Algorithm xxx: SFSDP: a Sparse Version of Full SemiDefinite Programming Relaxation for Sensor Network Localization Problems

SUNYOUNG KIM, Ewha W. University
MASAKAZU KOJIMA, Tokyo Institute of Technology
HAYATO WAKI, The University of Electro-Communications
MAKATO YAMASHITA, Tokyo Institute of Technology

SFSDP is a Matlab package for solving sensor network localization (SNL) problems. 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 problems than other software as well as three-dimensional anchor-free problems. SFSDP analyzes 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.

Categories and Subject Descriptors: G.1.6 [**Numerical Analysis**]: Optimization-Nonlinear programming

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Sensor network localization problems, semidefinite programming relaxation, sparsity exploitation, Matlab software package

ACM Reference Format:
Kim, S., Kojima, M., Waki, H., and Yamashita, M. 2011. Algorithm xxx: **SFSDP**: a **S**parse Version of **F**ull **S**emi**D**efinite **P**rogramming Relaxation for Sensor Network Localization Problems. ACM Trans. Math. Softw. 1, 1, Article 1 (January 2011), 19 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

We introduce a Matlab package SFSDP for solving sensor network localization (SNL) problems using semidefinite programming (SDP) relaxation. SNL 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 like predicting molecule structures and to graph rigidity.

For a network of n sensors, the SNL problem is to locate m sensors of unknown positions ($m < n$) that match the given distances if a subset of distances and $m_a = n-m$

This work is supported by the National Research Fund, under grant KOSEF 2009-007-1314, and by Grant-in-Aid for Scientific Research (B) 19310096, Grant-in-Aid for JSPS Fellows 20003236, and Grant-in-Aid for Young Scientists (B) 21710148.
Author's addresses: S. Kim, Department of Mathematics, Ewha W. University; M. Kojima and M. Yamashita, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology; H. Waki, Department of Computer Science, The University of Electro-Communications.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

sensors of known position (called anchors) are provided. Various approaches [Alfakih et al. 1999; Doherty et al. 2001; Eren et al. 2004; Ganesan et al. 2002; Howard et al. 2001; Tseng 2007] have been proposed for the problem to approximate the solution. Full semidefinite programming (FSDP) relaxation was introduced by Biswas and Ye in [Biswas and Ye 2004], and a number of solution methods based on SDP relaxation have followed [Biswas and Ye 2006; Biswas et al. 2006; Biswas et al. 2006; Nie 2009; Wang et al. 2008].

The SNL problem was formulated as a quadratic optimization problem (QOP) by Biswas and Ye [Biswas and Ye 2004], and was solved with SDP relaxation. We call their method FSDP relaxation in this paper. Solving nonlinear optimization problems using SDP relaxations has become a popular approach because of the accuracy of the computed solutions and the efficiency of the computation. Software packages based on the primal-dual interior-point methods [Fujisawa et al. 2008; Tütüncü et al. 2003; Strum 1999] are used to solve SDP relaxations.

For the SNL problem with a large number of sensors, distributed methods in [Biswas and Ye 2006] were introduced, and a method combined with a gradient method [Lian et al. 2004] was proposed to improve accuracy. The second-order cone programming (SOCP) relaxation was studied first in [Doherty et al. 2001] and then in [Tseng 2007]. The solution obtained by the SOCP relaxation is inaccurate compared to SDP relaxation [Tseng 2007]. Edge-based SDP (ESDP) and node-based SDP (NSDP) relaxations were introduced in [Wang et al. 2008] to improve the computational efficiency of Biswas-Ye's FSDP relaxation in [Biswas and Ye 2004]. These SDP relaxations are weaker than the FSDP relaxation in theory, however, computational results show that the quality of solution is comparable to that of FSDP. It has also been shown that much larger problems can be handled.

