrt{z} \geq z$ , implying  $\sqrt{z} - \omega - \beta/(2\sqrt{z}) \geq q'_-(z) = 0$ , which is equivalent to  $z \geq \gamma_{\omega, \beta} > 0$ . Thus we must have  $z \geq \min\{1, \gamma_{\omega, \beta}\}$  and it holds  $x_-^* = \Pi_{B_-}(z) = \min\{\delta^2, \max\{a, z\}\} \geq \min\{\delta^2, 1, \gamma_{\omega, \beta}\}$ , which is the claimed lower bound for  $x_-^*$ .  $\square$

### 8.4 Proof of Proposition 3

**Proof** The first and the second derivatives of  $q_+(x)$  are

$$q'_+(x) = x - \omega + \frac{\beta}{2\sqrt{x}}, \quad q''_+(x) = 1 - \frac{\beta}{4\sqrt{x^3}}, \quad \forall x > 0.$$

It is easy to verify that for  $x \geq \delta^2$  and  $\beta < 4\delta^3$

$$q''_+(x) \geq 1 - \frac{\beta}{4\delta^3} > 0,$$

which implies that  $q(x)$  is strictly convex on  $[\delta^2, \infty)$ .

We consider two cases. Case 1:  $\tau \geq 0$ . This implies  $\omega \leq 3u^{2/3}$ . It follows that for  $x > 0$

$$\begin{aligned} q'_+(x) &= x - \omega + \frac{\beta}{4\sqrt{x}} + \frac{\beta}{4\sqrt{x}} \\ &\geq 3 \left[ x \frac{\beta}{4\sqrt{x}} \frac{\beta}{4\sqrt{x}} \right]^{1/3} - \omega = 3 \left[ \frac{\beta^2}{4^2} \right]^{1/3} - \omega = 3u^{2/3} - \omega \geq 0. \end{aligned}$$

This implies that  $q_+(x)$  is non-decreasing and hence  $x_+^* = \delta^2$ .

Case 2:  $\tau < 0$ , which implies  $\omega > 3u^{2/3}$ . Consider the problem:

$$\min q_+(x) \quad \text{s.t. } x \geq 0. \tag{62}$$

We will apply Lemma 3 to the problem (62) and show that exactly one of its two positive stationary points falls within the interval  $[\delta^2, \infty)$ . We will further show that this stationary point is defined by (46) for the case  $\tau < 0$ . Since  $q_+(x)$  is convex over this interval, the optimal solution of the problem (45) is just the projection of this stationary point onto the interval  $B_+ = [\delta^2, b]$ . This would complete the proof.

Comparing the problem (62) with the problem (32), the corresponding quantities are

$$v = 2\beta, \quad t = \omega, \quad \bar{x} = (v/8)^{2/3} = (\beta/4)^{2/3} = u^{2/3} \quad \text{and} \quad \bar{t} = 3\bar{x}.$$

It is obvious that  $t = w > 3u^{2/3} = 3\bar{x}$  (the condition of Lemma 3 is satisfied). Lemma 3 implies that the problem (62) has two positive stationary points, which must satisfy the optimality condition  $q'_+(\hat{x}) = 0$ , leading to

$$\hat{x} - \omega + \frac{\beta}{2\sqrt{\hat{x}}} = 0.$$

Let  $\hat{y} := \sqrt{\hat{x}}$ , we then have

$$\hat{y}^3 - \omega\hat{y} + \frac{\beta}{2} = 0. \quad (63)$$

This is the well-known depressed cubic equation, whose solution (i.e., Cardan formula) has a long history [7, Chp. 7].

