Tackling the Flip Ambiguity in Wireless Sensor Network Localization and Beyond

Shuanghua Bai*, Houduo Qi

School of Mathematical Sciences, University of Southampton, Highfield, Southampton, SO17 1BJ, UK

Abstract

There have been significant advances in range-based numerical methods for sensor network localizations over the past decade. However, there remain a few challenges to be resolved to satisfaction. Those issues include, for example, the flip ambiguity, high level of noises in distance measurements, and irregular topology of the concerning network. Each or a combination of them often severely degrades the otherwise good performance of existing methods. Integrating the connectivity constraints is an effective way to deal with those issues. However, there are too many of such constraints, especially in a large and sparse network. This presents a challenging computational problem to existing methods. In this paper, we propose a convex optimization model based on the Euclidean Distance Matrix (EDM). In our model, the connectivity constraints can be simply represented as lower and upper bounds on the elements of EDM, resulting in a standard 3-block quadratic conic programming, which can be efficiently solved by a recently proposed 3-block alternating direction method of multipliers. Numerical experiments show that the EDM model effectively eliminates the flip ambiguity and retains robustness in terms of being resistance to irregular wireless sensor network topology and high noise levels.

Keywords: Euclidean distance matrix, range-based node localization, convex optimization, alternating direction method of multipliers, wireless sensor networks.

1. Introduction

Wireless Sensor Networks (WSNs) consist of a collection of spatially distributed autonomous sensor nodes, which can sense, measure, gather and transmit information from a geographical area [1]. WSNs play an important role in a variety of applications such as environmental/earth sensing and industrial monitoring, in which an accurate realization of sensor positions with respect to a global coordinate system is highly desirable for the data gathered to be geographically meaningful. One way to locate sensor nodes is by equipping each of them with a global positioning system (GPS) device, which could be very costly and significantly energy consuming for networks with numerous sensors. Thus, a common approach is to firstly acquire only a small portion of sensor positions by manual deployment or by GPS, and they are known as *anchor* nodes. It is then to locate the rest of nodes in the network using the connectivity or pairwise distance information from typical range measurements such as time-of-arrival (ToA). We refer to the book by Akyildiz and Vuran [2] for the details of hardware technologies. Localization with only connectivity information is categorized as range-free localization. It often provides less accurate positions than the range-based localization, in which both connectivity and pairwise distance information are used. The main focus of this paper lies in the latter category as we assume that partial information on pairwise distances is provided.

Suppose there are n sensors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in $\mathbb{R}^r$ ($r = 2$ or 3). Some of them may be anchors. Assume that Euclidean distance among some of the sensors can be observed:

$$\widehat{d}_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\| + \epsilon_{ij}, \tag{1}$$

where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^r$, ϵ_{ij} are errors, and $\widehat{d}_{ij}$ are observed distances. The sensor network localization (SNL) problem is to recover the sensor positions through those distances. The most popular method may be the classical Multidimensional Scaling (cMDS), well documented in [3, 4], and a recent survey [5] and tutorial [6]. It works well when a large number of $\widehat{d}_{ij}$ are close to their true distances (i.e., ϵ_{ij}s are relatively small). Otherwise, one has to opt for many of its variants. Well-known ones include SMACOF [7], weighted-MDS [8], and MDS-MAP [9, 10]. In particular, MDS-MAP often works when nodes are positioned relatively uniformly in the space, but does not perform well on networks with irregular topology, where the shortest path distance does not correlate well with the true Euclidean distance. To compensate for this drawback, several "patching" algorithms, such as MDS-MAP(P) [11], PATCHWORK [12] and As-Rigid-As-Possible (ARAP) [13], are proposed, which starts by first localizing small patches and then stitching them together to recover the global coordinates. Gepshtein *et*

*Corresponding author

Email addresses: sb11g12@soton.ac.uk (Shuanghua Bai), hdqi@soton.ac.uk (Houduo Qi)

al. proposed an anchor-based sensor networks localization scheme called ADESR in [14] that utilizes a dual spectral embedding. It is experimentally shown that ADESR outperforms ARAP in terms of robustness to noise and localization accuracy, but it may suffer from the increasing computational complexity.

Another important class of methods is the Semi-Definite Programming (SDP) relaxation-based, initially introduced by Biswas and Ye [15] to SNL. The major modelling procedure in many papers in this class of methods is first to formulate the SNL problem as a quadratic optimization problem and then to relax it as SDP, which can be efficiently solved by interior point methods such as SDPT3 in [16]. Important techniques have been introduced in order to improve the quality and/or computational efficiency of various relaxations. For example, techniques of regularization and refinement have been employed in [17]. Edge-based SDP (ESDP) and node-based SDP (NSDP) relaxations of the full SDP (FSDP) [15] are studied in [18]. A sparse version of FSDP has been implemented as SFSDP by Kim *et al.* in [19] and it requires much less computational time than most of the SDP relaxation methods. SDP relaxation based methods work well on most types of networks. However, numerical simulation shows that they are very sensitive to networks with randomly deployed anchor nodes, and often suffer from flip ambiguity, which may induce large errors to the final localization.

As lightly touched above and already reported in existing literature, the issues such as high level of noises (i.e., ϵ_{ij} in (1) are relatively large), irregular topology (e.g., sensors are not uniformly distributed over a convex region) and flip ambiguity can severely degrade the performance of existing methods. In particular, flip ambiguity is one of the major challenges brought up by ranging errors. It arises when the neighbours of a node lie almost in a line such that the node can be mirror reflected across the line while still satisfying the distance constraints from its neighbours [20], as illustrated in Fig. 1. In this small network, node A could be mirror reflected to the position of A' when there exists error in the distance measurements, while the distances under the same connectivity states, i.e. $A'B$, $A'C$, $A'D$, can still remain unchanged, thereby causing a large localization error. Moreover, the error may transmit and a large part of the sensors could be folded over. For a larger network example, see Fig. 4b.

Several heuristic methods have been purposefully proposed to tackle the issue of flip ambiguity such as the two-phase simulated annealing algorithm by Kannan *et al.* [21] and the two-step tabu search algrithm by Shekofteh *et al.* [22]. These methods can be robust and efficient if the parameters are well controlled. However, tuning parameters for networks of various sizes and topologies is very time consuming with no guarantee of success.

The strategy we propose here is more direct and is of geometric. Take the case in Fig. 1 for example, we note that point A has no connection with both points E and F. If we enforce the (dis)connectivity constraints

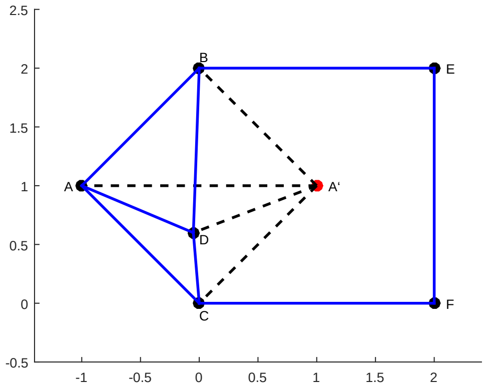

Figure 1: Illustration of flip ambiguity, a small network with 6 nodes, communication radius $R = 2.1$, blue lines indicate the existence of communication between two nodes.

$$\|\mathbf{x}_A - \mathbf{x}_E\| > R \quad \text{and} \quad \|\mathbf{x}_A - \mathbf{x}_F\| > R, \tag{2}$$

then the possibility of point A flipping to point A' would be removed because A' would fall within the communication range of points E and F. For this strategy to be effective, the following three elements have to be taken into consideration. (i) There are a large number of such connectivity constraints as in (2). The order is $O(n^2)$. The more sparse the network is , the more of such constraints there are. (ii) The connectivity constraints such as (2) should be satisfied in the embedding space $\mathbb{R}^r$, which is a low-dimensional space. This essentially introduces nonconvexity to the problem. (iii) The computational model of localization, as done in SDP relaxation methods reviewed above, should be of convex optimization in order to develop fast algorithms. The consideration of the three elements all together will result in robust localization and this is where our research departs from most of the existing ones.

The important concept that we will employ is the Euclidean Distance Matrix (EDM), which allow us to effectively address the three elements above. Namely, the connectivity constraints will be simply represented as lower and/upper bounds constraints. It is those bound constraints that play an important role in mitigating flip ambiguity and obtaining robustness under presence of large noises in the distance measurements. The nonconvexity concerning the embedding space is formulated as a rank constraint, which is further approximated by a (heuristic) linear function to induce a low rank solution. The final computational model belongs to the class of convex EDM optimization, which in various forms has long been studied in numerical optimization and found many applications including molecular conformation and sensor network localization, see, e.g., [23, 24, 25, 26]. Both primal and dual problem of our model has 3 separate block-variables. It is this nice and tidy structure that makes it possible to implement an efficient algorithm of Alternating Direction Method of Multiplier (ADMM). The algorithm is derived based on a recently proposed Schur complement based semi-proximal ADMM in [27], which allows us to handle millions of inequality constraints efficiently when the sensor number grows to thousands. Numerical comparison will demonstrate that, with less computational complexity, our approach outperforms the existing state-of-art methods in terms of flip ambiguity elimination, being resistance to irregular WSN topology and presence of large noises in ranging measurements.

The rest of the paper is organized as follows. In the next section we briefly introduce necessary background on Euclidean distance matrix and classical multidimensional scaling. Our EDM-based localization scheme is described in Sect. 3, where we describe the convex optimization model, deal with the rank constraint as well as discuss the global coordinate recovery. In Sect 4, we introduce a convergent 3-block ADMM algorithm for our convex optimization model to retrieve missing distance information. Numerical comparison with existing state-of-art methods is reported in Sect. 5, which demonstrates the robustness of our model in handling the issues reviewed above. We conclude in Sect. 6.

2. Background on EDM Optimization

In this section, we give an elementary review on EDM optimization, to which our ultimate model belong. More materials can be found in [3, 4, 28]. Basically speaking, EDM optimization aims to solve the inverse problem of constructing the coordinates of n points from their distance information (the direct problem from coordinates to distances is straightforward). Let us formalize this inverse problem. Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be n points in $\mathbb{R}^r$. Let $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ be their pairwise Euclidean distances. The $n \times n$ Euclidean distance matrix D from those n points is defined to be $D_{ij} = d_{ij}^2$. That is, the elements of D are the squared Euclidean distances. In practice, we are often given a distance matrix and we seek the coordinates of points that can generate this distance matrix D (the inverse problem).