SFSDP is an implementation of the SDP relaxation proposed in Kim et al. [Kim et al. 2009a], which is called the sparse FSDP relaxation as opposed to the FSDP relaxation in [Biswas and Ye 2004]. Both SDP relaxations are derived from the SNL problem formulated as a QOP. When solving an SDP relaxation problem by the primal-dual interior-point method, the size of SDP relaxation problem generated from an SNL problem is one of the important factors that determines the computational efficiency. As we try to solve an SNL problem with an increasing number of sensors or in higher dimension, it is obvious that the size of SDP relaxation increases. Thus, reducing the size of SDP relaxation is essential to solve large SNL problems. This motivates us to utilize the sparsity of the problem, in particular, the aggregated and correlative sparsity [Fukuda et al. 2000; Kobayashi et al. 2008; Nakata et al. 2003] of SNL problems.

When we want to decrease the size of FSDP relaxation, the quality of the computed solution becomes an important issue. For a QOP (such as the SNL problem) it is shown in [Waki et al. 2006] that the solution quality of the sparse SDP relaxation is equivalent to that of FSDP relaxation. In fact, the quality of obtained solution by SFSDP remains equivalent to that by FSDP (see Proposition 3.3 of [Kim et al. 2009a]). As a result, SFSDP can handle larger SNL problems, e.g., up to 20000 sensors for 2-dimensional problems, than FSDP without deteriorating the quality of solutions. SpaseLoc [Carter et al. 2006] solves a large SNL problem by applying FSDP to a sequence of subproblems. Since SFSDP solves the SNL problem as one problem, the accuracy of the solution is maintained throughout the solution process.

For the SNL problem, distance data usually contain noise. SFSDP can solve the problem with both exact and noisy distances. It is designed for users who want to solve their own SNL problems and to experiment with various test problems generated by SFSDP. One feature of SFSDP is that users can select a primal-dual interior-point solver, either SDPA [Fujisawa et al. 2008] (available at [SDPA Homepage 2009]) or SeDuMi [Strum 1999] (available at [SeDuMi Homepage]). SDPA is known to be faster

for solving large problems, hence, shorter execution times can be expected if SDPA is used. When an SNL problem is given, SFSDP analyzes the input data, and calls SDPA or SeDuMi to solve the SDP relaxation problem. It also displays the figures of location of sensors at the end of computation.

The main features and capabilities of SFSDP are presented in the rest of the paper. We provide some background information on SNL problems in Section 2. In Section 3, we discuss FSDP and sparse SDP relaxations of the problem following a description of how to extract the aggregated and correlative sparsity. Section 4 considers implementation issues for constructing the sparse SDP relaxations. Numerical results are presented in Section 5 and concluding remarks in Section 6.

2. SNL PROBLEMS

We consider a problem with m sensors and m_a $(= n - m)$ anchors to describe a form of the SNL problem that can be solved by SFSDP. Let $\rho > 0$ be a radio range, which determines the set $\mathcal{N}_x^\rho$ of pairs of sensors p and q such that their (Euclidean) distance d_{pq} is not greater than ρ, and the set $\mathcal{N}_a^\rho$ of 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 the unknown location of sensor p and $\boldsymbol{a}_r \in \mathbb{R}^\ell$ the known location of anchor r. SFSDP can solve the problem of $\ell = 2$ or 3.

Let $\mathcal{N}_x$ be a subset of $\mathcal{N}_x^\rho$ and $\mathcal{N}_a$ a subset of $\mathcal{N}_a^\rho$. By introducing a zero objective function and having the distance equations as constraints, we have the following form of SNL problem with exact distances

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

For problems with noise, we consider

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

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

We use two kinds of subsets $\mathcal{N}_x$ of $\mathcal{N}_x^\rho$ and $\mathcal{N}_a$ of $\mathcal{N}_a^\rho$ instead of $\mathcal{N}_x^\rho$ and $\mathcal{N}_a^\rho$ for two reasons. First, some of the distances d_{pq} (or the estimated distances $\hat{d}_{pq}$) $((p,q) \in \mathcal{N}_x^\rho)$ and d_{pr} (or the estimated distances $\hat{d}_{pr}$) $((p,r) \in \mathcal{N}_a^\rho)$ may be unavailable in practice. Second, for numerical efficiency, we use these smaller subsets even when $\mathcal{N}_x^\rho$ and $\mathcal{N}_a^\rho$ are available. Using the smaller $\mathcal{N}_x$ and $\mathcal{N}_a$ for the SNL problems (1) and (2), reduces the size of SDP relaxations although the computed accuracy of the locations of the sensors may deteriorate. Thus, the way $\mathcal{N}_x$ and $\mathcal{N}_a$ are selected is an important issue for both efficiency and accuracy. This will be discussed further in Section 4.1.