Since  $\omega > 3u^{2/3}$ , it follows from the Cardan formula (in terms of the trigonometric functions, see [51, Sect. 3]) that (63) has three real roots, namely

$$\hat{y}_1 := 2\sqrt{v}\cos(\theta/3), \quad \hat{y}_2 := 2\sqrt{v}\cos((4\pi + \theta)/3), \quad \hat{y}_3 := 2\sqrt{v}\cos((2\pi + \theta)/3)$$

with  $\cos(\theta) = -uv^{-3/2}$ . Moreover, the three roots satisfy that  $\hat{y}_1 \geq \hat{y}_2 \geq \hat{y}_3$ . According to Lemma 3, two of them are positive. That is,  $\hat{y}_1 > 0$ ,  $\hat{y}_2 > 0$  and

$$\hat{y}_2^2 < \bar{x} < \hat{y}_1^2.$$

Since  $\beta < 4\delta^3$ , we have

$$\bar{x} = u^{2/3} = (\beta/4)^{2/3} < \delta^2.$$

Therefore,  $\hat{y}_1^2$  is the only point that falls within the interval  $[\delta^2, \infty)$ . Since  $q_+(x)$  is strictly convex, the minimum of the problem (45) must be the projection of  $\hat{y}_1^2$  onto the interval  $B_+$ . Hence, for the Case 2, we must have  $x_+^* = \Pi_{B_+}(\hat{y}_1^2)$ . The proof is completed by noting that  $\hat{y}_1^2$  is just  $x_{\omega, \beta}^+$  defined in (46) for the case  $\tau < 0$ .  $\square$

### 8.5 Proof of Theorem 1

**Proof** (i) is a direct consequence of Propositions 2 and 3. We now prove (ii). For any  $x, y > 0$  and any  $\xi_x \in \partial q(x)$ , it follows that

$$\xi_x = x - \omega + \beta \zeta \quad \text{with } \zeta \in \partial \phi_\delta(x)$$

and

$$\begin{aligned} q(y) - q(x) &= \frac{1}{2}(y - \omega)^2 - \frac{1}{2}(x - \omega)^2 + \beta(|\sqrt{y} - \delta| - |\sqrt{x} - \delta|) \\ &= (x - \omega)(y - x) + \frac{1}{2}(x - y)^2 + \beta(|\sqrt{y} - \delta| - |\sqrt{x} - \delta|) \\ &\geq (x - \omega)(y - x) + \frac{1}{2}(x - y)^2 - \beta \zeta(x - y) - \frac{\beta(x - y)^2}{8\delta^3} \\ &= (x - \omega + \beta \zeta)(y - x) + \frac{4\delta^3 - \beta}{8\delta^3}(x - y)^2 \\ &> \xi_x(y - x), \end{aligned}$$

where the first inequality above used Lemma 1 and the last inequality used the fact  $4\delta^3 > \beta > 0$ . Swapping the role of  $x$  and  $y$  above yields

$$q(x) - q(y) > \xi_y(x - y) \quad \forall x, y > 0, \xi_y \in \partial q(y).$$

Therefore, we have

$$(\xi_x - \xi_y)(x - y) > 0 \quad \forall x, y > 0, \xi_x \in \partial q(x) \text{ and } \xi_y \in \partial q(y).$$

This together with Theorem 12.17 of [42] proves that  $q(x)$  is strictly convex over  $[a, b]$ . The rest in (ii) is just the first order optimality condition of the convex optimization problem (41) because we just proved the convexity of  $q(x)$  over  $[a, b]$ . Finally, we prove (iii). It follows from (42) that

$$\gamma_{\omega, \beta} = \left[ \frac{\omega + \sqrt{\omega^2 + 2\beta}}{2} \right]^2 = \left[ \frac{\beta}{\sqrt{\omega^2 + 2\beta} - \omega} \right]^2 \geq \left[ \frac{\beta}{\sqrt{C^2 + 2\beta} + C} \right]^2 := \kappa_0$$

and from Propositions 2 and 3 that

$$x_-^* \geq \min\{\delta^2, 1, \kappa_0\} \quad \text{and} \quad x_+^* \geq \delta^2.$$

