SFSDP: a **S**parse Version of **F**ull **S**emi**D**efinite **P**rogramming Relaxation for Sensor Network Localization Problems

Sunyoung Kim[⋆], Masakazu Kojima[†], Hayato Waki[‡], and Makoto Yamashita[♯]

July 2009

Abstract.
SFSDP is a Matlab package for solving a sensor network localization problem. These types of problems arise in monitoring and controlling applications using wireless sensor networks. SFSDP implements the semidefinite programming (SDP) relaxation proposed in Kim et al. [2009] for sensor network localization problems, as a sparse version of the full semidefinite programming relaxation (FSDP) by Biswas and Ye [2004]. To improve the efficiency of FSDP, SFSDP exploits the aggregated and correlative sparsity of a sensor network localization problem. As a result, SFSDP can handle much larger-sized problems than other softwares, and three-dimensional anchor-free problems. SFSDP can analyze the input data of a sensor network localization problem, solves the problem, and displays the computed locations of sensors. SFSDP also includes the features of generating test problems for numerical experiments.

Key words. Sensor network localization problems, semidefinite programming relaxation, sparsity exploitation, Matlab software package.

⋆ Department of Mathematics, Ewha W. University, 11-1 Dahyun-dong, Sudaemoon-gu, Seoul 120-750 Korea. The research was supported by KOSEF 2009-007-1314.
`skim@ewha.ac.kr`

† Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152-8552 Japan. This research was partially supported by Grant-in-Aid for Scientific Research (B) 19310096.
`kojima@is.titech.ac.jp`

‡ Department of Computer Science, The University of Electro-Communications, 1-5-1 Chofu-gaoka, Chofu-shi, Tokyo 182-8585 Japan. This research was partially supported by Grant-in-Aid for JSPS Fellows 20003236.
`hayato.waki@jsb.cs.uec.ac.jp`

♯ Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152-8552 Japan. M. Yamashita's research was supported by Grant-in-Aid for Young Scientists (B) 21710148.
`Makoto.Yamashita@is.titech.ac.jp`

1 Introduction

We introduce a Matlab package SFSDP for solving sensor network localization problems using semidefinite programming (SDP) relaxation. Sensor network localization problems arise in monitoring and controlling applications using wireless sensor networks such as inventory management and gathering environment data. Locating sensors accurately in a wireless sensor network is an important problem for the efficiency of applications. It is also closely related to distance geometry problems arising in predicting molecule structures and to graph rigidity.

For a network of n sensors, a sensor network localization problem is to locate m sensors of unknown position ($m < n$) that match the given distances if a subset of distances and $n-m$ sensors of known position (called anchors) are provided. Various approaches [1, 7, 8, 11, 12, 26] have been proposed for the problem to approximate the solution. Full semidefinite programming relaxation (FSDP) was introduced by Biswas and Ye in [3], and a number of solution methods based on SDP relaxation have followed [4, 5, 6, 21, 29].

The sensor network localization problem was formulated as a quadratic optimization problem (QOP) by Biswas and Ye [3], and was solved with SDP relaxation. We call their method as dense SDP relaxation in this paper. Solving nonlinear optimization problems using SDP relaxation has been a widely popular approach for the accuracy of approximated solutions and the efficiency of the computation. Software packages based on the primal-dual interior-point method [9, 27, 25] are used for solving SDP relaxation.

For the sensor network localization problem with a larger number of sensors, a distributed method in [4] was introduced, and a method combined with a gradient method [19] was proposed to improve the accuracy. The second-order cone programming (SOCP) relaxation was studied first in [7] and [26]. The solutions obtained by the SOCP relaxation are inaccurate compared to that by the SDP relaxation [26]. Edge-based SDP (ESDP) and node-based SDP (NSDP) relaxations were introduced in [29] to improve the computational efficiency of the original Biswas-Ye SDP relaxation in [3]. These SDP relaxations are weaker than the original SDP relaxation in theory, however, computational results show that the quality of solution is comparable to that of FSDP. It is also shown that much larger-sized problems can be handled.

SFSDP is an implementation of the SDP relaxation proposed in Kim et al. [13], which is called the sparse FSDP relaxation as opposed to the FSDP in [3]. Both SDP relaxations are derived from the sensor network localization problem formulated as a QOP. When solving a SDP relaxation problem using the primal-dual interior-point methods, the size of SDP relaxation problem generated from a sensor network localization problem is one of the important factors that decides the computational efficiency. As we try to solve a sensor network localization problem with an increasing number of sensors or in higher dimension, it is obvious that the size of SDP relaxation increases. It is, therefore, essential to reduce the size of SDP relaxation to solve a larger-sized sensor network localization problem. This motivates us to utilize the sparsity of problem, in particular, the aggregated and correlative sparsity [10, 17, 20] of sensor network localization problems.

When we want to decrease of the size of dense SDP relaxation, the quality of obtained solution becomes an important issue. For a QOP as the sensor network localization problem,

it is shown in [28] that the solution quality of the sparse SDP relaxation is equivalent to that of the dense SDP relaxation. In fact, the quality of obtained solution by SFSDP remains equivalent to that by FSDP. See Proposition 3.3 of [13]. As a result, SFSDP can handle larger-sized sensor network problems, e.g., up to 20000 sensors for 2-dimensional problems, than FSDP without deteriorating the quality of solution.

For the sensor network localization problem, distance data usually contain noise. SFSDP can solve the problem with both exact and noisy distance. It is designed for users who want to solve their own sensor network localization problems and to experiment with various test problems generated by SFSDP. One of the features of SFSDP is that users can select a primal-dual interior-point solver, either SDPA [9], available at [24], or SeDuMi [25], available at [23]. SDPA is known to be faster for solving large-sized problems, hence, shorter cpu time can be expected if SDPA is used. When a sensor network localization probem is given, SFSDP analyzes input data, and calls SDPA [9] or SeDuMi [25] to solve the SDP relaxation problem. It also displays the figures of location of sensors at the end of the execution.

Main features and capabilities of SFSDP are presented in the remainder of the paper. We provide some background information on sensor network localization problems in Section 2. In Section 3, the dense and sparse SDP relaxation of the problem are discussed after the description of how to extract the aggregated and correlative sparsity from the given problem. In Section 4, we illustrate how SFSDP is used for solving a sensor network localization problem with implementation details. Numerical results are presented in Section 5. Section 6 is for concluding remarks.