To transform problems (1) and (2) into SDP relaxation [Biswas and Ye 2004], we first introduce an $\ell \times m$ matrix variable $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{\ell\times m}$. The system of equations

(1) can then be written as

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

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 SNL without noise is obtained:

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

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

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

3. SDP RELAXATIONS OF THE SNL PROBLEM

We first describe the construction of FSDP for the SNL problem (4) with exact distances in Section 3.1, and then the exploitation of sparsity of FSDP, which leads to SFSDP, in Section 3.2. Most of the discussion is valid for the problem (5) with noisy distances.

3.1. The Biswas-Ye SDP Relaxation of the SNL Problem

Let

$$Y_{pq} = \sum_{i=1}^{\ell} X_{ip}X_{iq}, \quad \text{or } \boldsymbol{Y} = \boldsymbol{X}^T\boldsymbol{X} \in \mathbb{S}^m.$$

where $\mathbb{S}^m$ denotes the set of $m \times m$ symmetric matrices. Then, the problem (4) can be written as

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

We now relax the nonconvex equality constraint $\boldsymbol{Y} = \boldsymbol{X}^T\boldsymbol{X}$ to the matrix inequality constraint $\boldsymbol{Y} \succeq \boldsymbol{X}^T\boldsymbol{X}$. Here, $\boldsymbol{A} \succeq \boldsymbol{B}$ means $\boldsymbol{A} - \boldsymbol{B}$ is positive semidefinite for $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{S}^m$. Let $\boldsymbol{I}_\ell$ denotes the $\ell \times \ell$ identity matrix and $\boldsymbol{O}$ the zero matrix. 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 [Biswas and Ye 2004] (called the FSDP relaxation) of the SNL problem (4) without noise

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

Similarly, we obtain the FSDP relaxation of the SNL problem (5) with noise

$$\left.\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(p,q) \in \mathcal{N}_x} (\xi_{pq}^+ + \xi_{pq}^-) + \sum_{(p,r) \in \mathcal{N}_a} (\xi_{pr}^+ + \xi_{pr}^-) \\ \text{subject to} & \hat{d}_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} + \xi_{pq}^+ - \xi_{pq}^- \ \ (p,q) \in \mathcal{N}_x, \\ & \hat{d}_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\displaystyle\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp} + \xi_{pr}^+ - \xi_{pr}^- \ \ (p,r) \in \mathcal{N}_a, \\ & \xi_{pq}^+ \geq 0, \ \xi_{pq}^- \geq 0 \ \ (p,q) \in \mathcal{N}_x, \ \ \xi_{pr}^+ \geq 0, \ \xi_{pr}^- \geq 0 \ \ (p,r) \in \mathcal{N}_a, \\ & \begin{pmatrix} \boldsymbol{I}_\ell & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix} \succeq \boldsymbol{O}. \end{array}\right\} \tag{7}$$

3.2. Exploiting Sparsity

The sparsity of the SNL problem (4) and its SDP relaxation (6) is first extracted and then exploited. To describe the procedure, we introduce a graph $G(V_x, \mathcal{N}_x)$ consisting of the node set $V_x = \{1, 2, \ldots, m\}$ (the index set of sensors) and the edge set $\mathcal{N}_x$. Let $G(V_x, \widetilde{\mathcal{N}}_x)$ be a chordal extension of $G(V_x, \mathcal{N}_x)$, and $C_1, \ldots, C_k$ be the maximal cliques of $G(V_x, \widetilde{\mathcal{N}}_x)$. For the definition of chordal graphs and maximal cliques, we refer to [Blair and Peyton 1993; Golumbic 1980]. Recall that $\mathcal{N}_x$, a subset of $\mathcal{N}_x^\rho$, is chosen before constructing the SNL problem (4). Assume that the sizes of the maximum cliques $C_1, \ldots, C_k$ are small. Selecting $\mathcal{N}_x$ that satisfies this assumption will be discussed in Section 4.