Mathematically, for a given matrix D to be an EDM, it must satisfy two conditions. Let us explain them and we need some standard notation. Let $\mathcal{S}^n$ denote the space of $n \times n$ symmetric matrices equipped with the standard inner product $\langle A, B \rangle = \mathrm{tr}(AB)$ for $A, B \in \mathcal{S}^n$ ($\mathrm{tr}(X)$ is the trace of X). Let $\| \cdot \|$ denote the induced Frobenius norm. Let $\mathcal{S}^n_+$ denote the cone of positive semidefinte matrices in $\mathcal{S}^n$ (often abbreviated as $X \succeq 0$ for $X \in \mathcal{S}^n_+$). Throughout the paper, we use ":=" to mean "define". The so-called *hollow* subspace $\mathcal{S}^n_h$ is defined by

$$S^n_h := \{ A \in \mathcal{S}^n | \quad \mathrm{diag}(A) = 0 \},$$

where $\mathrm{diag}(A)$ is the vector formed by the diagonal elements of A. The two conditions for a matrix $D \in \mathcal{S}^n$ to be an EDM are

$$D \in \mathcal{S}^n_h \text{ and } -JDJ \succeq 0, \text{ with } J := J_n = I - ee^T/n, \quad (3)$$

where I is the identity matrix in $\mathcal{S}^n$, e is the vector of all ones in $\mathbb{R}^n$ (all vectors are treated as columun vectors in the paper) and J_n is the centralization matrix of dimension n. The origin of this result can be traced back to Schoenberg [29] and an independent work [30] by Young and Householder. Moreover, a set of coordinates can be obtained through the decomposition:

$$-JDJ/2 = X^T X \quad (4)$$

where $X \in \mathbb{R}^{r \times n}$. We note that the decomposition is possible because the matrix $(-JDJ)$ is positive semidefinite according to (3). Let $\mathbf{x}_i$ denote the ith column of X. Then those points recover the distances in D. That is, $D_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ for all i, j. The embedding dimension r is actually the rank of the matrix (JDJ).

The results in (3) and (4) are true when D is a true EDM. What should one do if D is not a true EDM? Here and forth, we use $\widehat{D}$ to denote that it is not a true EDM. As reviewed in Introduction, the most popular method is the cMDS, which simply computes the nearest positive semidefinite matrix from $(-J\widehat{D}J)$ and is obtained through the following optimization:

$$\min \ \|J(D - \widehat{D})J\|^2 \ \text{ s.t. } -JDJ \succeq 0 \text{ and } D \in \mathcal{S}^n_h. \quad (5)$$

The optimal solution is just the orthogonal projection of $(-J\widehat{D}J)$ onto $\mathcal{S}^n_+$ and is denoted by $\Pi_{\mathcal{S}^n_+}(-J\widehat{D}J))$. cMDS then uses this projection in replace of $(-JDJ)$ in (4) to get the embedding points in X. This works well when $\widehat{D}$ is close to a true EDM. Otherwise it may perform poorly in terms of embedding quality due to the embedding dimension thus obtained being too high.

The nearness in the objective of (5) between the true EDM D and the observed matrix $\widehat{D}$ is measured under the transformation $J(D-\widehat{D})J$. A more accurate measurement is between D and $\widehat{D}$ themselves. This leads to the widely studied nearest EDM problem (see, e.g., [31, 25] and the references therein):

$$\min \ \|D - \widehat{D}\|^2 \ \text{ s.t. } -JDJ \succeq 0 \text{ and } D \in \mathcal{S}^n_h. \quad (6)$$

The price for having this more accurate model (6) is that it does not have a close-form solution anymore and has to rely on iterative methods for its solution.

Fortunately, fast algorithms are available for (8). For example, the alternating projection method in [31] and the Newton method in [25] both make use of the nice characterization: $(-JDJ) \succeq 0$ if and only if

$$-D \in \mathcal{K}^n_+ := \left\{ A \in \mathcal{S}^n : \mathbf{v}^T A \mathbf{v} \geq 0, \ \forall \ \mathbf{v} \in e^\perp \right\}, \quad (7)$$

where $e^\perp$ denotes the orthogonal space to the vector e. The set $\mathcal{K}^n_+$ is known as the almost positive semidefinite cone.

Therefore, the problem (6) can be seen as the orthogonal projection problem onto the intersection of the two cones $(-\mathcal{K}^n_+)$ and $\mathcal{S}^n_h$:

$$\min \ \|D - \widehat{D}\|^2 \ \text{ s.t. } \ D \in (-\mathcal{K}^n_+) \cap \mathcal{S}^n_h. \tag{8}$$

Our model in this paper largely follows this line of development, but we will have to accommodate the large number of additional constraints of the type (2) in a low-dimensional space and to handle the fact that some distances in $\widehat{D}$ are missing. We will develop our model in the next section.

3. EDM-Based Localization Scheme

In this section, we first state the SNL problem, especially on what types of constraints we intend to address. We then describe how we reformulate the problem as a convex EDM optimization. The final part is about placing the recovered sensors to the coordinate systems used by the existing anchors. If there exist no anchors, this part will not be necessary.

3.1. Problem Statement

Assume a sensor network in $\mathbb{R}^r$ ($r = 2$ or 3) has n nodes in total, with m known anchor nodes and $n - m$ sensor nodes whose locations are unknown (m may take 0). Let $\mathbf{x}_k = \mathbf{a}_k \in \mathbb{R}^r$, $k = 1, 2, \ldots, m$ denote the location of kth anchor node, and $\mathbf{x}_i \in \mathbb{R}^r$, $i = m + 1, \ldots, n$ denote the location of ith sensor node. The maximum communication range is R, which determines two index sets $\mathcal{N}_x$ and $\mathcal{N}_a$ that indicates the connectivity states of nodes. For any $(i, j) \in \mathcal{N}_x$, the Euclidean distance d_{ij} between sensor nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ is not greater than R. Hence, the two sensor nodes are able to transmit signal between each other. Similarly, for any $(i, k) \in \mathcal{N}_a$, a sensor $\mathbf{x}_i$ and an anchor node $\mathbf{a}_k$ can communicate with each other and their Euclidean distance $d_{ik} \le R$. Therefore, we have

$$\mathcal{N}_x := \{(i,j) : m+1 \le i < j \le n, \|\mathbf{x}_i - \mathbf{x}_j\| \le R\},$$
$$\mathcal{N}_a := \{(i,k) : m+1 \le i \le n, 1 \le k \le m, \|\mathbf{x}_i - \mathbf{a}_k\| \le R\}.$$

To indicate a pair of nodes that are too far away to communicate with each other, we define

$$\overline{\mathcal{N}}_x := \{(i,j) : m+1 \le i < j \le n, \|\mathbf{x}_i - \mathbf{x}_j\| > R\},$$
$$\overline{\mathcal{N}}_a := \{(i,k) : m+1 \le i \le n, 1 \le k \le m, \|\mathbf{x}_i - \mathbf{a}_k\| > R\}.$$

For any sensor nodes $\mathbf{x}_i$ and $\mathbf{x}_j$, $(i, j) \in \mathcal{N}_x \cup \mathcal{N}_a$, a noisy range measurement $\widehat{d}_{ij}$ is taken. We assume that the distance estimations are symmetric, i.e., $\widehat{d}_{ij} = \widehat{d}_{ji}$. Then the range-based sensor network localization problem can be described as to recover $\mathbf{x}_{m+1}, \mathbf{x}_{m+2}, \ldots, \mathbf{x}_n$ such that

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 \approx \widehat{d}^2_{ij}, \quad \forall (i,j) \in \mathcal{N}_x, \tag{9a}$$
$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 > R^2, \quad \forall (i,j) \in \overline{\mathcal{N}}_x, \tag{9b}$$
$$\|\mathbf{x}_i - \mathbf{a}_k\|^2 \approx \widehat{d}^2_{ik}, \quad \forall (i,k) \in \mathcal{N}_a, \tag{9c}$$
$$\|\mathbf{x}_i - \mathbf{a}_k\|^2 > R^2, \quad \forall (i,k) \in \overline{\mathcal{N}}_a. \tag{9d}$$

(9a) and (9c) come from the incomplete distance information, and (9b) and (9d) come from the connectivity information due to the limitation of radio range. That is, if there is no distance information measured between two nodes, then their Euclidean distance is greater than R. Many existing localization schemes neglect all the inequality constraints (9b) and (9d). However, as some of the existing research demonstrated, those bound constraints can actually improve the robustness and accuracy of localization. In particular, Biswas and Ye in [15] suggest to select some of these lower bound constraints based on an iterative active-constraint generation technique. We note that the total number of the constraints is order of $O(n^2)$.

3.2. EDM Reformulation

We have prepared ourselves to use EDM in Section 2. Let D be such an EDM that it satisfies

$$D_{ij} \approx \widehat{d}^2_{ij}, \quad \forall (i,j) \in \mathcal{N}_x \text{ or } \mathcal{N}_a, \tag{10a}$$
$$D_{ij} > R^2, \quad \forall (i,j) \in \overline{\mathcal{N}}_x \text{ or } \overline{\mathcal{N}}_a, \tag{10b}$$
$$D_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|^2, \quad \forall i,j \in \{1, 2, \ldots, m\}, \tag{10c}$$
$$\operatorname{rank}(-JDJ) = r. \tag{10d}$$

The first two constraints in (10) are to accommodate the constraints in (9). The third constraint includes the fixed distances among the m anchors and the last constraint enforces that the embedding points from D should be in the low-dimensional space $\mathbb{R}^r$. Our EDM optimization seeks for a best D from all matrices of satisfying those constraints in (10).

For the constraints in (10a), we use the principle of least-squares to get our objective. For other constraints, we keep them. This leads to the following optimization problem:

$$\begin{array}{ll} \min & \sum_{(i,j) \in \mathcal{N}_x \cup \mathcal{N}_a} \left(D_{ij} - \widehat{d}^2_{ij} \right)^2 \\ \text{s.t.} & D_{ij} \ge R^2, \ \forall (i,j) \in \overline{\mathcal{N}}_x \cup \overline{\mathcal{N}}_a \\ & D_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|^2, \ \forall \, i,j \in \{1, 2, \ldots, m\} \\ & \operatorname{rank}(-JDJ) = r, \ D \in (-\mathcal{K}^n_+) \cap \mathcal{S}^n_h. \end{array} \tag{11}$$

For this problem to be well defined, we consider the graph $\mathcal{G}$ formed by n nodes with its edges being $(i, j) \in \mathcal{N}_x \cup \mathcal{N}_a$. We assume that $\mathcal{G}$ is connected. Otherwise, problem (11) can be separated into smaller subproblems each corresponding to a connected subgraph in $\mathcal{G}$. The following result ensures that our problem is well defined and its effective region which contains all optimal solutions can be bounded.