2 Sensor Network Localization Problems

We consider a problem with m sensors and m_a $(= n - m)$ anchors to describe a form of the sensor network localization problem that can be solved by SFSDP. Let $\rho > 0$ be a radio range, which determines the set $\mathcal{N}_x^\rho$ for pairs of sensors p and q such that their (Euclidean) distance d_{pq} is not greater than ρ, and the set $\mathcal{N}_a^\rho$ for pairs of a sensor p and an anchor r such that their distance d_{pr} does not exceed ρ;

$$\left.\begin{array}{rcl} \mathcal{N}_x^\rho & = & \{(p,q) : 1 \leq p < q \leq m, \|\boldsymbol{x}_p - \boldsymbol{x}_q\| \leq \rho\}, \\ \mathcal{N}_a^\rho & = & \{(p,r) : 1 \leq p \leq m,\ m+1 \leq r \leq n, \|\boldsymbol{x}_p - \boldsymbol{a}_r\| \leq \rho\}, \end{array}\right\}$$

where $\boldsymbol{x}_p \in \mathbb{R}^\ell$ denotes unknown location of sensor p and $\boldsymbol{a}_r \in \mathbb{R}^\ell$ known location of anchor r. Let $\mathcal{N}_x$ be a subset of $\mathcal{N}_x^\rho$ and $\mathcal{N}_a$ a subset of $\mathcal{N}_a^\rho$. SFSDP can solve the problem of $\ell = 2$ or 3.

By introducing zero objective function and the distance equations as constraints, we have the following form of sensor network localization problem with exact distance.

$$\left.\begin{array}{lll} \text{minimize} & 0 & \\ \text{subject to} & d_{pq}^2 = \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 & (p,q) \in \mathcal{N}_x, \\ & d_{pr}^2 = \|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 & (p,r) \in \mathcal{N}_a. \end{array}\right\} \tag{1}$$

For the problems with noise, we consider the following.

$$\left.\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(p,q)\in\mathcal{N}_x}\left(\epsilon^+_{pq}+\epsilon^-_{pq}\right)+\sum_{(p,r)\in\mathcal{N}_a}\left(\epsilon^+_{pr}+\epsilon^-_{pr}\right) \\ \text{subject to} & \hat{d}^2_{pq}=\|\boldsymbol{x}_p-\boldsymbol{x}_q\|^2+\epsilon^+_{pq}-\epsilon^-_{pq}\ \ (p,q)\in\mathcal{N}_x, \\ & \hat{d}^2_{pr}=\|\boldsymbol{x}_p-\boldsymbol{a}_r\|^2+\epsilon^+_{pr}-\epsilon^-_{pr}\ \ (p,r)\in\mathcal{N}_a, \\ & \epsilon^+_{pq}\geq 0,\ \epsilon^-_{pq}\geq 0,\ (p,q)\in\mathcal{N}_x, \\ & \epsilon^+_{pr}\geq 0,\ \epsilon^-_{pr}\geq 0, (p,r)\in\mathcal{N}_a, \end{array}\right\} \tag{2}$$

where $\epsilon^+_{pq}+\epsilon^-_{pq}$ (or $\epsilon^+_{pr}+\epsilon^-_{pr}$) indicates 1-norm error in the estimated distance $\hat{d}_{pq}$ between sensors p and q (or an estimated distance $\hat{d}_{pr}$ between sensor p and anchor r).

To transform the problems (1) and (2) to SDP relaxation [3], we first introduce an $\ell\times m$ matrix variable $\boldsymbol{X}=(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_m)\in\mathbb{R}^{\ell\times m}$. Then, the system of equations above can be written as

$$\left.\begin{array}{ll} \displaystyle d^2_{pq}=\sum_{i=1}^{\ell}X^2_{ip}-2\sum_{i=1}^{\ell}X_{ip}X_{iq}+\sum_{i=1}^{\ell}X^2_{iq} & (p,q)\in\mathcal{N}_x, \\ \displaystyle d^2_{pr}=\sum_{i=1}^{\ell}X^2_{ip}-2\sum_{i=1}^{\ell}X_{ip}a_{ir}+\|\boldsymbol{a}_r\|^2 & (p,r)\in\mathcal{N}_a, \end{array}\right\} \tag{3}$$

where X_{ip} denotes the (i,p)th element of the matrix $\boldsymbol{X}$ or the ith element of $\boldsymbol{x}_p$. Now, a QOP for the sensor network localization without noise is obtained.

$$\text{minimize 0 subject to the equality constraints (3).} \tag{4}$$

Using the matrix variable $\boldsymbol{X}$, the problem (2) becomes

$$\left.\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(p,q)\in\mathcal{N}_x}(\xi^+_{pq}+\xi^-_{pq})+\sum_{(p,r)\in\mathcal{N}_a}(\xi^+_{pr}+\xi^-_{pr}) \\ \text{subject to} & \displaystyle\hat{d}^2_{pq}=\sum_{i=1}^{\ell}X^2_{ip}-2\sum_{i=1}^{\ell}X_{ip}X_{iq}+\sum_{i=1}^{\ell}X^2_{iq}+\xi^+_{pq}-\xi^-_{pq}\ \ (p,q)\in\mathcal{N}_x, \\ & \displaystyle\hat{d}^2_{pr}=\sum_{i=1}^{\ell}X^2_{ip}-2\sum_{i=1}^{\ell}X_{ip}a_{ir}+\|\boldsymbol{a}_r\|^2+\xi^+_{pr}-\xi^-_{pr}\ \ (p,r)\in\mathcal{N}_a, \\ & \xi^+_{pq}\geq 0,\ \xi^-_{pq}\geq 0\ \ (p,q)\in\mathcal{N}_x,\ \ \xi^+_{pr}\geq 0,\ \xi^-_{pr}\geq 0\ \ (p,r)\in\mathcal{N}_a. \end{array}\right\} \tag{5}$$

When a sensor network problem of the form (4) or (5) has many equality constraints that may be redundant, the resulting SDP relaxation problem can be too large to solve. To deal with such a problem, SFSDP replaces $\mathcal{N}_x$ and $\mathcal{N}_a$ by smaller subsets of them, $\widetilde{\mathcal{N}}_x$ and $\widetilde{\mathcal{N}}_a$, respectively, before applying the sparse SDP relaxation to the problem (4) or (5). Then, the resulting SDP relaxation problem becomes smaller and sparser. This process, which will be discussed in details in Section 4.1, is a key to solving a large scale sensor network localization problem efficiently by SFSDP. We assume that either (i) (noisy) distance is available between a fairly large number of sensors and anchors in the original problem (4) or (5) to extract a smaller-sized subproblem satisfying the sparsity (the aggregated and correlative sparsity), or (ii) the original problem itself is sparse. If the radio range ρ is large and if we take $\mathcal{N}^\rho_x$ and $\mathcal{N}^\rho_a$ (or their subsets large enough) for $\mathcal{N}_x$ and $\mathcal{N}_a$, respectively, the assumption (i) is usually satisfied. We should note, however, that SFSDP may fail to solve the problem efficiently if neither (i) nor (ii) is satisfied.