Therefore,

$$\text{dcroot}_B[\omega, \beta, \delta] \geq \min\{\delta^2, 1, \kappa_0\} := \kappa.$$

We finish our proof. □

## 8.6 Proof of Proposition 4

**Proof** (i) Let us pick a pair  $(i, j)$  such that  $W_{ij} > 0$ , which implies  $\delta_{ij} > 0$  (Assumption 2). It follows from (50) that

$$D_{ij}^k = \text{dCroot}_{[L_{ij}, U_{ij}]} \left[ (Z_K^{k-1})_{ij}, W_{ij}/\rho, \delta_{ij} \right],$$

where  $Z_K^{k-1} := -\Pi_{\mathcal{K}_+^n(r)}(-D^{k-1})$ . Since  $\mathcal{B}$  is bounded (Assumption 1) and  $D^k \in \mathcal{B}$ , the sequence  $\{D^k\}$  is bounded. Lemma 2 implies

$$\| -\Pi_{\mathcal{K}_+^n(r)}(-D^{k-1}) \| \leq 2\|D^{k-1}\| \leq 2\|U\|,$$

which further implies  $|(Z_K^{k-1})_{ij}| \leq 2\|U\|$  for all  $k = 1, \dots$ . Let  $\beta_{ij} := W_{ij}/\rho$ . Then

$$0 < \beta_{ij} < 4\delta_{ij}^3$$

owing to  $\rho > \rho_o(W, \Delta)$ . It follows from Theorem 1(iii) that there exists  $\kappa_{ij} > 0$  such that  $D_{ij}^k \geq \kappa_{ij}$  for all  $k = 1, 2, \dots$ . The choice of  $c_1$  by

$$c_1 := \min\{\kappa_{ij} : (i, j) \text{ such that } W_{ij} > 0\} > 0.$$

satisfies the bound in (i).

(ii) We write  $f(D)$  in terms of  $D_{ij}$ :

$$f(D) = \sum_{i,j} W_{ij} |\sqrt{D_{ij}} - \delta_{ij}| = \sum_{i,j} W_{ij} \phi_{\delta_{ij}}(D_{ij}). \quad (64)$$

We let  $\partial_{ij} f(D)$  denote the subdifferential of  $f$  with respect to its  $(i, j)$ th element  $D_{ij}$ . We consider two cases. **Case 1:**  $W_{ij} = 0$ . This implies that  $f(D)$  is a constant function ( $\equiv 0$ ) of  $D_{ij}$  and hence  $f(D)$  is continuously differentiable with respect to  $D_{ij}$ . Consequently,  $\partial_{ij} f(D^k) = \{0\}$ .

**Case 2:**  $W_{ij} > 0$ , which implies  $\delta_{ij} > 0$  (Assumption 2). It follows from (i) that there exists  $c_1 > 0$  such that  $D_{ij}^k \geq c_1$  for all  $k = 1, 2, \dots$ . The Eqs. (64) and (25) yield

$$\partial_{ij} f(D^k) = W_{ij} \text{sign} \left[ \sqrt{D_{ij}^k} - \delta_{ij} \right] / \left[ 2\sqrt{D_{ij}^k} \right],$$

which implies that for any  $\xi_{ij}^k \in \partial_{ij} f(D^k)$  there exists  $\zeta_{ij}^k \in \text{sign}((D_{ij}^k)^{1/2} - \delta_{ij})$  such that

$$|\xi_{ij}^k| = W_{ij} |\zeta_{ij}^k| / \left[ 2\sqrt{D_{ij}^k} \right] \leq W_{ij} / \sqrt{4c_1}.$$