Proposition 3.1. *Assume that $\mathcal{G}$ is connected. Then the optimal solution of (11) is attained. Moreover, the effective feasible region that contains all optimal solutions is bounded.*

Proof. Consider the following embedding points $\mathbf{x}_i = \mathbf{a}_i$, for $i = 1, \ldots, m$ and $\mathbf{x}_i = i(R + c)e$, for $i = m + 1, \ldots, n$, where $e \in \mathbb{R}^r$ is the vector of all ones and $c := \max\{\|\mathbf{a}_i\|\}$. Let D be the corresponding EDM generated by those points in $\mathbb{R}^r$. It is easy to verify that all constraints in (11) are satisfied. For example, for $(i,j) \in \overline{\mathcal{N}}_x$, we have

$$\|\mathbf{x}_i - \mathbf{x}_j\| = (R+c)\sqrt{r} \geq R,$$

and for $(i,k) \in \overline{\mathcal{N}}_a$, we have by triangle inequality

$$\|\mathbf{x}_i - \mathbf{x}_k\| \geq \|\mathbf{x}_i\| - \|\mathbf{x}_k\| \geq i\sqrt{r}(R+c) - c \geq R.$$

The rank of $(-JDJ)$ obviously is r. Hence, the feasible region is nonempty.

Let $f_{\rm opt}$ be the optimal objective value of (11). Then

$$f_{\rm opt} \leq f_{\rm bound} := \sum_{(i,j)\in\mathcal{N}_x\cup\mathcal{N}_a} \left(D_{ij} - \widehat{d}_{ij}^2\right)^2,$$

where D is any (fixed) feasible point. Let

$$\Omega := \left\{ D \in \mathcal{S}^n \mid D \geq 0,\ D_{ij} \leq \widehat{d}_{ij}^2 + c, \forall (i,j) \in \mathcal{N}_x \cup \mathcal{N}_a \right\},$$

where $c \geq \sqrt{f_{\rm bound}}$ is a sufficiently large number so that it intersects the feasible region of (11). It is easy to see that any optimal solution (if exists) would have to be in Ω. Otherwise the optimal objective will be larger than $f_{\rm bound}$. Hence, the effective region $\mathcal{F} \cap \Omega$ contains all the optimal solutions, where $\mathcal{F}$ denotes the feasible region of (11). Now consider any matrix D in this effective region. For any pair of nodes (i,j), the distance d_{ij} is bounded by the shortest path in the graph $\mathcal{G}$ since $\mathcal{G}$ is connected. This means that D is bounded by the elements in Ω. Hence, the effective region is bounded and closed. The optimal solution of (11) is attained. □

Let us temporarily ignore the rank constraint (10d). Then problem (11) becomes convex. We put it in a more general and compact form. Define symmetric matrices $\widehat{D}, H \in \mathcal{S}^n$ respectively by

$$\widehat{D}_{ij} := \begin{cases} \widehat{d}_{ij}^2, & \text{if } (i,j) \in \mathcal{N}_x \ \cup \ \mathcal{N}_a, \\ 0, & \text{otherwise,} \end{cases}$$

$$H_{ij} := \begin{cases} 1, & \text{if } (i,j) \in \mathcal{N}_x \ \cup \ \mathcal{N}_a, \\ 0, & \text{otherwise,} \end{cases} \tag{12}$$

and two matrices $L, U \in \mathcal{S}^n$ being lower bound and upper bound for D respectively as

$$L_{ij} := \begin{cases} \|\mathbf{a}_i - \mathbf{a}_j\|^2, & \text{if } i,j \in \{1,2,\ldots,m\} \\ R^2, & \text{if } (i,j) \in \overline{\mathcal{N}}_x \ \cup \ \overline{\mathcal{N}}_a, \\ 0, & \text{otherwise,} \end{cases}$$

$$U_{ij} := \begin{cases} \|\mathbf{a}_i - \mathbf{a}_j\|^2, & \text{if } i,j \in \{1,2,\ldots,m\} \\ R^2, & \text{if } (i,j) \in \mathcal{N}_x \ \cup \ \mathcal{N}_a, \\ M^2, & \text{otherwise,} \end{cases} \tag{13}$$

where M is a large number (e.g., $M = n\max\{\widehat{d}_{ij}, R, \|\mathbf{a}_i\|, c\}$, where c is the constant used in the proof of Prop. 3.1). Let "$\circ$" denote the Hadamard (componentwise) product of two matrices of same size, the convex relaxation of problem (11) takes the form:

$$\begin{array}{ll} \min_{D\in\mathcal{S}^n} & \frac{1}{2}\|H \circ (D - \widehat{D})\|^2 \\ \text{s.t.} & L \leq D \leq U, \\ & D \in \mathcal{S}^n_h, \ \ -D \in \mathcal{K}^n_+. \end{array} \tag{14}$$

Compared to the nearest EDM problem (8), (14) has extra lower and upper bounds constraints and the number of them is of the order $O(n^2)$. Furthermore, the objective in (14) is not strongly convex anymore as some of the weights H_{ij} are 0 (in (8), all weights are 1). Existing methods for (8) are not capable of solving this problem anymore due to the issues above.

3.3. Dealing with the Rank Constraint

Another potential issue for the convex model (14) is from dropping the rank constraint. In order for those distances in D to be "closer" to the distances in $\widehat{D}$, the points would have to be embedded in a higher dimensional space. The found positions can then be seen as the projections from this higher dimensional embedding, resulting in a network in which the sensors always tend to be crowded around the center. Weinberger *et al.* in [32] proposed a strategy for the embedding points $\mathbf{x}_i$ to be pushed away from each other. In fact, the term $\sum_{i,j}\|\mathbf{x}_i - \mathbf{x}_j\|^2$ (called the variance in [32]) is maximized. Biswas *et al.* also used the similar idea in [17] to add a regularization term in their SDP model. One benefit of maximizing this term is to alleviate the drawback of those optimization models in that they often lead to "*crowding embedding*" around their geometric center. See [26, Sect. 6.2(e)] for a numerical demonstration and also see [6, Eq.(36)] for more comments on its use from a heuristic viewpoint.

Let us explain this term a bit more. In the SDP models in both [32, 17], the positive semidefinite matrix $Y \in \mathcal{S}^n_+$ takes the form:

$$Y = X^TX, \qquad \text{with } X := [\mathbf{x}_1, \ldots, \mathbf{x}_n],$$

under the condition that those points are centralized

$$\mathbf{x}_1 + \ldots + \mathbf{x}_n = 0. \tag{15}$$

Then it is easy to verify that

$$\sum_{i,j}\|\mathbf{x}_i - \mathbf{x}_j\|^2 = 2n\sum_{i=1}^n \|\mathbf{x}_i\|^2 = 2n\langle I,\ Y\rangle = 2n\text{tr}(Y).$$

Using the above fact, the minimization SDP models in [32, 17] add the term $-\text{tr}(Y)$ to their objective. In other words, they try to maximize the trace. In convex optimization, it is widely known [33] that maximizing trace is about maximizing the rank, a contradiction to the original purpose of minimizing the rank.

Now coming back to the distances, we know from (4) that the embedding points in X satisfy

$$2\mathrm{tr}(Y) = 2\mathrm{tr}(X^TX) = \mathrm{tr}(-JDJ) = -\langle J,\ D\rangle, \qquad (16)$$

where we used the fact $J^2 = J$. We also note that it follows from (4) that the centralization condition (15) is met for those points in X.

Based on the reasoning above, maximizing the variance (trace) can only be interpreted as pushing the embedding points far from each other and cannot be used to minimize rank. We introduce below a new technique to reduce the rank. Our intention here, under the condition that the variance $\mathrm{tr}(-JDJ)$ is made as large as possible, is to decrease the rank of $(-JDJ)$. Let $V \in \mathcal{S}^n$ be a matrix of rank r. It follows from Fan's inequality (see, e.g., [34, Thm. 1.2.1]) that

$$\langle V,\ -JDJ\rangle \le \sum_{i=1}^{r} \mu_i(V)\lambda_i(-JDJ),$$

where $\mu_i(V) \ge \ldots \ge \mu_r(V)$ and $\lambda_1(-JDJ) \ge \ldots \ge \lambda_n(-JDJ)$ are respectively the eigenvalues of V and $(-JDJ)$ in nonincreasing order. And the equality holds if and only if both V and $(-JDJ)$ achieve simultaneous ordered spectral decomposition. If we maximize the term $\langle V,\ -JDJ\rangle$ under the constraints in (11) (recall its feasible region is bounded from Prop. 3.1), then the matrix $(-JDJ)$ is likely to be a low-rank matrix.

Let us consider the possible choice of V. Suppose each dimension defined by the eigenvectors of matrix V is equally important. This translates to the leading eigenvalues being equal. We set them to be 1 for simplicity:

$$\mu_i(V) = 1, \qquad i = 1,\ \ldots,\ r.$$

This leaves the eigen-space corresponding to the r leading eigenvalues to be approximated. Suppose we have an initial solution $\widetilde{D}$, which, for example, may be obtained by the shortest path distances in $\mathcal{G}$, as used in MDS-MAP. Let

$$V = P_1P_1^T \qquad \text{with} \quad P_1 = [\mathbf{p}_1, \ldots, \mathbf{p}_r],$$

where $\mathbf{p}_i$ is the orthogonal eigenvector of $(-J\widetilde{D}J)$ corresponding to its ith largest positive eigenvalue. Since e is an eigenvector of $(-J\widetilde{D}J)$ due to $Je = 0$, we have

$$JVJ = \Big(I - \frac{1}{n}ee^T\Big)P_1P_1^T\Big(I - \frac{1}{n}ee\Big) = P_1P_1^T = V.$$

Hence,

$$\langle V,\ -JDJ\rangle = \langle JVJ,\ -D\rangle = \langle V,\ -D\rangle. \qquad (17)$$

By subtracting the regularization term (16) (to achieve maximal variance) and the term (17) (to reduce the rank) from the objective function of (14), we have our final convex optimization model:

$$\begin{array}{lll} \min_{D\in\mathcal{S}^n} & \frac{1}{2}\|H\circ(D-\widehat{D})\|^2 + \nu\,\langle J, D\rangle + \langle V, D\rangle & \\ \text{s.t.} & L \le D \le U, & (18) \\ & D \in \mathcal{S}^n_h,\ \ -D \in \mathcal{K}^n_+, & \end{array}$$

where $\nu \ge 1$ is the parameter that balances the trade-off between preserving the local distances, maximizing the variance, and reducing the rank. Our numerical experiment will show that the solution compared to (14) always enjoys the low rank property.