3 SDP Relaxations of the Sensor Network Localization Problem

We describe the construction of FSDP and SFSDP for the sensor network localization problem with exact distances. Most of the discussion is valid for (5) for the problem with noisy distances. Let

$$Y_{pq} = \sum_{i=1}^{\ell} X_{ip}X_{iq} \ \ (p,q) \in \mathcal{N}_x, \ \text{or}, \ \ \boldsymbol{Y} = \boldsymbol{X}^T\boldsymbol{X} \in \mathcal{S}^m.$$

Then, the problem (4) can be written as

$$\left.\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \ \ (p,q) \in \mathcal{N}_x, \\ & d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp} \ (p,r) \in \mathcal{N}_a, \\ & \boldsymbol{Y} = \boldsymbol{X}^T\boldsymbol{X}. \end{array}\right\}$$

We now relax the nonconvex equality constraint $\boldsymbol{Y} = \boldsymbol{X}^T\boldsymbol{X}$ to the matrix inequality constraint $\boldsymbol{Y} \succeq \boldsymbol{X}^T\boldsymbol{X}$. Then, using the relation

$$\boldsymbol{Y} \succeq \boldsymbol{X}^T\boldsymbol{X} \iff \begin{pmatrix} \boldsymbol{I}_\ell & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix} \succeq \boldsymbol{O},$$

we obtain the Biswas-Ye SDP relaxation [3], called full SDP (FSDP) relaxation,

$$\left.\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \ \ (p,q) \in \mathcal{N}_x, \\ & d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp} \ (p,r) \in \mathcal{N}_a, \\ & \boldsymbol{O} \preceq \begin{pmatrix} \boldsymbol{I}_\ell & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix}, \end{array}\right\} \tag{6}$$

where $\boldsymbol{I}_\ell$ denotes the $\ell \times \ell$ identity matrix. Similary, we can obtain the SDP relaxation of (5).

Now, we briefly present the derivation of the sparse SDP relaxation of the sensor network localization problem. For details, we refer to [13]. Let Λ denote a finite index set and

$$\boldsymbol{Z} \ = \ \begin{pmatrix} \boldsymbol{W} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix}. \tag{7}$$

After rewriting the SDP (6) in an equality standard primal SDP of the form

$$\text{minimize } \boldsymbol{A}_0 \bullet \boldsymbol{Z} \text{ subject to } \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t \ (t \in \Lambda) \ \boldsymbol{Z} \succeq \boldsymbol{O}, \tag{8}$$

the conversion method [20] is applied to (8) as follows. Consider the index set $\mathcal{V}$ of rows (and columns) of the matrix variable $\boldsymbol{Z}$. Let us assume that the rows and columns of

matrix $\boldsymbol{Z}$ in (7) are indexed in the lexicographical order as $10, \ldots, \ell 0, *1, \ldots, *m$. Then, $\mathcal{V} = \{10, \ldots, \ell 0, *1, \ldots, *m\}$ for (6), where $*$ denotes a fixed symbol or integer larger than ℓ, and each element of $\boldsymbol{A}_t$ can be written as $[\boldsymbol{A}_t]_{ipjq}$ with $ip \in \mathcal{V}$ and $jq \in \mathcal{V}$.

We introduce the *aggregated sparsity pattern* $\mathcal{E}$ of the data matrices as in [10] for the description of sparsity exploitation in the sparse SDP relaxation with a chordal graph.

$$\mathcal{E} \;=\; \{(ip, jq) \in \mathcal{V} \times \mathcal{V} : ip \neq jq, [\boldsymbol{A}_t]_{ipjq} \neq 0 \text{ for some } t \in \Lambda\}$$

A geometrical representation of the aggregated sparsity pattern is considered with a graph $G(\mathcal{V}, \mathcal{E})$. To construct a chordal extension $G(\mathcal{V}, \overline{\mathcal{E}})$ of $G(\mathcal{V}, \mathcal{E})$ by simulating a chordal extension from $G(V, E)$ to $G(V, \overline{E})$, we let

$$\begin{aligned}
\overline{\mathcal{E}} &= \{(i0, j0) : 1 \leq i < j \leq \ell\} \cup \{(i0, *p) : 1 \leq i \leq \ell,\ 1 \leq p \leq m\} \\
&\quad \cup \{(*p, *q) : (p, q) \in \overline{E}\}, \\
\overline{\mathcal{C}}_h &= \{10, \ldots, \ell 0\} \cup \{*p : p \in C_h\} \quad (1 \leq h \leq k),
\end{aligned}$$

where $G(V, E)$ denotes the graph with the node set $V = \{1, 2, \ldots, m\}$ (the index set of sensors) and $E = \mathcal{N}_x$, and $C_1, \ldots, C_k$ the maximal cliques of the chordal extension $G(V, \overline{E})$. Using the information on the chordal graph $G(\mathcal{V}, \overline{\mathcal{E}})$ and its maximal cliques $\overline{\mathcal{C}}_1, \ldots, \overline{\mathcal{C}}_k$, application of the conversion method [20] to (8) leads to an SDP problem

$$\left.\begin{array}{ll}
\text{minimize} & \boldsymbol{A}_0 \bullet \boldsymbol{Z} \\
\text{subject to} & \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t \ (t \in \Lambda),\ \boldsymbol{Z}_{\overline{\mathcal{C}}_h, \overline{\mathcal{C}}_h} \succeq \boldsymbol{O} \ (1 \leq h \leq k),
\end{array}\right\} \tag{9}$$

where $\boldsymbol{Z}_{\overline{\mathcal{C}}_h, \overline{\mathcal{C}}_h}$ denotes a submatrix of $\boldsymbol{Z}$ consisting of the elements $\boldsymbol{Z}_{ipjq}$ $(ip \in \overline{\mathcal{C}}_h,\ jq \in \overline{\mathcal{C}}_h)$.

Rewriting (9), we obtain the sparse SDP relaxation as follows.

$$\left.\begin{array}{ll}
\text{minimize} & 0 \\
& d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \quad (p, q) \in \mathcal{N}_x, \\
& d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\displaystyle\sum_{i=1}^{\ell} X_{ip} a_{ir} + Y_{pp} \ (p, r) \in \mathcal{N}_a, \\
& \boldsymbol{O} \preceq \begin{pmatrix} \boldsymbol{I}_\ell & (\boldsymbol{x}_p : p \in C_h) \\ (\boldsymbol{x}_p : p \in C_h)^T & \boldsymbol{Y}_{C_h, C_h} \end{pmatrix} \quad (1 \leq h \leq k),
\end{array}\right\} \tag{10}$$

where $(\boldsymbol{x}_p : p \in C_h)$ denotes the $\ell \times \#C_h$ matrix variable with $\boldsymbol{x}_p$ $(p \in C_h)$ and $\boldsymbol{Y}_{C_h, C_h}$ a submatrix of $\boldsymbol{Y}$ with elements $\boldsymbol{Y}_{pq}$ $(p \in C_h,\ q \in C_h)$.