In other words,  $\partial_{ij} f(D^k)$  is bounded by  $W_{ij}/\sqrt{c_1}$ , which is independent of the index  $k$ . It follows directly from the definition of subdifferential [42, Chp. 8.3] that

$$\partial f(D^k) \subseteq \bigotimes \partial_{ij} f(D^k)$$

in the sense that for any  $\Gamma^k \in \partial f(D^k)$ , there exist  $\xi_{ij}^k \in \partial_{ij} f(D^k)$  such that

$$\Gamma_{ij}^k = \xi_{ij}^k, \quad i, j = 1, \dots, n.$$

Consequently, we have for all  $k = 1, 2, \dots$ ,

$$\|\Gamma^k\| \leq n \max_{i,j} |\xi_{ij}^k| \leq n W_{ij}/(2\sqrt{c_1}) \leq n \max_{i,j} W_{ij}/(2\sqrt{c_1}) =: c_2 > 0.$$

This completes the proof for (ii).

(iii) Since  $\rho > \rho_0$ , for each pair  $(i, j)$  we have  $\beta_{ij} := W_{ij}/\rho < 4\delta_{ij}^3$ . It then follows from Theorem 1(ii) that each separable function  $f_{ij}^k(D_{ij})$  is convex and hence the function  $f_\rho^k(D)$  is convex over  $D \in \mathcal{B}$ . Consequently, subproblem (48) is convex. The first-order necessary and sufficient optimality condition is just (52).  $\square$

## References

1. Agarwal, A., Phillips, J.M., Venkatasubramanian, S.: Universal multi-dimensional scaling, In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1149–1158, ACM (2010)
2. An, L.T.H., Tao, P.D.: Large-scale molecular optimization from distance matrices by a dc optimization approach. *SIAM J. Optim.* **14**, 77–114 (2003)
3. Bai, S., Qi, H.-D.: Tackling the flip ambiguity in wireless sensor network localization and beyond. *Digit. Signal Process.* **55**, 85–97 (2016)
4. Berman, H.M., Westbrook, J., Feng, Z., Gillilan, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. *Nucleic Acids Res.* **28**, 235–242 (2000)
5. Biswas, P., Liang, T.-C., Toh, K.-C., Wang, T.-C., Ye, Y.: Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Trans. Auto. Sci. Eng.* **3**, 360–371 (2006)
6. Borg, I., Groenen, P.J.F.: *Modern Multidimensional Scaling: Theory and Applications*. Springer Series in Statistics, 2nd edn. Springer, Berlin (2005)
7. Burton, D.M.: *The History of Mathematics*, 7th edn. McGraw-Hill, New York City (2011)
8. Cayton, L., Dasgupta, S.: Robust Euclidean embedding. In: Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, pp. 169–176 (2006)
9. Chen, Y.Q., Xiu, N.H., Peng, D.T.: Global solutions of non-Lipschitz  $S_2 - S_p$  minimization over the positive semidefinite cone. *Optim. Lett.* **8**, 2053–2064 (2014)
10. Cox, T.F., Cox, M.A.A.: *Multidimensional Scaling*, 2nd edn. Chapman and Hall/CRC, Boca Raton (2001)
11. Ding, C., Qi, H.-D.: Convex optimization learning of faithful Euclidean distance representations in nonlinear dimensionality reduction. *Math. Program.* **164**, 341–381 (2017)
12. Drusvyatskiy, D., Krislock, N., Voronin, Y.L., Wolkowicz, H.: Noisy Euclidean distance realization: robust facial reduction and the Pareto frontier. *SIAM J. Optim.* **27**(4), 2301–2331 (2017)
13. de Leeuw, J.: Applications of Convex analysis to multidimensional scaling. In: Barra, J., Brodeau, F., Romier, G., van Cutsem, B. (eds.) *Recent Developments in Statistics*, pp. 133–145. North Holland Publishing Company, Amsterdam, The Netherlands (1977)