We end this part by a brief technical note that explains why (18) is likely to induce a low-rank solution. We regroup the two regularization terms in (18) as follows:

$$\begin{aligned} q(D) :&= \nu\,\langle J, D\rangle + \langle V, D\rangle \\ &= (\nu+1)\,\langle J, D\rangle + \Big(-\langle J, D\rangle + \langle V, D\rangle\Big) \\ &= (\nu+1)\,\langle J,\ D\rangle + (\langle -J,\ D\rangle + \langle JVJ,\ D\rangle) \\ &\qquad \text{(using } J^2 = J \text{ and (17))} \\ &= (\nu+1)\,\langle J,\ D\rangle + \underbrace{\langle -J(I-V)J,\ D\rangle}_{=:q_0(D)} \end{aligned}$$

We now consider another function:

$$p(D) := \langle I,\ -JDJ\rangle - \sum_{i=1}^{r}\lambda_i(-JDJ).$$

This function has a nice property that $p(D) \ge 0$ for all D being EDM and $P(D) = 0$ if and only if D has embedding dimension r (i.e., the rank of $(-JDJ)$ is r). We call $p(D)$ a rank function of D. Hence, one may like to minimize $p(D)$ in order to achieve a low-dimensional embedding. However, the rank function $p(D)$ is a concave function (the negative of the sum of the largest leading eigenvalues of a symmetric matrix is a concave function of the matrix). It turns out that $p(D)$ can be easily majorized at the initial point $\widetilde{D}$:

$$p(D) \le p(\widetilde{D}) + \langle J(I-V)J,\ \widetilde{D}\rangle + q_0(D).$$

This is because $(-J(I-V)J)$ is a subgradient of $p(\cdot)$ at $\widetilde{D}$. Therefore, one would like to minimize $q_0(D)$ instead of $p(D)$ itself to get a convex relaxation of $p(D)$. This leads to the regularized term $q_0(D)$ hidden in (18), where D is required to be of EDM.

3.4. Global Coordinates Recovery

After obtaining a solution $\overline{D}$ of (18), cMDS is then applied to obtain the coordinates of the sensors. If there are anchors ($m > 0$), then a matching procedure must be carried out to place the found coordinates to the coordinate system used by the anchors. This matching procedure is known as the Procrustes analysis on the obtained points.

We include the detail below for completion (more on this, see [35]).

Suppose $\overline{X}$ is obtained by cMDS (4) with D replaced by $\overline{D}$. Denote

$$\overline{X} = [\overline{X}_1, \overline{X}_2], \text{ with } \overline{X}_1 \in \mathbb{R}^{r\times m}, \ \overline{X}_2 \in \mathbb{R}^{r\times(n-m+1)},$$

where $\overline{X}_1$ contains m points corresponding to the m anchors. whose true positions are collected in $A = [\mathbf{a}_1, \ldots, \mathbf{a}_m]$. Our first task is to match $\overline{X}_1$ to A. Since the distances among the anchors are kept as hard constraints in (18), we must have (ignore the computational errors)

$$\|\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j\| = \|\mathbf{a}_i - \mathbf{a}_j\|, \quad i,j = 1,\ldots,m.$$

This means that those points can be matched perfectly through rotation. To do this, we first centralize the points in $\overline{X}_1$ and A by

$$\overline{X}_1^c := \overline{X}_1 J_m \quad \text{and} \quad A^c := AJ_m.$$

Denote

$$\overline{\mathbf{a}} := \frac{1}{m}\sum_{i=1}^m \mathbf{a}_i \qquad \text{and} \qquad \overline{\mathbf{x}} := \frac{1}{m}\sum_{i=1}^m \overline{\mathbf{x}}_i.$$

We then find the best rotation matrix Q by solving

$$\min_{Q\in\mathbb{R}^{r\times r}} f = \|Q\overline{X}_1^c - A^c\|, \quad \text{s.t. } Q^TQ = I_r. \qquad (19)$$

Problem (19) has a closed form solution $Q = U_\Sigma V_\Sigma^T$, where U_Σ and V_Σ are from the singular-value-decomposition of $A^c(\overline{X}_1^c)^T = U_\Sigma \Sigma V_\Sigma^T$. Then the anchors will be fully recovered through

$$\mathbf{a}_i = Q\overline{\mathbf{x}}_i + \overline{\mathbf{a}}, \quad i = 1,\ldots,m.$$

The remaining points will be mapped to

$$\mathbf{x}_i = Q(\overline{\mathbf{x}}_i - \overline{\mathbf{x}}) + \overline{\mathbf{a}}, \quad i = m+1,\ldots,n. \qquad (20)$$

We summarize our EDM-based localization scheme as the algorithm EDM-SNL below.

Algorithm 1 EDM-SNL

```
1: Distance Reconstruction: Set $\widehat{D}$ and $H$ by (12), $L$ and $U$ by (13) as input data, solve (18) to get estimated full distance matrix $\overline{D}$.
2: cMDS: Compute the sensor positions $\overline{X}$ by (4) by replacing $D$ by $\overline{D}$.
3: if Anchors exist then
4:     Solve (19) to get rotation matrix $Q$ and compute sensor positions by (20). Return $X_2$ as recovered sensor coordinates.
5: end if
```

In the next section, we will show that problem (18) in the first step of the above algorithm can be efficiently solved by an alternating minimization method.

4. A Convergent 3-Block ADMM Algorithm

The key differences of the problem (18) from (8) include possible zero H weights in the former (vs the unity weights in the latter) and a large number of lower and upper bounds constraints. Coupled with the conic constraints $D \in (-\mathcal{K}_+^n) \cap \mathcal{S}_h^n$, those key differences make the existing methods for (8) not applicable anymore. Here, we choose to apply a recent ADMM (alternating direction method of multipliers) of Sun *et al.* in [27] for general convex quadratic programming. To describe this method, we need to reformulate (18).

4.1. Reformulation by Vectorization

Let $\mathcal{N} := \mathcal{N}_x \cup \mathcal{N}_a$. Note that $\mathcal{N}_x \cap \mathcal{N}_a = \emptyset$, so we have $|\mathcal{N}| = |\mathcal{N}_x| + |\mathcal{N}_a|$, where $|\mathcal{N}|$ is the cardinality of set $\mathcal{N}$. To simplify our notation, we denote $|\mathcal{N}| = N$ and

$$\mathcal{N} = \{(i_1,j_1),(i_2,j_2),\ldots,(i_N,j_N)\}.$$

Define the linear operator $\mathcal{A}_1 : \mathcal{S}^n \to \mathbb{R}^N$ by

$$\mathcal{A}_1(X) := (\langle A_1, X\rangle, \langle A_2, X\rangle, \ldots, \langle A_N, X\rangle)^T, \qquad (21)$$

where A_k is the symmetric selection matrix defined as

$$A_k := \frac{1}{2}(e_{i_k}e_{j_k}^T + e_{j_k}e_{i_k}^T), \ (i_k,j_k) \in \mathcal{N},$$

and e_i is the ith column vector of the identity matrix. Apparently, we have $\langle A_k, D\rangle = D_{i_kj_k}$ (hence the name of the selection matrix). Furthermore, let $\mathbf{b}_1 = \mathcal{A}_1(\widehat{D})$. We then have

$$\|H \circ (D - \widehat{D})\|^2 = \|\mathcal{A}_1(D) - \mathbf{b}_1\|^2.$$

We further define linear operator $\mathcal{A}_2 : \mathcal{S}^n \to \mathbb{R}^n$, the matrix $C \in \mathcal{S}^n$, and closed convex set $\mathcal{C}$ respectively as

$$\mathcal{A}_2(X) := \mathrm{diag}(X), \quad C := \nu J + V,$$

and

$$\mathcal{C} := \{W \in \mathcal{S}^n : L \le W \le U\},$$

and the problem (18) can be equivalently written as

$$\begin{array}{ll} \min_{D\in\mathcal{S}^n} & \frac{1}{2}\|\mathcal{A}_1(D) - \mathbf{b}_1\|^2 + \langle C, \ D\rangle \\ \text{s.t.} & \mathcal{A}_2(D) = 0, \\ & D \in \mathcal{C}, \ -D \in \mathcal{K}_+^n. \end{array} \qquad (22)$$

4.2. Lagrangian Dual Problem

Directly solving problem (22) by ADMM is applicable, however, it has been discovered to be much less efficient than working on its dual problem. To derive the Lagrangian dual of (22), we introduce a new variable $\mathbf{t}$:=

$A_1(D)-\mathbf{b}_1\in\mathbb{R}^N$ and a slack variable $W\in\mathcal{S}^n$, then (22) is equivalent to

$$\begin{array}{ll} \min\limits_{D\in\mathcal{S}^n,\mathbf{t}\in\mathbb{R}^N,W\in\mathcal{S}^n} & \frac{1}{2}\|\mathbf{t}\|^2+\langle C,D\rangle+\delta_{\mathcal{C}}(W)+\delta_{\mathcal{K}^n_+}(-D)\\ \text{s.t.} & \mathcal{A}(D)-B\mathbf{t}=\mathbf{b},\\ & D=W,\end{array} \tag{23}$$

where $\mathcal{A}:\mathcal{S}^n\to\mathbb{R}^N\times\mathbb{R}^n$ is the linear mapping defined by

$$\mathcal{A}(D):=(\mathcal{A}_1(D);\mathcal{A}_2(D)),\quad \mathbf{b}:=(\mathbf{b}_1;\mathbf{0}^n)\in\mathbb{R}^{N+n},$$

and $B:=[I_N;\mathbf{0}^{n\times N}]\in\mathbb{R}^{(N+n)\times N}$, I_N is the identity matrix with order N, $\mathbf{0}^{n\times N}$ is the matrix of all zeroes with order $n\times N$. For any given set Ω, $\delta_\Omega(\cdot)$ is the indicator function over Ω such that $\delta_\Omega(u)=0$ if $u\in\Omega$ and ∞ otherwise. Let the Lagrangian function of the above problem be defined by

$$\begin{aligned} L(D,\mathbf{t},W;\mathbf{y},Z):=&\ \frac{1}{2}\|\mathbf{t}\|^2+\langle C,D\rangle+\delta_{\mathcal{C}}(W)+\delta_{\mathcal{K}^n_+}(-D)\\ &-\langle\mathcal{A}(D)-B\mathbf{t}-\mathbf{b},\mathbf{y}\rangle-\langle D-W,Z\rangle, \end{aligned}$$

where $\mathbf{y}\in\mathbb{R}^{N+n}$ and $Z\in\mathcal{S}^n$ are the Lagrangian multipliers corresponding to the two constraints. The Lagrangian dual problem is

$$\max_{\mathbf{y},Z}\left\{\min_{D,\mathbf{t},W}L(D,\mathbf{t},W;\mathbf{y},Z)\right\}. \tag{24}$$