Note that (10) is not the standard form of SDP since some of the variables $\boldsymbol{x}_p$ $(p \in V)$ and Y_{pq} $((p, q) \in \mathcal{N}_x)$ appear more than once in the positive semidefinite constraints. To convert (10) to the standard form of SDP, we use the domain-space conversion method [15, 16], which was succefully implemented in SparsePOP, a Matlab package for polynomial optimization problems [28]. The resulting SDP involves k small-sized positive semidefinite matrix variables induced from the k positive semidefinite constraints in (10). By contrast, the SDP (6) involves a single large-sized $(\ell + m) \times (\ell + m)$ matrix variable $\boldsymbol{Z}$ given in (7). Furthermore, the resulting SDP is expected to satisfy the correlative sparsity, which characterizes the sparsity of the Schur complement matrix. We note that the Schur complement matrix is

the coefficient matrix of a system of linear equations that is solved by the Cholesky factorization for a search direction, at every iteration of the primal-dual interior-point method. These two properties on the resulting SDP, multiple but small-sized positive-semidefinite matrix variables and the correlative sparsity, greatly enhance the computational efficiency. See Kobayashi et al [16] for more details of the correlative sparsity.

Let us denote

$$L_d \;=\; \left\{ \boldsymbol{X} \in \mathbb{R}^{\ell \times m} : \begin{array}{l} (\boldsymbol{X}, \boldsymbol{Y}) \;\text{ is a solution of (6)} \\ \text{for some } \boldsymbol{Y} \in \mathcal{S}^m \end{array} \right\}.$$

In addtion, let L_s denote the set of solutions of the sparse SDP relaxation (10). We note that $L_d = L_s$ is shown in [13], which indicates that the same quality of solutions can be obtained by the sparse SDP relaxation as FSDP. Hence, the sparse SDP relaxation can achieve better computational performance for the same quality of solutions as FSDP.

The ESDP and NSDP relaxations of sensor network localization problems were proposed in [29] as further relaxations of the Biswas-Ye SDP relaxation. In essence, a generalization of the sparse SDP relaxation (10) includes NSDP and ESDP. In other words, if we denote L_n and L_e the solution sets of the NSDP and ESDP relaxation, respectively, then, by construction, we know $L_d \subseteq L_n \subseteq L_e$. It was shown in [29] that if the underlying graph $G(V, \mathcal{E})$ is chordal, then $L_d = L_n$. In this case, we know that $G(V, \mathcal{E}) = G(V, \overline{\mathcal{E}})$ and that $L_d = L_s = L_n$ also follows from Proposition 3.3 in [13]. For details, we refer to [13]. We used ESDP for the numerical experiments shown in Section 5 because it is known to be more efficient than NSDP.

We mention that the technique used in SFSDP for exploiting the sparsity for the QOP formulation is a fairly general technique for exploiting the sparsity in SDPs. When our technique is applied to the sensor network localization problem, a chordal extension of the entire underlying graph (not necessarily chordal) is artificially constructed, but not any subgraph of the graph from the given sensor network. This extended chordal graph does not correspond to any sensor network localization problem with distances given to all the edges. The important feature of this technique is to convert a sparse SDP into another equivalent SDP which can be solved efficiently by exploiting its sparsity.

From the construction of SDP relaxations, FSDP and SFSDP can be expected to attain solutions of more accuracy than ESDP and NSDP. Thus, if sparsity is utilized to enhance the computational efficiency, the performance of SFSDP can be expected better than other SDP relaxations.

4 Implementation

Both the sparse and dense SDP relaxation can be implemented using SFSDP for solving the sensor network localization problem in 2 or 3 dimensions. Anchor-free problems can also be solved with SFSDP.

4.1 Varying the Problem Size for Accuracy and Efficiency

SFSDP provides a way to reduce the size of given problem for the efficiency of the computation or to gradually decrease the degree of exploiting the sparsity of sensor network localization problem for the accuracy of solution.

Let $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ be a graph associated with the system of sensor network localization equations (3), where $N = \{1, \ldots, n\}$ denotes the node set consisting of all sensors and anchors. Consider subgraphs $G(N, E')$ of $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ with the same node set N and an edge subset E' of $\mathcal{N}_x \cup \mathcal{N}_a$. Let $\deg(p, E')$ be the degree of a node $p \in N$ in a subgraph $G(N, E')$, i.e., the number of edges incident to a node $p \in N$. In the ℓ-dimensional sensor network localization problem, $\deg(p, E) \geq \ell + 1$ for every sensor node p is a necessary condition to determine their locations when they are located in generic positions. Based on this, we consider the family $\mathcal{G}_\kappa$ of subgraphs $G(N, E')$ of $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ such that $\deg(p, E')$ is not less than $\min\{\deg(p, \mathcal{N}_x \cup \mathcal{N}_a), \kappa\}$ for every sensor node p, where κ is a positive integer not less than $\ell + 1$.

In SFSDP, a positive integer parameter κ not less than $\ell+1$ is used for selecting subsets $\widetilde{\mathcal{N}}_x$ and $\widetilde{\mathcal{N}}_a$ from $\mathcal{N}_x$ and $\mathcal{N}_a$ to reduce the size of the problem (4) or (5). A value for the parameter named pars.minDegree can be set for this purpose. Obviously, the minimum value for pars.minDegree is $\ell + 1$. We note that a minimal subgraph $G(N, \widetilde{E})$ is chosen from the family $\mathcal{G}_\kappa$, and $\mathcal{N}_x$ and $\mathcal{N}_a$ are replaced by $\widetilde{\mathcal{N}}_x = \mathcal{N}_x \cap \widetilde{E}$ and $\widetilde{\mathcal{N}}_a = \mathcal{N}_a \cap \widetilde{E}$, respectively, in (3). Since this method of reducing the size of (3) is based on heuristics, it may weaken the quality of SDP relaxation. See Section 4.1 of [13].

With an increasingly larger value for κ or pars.minDegree, we can expect to obtain more accurate locations of the sensors, although it takes longer to solve the sparse SDP relaxation. Some applications of the sensor network localization problem, *e.g.*, molecular conformation, have not enough distance information to obtain accurate solutions. For these applications, pars.minDegree in SFSDP can be set not to reduce the size of the problem. If pars.minDegree is increased, a stronger SDP relaxation resulting in longer cpu time is expected. In SFSDP, if it is equal to or larger than 100, then no reduction described previously is conducted. The default value for pars.minDegree is $\ell + 2$.

4.2 Selecting Objective Functions

SFSDP provides four objective functions to choose. Depending on whether the problem contains noise and whether anchors exist, one of four objective functions can be selected by setting a value of 0-3 for the parameter called pars.objSW.

The zero objective function is for the problem with at least $\ell + 1$ anchors and exact distance as (4), For the problem with at least $\ell+1$ anchors and noisy distance, the objective function of (5)

$$\sum_{(p,q)\in\widetilde{\mathcal{N}}_x} \left(\xi^+_{pq} + \xi^-_{pq}\right) + \sum_{(p,r)\in\widetilde{\mathcal{N}}_a} \left(\xi^+_{pr} + \xi^-_{pr}\right). \tag{11}$$

is to be minimized.