Now, we derive the sparse SDP relaxation of the SNL problem; for details, we refer to [Kim et al. 2009a]. Let $\boldsymbol{Z} \in \mathbb{S}^{\ell+m}$ be a variable matrix of the form

$$\boldsymbol{Z} \ = \ \begin{pmatrix} \boldsymbol{W} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix}, \ \boldsymbol{W} \in \mathbb{S}^\ell, \ \boldsymbol{X} \in \mathbb{R}^{\ell \times m}, \ \boldsymbol{Y} \in \mathbb{S}^m. \tag{8}$$

We can rewrite the SDP (6) as a standard primal SDP form

$$\text{minimize } \boldsymbol{A}_0 \bullet \boldsymbol{Z} \text{ subject to } \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t \ (t \in \Lambda) \ \boldsymbol{Z} \succeq \boldsymbol{O}, \tag{9}$$

where Λ denotes a finite index set. The constraint $\boldsymbol{W} = \boldsymbol{I}_\ell$ is included in the equality constraint $\boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t$. We then apply the conversion method [Nakata et al. 2003] to (9) as follows. Consider the index set $\mathcal{V}$ of rows (and columns) of the matrix variable $\boldsymbol{Z}$ and assume that the rows and columns of matrix $\boldsymbol{Z}$ in (8) are indexed in the lexicographical order as $1\ldots, \ell 0, *1, \ldots, *m$. Then, $\mathcal{V} = \{1\ldots, \ell 0, *1, \ldots, *m\}$ 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 define the *aggregated sparsity pattern* $\mathcal{E}$ of the data matrices as in [Fukuda et al. 2000] for the description of sparsity exploitation in the sparse SDP relaxation

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

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

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

Using the information on the chordal graph $G(\mathcal{V},\widetilde{\mathcal{E}})$ and its maximal cliques $\widetilde{\mathcal{C}}_1,\ldots,\widetilde{\mathcal{C}}_k$, application of the conversion method [Nakata et al. 2003] to (9) leads to an SDP problem

$$\left.\begin{array}{ll}\text{minimize} & \boldsymbol{A}_0 \bullet \boldsymbol{Z}\\ \text{subject to} & \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t\ (t \in \Lambda),\ \boldsymbol{Z}_{\widetilde{\mathcal{C}}_h,\widetilde{\mathcal{C}}_h} \succeq \boldsymbol{O}\ (1 \le h \le k),\end{array}\right\} \tag{10}$$

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

Rewriting (10), we obtain the sparse SDP relaxation of the SNL problem (4) without noise

$$\left.\begin{array}{ll}\text{minimize} & 0\\ \text{subject to} & d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq}\ \ (p,q) \in \mathcal{N}_x,\\ & d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp}\ (p,r) \in \mathcal{N}_a,\\ & \begin{pmatrix} \boldsymbol{I}_\ell & (\boldsymbol{x}_p : p \in C_h)\\ (\boldsymbol{x}_p : p \in C_h)^T & \boldsymbol{Y}_{C_h,C_h}\end{pmatrix} \succeq \boldsymbol{O}\ \ (1 \le h \le k),\end{array}\right\} \tag{11}$$

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 (11) is not the standard SDP form because some of the variables $\boldsymbol{x}_p$ $(p \in V_x)$ and Y_{pq} $((p,q) \in \mathcal{N}_x)$ appear more than once in the positive semidefinite constraints. To convert (11) to the standard SDP form, we use the domain-space conversion method [Kim et al. 2011; 2009], which was successfully implemented in SparsePOP, a Matlab package for polynomial optimization problems [Waki et al. 2006]. The resulting SDP involves k small positive semidefinite matrix variables induced from the k positive semidefinite constraints in (11). In contrast, the SDP (6) involves a single large $(\ell+m)\times(\ell+m)$ matrix variable $\boldsymbol{Z}$ given in (8). 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 needs to be solved for a search direction by Cholesky factorization at every iteration of the primal-dual interior-point methods. These two properties for the resulting SDP, multiple (but small) positive-semidefinite matrix variables and the correlative sparsity, greatly enhance the computational efficiency (see [Kobayashi et al. 2008] for more details of the correlative sparsity).