14. de Leeuw, J., Mair, P.: Multidimensional scaling using majorization: Smacof in R. *J. Stat. Softw.* **31**, 1–30 (2009)
15. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: *Proceedings of the 3rd IPSN*, pp. 46–54. Berkeley, CA (2004)
16. Drusvyatskiy, D., Krislock, N., Voronin, Y.-L., Wolkowicz, H.: Noisy Euclidean distance realization: robust facial reduction and the Pareto frontier. *SIAM J. Optim.* **27**, 2301–2331 (2017)
17. France, S.L., Carroll, J.D.: Two-way multidimensional scaling: a review. *IEEE Trans. Syst. Man Cyber. Part C* **41**, 644–661 (2011)
18. Gao, Y.: *Structured Low Rank Matrix Optimization Problems: a Penalty Approach*, PhD Thesis, National University of Singapore (2010)
19. Gaffke, N., Mathar, R.: A cyclic projection algorithm via duality. *Metrika* **36**, 29–54 (1989)
20. Glunt, W., Hayden, T.L., Hong, S., Wells, J.: An alternating projection algorithm for computing the nearest Euclidean distance matrix. *SIAM J. Matrix Anal. Appl.* **11**, 589–600 (1990)
21. Glunt, W., Hayden, T.L., Raydan, R.: Molecular conformations from distance matrices. *J. Comput. Chem.* **14**, 114–120 (1993)
22. Gower, J.C.: Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika* **53**(1966), 325–338 (1966)
23. Heiser, W.J.: Multidimensional scaling with least absolute residuals. In: *Proceedings of the First Conference of the International Federation of Classification Societies (IFCS)*, pp. 455–462. Germany, Aachen (1987)
24. Jiang, K.F., Sun, D.F., Toh, K.C.: *Solving Nuclear Norm Regularized and Semidefinite Matrix Least Squares Problems with Linear Equality Constraints*, *Discrete Geometry and Optimization*, pp. 133–162. Springer International Publishing, Berlin (2013)
25. Jiang, K.F., Sun, D.F., Toh, K.-C.: A partial proximal point algorithm for nuclear norm regularized matrix least squares problems. *Math. Progr. Comput.* **6**, 281–325 (2014)
26. Kanzow, C., Qi, H.-D.: A QP-free constrained Newton-type method for variational inequality problems. *Math. Progr.* **85**, 81–106 (1999)
27. Kim, S., Kojima, M., Waki, H., Yamashita, M.: Algorithm 920: SFSDP: a sparse version of full semidefinite programming relaxation for sensor network localization problems. *ACM Trans. Math. Softw.* **38**(4), 27:1–27:19 (2012)
28. Korkmaz, S., Van der Veen, A.J.: Robust localization in sensor networks with iterative majorization techniques. *ICASSP 2049–2052* (2009)
29. Kovnatsky, A., Glashoff, K., Bronstein M.M.: MADMM: a generic algorithm for non-smooth optimization on manifolds. In: *European Conference on Computer Vision*, Springer, Cham., pp. 680–696 (2016)
30. Kruskal, J.B.: Nonmetric multidimensional scaling: a numerical method. *Psychometrika* **29**, 115–129 (1964)
31. Mandanas, F.D., Kotropoulos, C.L.: Robust multidimensional scaling using a maximum correntropy criterion. *IEEE Trans. Signal Process.* **65**, 919–932 (2017)
32. Micchelli, C.A.: Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constr. Approx.* **2**, 11–22 (1986)
33. More, J.J., Wu, Z.: Global continuation for distance geometry problems. *SIAM J. Optim.* **7**, 814–836 (1997)
34. Nocedal, G., Wright, S.J.: *Numerical Optimization*, 2nd edn. Springer, Berlin (2006)
35. Oğuz-Ekim, P., Gomes, J.P., Xavier, J., Oliveira, P.: Robust localization of nodes and time-recursive tracking in sensor networks using noisy range measurements. *IEEE Trans. Signal Process.* **59**, 3930–3942 (2011)
36. Peng, D.T., Xiu, N.H., Yu, J.:  $S_{1/2}$  regularization methods and fixed point algorithms for affine rank minimization problems. *Comput. Optim. Appl.* **67**, 543–569 (2017)
37. Piovesan, N., Erseghe, T.: Cooperative localization in WSNs: a hybrid convex/non-convex solution. *IEEE Trans. Signal and Information Processing over Networks*. <https://doi.org/10.1109/TSIPN.2016.2639442>. (IEEE early access article, 2016)
38. Pong, T.K.: Edge-based semidefinite programming relaxation of sensor network localization with lower bound constraints. *Comput. Optim. Appl.* **53**, 23–44 (2012)
39. Qi, H.-D.: A semismooth Newton method for the nearest Euclidean distance matrix problem. *SIAM J. Matrix Anal. Appl.* **34**, 67–93 (2013)