Functions	Description
SFSDPplus.m	A function that analyzes input data given and calls SFSDP.m.
SFSDP.m	A solver that solves the problem and returns the location of sensors, and can be called directly by users.
generateProblem.m	A function that generates a sensor network localization problem with the given parameters.
test_SFSDP.m	A function that tests the problem generated by generateProblem.m using SFSDP.m.

Table 1: Functions provided in SFSDP

If there are no anchors for the problem with exact distance, the objective function with a regularization term provides a more accurate solution. For this, we provide a function of the regularization term [2, 18] given by

$$-\sum_{(p,q)\in\overline{E}} \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2. \tag{12}$$

Recall that $\overline{E}$ denotes the edge set of the chordal extension $G(V,\overline{E})$ of the graph of $G(\{1,\ldots,m\},\widetilde{\mathcal{N}}_x)$, which was introduced for constructing the sparse SDP relaxation (10) in Section 3. For the problem with noisy distance and no anchors, the objective function of the sum of the 1-norm error of (11) and the regularization term described in (12), i.e., (11) + (12), leads to a more accurate solution.

4.3 An Exmple for Executing SFSDP

SFSDP includes four Matlab functions. Their names and functionality are briefly described in Table 1.

SFSDP needs the dimension of the problem, the number of sensors, the number of anchors, the locations of anchors if they are available, and distance information, for the input, and outputs the location of sensors in the form of a matrix called xMatrix. The detailed desciption is given in Table 2. The output matrix, xMatrix, has the same size and structure as the corresponding matrix for input.

The simplest use of SFSDP Matlab functions is

```
>>load 'd2n01s500a100.mat'
>>[xMatrix,info] =SFSDPplus(sDim,noOfSensors,noOfAnchors, xMatrix0,...
      distanceMatrix0,pars);
```

In this example, "d2n01s500a100.mat", which contains necessary data for input arguments, is loaded before executing SFSDP. At the end of execution, SFSDP refines the solution with the function refineposition.m, which is a Matlab implementation of a gradient method by Toh [19], and displays the figures with the computed solution as shown in Figure 1. For details, users can refer the user's guide [14].

Variable name	Description
sDim	The dimension of space where sensors and anchors are located.
noOfSensors	The number of sensors
noOfAnchors	The number m_a of anchors; the last m_a columns of xMatrix0
xMatrix0	sDim$\times n$ matrix of the location of sensors and anchors in the sDim-dimensional space, where n is the total number of sensors and anchors, and anchors are placed in the last m_a columns, or sDim$\times m_a$ matrix of anchors in the sDim-dimensional space, where m_a denotes the number of anchors. If noOfAnchors == 0, then xMatrix0 can be [].
distanceMatrix0	The sparse (and noisy) distance matrix between xMatrix0(:,i) and xMatrix0(:,j). distanceMatrix0(i,j) = (noisy) distance between xMatrix0(:,i) and xMatrix0(:,j) if $i < j$, distanceMatrix0$(i,j) = 0$ if $i >= j$.

Table 2: Input for SFSDP

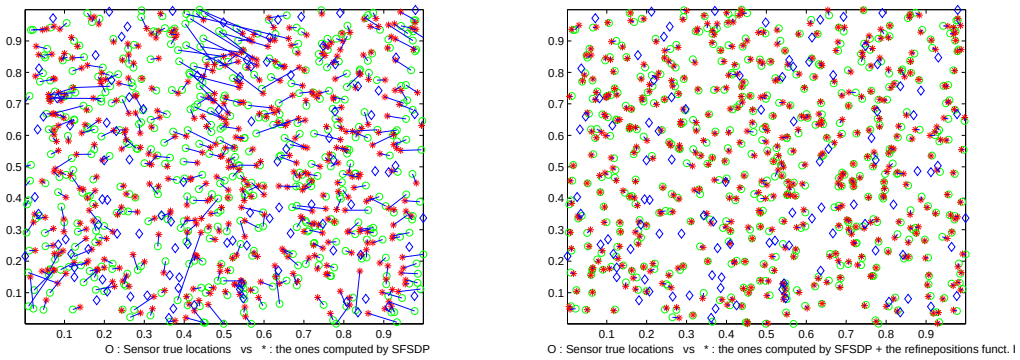

Figure 1: A 2-dimensional problem with 500 sensors and 100 anchors and noisy distance. Before and after the refinement using a gradient method. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a deviation from the true and computed location.

5 Numerical Results

One of the features of SFSDP is that either SDPA or SeDuMi can be used for solving the SDP relaxation. We first compare the performance of

- ESDP using SeDuMi.
- FSDP using SeDuMi.
- FSDP using SDPA.
- SFSDP using SeDuMi.
- SFSDP using SDPA.

applied to two-dimensional sensor network localization problems with 1000 to 6000 sensors randomly distributed in the unit square $[0,1]^2$ and 4 anchors placed at the corners of $[0,1]^2$. In addition, two-dimensional problems up to 18000 sensors and 2000 anchors randomly placed are tested with varying radio range and noisy factor. We note that the source code of ESDP was not available, thus, it could not be tested with SDPA. This comparison exhibits that SFSDP using SDPA is more efficient than the other methods. After the comparison, the performance of SFSDP with SDPA is further tested on larger-scale two-dimensional problems with randomly distributed sensors and anchors in $[0,1]^2$, three-dimensional problems with 1000 to 4000 sensors randomly distributed in the unit cube $[0,1]^3$ and 8 anchors placed at the corners of $[0,1]^3$, and three-dimensional anchor-free problems with 1000 to 4000 sensors randomly distributed in $[0,1]^3$. Numerical experiments were performed on 3.00GHz Quad-Core Intel Xeon X5365 with 48GB memory. For efficiency comparison, elapsed time using 4 threads is shown in Tables.

Throughout our numerical experiments, we generated sensors $\boldsymbol{a}_p$ $(p = 1, 2, \ldots, m)$ randomly in the unit square $[0,1]^2$ for two-dimensional problems, or in the unit cube $[0,1]^3$ for three-dimensional problems with a radio range $\rho > 0$. We placed anchors at the corners of the unit square $[0,1]^2$ or the unit cube $[0,1]^3$ except for the numerical experiment shown in Section 5.2, where 10% of the sensors were chosen randomly as anchors. We perturbed the distances to create problems with noisy distances:

$$\begin{array}{l} \hat{d}_{pq} = (1 + \sigma\epsilon_{pq})\|\boldsymbol{a}_p - \boldsymbol{a}_q\| \ ((p,q) \in \widetilde{\mathcal{N}}_x), \\ \hat{d}_{pr} = (1 + \sigma\epsilon_{pr})\|\boldsymbol{a}_p - \boldsymbol{a}_r\| \ ((p,r) \in \widetilde{\mathcal{N}}_a). \end{array} \tag{13}$$

Here, $\sigma \geq 0$ denotes a noisy factor, and ϵ_{pq} and ϵ_{pr} are chosen from the standard normal distribution $N(0,1)$. As in [3, 4, 5, 26, 29], the root mean square distance (rmsd)

$$\left(\frac{1}{m} \sum_{p=1}^{m} \|\boldsymbol{x}_p - \boldsymbol{a}_p\|^2 \right)^{1/2}$$

is used to measure the accuracy of locations of m sensors computed by SDPA or SeDuMi, and the accuracy of refined solutions by the gradient method. The values of rmsd after the refinement are included in the parentheses. We note that ESDP available at [30] provides rmsd only after refining the locations of sensors.