By solving the insider optimization problem, the Lagrangian dual problem can be derived as

$$\begin{array}{ll} \max_{\mathbf{y},Z,S} & -\frac{1}{2}\|B^T\mathbf{y}\|^2+\langle\mathbf{b},\mathbf{y}\rangle-\delta^*_{\mathcal{C}}(-Z)\\ \text{s.t.} & Z+\mathcal{A}^*\mathbf{y}-S-C=0,\\ & S\in(\mathcal{K}^n_+)^*, \end{array} \tag{25}$$

where $(\mathcal{K}^n_+)^*$ is the dual cone of $\mathcal{K}^n_+$ and $\delta^*_{\mathcal{C}}(\dot{})$ is the conjugate function of $\delta_{\mathcal{C}}(\cdot)$ given by

$$\delta^*_{\mathcal{C}}(-Z)=\sup_{W\in\mathcal{C}}\langle -Z,W\rangle=-\inf_{W\in\mathcal{C}}\langle Z,W\rangle. \tag{26}$$

We note that the insider optimization problem in (24) also gives the relationship between $\mathbf{t}$ and $\mathbf{y}$: $\mathbf{t}=-B^T\mathbf{y}$.

4.3. An ADMM Algorithm

Problem (25) is well structured with three separable block variables. The ADMM of Sun *et al.* in [27] can be directly applied to get the optimal solution. The algorithm actually alternatively minimizes the associated Augmented Lagrangian defined by (note: here we cast (25) as a minimization problem)

$$\begin{aligned} L_\sigma(\mathbf{y},Z,S;D):=&\ \frac{1}{2}\|B^T\mathbf{y}\|^2-\langle\mathbf{b},\mathbf{y}\rangle+\delta^*_{\mathcal{C}}(-Z)\\ &+\langle Z+\mathcal{A}^*\mathbf{y}-S-C,D\rangle\\ &+\frac{\sigma}{2}\|Z+\mathcal{A}^*\mathbf{y}-S-C\|^2, \end{aligned}$$

where $\sigma>0$ is a given parameter and $D\in\mathcal{S}^n$ is the Lagrangian multiplier corresponding to the equality constraint in (25). The proposed ADMM algorithm is described below.

Let $\sigma>0$ and $\tau\in(0,\infty)$, set $\left(\mathbf{y}^0,Z^0,S^0,D^0\right)\in\mathbb{R}^{N+n}\times\mathcal{S}^n\times\mathcal{S}^n\times\mathcal{S}^n$ as initial point. For iteration $k=1,2,\ldots$ perform the kth iteration as follows

$$\left\{\begin{array}{l} \bar{\mathbf{y}}^k=\arg\min_{\mathbf{y}\in\mathbb{R}^{N+n}}L_\sigma(y,Z^k,S^k;D^k),\\ Z^{k+1}=\arg\min_{Z\in\mathcal{S}^n}L_\sigma(\bar{\mathbf{y}}^k,Z,S^k;D^k),\\ \mathbf{y}^{k+1}=\arg\min_{y\in\mathbb{R}^{N+n}}L_\sigma(\mathbf{y},Z^{k+1},S^k;D^k),\\ S^{k+1}=\arg\min_{S\in(\mathcal{K}^n_+)^*}L_\sigma(\mathbf{y}^{k+1},Z^{k+1},S;D^k),\\ D^{k+1}=D^k+\tau\sigma\left(Z^{k+1}+\mathcal{A}^*\mathbf{y}^{k+1}-S^{k+1}-C\right),\\ \mathbf{t}^{k+1}=-B^T\mathbf{y}^{k+1}. \end{array}\right. \tag{27}$$

The convergence analysis of the algorithm above as well as the necessity of having the first update of $\bar{\mathbf{y}}$ can be found in [27]. We omit the details here for simplicity.

For any given set $\Omega\in\mathcal{S}^n$, we let $\Pi_\Omega(A)$ denote the orthogonal projection of $A\in\mathcal{S}^n$ onto Ω. The subproblems in (27) all have closed-form solutions involving orthogonal projections. We omit the detailed (technical) computations for them (they can be verified directly):

$$\begin{aligned} \bar{\mathbf{y}}^k&=\left(BB^T+\sigma\mathcal{A}\mathcal{A}^*\right)^{-1}\left(\mathbf{b}-\mathcal{A}(Z^k+\widehat{W}^k)\right)\\ Z^{k+1}&=\widehat{Z}^k+\sigma^{-1}\Pi_{\mathcal{C}}(-\sigma\widehat{Z}^k)\\ \mathbf{y}^{k+1}&=\left(BB^T+\sigma\mathcal{A}\mathcal{A}^*\right)^{-1}\left(\mathbf{b}-\mathcal{A}(Z^{k+1}+\widehat{W}^k)\right)\\ S^{k+1}&=\Pi_{(\mathcal{K}^n_+)^*}\left(Z^{k+1}+\mathcal{A}^*\mathbf{y}^{k+1}-C+\sigma^{-1}D^k\right) \end{aligned}$$

where $\widehat{W}^k:=\sigma^{-1}D^k-S^k-C$, and $\widehat{Z}^k:=S^k-\mathcal{A}^*\bar{\mathbf{y}}^k-\sigma^{-1}D^k+C$. Because $\mathcal{C}$ is a box-type constraints (lower and upper bounds), the projection $\Pi_{\mathcal{C}}(A)$ is easy to compute. For the projection $\Pi_{(\mathcal{K}^n_+)^*}(A)$, we use the Moreau formula:

$$\Pi_{(\mathcal{K}^n_+)^*}(A)=A+\Pi_{\mathcal{K}^n_+}(-A),$$

and the Gaffke-Mather formula [36]:

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

Hence, $\Pi_{(\mathcal{K}^n_+)^*}(A)=\Pi_{\mathcal{S}^n_+}(JAJ)$, which can be calculated through the eigen-decomposition of JAJ. We note that the matrix $(BB^T+\sigma\mathcal{A}\mathcal{A}^*)$ is a positive definite diagonal matrix in $\mathcal{S}^{N+n}$. The computation complexity of JAJ is in the order $O(n^2)$ because of the structure in J. Hence, the major computational complexity in (27) is dominated by the eigen-decomposition of JAJ in computing S^{k+1}.

We directly adopt the measurement in [27] on the accuracy of an approximate optimal solution for (22) and its dual (25) by using the following relative residual obtained from their general optimality conditions (KKT conditions):

$$\eta_e(\mathbf{t},\mathbf{y},D,Z,S)=\max\left\{\eta_P,\eta_D,\eta_Z,\eta_{C_1},\eta_{C_2}\right\}, \tag{28}$$

where

$$\eta_P = \frac{\|\mathcal{A}(D) - B\mathbf{t} - \mathbf{b}\|}{1 + \|\mathbf{b}\|}, \quad \eta_D = \frac{\|Z + \mathcal{A}^*\mathbf{y} - S - C\|}{1 + \|C\|},$$

$$\eta_Z = \frac{\|D - \Pi_{\mathcal{C}}(D - Z)\|}{1 + \|D\| + \|Z\|}, \quad \eta_{C_1} = \frac{|\langle S, D\rangle|}{1 + \|S\| + \|D\|},$$

$$\eta_{C_2} = \frac{\|D + \Pi_{\mathcal{K}^n_+}(-D)\|}{1 + \|D\|}.$$

We note that η_P measures the violation of the first equation constraint in the primal problem (23); η_D for the violation of the equation constraint in the dual problem (25); η_Z for the violation of D belonging to $\mathcal{C}$; η_{C_1} measures the complementarity condition between S and D; and finally η_{C_2} measures the violation of $-D$ belonging to $\mathcal{K}^n_+$. We terminate the algorithm if the maximum of the violations is below certain level, i.e.,

$$\eta_e(\mathbf{t}^k, \mathbf{y}^k, D^k, Z^k, S^k) \le \text{tol}, \tag{29}$$

where tol is a given tolerance.

5. Experimental Results

In this section, we conducted numerical experiments using MATLAB (R2015a) on a desktop of 4GB memory and Intel(R) Core(TM) i5-2500 3.3GHz CPU to evaluate the performance of the proposed EDM-SNL, whose parameters are set as

$$\nu = 10, \ \text{tol} = 10^{-3},$$

and the initial points $\mathbf{y}^0, Z^0, S^0, D^0$ are all zeros in the corresponding space. Below we first briefly discuss the numerical methods that we plan to compare with and why we choose them. We then generate three classes of test problems having regular and irregular topologies. We finally present the comparison results, which show that EDM-SNL is overall more robust and faster on the tested problems.

5.1. Benchmark Methods

We select two representative state-of-the-art methods: ARAP in [13] and SFSDP in [19] as benchmark methods. ARAP is a MDS-based localization scheme that "stitches" together local structures of sensor nodes. Shown by the authors, ARAP outperforms other MDS-based methods such as SMACOF and MDS-MAP(P, R). SFSDP is a sparse version of the full SDP localization scheme proposed by Biswas and Ye [15] and it achieves results with similar accuracy but using much less computation time. There are other methods that also produce comparable results, but their codes are not publicly available.

For ARAP, we use its default parameters. However, ARAP is mainly for the anchor-free localization problem (i.e., $m = 0$ in our case). But it can be used to deal with the case $m \neq 0$. One would have to do an extra step to place the generated localizations by ARAP to the coordinate system used by the anchors. The step is very similar to the one described in Subsect. 3.4. For SFSDP, we keep all default parameters except the SDP solver option and the objective function option. We set *pars.SDPsolver* = 'sedumi' because by our observation, SeDuMi always gives us a solution with better accuracy than SDPA (a sover used by SFSDP for SDPs). We set *pars.objSW* = 3 to add the regularization term (the trace term) to the SDP objective function. It is reported in [17] that the use of regularization in SDP provides notable improvement for the networks with random anchor distribution. We believe in these settings, SFSDP would have the best performance in terms of localization accuracy.