40. Qi, H.-D., Yuan, X.M.: Computing the nearest Euclidean distance matrix with low embedding dimensions. *Math. Progr.* **147**, 351–389 (2014)
41. Qi, H.-D., Xiu, N.H., Yuan, X.M.: A Lagrangian dual approach to the single source localization problem. *IEEE Trans. Signal Process.* **61**, 3815–3826 (2013)
42. Rockafellar, R.T., Wets, R.J.-B.: *Variational Analysis*, 3rd edn. Springer, Berlin (2009)
43. Schoenberg, I.J.: Remarks to Maurice Frechet’s article Sur la definition axiomatique d’une classe d’espaces vectoriels distances applicables vectoriellement sur l’espace de Hilbert. *Ann. Math.* **36**, 724–732 (1935)
44. Shang, Y., Ruml, W., Zhang, Y., Fromherz, M.P.J.: Localization from mere connectivity. In: *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing, MobiHoc 03*, ACM, New York, NY, USA, pp. 201–212 (2003)
45. Soares, C., Xavier, J., Gomes, J.: Simple and fast convex relaxation method for cooperative localization in sensor networks using range measurements. *IEEE Trans. Signal Process.* **63**(17), 4532–4543 (2015)
46. Sun, Y., Babu, P., Palomar, D.P.: Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Trans. Signal Process.* **65**, 794–816 (2017)
47. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. *Science* **290**, 2319–2323 (2000)
48. Toh, K.C.: An inexact path-following algorithm for convex quadratic SDP. *Math. Progr.* **112**, 221–254 (2008)
49. Torgerson, W.S.: Multidimensional scaling: I. Theory and method. *Psychometrika* **17**, 401–419 (1952)
50. Wang, Z., Zheng, S., Ye, Y., Boyd, S.: Further relaxations of the semidefinite programming approach to sensor network localization. *SIAM J. Optim.* **19**, 655–673 (2008)
51. Xing, F.C.: Investigation on solutions of cubic equations with one unknown. *J. Central Univ. Nat. (Natural Sci. Ed.)* **12**, 207–218 (2003)
52. Xu, Z., Chang, X., Xu, F., Zhang, H.:  $L_{1/2}$  regularization: a thresholding representation theory and a fast solver. *IEEE Trans. Neural Netw. Learn. Sys.* **23**, 1013–1027 (2012)
53. Young, G., Householder, A.S.: Discussion of a set of points in terms of their mutual distances. *Psychometrika* **3**, 19–22 (1938)
54. Zhang, L., Liu, L., Gotsman, C., Gortler, S.J.: An as-rigid-as-possible approach to sensor network localization. *ACM Trans. Sen. Netw.* **6**, 35:1–35:21 (2010)
55. Zhang, L., Wahba, G., Yuan, M.: Distance shrinkage and Euclidean embedding via regularized kernel estimation. *J. R. Stat. Soc.: Ser. B* **78**, 849–867 (2016)
56. Zhou, S.L., Xiu, N.H., Qi, H.D.: A fast matrix majorization-projection method for penalized stress minimization with box constraints. *IEEE Trans. Signal Process.* **66**, 4331–4346 (2018)

**Publisher’s Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.