SDP(λ|κ)	SDP solver	#Sensors	Rmsd	Elapsed time		
ESDP($\lambda = 5$)	SeDuMi	1000	(2.71e-2)		60.7	
		2000	(1.60e-2)		217.0	
		4000	(2.30e-2)		1103.1	
		6000	(1.89e-2)		2774.7	
FSDP($\kappa = 4$)	SeDuMi	1000	4.98e-2 (8.56e-3)	1.1	3461.0	5.7
		2000	out of memory			
FSDP($\kappa = 4$)	SDPA	1000	4.98e-2 (9.37e-3)	1.11	387.3	2.87
		2000	4.46e-2 (7.72e-3)	5.5	2344.9	13.3
		4000	out of memory			
SFSDP($\kappa = 4$)	SeDuMi	1000	4.97e-2 (8.56e-3)	1.1	58.7	6.6
		2000	4.46e-2 (6.23e-3)	6.2	264.0	29.6
		4000	4.32e-2 (6.78e-3)	26.2	390.3	35.6
		6000	4.36e-2 (5.81e-3)	63.0	648.4	143.4
SFSDP($\kappa = 4$)	SDPA	1000	4.97e-2 (8.57e-3)	1.1	21.4	5.7
		2000	4.46e-2 (6.23e-3)	5.5	39.0	28.6
		4000	4.31e-2 (6.76e-3)	25.9	91.7	36.9
		6000	4.34e-2 (5.68e-3)	62.6	117.7	186.8

Table 3: Two-dimensional problems with the noisy factor $\sigma = 0.1$, radio range ρ =0.1, and rand seed 100. Three numbers in column "Elapsed time" indicate the elapsed time for generating a SDP, solving the SDP, and refining a solution by the gradient method.

5.1 Comparison of SFSDP with FSDP and ESDP for Two-dimensional Problems

We fixed 4 anchors at the corner of the unit square $[0, 1]^2$ with the radio range $\rho = 0.1$. In Table 3, the rmsd and elapsed time for ESDP, FSDP and SFSDP are compared. We denote λ an upper bound for the degree of any sensor node in ESDP, and κ a lower bound for the degree of any sensor node described in Section 4.1. See also Section 4.1 of [13] for their precise definitions and the comparison. FSDP and SFSDP were tested with either SDPA or SeDuMi. ESDP was tested only with SeDuMi since the source code of ESDP was not available. We notice that FSDP could not handle larger problems than 2000 sensors because of out-of-memory error. In all tested problems, we observe that SDPA provides a solution much faster than SeDuMi with comparable values of rmsd. We see that SFSDP with SDPA performs better than FSDP and ESDP.

5.2 Two-dimensional Larger-scale Problems

In Table 4, we show the numerical results for problems with 9,000 sensors and 1,000 anchors, both randomly generated in the unit square $[0, 1]^2$ as in [22]. In addition, problems with 18,000 sensors and 2000 anchors, both randomly generated in the unit square $[0, 1]^2$, are tested. The noisy factor, σ, was varied from 0.01 to 0.2 when generating the problems. We observe that SFSDP solves these problems efficiently with accurate values of rmsd. We note

#Sensors	#Anchors	ρ	σ	Rmsd	Elapsed time		
9000	1000	0.02	0.01	1.85e-2 (2.36e-3)	38.9	65.4	30.7
9000	1000	0.02	0.1	1.88e-2 (3.13e-3)	61.9	111.7	16.2
9000	1000	0.02	0.2	1.93e-2 (4.31e-3)	59.5	100.6	23.5
18000	2000	0.015	0.01	1.71e-2 (1.47e-3)	346.4	168.2	51.5
18000	2000	0.015	0.1	1.73e-2 (1.73e-3)	324.1	178.1	66.1
18000	2000	0.015	0.2	1.75e-2 (2.38e-3)	332.9	117.4	63.9

Table 4: Numerical results using SFSDP($\kappa = 4$) and SDPA for two-dimensional problems generated with random seed 100. Three numbers in column "Elapsed time" indicate the elapsed time for generating a SDP, solving the SDP, and refining a solution by the gradient method.

that the test problems shown in [22] involve σ up to 0.01.

5.3 Three-dimensional Problems

Numerical results for three-dimensional problems are shown in Table 5. We varied the number of sensors from 1000 to 4000. ESDP downloaded from [30] could not handle three-dimensional problems.

For the problems with 1000 sensors, the value for ρ is increased from 0.25 to 0.4. We notice that the elapsed time for solving the SDP relaxation decreases, as the value for ρ increases. Or, it takes longer to solve the SDP relaxation from the problem with a smaller value of ρ. As described in Section 4.1, SFSDP chooses a subgraph $G(N, \widetilde{E})$ of the graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ associated with a given sensor network problem to be solved. As we have more edges incident to every node, we have more flexibility for choosing a subgraph $G(N, \widetilde{E})$ such that $G(V, \mathcal{N}_x \cap \widetilde{E})$ can be extended to a sparse chordal graph having maximal cliques $\mathcal{C}_h$ $(h = 1, 2, \ldots, k)$ with smaller sizes. Recall that the maximal cliques $\mathcal{C}_h$ $(h = 1, 2, \ldots, k)$ determine the positive semidefinite matrix inequalities in the sparse SDP relaxation problem (10). Thus, smaller size maximal cliques $\mathcal{C}_h$ $(h = 1, 2, \ldots, k)$ provide more efficiency for solving (10). By contrast, if enough edges incident to each node of the graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ are not available, SFSDP may encounter a difficulty in choosing such a subgraph $G(N, \widetilde{E})$. For this reason, we see the increase of elapsed time with decreasing ρ in Table 5.

SFSDP successfully solves three-dimensional problems with 2000 and 4000 sensors efficiently, resulting accurate values of rmsd, as displayed in Table 5.