Another important feature of SFSDP is its refinement step, which was previously used by [17]. The refinement is a heuristic step that uses the steepest gradient method to improve the quality of the final localization. The general view on the refinement step is that it more often than not improves the quality, but not always. For more detail on this, see [17]. We will report both the results with and without the refinement step for all algorithms. However, it will be seen in Section 5.4 that when the network is in a square region, ARAP actually benefits little from the refinement step.

5.2. Test Examples

We follow the standard framework in generating the noisy distances:

$$\widehat{d}_{ij} = d_{ij} \times |1 + nf \times randn|, \quad \forall\ (i,j) \in \mathcal{N}_x \cup \mathcal{N}_a,$$

where d_{ij} is the true Euclidean distance between nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ which will be generated soon, $0 \le nf \le 1$ is the noise factor, and $randn$ is the standard normal random variable. We generate $\mathbf{x}_i$ in the following three examples, the first having square region layout and the rest two having irregular topologies.

Example 5.1. *(Square Network) In a square region of* $[0, 100]^2$*, we randomly generate n points* $\mathbf{x}_i$ *following the uniform distribution. Two types of anchor positions are considered: placed at corners or randomly.*

Example 5.2. *(Corridor Network) We generate n points* $\mathbf{x}_i$ *following a uniform distribution in a region that looks like a corridor, as shown in Fig. 2. The corridors are formed by creating two rectangular gaps inside the square* $[0, 100]^2$*.*

Example 5.3. *(EDM Network) We randomly generated n points* $\mathbf{x}_i$ *in a region whose shape is similar to the letter "E", "D" and "M". The ground truth network is depicted in Fig.3.*

To measure the accuracy of the estimated positions, we simply adopt the commonly used root mean square

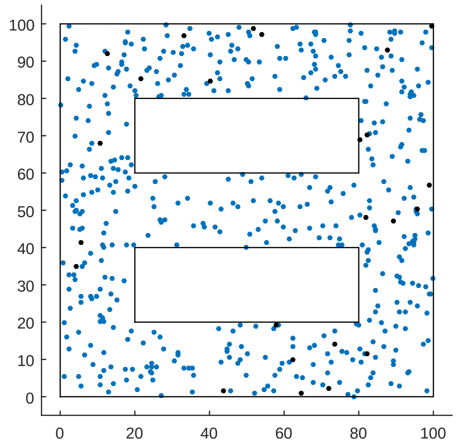

Figure 2: Ground truth *Corridor*network with $n = 494$ nodes in total, among which are $m = 24$ anchor nodes randomly distributed (colored in black).

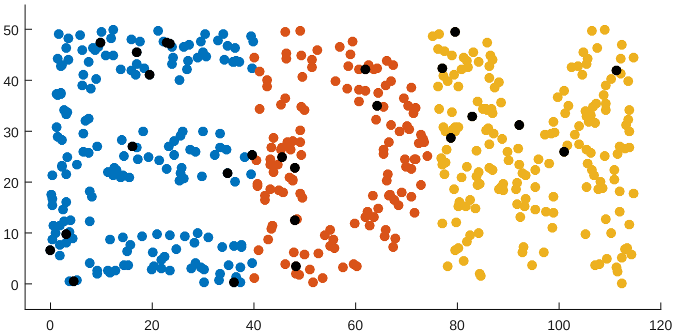

Figure 3: Ground truth *EDM*network with $n = 511$ nodes in total, among which are $m = 25$ anchor nodes randomly distributed (colored in black).

distance (RMSD)

$$\mathrm{RMSD} := \frac{1}{\sqrt{n-m}} \left(\sum_{i=m+1}^{n} \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|^2 \right)^{\frac{1}{2}}, \tag{30}$$

where $\hat{\mathbf{x}}_i$ is the estimated position and $\mathbf{x}_i$ is the ground-truth sensor position.

5.3. Performance Comparison on Quality of Localizations

Before we report more detailed numerical comparison, we would like to emphasize one important fact that the flip ambiguity does not always occur. Moreover, it is impossible to know beforehand whether it is going to happen given a set of distance observations in $\widehat{D}$. To make it worse, its occurrence seems to be dependent on algorithms and their initial points being used. However, when occurred, it would severely degrade the localization of the whole network. A simulation result from randomly generated network is depicted in Fig.4 to illustrate the importance of flip ambiguity mitigation. As shown in Fig.4b, the lower left and upper right parts of the sensors are completely folded over across the dotted line towards the center of the region, and causes a large localization error by SFSDP, while EDM-SNL recovers the sensor positions correctly in Fig. 4a.

Now we report our observation on the general performance of all methods using the three test problems. The results are the average over 50 randomly generated instances for each problem. Fig.5 depicts the variation of localization error (i.e. RMSD normalized by communication range R) from each method for Example 5.1 (with $n = 200$ and various anchor positions) when the noise in the distance measurements increases.

In Fig. 5a, four anchors are placed at four corners of the square region, i.e., $[0, 0]$, $[1, 100]$, $[100, 0]$ and $[100, 100]$. It can be observed that when the noise factor is less than 0.3, all three methods performed similarly well. However, as the noise keeps increasing, EDM-SNL outperforms both ARAP and SFSDP in terms of localization accuracy, and controls the localization error around $50\%R$ even when the noise factor is as large as 0.9. When the anchor distribution type is changed to random, as shown in Fig. 5b, SFSDP got a large impact on its localization accuracy, while EDM-SNL and ARAP are steadily getting worse when the noise level gets bigger. Overall, EDM-SNL performed best. As expected, with more anchors being used, all three methods obtained better quality of localization and this can be observed from Fig. 5c and Fig. 5d.

Now we test all the methods on Example 5.2 and Example 5.3, both of which have irregular topology. It is always important for a localization system to be resistance to irregular wireless sensor networks, because many applications require sensor network to be restricted in a certain region. The main challenge of this problem comes from its multiscale structure and nonconvex region. Such characteristics of irregular layout would cause large localization error for ARAP in its patching and stitching procedures when the distance measurements contain large noise. For example, for the corridor network in Example 5.2 (it contains total 494 nodes randomly (uniformly) distributed in a nonconvex region and $m = 24$ of them are anchors, and the communication radius of sensor/anchor nodes is $R = 12$), the localization error by three methods under different noise factors are compared in Fig. 6. It can be observed that when the noise factor is less than 0.3, EDM-SNL and ARAP both work well and ARAP is slightly better. However, when the noise factor goes beyond 0.3, the performance of ARAP degrades dramatically, whereas EDM-SNL is still stable and kept the localization error under $30\%R$.

We further tested Example 5.3 with $n = 511$ and $m = 25$. The communication radius of sensor/anchor nodes is $R = 10$. Localization error comparison is shown in Fig. 7. Like the results for Example 5.2, when the noise factor is less than 20%, all three methods perform well and ARAP is slightly better. When the noise is greater than 0.5, our method still controls the error under $20\%R$, while ARAP and SDP recovered sensor positions with relatively large

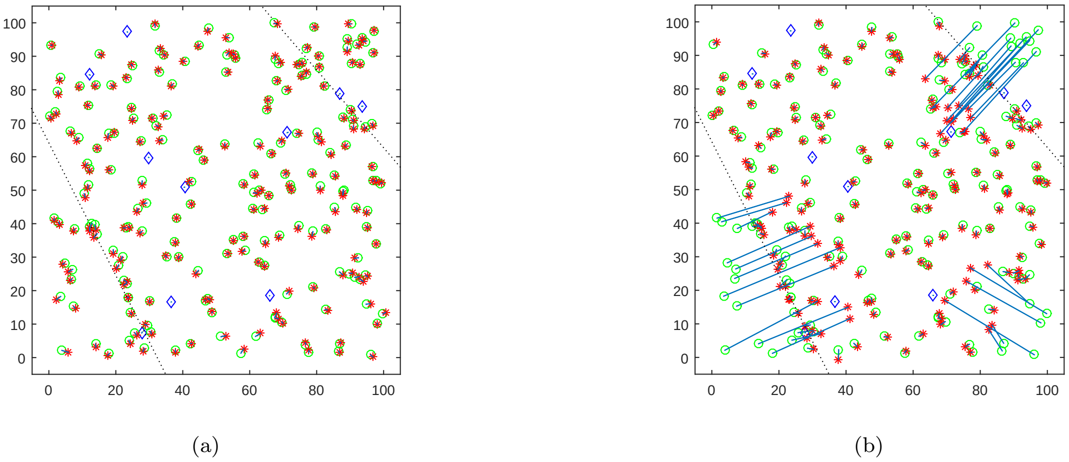

Figure 4: Network generated in Example 5.1 with $n = 200$ and $m = 10$ anchors randomly distributed. Noise factor $nf = 0.1$. Communication radius $R = 20$. (a) EDM-SNL: Localization Error $= 3.93\%R$. (b) SFSDP: Localization Error $= 52.62\%R$. (Blue diamond: anchor position. Green circle: original sensor position. Red star: estimate sensor position. Blue line: error offset between original and estimate sensor position.)