5.4 Anchor-free Problems in 3 dimensions

SFSDP handles anchor-free problems in ℓ dimensions, $\ell = 2$ or 3, by first fixing $\ell + 1$ sensors as anchors, and then applying the sparse SDP relaxation to the resulting problem. More precisely, if an anchor-free sensor network localization problem with n sensors in the ℓ-dimensional space is given, SFSDP first chooses $\ell + 1$ sensors, e.g., sensors $n - \ell, n - \ell + 1, \ldots, n$, which are adjacent to each other and forms a nondegenerate ℓ-simplex. Then,

#Sensors	ρ	σ	Rmsd	Elapsed time		
1000	0.25	0.0	6.45e-3 (9.88e-4)	4.1	145.3	3.6
1000	0.25	0.01	2.53e-2 (5.79e-3)	4.3	178.7	7.5
1000	0.25	0.1	1.05e-1 (3.04e-2)	4.1	159.3	3.1
1000	0.3	0.0	3.11e-3 (3.61e-4)	1.7	52.6	1.3
1000	0.3	0.01	2.67e-2 (4.10e-3)	2.4	61.7	11.8
1000	0.3	0.1	1.00e-1 (1.75e-2)	2.4	58.7	7.0
1000	0.4	0.0	2.32e-3 (2.10e-4)	4.8	11.2	2.2
1000	0.4	0.01	2.85e-2 (3.15e-3)	5.1	15.5	18.8
1000	0.4	0.1	1.04e-1 (1.74e-2)	5.2	13.8	31.3
2000	0.3	0.0	2.81e-3 (1.94e-4)	7.1	83.5	3.5
2000	0.3	0.01	2.62e-2 (4.47e-3)	11.3	108.0	19.8
2000	0.3	0.1	9.84e-2 (1.53e-2)	11.5	115.3	20.9
4000	0.3	0.0	4.54e-3 (2.05e-4)	29.6	145.9	20.2
4000	0.3	0.01	2.65e-2 (3.76e-3)	48.8	156.6	35.4
4000	0.3	0.1	9.86e-2 (1.84e-2)	50.4	146.4	44.8

Table 5: Numerical results with SFSDP($\kappa = 5$) on 3-dimensional problems with m sensors randomly generated in $[0, 1]^3$, 8 anchors fixed at the corner of $[0, 1]^3$. Three numbers in column "Elapsed time" indicate the elapsed time for generating a SDP, solving the SDP, and refining a solution by the gradient method.

we temporarily fix their locations, say $\boldsymbol{x}_r = \boldsymbol{a}_r$ $(r = n - \ell, n - \ell + 1, \ldots, n)$, as anchors, and apply the sparse SDP relaxation described in Sections 3 and 4 to the resulting sensor network problem with $m = n-\ell+1$ sensors and $m_a = \ell+1$ anchors. SFSDP, then, computes the location of sensors $\boldsymbol{x}_p$ $(p = 1, 2, \ldots, m)$ relative to the sensors fixed as anchors. The gradient method is applied to refine the locations $\boldsymbol{x}_p$ $(p = 1, 2, \ldots, m)$.

We can measure the accuracy of computed solution when the true locations $\boldsymbol{a}_p$ $(p = 1, 2, \ldots, n)$ of all sensors are known. In the numerical experiment whose results are shown in Table 6, the rmsd of the computed locations of sensors $\boldsymbol{x}_p$ $(p = 1, 2, \ldots, n)$ is evaluated after applying a linear transformation (translation, reflection, orthogonal rotation, and scaling) T, provided by a MATLAB program procrustes.m [2, 18]. This function minimizes the total squared errors $\sum_{p=1}^{n} \|T(\boldsymbol{x}_p) - \boldsymbol{a}_p\|^2$ between the true and the transformed approximate locations of sensors. We also observe that SFSDP can find the location of the sensors with accuracy, as shown in the values of rmsd.

As in Figures 2 and 3, SFSDP displays three figures, the locations of sensors from SDPA, after applying the linear transformation to them, and after refining them using the gradient method and applying the linear transformation to the refined locations of sensors.

6 Concluding Remarks

We have described the Matlab package SFSDP. It is designed to solve larger-sized sensor network localization problems than other available softwares. SFSDP can be used for the

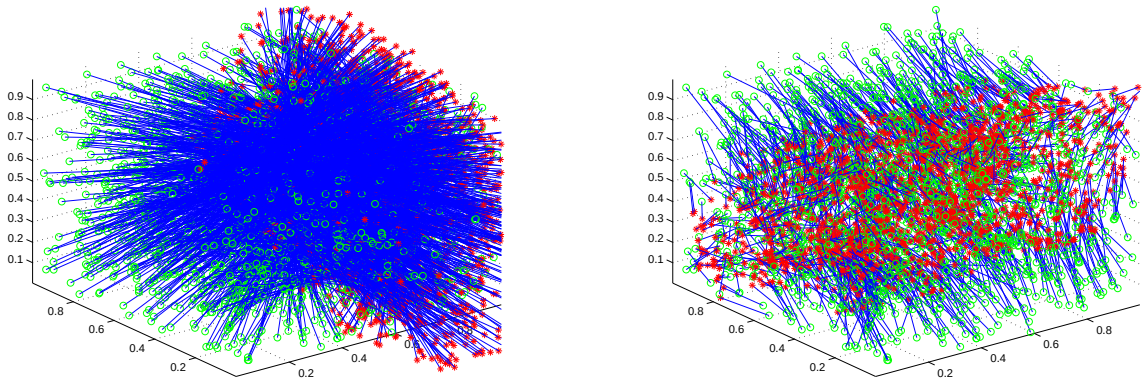

Figure 2: A 3-dimensional anchor-free problem with 2000 sensors, $\rho = 0.3$, and $\sigma = 0.1$. The locations of sensors from SFSDP($\kappa = 4$) on the left and the locations of sensors after applying the linear transformation (translation, reflection, orthogonal rotation, and scaling) which minimizes the squared errors between the true and the transformed locations of sensors on the right. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a deviation from true and computed locations.

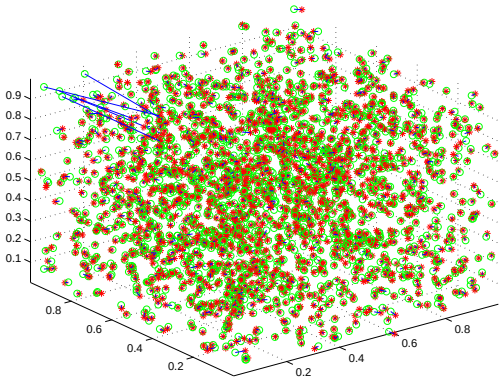

Figure 3: A 3-dimensional anchor-free problem with 2000 sensors, $\rho = 0.3$, and $\sigma = 0.1$. After applying the local refinement (a gradient method) followed the linear transformation (translation, reflection, orthogonal rotation, and scaling) which minimizes the squared errors between the true and the transformed locations. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a deviation from true and computed locations.