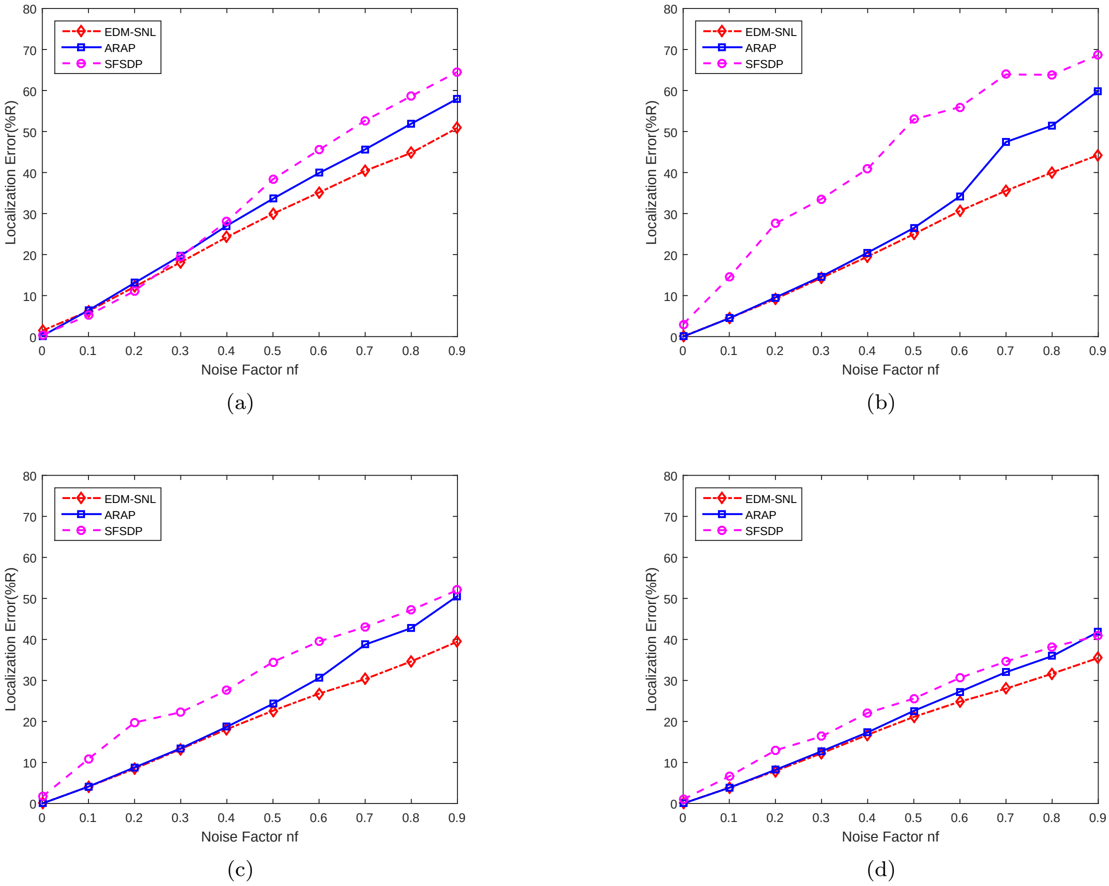

Figure 5: Variation of localization error with varying noise factor and anchor distribution type. Networks of totally 200 nodes with communication radius $R = 20$. (a) 4 anchor nodes distributed at corners. (b) 10 anchor nodes randomly (uniformly) distributed. (c) 20 anchor nodes randomly (uniformly) distributed. (d) 40 anchor nodes randomly (uniformly) distributed.

error. Qualitative results are shown in Fig. 8. When the noise factor goes to as large as 0.6, EDM-SNL is still capable of generating sensor nodes that form the shape of the *EDM* network, especially for the rightmost letter (‘M’),

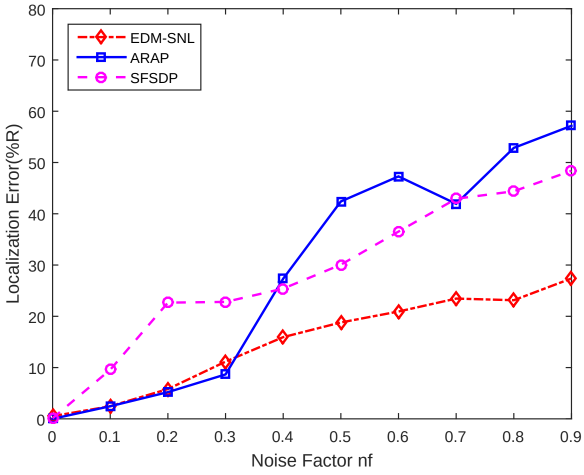

Figure 6: Variation of localization error with varying noise factor. Networks of totally 494 nodes, among which are 24 anchor nodes distributed randomly. Communication radius $R = 12$.

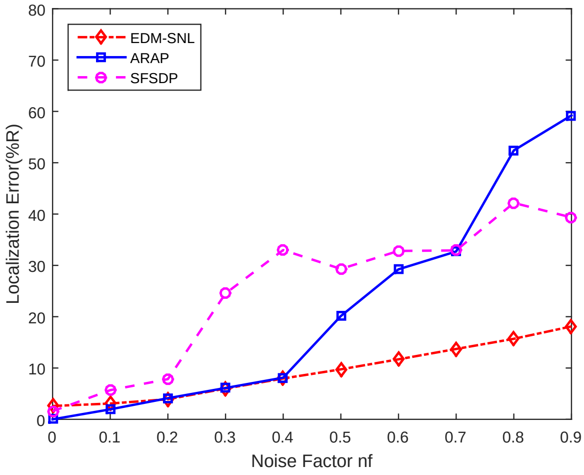

Figure 7: Variation of localization error with varying noise factor. Networks of totally 511 nodes, among which are 25 anchor nodes distributed randomly. Communication radius $R = 10$.

leading to the superior localization results. One of the reasons that make EDM-SNL robust to large noise measurement is that the upper bound constraint in (14) makes sure the distance between two sensor nodes that have communication with each other is less than the communication radius R, no matter how large the input distance is detected.

5.4. Performance Comparison on Computation Time

We use Example 5.1 and Example 5.2 to test the computation efficiency of our EDM-based localization approach for networks with regular and irregular topology respectively, we run simulations on networks with various number of sensor and anchor nodes randomly (uniformly) distributed. The total number of (sensor/anchor) nodes n varies as $n = 200, 500$ and 1000, and the number of anchor nodes $m = 5\%, 10\%$ and 20% of n. The noise factor nf is fixed at $nf = 0.4$ and the communication radius R varies in order to control the average degree of nodes. For each specific network, we randomly generated 50 different inputs (sparse and noisy distance matrices) and the average results are reported in Table 1 for square networks and Table 2 for corridor networks, which contain the execution time and the corresponding localization error for all three methods before and after refinement step.

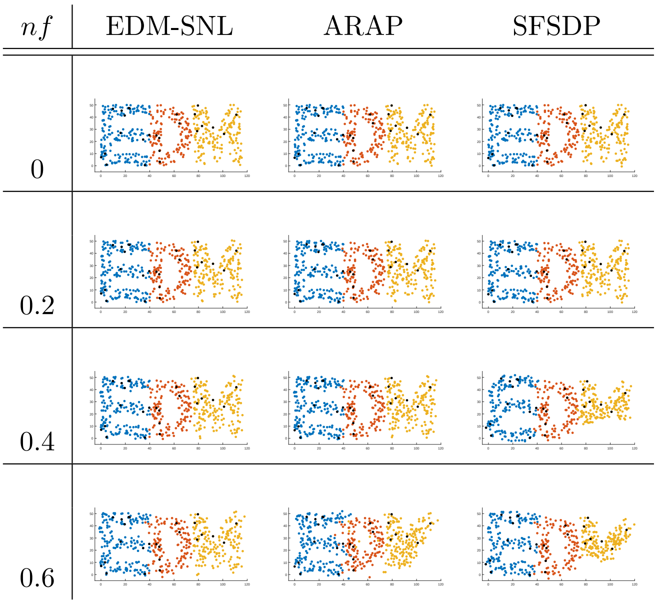

Figure 8: Qualitative localization results of *EDM* network, comparing EDM-SNL, ARAP, SFSDP with different noise level

From both Table 1 and Table 2, it can be observed that the refinement step usually reduces the localization error by some extent, except for the EDM-SNL when $n = 500$ and $m = 5\%n$, $10\%n$ in Table 1, and the additional computation time is only around 1 second. ARAP benefits the least from refinement among three methods since it actually uses a stress majorization step, which is similar to the refinement we used here, in localizing each small patch to improve the local embedding in its whole process.

The computation time of both the EDM-SNL and SFSDP will benefit from the increasing number of anchor nodes since more prior knowledge is used in the localization scheme. It can be observed from Table 1 and Table 2 that our approach is the fastest among three methods with comparable values of RMSD for all cases tested. For small networks, e.g. $n = 200$, our approach only took around 7 seconds to achieve a localization result with the lowest error among three methods. When the number of nodes goes to thousand, e.g. $n = 1000$ in Table 2, EDM-SNL used approximate 100 seconds less time than ARAP for the case $m = 20\%n$, and the difference of localization error from two schemes is only as small as 0.07.

Table 1: Execution Time Results of Square Network

Test Problems		Localization					
$n(R)$	m	Scheme	RMSD	RMSD.Re	Time	Time.Re	Total Time
200(20)	5%n	EDM-SNL	4.18E+00	3.74E+00	6.34	0.10	6.44
		ARAP	4.98E+00	3.90E+00	12.90	0.12	13.02
		SFSDP	1.47E+01	7.11E+00	38.71	0.14	38.85
	10%n	EDM-SNL	3.76E+00	3.44E+00	4.97	0.08	5.05
		ARAP	4.96E+00	3.57E+00	12.91	0.07	12.97
		SFSDP	1.30E+01	5.81E+00	34.28	0.08	34.36
	20%n	EDM-SNL	3.15E+00	3.12E+00	2.91	0.04	2.94
		ARAP	4.79E+00	3.35E+00	12.97	0.06	13.02
		SFSDP	1.12E+01	4.01E+00	24.40	0.04	24.44
500(15)	5%n	EDM-SNL	1.89E+00	1.98E+00	50.29	0.20	50.49
		ARAP	2.16E+00	1.99E+00	64.09	0.15	64.24
		SFSDP	1.19E+01	3.04E+00	127.47	0.43	127.91
	10%n	EDM-SNL	1.92E+00	1.89E+00	32.28	0.12	32.40
		ARAP	2.14E+00	1.90E+00	64.76	0.13	64.89
		SFSDP	9.81E+00	3.09E+00	125.10	0.22	125.32
	20%n	EDM-SNL	1.48E+00	1.81E+00	29.68	0.07	29.75
		ARAP	2.13E+00	1.81E+00	64.60	0.09	64.68
		SFSDP	8.85E+00	2.03E+00	88.08	0.14	88.22
1000(12)	5%n	EDM-SNL	3.24E+00	2.15E+00	216.08	0.56	216.64
		ARAP	1.48E+00	1.34E+00	223.56	0.26	223.82
		SFSDP	9.61E+00	2.44E+00	457.10	0.88	457.98
	10%n	EDM-SNL	3.03E+00	1.76E+00	168.87	0.49	169.36
		ARAP	1.47E+00	1.28E+00	223.36	0.24	223.60
		SFSDP	1.10E+01	1.92E+00	345.86	0.76	346.62
	20%n	EDM-SNL	3.72E+00	1.22E+00	146.34	0.21	146.55
		ARAP	1.47E+00	1.22E+00	223.30	0.16	223.46
		SFSDP	1.04E+01	1.56E+00	275.52	0.27	275.79

6. Conclusions and Future Work

In this paper, we introduced a new scheme for wireless sensor network localization problem based on the Euclidean distance matrix. A conic programming built upon the $\mathcal{K}^n_+$ cone is solved by a recently developed alternating direction method of multipliers to retrieve missing distances between nodes. Then classical multidimensional and Procrustes analysis are applied to recover the node positions. By modelling the problem based on EDM, inequality constraints are integrated in a simple closed convex set, resulting in a localization scheme that acquires both efficiency and robustness towards the existence of large noise. Numerical experiments showed the EDM-based Localization scheme outperforms existing state-of-the-art ARAP and SFSDP schemes.