#Sensors	ρ	σ	Rmsd	Elapsed time		
1000	0.3	0.0	5.26e-3 (3.83e-5)	2.3	52.3	0.9
1000	0.3	0.01	9.90e-2 (1.16e-3)	3.0	52.4	2.0
1000	0.3	0.1	2.58e-1 (9.25e-2)	2.9	46.1	4.8
2000	0.3	0.0	3.76e-3 (7.73e-5)	9.1	96.1	2.3
2000	0.3	0.01	7.15e-2 (1.14e-3)	12.8	101.5	7.6
2000	0.3	0.1	1.59e-1 (2.02e-2)	13.4	92.7	6.9
4000	0.3	0.0	3.42e-3 (3.29e-5)	36.4	157.4	12.6
4000	0.3	0.01	5.89e-2 (1.18e-3)	56.1	137.8	26.2
4000	0.3	0.1	1.66e-1 (1.37e-2)	56.0	158.5	29.9

Table 6: Numerical results with SFSDP($\kappa = 5$) on 3-dimensional anchor-free problems with n sensors randomly generated in $[0, 1]^3$ and radio range $\rho = 0.3$. Three numbers in column "Elapsed time" indicate the elapsed time for generating a SDP, solving the SDP, and refining a solution.

problems with various anchor locations and anchor-free problems in 2 or 3 dimensions.

The sensor network localization problem has a number of applications where computational efficiency is an important issue. SDP approach has been known to be effective in locating sensors, however, solving large-scale problems with this approach has been a challenge.

From numerical results in Section 5, SFSDP demonstrates the computational advantages over other methods in solving the large-sized sensor network localization problems. These come from utilizing the aggregated and correlative sparsity of the problem, which reduces the size of SDP relaxation. SFSDP incorporated with SDPA provides a solution faster than that with SeDuMi.

One of the advantages of SFSDP is that it is equipped with both SeDuMi and SDPA. It is our experience that the accuracy of solution by SDPA is not as good as that by SeDuMi for some problems. However, SDPA may serve better, as observed in Section 5, for problems such as sensor network localization problems with the existence of noise frequent.

We hope to improve the performance of the sparse SDP relaxation implemented in SFSDP, in particular, for the case when the original problem does not provide enough distance information between sensors.

Acknowledgments

The authors would like to thank Professor Yinyu Ye for the original version of FSDP, and Professor Kim Chuan Toh for Matlab programs refineposition.m and procrustes.m, and for helpful comments.

References

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

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

[3] P. Biswas and Y. Ye (2004) "Semidefinite programming for ad hoc wireless sensor network localization," in *Proceedings of the third international symposium on information processing in sensor networks*, ACM press, 46–54.

[4] P. Biswas and Y. Ye (2006) "A distributed method for solving semidefinite programs arising from Ad Hoc Wireless Sensor Network Localization," in *Multiscale Optimization Methods and Applications*, 69–84, Springer.

[5] P. Biswas, T.-C. Liang, T.-C. Wang, Y. Ye (2006) "Semidefinite programming based algorithms for sensor network localization," *ACM Tran. Sensor Networks*, **2**, 188–220.

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

[7] L. Doherty, K. S. J. Pister, and L. El Ghaoui (2001) "Convex position estimation in wireless sensor networks," *Proceedings of 20th INFOCOM*, **3**, 1655–1663.

[8] T. Eren, D. K. Goldenberg, W. Whiteley, Y. R. Wang, A. S. Morse, B. D. O. Anderson, and P. N. Belhumeur (2004) "Rigidity, computation, and randomization in network localization," *in Proceedings of IEEE Infocom.*

[9] K. Fujisawa, M. Fukuda, K. Kobayashi, M. Kojima, K. Nakata, M. Nakata and M. Yamashita (2008), "SDPA (SemiDefinite Programming Algorithm) User's Manual — Version 7.0.5," Research Report B-448, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

[10] M. Fukuda, M. Kojima, K. Murota and K. Nakata (2000) "Exploiting sparsity in semidefinite programming via matrix completion I: General framework," *SIAM J. Optim.*, **11**, 647–674.

[11] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S.Wicker (2002) "An empirical study of epidemic algorithms in large scale multihop wireless network," IRB-TR-02-003, Intel Corporation.

[12] A. Howard, M. Matarić and G. Sukhatme (2001) "Relaxation on a mesh: a formalism for generalized localization," In *IEEE/RSJ International conference on intelligent robots and systems*, Wailea, Hawaii, 1055–1060.

[13] S. Kim, M. Kojima and H. Waki (2009) " Exploiting sparsity in SDP relaxation for sensor network localization," *SIAM J. Optim.*, **20**, (1) 192–215.

[14] S. Kim, M. Kojima and H. Waki (2009) " User's manual for SFSDP: a Sparse Version of Full SemiDefinite Programming Relaxation for Sensor Network Localization Problems," Research Report B-449, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

[15] S. Kim, M. Kojima, M. Mevissen, and M. Yamashita (2009) " Exploiting Sparsity in Linear and Nonlinear Matrix Inequalities via Positive Semidefinite Matrix Completion," Research Report B-452, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

[16] S. Kim, M. Kojima, M. Mevissen, and M. Yamashita (2009) " User's Manual for SparseCoLO: Conversion Methods for SPARSE COnic-form Linear Optimization Problems," Research Report B-453, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

[17] K. Kobayashi, S. Kim and M. Kojima (2008) Correlative sparsity in primal-dual interior-point methods for LP, SDP and SOCP, *Appl. Math. Opt.,* **58** (1) 69–88.

[18] N.-H. Z. Leung and K.-C. Toh (2008) ,"An SDP-based divide-and-conquer algorithm for large scale noisy anchor-free graph realization," preprint, National University of Singapore.

[19] T.-C. Lian, T.-C. Wang, and Y. Ye (2004) "A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization," Technical report, Dept. of Management Science and Engineering, Stanford University.

[20] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima and K. Murota (2003) "Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results," *Math. Program.*, **95**, 303–327.

[21] J. Nie (2009) "Sum of squares method for sensor network localization," *Comput. Opt. Appl.,* **43**, 151–179.

[22] T. K. Pong and P. Tseng (2009) "(Robust) Edge-Based Semidefinite Programming Relaxation of Sensor Network Localization," preprint.

[23] SeDuMi Homepage, http://sedumi.mcmaster.ca

[24] SDPA Homepage, http://sdpa.indsys.chuo-u.ac.jp/sdpa/

[25] J. F. Strum (1999) "SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optim. Methods Soft.*, **11 & 12**, 625–653.

[26] P. Tseng (2007) "Second order cone programming relaxation of sensor network localization," *SIAM J. Optim.*, **18**, 156-185.

[27] R. H. Tutuncu, K. C. Toh, and M. J. Todd (2003) "Solving semidefinite-quadratic-linear programs using SDPT3", *Math. Program.* **95**, 189–217.

[28] H. Waki, S. Kim, M. Kojima and M. Muramatsu (2006) "Sums of Squares and Semidefinite Programming Relaxations for Polynomial Optimization Problems with Structured Sparsity," *SIAM J. Optim,* **17**, 218–242.

[29] Z. Wang, S. Zheng, S. Boyd, and Y. Ye (2008) "Further relaxations of the SDP approach to sensor network localization," *SIAM J. Optim.*, **19** (2) 655–673.

[30] Y. Ye's website, http://www.stanford.edu/∼yyye.