The major computation in EDM-based localization scheme is the spectral decomposition of a symmetric matrix, which could be fatal to the scheme when the number of nodes in the network goes beyond thousands. This brings us to the main issue we are going to investigate in future: For the large scale networks consisting tens of thousands nodes, following the paradigm of "think globally, fit locally", the Euclidean distance matrix could be completed part by part, thus a robust EDM-based localization scheme for large scale networks could be developed.

Acknowledgments

We would like to thank one of the referees for bringing the references [5, 6] to our attention. We also thank the associate editor for the useful comments that have helped to revise the paper.

Table 2: Execution Time Results of Corridor Network

Test Problems		Localization					
$n(R)$	m	Scheme	RMSD	RMSD.Re	Time	Time.Re	Total Time
200(20)	5%n	EDM-SNL	4.41E+00	4.35E+00	7.09	0.12	7.21
		ARAP	1.25E+01	9.20E+00	12.18	0.14	12.32
		SFSDP	1.66E+01	9.29E+00	35.64	0.21	35.85
	10%n	EDM-SNL	3.76E+00	3.74E+00	4.85	0.08	4.93
		ARAP	1.21E+01	5.68E+00	12.13	0.11	12.24
		SFSDP	1.40E+01	7.24E+00	31.32	0.10	31.41
	20%n	EDM-SNL	3.50E+00	3.33E+00	3.35	0.06	3.41
		ARAP	1.20E+01	4.63E+00	12.15	0.06	12.22
		SFSDP	1.57E+01	4.86E+00	26.70	0.08	26.78
500(15)	5%n	EDM-SNL	3.11E+00	2.02E+00	54.70	0.42	55.12
		ARAP	2.63E+00	2.00E+00	71.86	0.35	72.21
		SFSDP	1.39E+01	4.91E+00	149.00	0.71	149.71
	10%n	EDM-SNL	3.56E+00	1.78E+00	36.36	0.28	36.65
		ARAP	2.59E+00	1.78E+00	71.55	0.28	71.82
		SFSDP	1.09E+01	2.51E+00	150.56	0.53	151.09
	20%n	EDM-SNL	2.15E+00	1.72E+00	32.01	0.13	32.13
		ARAP	2.57E+00	1.70E+00	71.71	0.13	71.84
		SFSDP	8.31E+00	2.01E+00	105.00	0.23	105.22
1000(12)	5%n	EDM-SNL	3.69E+00	1.44E+00	209.28	0.68	209.96
		ARAP	1.58E+00	1.49E+00	276.90	0.42	277.32
		SFSDP	9.81E+00	1.90E+00	428.95	1.13	430.08
	10%n	EDM-SNL	3.62E+00	1.23E+00	147.77	0.41	148.18
		ARAP	1.58E+00	1.45E+00	276.83	0.28	277.12
		SFSDP	7.77E+00	1.52E+00	421.06	0.54	421.60
	20%n	EDM-SNL	5.92E+00	1.18E+00	161.07	0.29	161.36
		ARAP	1.57E+00	1.11E+00	276.84	0.19	277.03
		SFSDP	1.09E+01	1.42E+00	312.34	0.38	312.73

References

[1] N. Patwari, J. Ash, S. Kyperountas, A. Hero, R. Moses, N. Correal, Locating the nodes: cooperative localization in wireless sensor networks, Signal Processing Magazine, IEEE 22 (4) (2005) 54–69.

[2] I. F. Akyildiz, M. C. Vuran, Wireless sensor networks, Vol. 4, John Wiley & Sons, 2010.

[3] T. F. Cox, M. A. Cox, Multidimensional scaling, CRC Press, 2000.

[4] I. Borg, P. J. Groenen, Modern multidimensional scaling: Theory and applications, Springer Science & Business Media, 2005.

[5] L. Liberti, C. Lavor, N. Maculan, A. Mucherino, Euclidean distance geometry and applications, Siam Review 56 (1) (2014) 3–69.

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

[7] J. De Leeuw, P. Mair, Multidimensional scaling using majorization: Smacof in r, Department of Statistics, UCLA.

[8] J. A. Costa, N. Patwari, A. O. Hero, III, Distributed weighted-multidimensional scaling for node localization in sensor networks, ACM Trans. Sen. Netw. 2 (1) (2006) 39–64.

[9] Y. Shang, W. Ruml, Y. Zhang, M. P. J. Fromherz, Localization from mere connectivity, in: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing, MobiHoc '03, ACM, New York, NY, USA, 2003, pp. 201–212.

[10] A. Karbasi, S. Oh, Robust localization from incomplete local information, Networking, IEEE/ACM Transactions on 21 (4) (2013) 1131–1144.

[11] Y. Shang, W. Ruml, Improved mds-based localization, in: INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, Vol. 4, 2004, pp. 2640–2651 vol.4.

[12] Y. Koren, C. Gotsman, M. Ben-Chen, Patchwork: Efficient localization for sensor networks by distributed global optimization, Tech. rep., Citeseer (2005).

[13] L. Zhang, L. Liu, C. Gotsman, S. J. Gortler, An as-rigid-as-possible approach to sensor network localization, ACM Trans. Sen. Netw. 6 (4) (2010) 35:1–35:21.

[14] S. Gepshtein, Y. Keller, Sensor network localization by augmented dual embedding, Signal Processing, IEEE Transactions on 63 (9) (2015) 2420–2431.

[15] P. Biswas, Y. Ye, Semidefinite programming for ad hoc wireless sensor network localization, in: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, IPSN '04, ACM, New York, NY, USA, 2004, pp. 46–54.

[16] K. C. Toh, M. J. Todd, R. H. Ttnc, Sdpt3 a matlab software package for semidefinite programming, version 1.3, Optimization Methods and Software 11 (1-4) (1999) 545–581.

[17] P. Biswas, T.-C. Liang, K.-C. Toh, Y. Ye, T.-C. Wang, Semidefinite programming approaches for sensor network localization with noisy distance measurements, Automation Science and Engineering, IEEE Transactions on 3 (4) (2006) 360–371.

[18] Z. Wang, S. Zheng, Y. Ye, S. Boyd, Further relaxations of the semidefinite programming approach to sensor network localization, SIAM Journal on Optimization 19 (2) (2008) 655–673.

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

[20] A. Kannan, B. Fidan, G. Mao, Analysis of flip ambiguities for robust sensor network localization, Vehicular Technology, IEEE Transactions on 59 (4) (2010) 2057–2070.

[21] A. Kannan, G. Mao, B. Vucetic, Simulated annealing based wireless sensor network localization with flip ambiguity mitigation, in: Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd, Vol. 2, 2006, pp. 1022–1026.

[22] S. Shekofteh, M. Khalkhali, M. Yaghmaee, H. Deldari, Localization in wireless sensor networks using tabu search and simulated annealing, in: Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on, Vol. 2, 2010, pp. 752–757.

[23] W. Glunt, T.-L. Hayden, R. Raydan, Moldecular conformations from distance matrices, J. Comput. Chem. 14 (1993) 114–120.

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

[25] H.-D. Qi, A semismooth newton method for the nearest euclidean distance matrix problem, SIAM Journal on Matrix Analysis and Applications 34 (1) (2013) 67–93.

[26] H.-D. Qi, X. Yuan, Computing the nearest euclidean distance matrix with low embedding dimensions, Math. Program. 147 (1-2) (2014) 351–389.

[27] X. Li, D. Sun, K.-C. Toh, A schur complement based semi-proximal admm for convex quadratic conic programming and extensions, Mathematical Programming (2014) 1–41.

[28] J. Dattorro, Convex optimization and Euclidean distance geometry, Lulu. com, 2008.

[29] I. J. Schoenberg, Remarks to maurice frechet's article "sur la definition axiomatique d'une classe d'espace distances vectoriellement applicable sur l'espace de hilbert, Annals of Mathematics 36 (3) (1935) pp. 724–732.

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

[31] W. Glunt, T.-L. Hayden, S. Hong, J. Wells, An alternating projection algorithm for computing the nearest euclidean distance matrice, SIAM J. Matrix Anal. Appl. 11 (1990) 589–600.

[32] K. Weinberger, L. Saul, Unsupervised learning of image manifolds by semidefinite programming, International Journal of Computer Vision 70 (1) (2006) 77–90.

[33] E. J. Candès, B. Recht, Exact matrix completion via convex optimization, Foundations of Computational mathematics 9 (6) (2009) 717–772.

[34] J. M. Borwein, A. S. Lewis, Convex Analysis and Nonlinear Optimization, Springer., 2000.

[35] J. C. Gower, Generalized procrustes analysis, Psychometrika 40 (1) (1975) 33–51.

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

Shuanghua Bai received his BSc and MSc in Information and Computing Science (2010) and Operational Research and Control Theory (2012) respectively from Beijing Jiaotong University. He is currently pursuing the PhD degree in Operational Research at University of Southampton, Southampton, UK. His current research interests include matrix optimization in wireless sensor network localization and dimensionality reduction in machine learning.

Houduo Qi received his BSc in Statistics from Peking University (1990), MSc and PhD in Operational Research respectively from Qufu Normal University (1993) and Chinese Academy of Sciences (CAS) (1996). He has been a postdoctoral fellow at the institutions including CAS; University of New South Wales (QEII Fellow) and Hong Kong Polytechnic University. From 2004, he joined University of Southampton as a lecturer and then associate professor (2010) in operational research. His current research is on matrix optimization. He has been an associate editor for Asia-Pacific Journal of Operational Research since 2011 and Mathematical Programming Computation since 2013.