Let us denote

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

and let L_s denote the set of solutions of the sparse SDP relaxation (11). From [Kim et al. 2009a] we have $L_d = L_s$ 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 SNL problems were proposed in [Wang et al. 2008] as further relaxations of the FSDP relaxation. In essence, a generalization of the sparse SDP relaxation (11) includes NSDP and ESDP. In other words, if we denote the solution sets of the NSDP and ESDP relaxation by L_n and L_e, respectively, then, by construction, we know $L_d \subseteq L_n \subseteq L_e$. It was shown in [Wang et al. 2008] that if the underlying graph $G(V_x, \mathcal{N}_x)$ is chordal, then $L_d = L_n$. In this case, we know $G(V_x, \mathcal{N}_x) = G(V_x, \widetilde{\mathcal{N}}_x)$ and that $L_d = L_s = L_n$ also follows from Proposition 3.3 in [Kim et al. 2009a]. For details, we refer to [Kim et al. 2009a]. 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 described in this section is a fairly general technique for exploiting the sparsity of SDPs. The important feature of this technique is to convert a sparse SDP into another equivalent SDP that can be solved efficiently by exploiting its sparsity. From the construction of SDP relaxations, FSDP and SFSDP can be expected to attain more accurate solutions than ESDP and NSDP. Moreover, if the maximal cliques $C_1, C_2, \ldots, C_k$ of $G(V_x, \widetilde{\mathcal{N}}_x)$ are small, then the performance of SFSDP will be better than other SDP relaxations.

We conclude this section by describing the sparse SDP relaxation of the SNL problem (5) with noise

$$\left.\begin{array}{ll}
\text{minimize} & \displaystyle\sum_{(p,q)\in\mathcal{N}_x} (\xi^+_{pq} + \xi^-_{pq}) + \sum_{(p,r)\in\mathcal{N}_a} (\xi^+_{pr} + \xi^-_{pr}) \\
\text{subject to} & \hat{d}^2_{pq} = Y_{pp} + Y_{qq} - 2Y_{pq} + \xi^+_{pq} - \xi^-_{pq} \ \ (p,q)\in\mathcal{N}_x, \\
& \hat{d}^2_{pr} = \|\boldsymbol{a}_r\|^2 - 2\displaystyle\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp} + \xi^+_{pr} - \xi^-_{pr} \ \ (p,r)\in\mathcal{N}_a, \\
& \xi^+_{pq} \geq 0,\ \xi^-_{pq} \geq 0 \ \ (p,q)\in\mathcal{N}_x,\ \ \xi^+_{pr} \geq 0,\ \xi^-_{pr} \geq 0 \ \ (p,r)\in\mathcal{N}_a, \\
& \begin{pmatrix} \boldsymbol{I}_\ell & (\boldsymbol{x}_p : p\in C_h) \\ (\boldsymbol{x}_p : p\in C_h)^T & \boldsymbol{Y}_{C_h,C_h} \end{pmatrix} \succeq \boldsymbol{O} \ \ (1\leq h\leq k).
\end{array}\right\} \qquad (12)$$

4. SOME IMPLEMENTATION ISSUES

Let $\overline{\mathcal{N}}_x \subset \mathcal{N}^\rho_x$ be input set of pairs of sensors and $\overline{\mathcal{N}}_a \subset \mathcal{N}^\rho_a$ input set of pairs of a sensor and an anchor, and the (noisy) distance information be given for $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$. Although $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$ can be used for $\mathcal{N}_x$ and $\mathcal{N}_a$ to construct the sparse SDP relaxations (11) and (12) , we take a subset of $\overline{\mathcal{N}}_x$ for $\mathcal{N}_x$ and a subset of $\overline{\mathcal{N}}_a$ for $\mathcal{N}_a$ to reduce their sizes. In general, more accurate locations of sensors are obtained in longer computational time as the size of $\mathcal{N}_x \subset \overline{\mathcal{N}}_x$ and $\mathcal{N}_a \subset \overline{\mathcal{N}}_a$ increases. SFSDP incorporates a method for selecting $\mathcal{N}_x$ from $\overline{\mathcal{N}}_x$ and $\mathcal{N}_a$ from $\overline{\mathcal{N}}_a$ so that the resulting sparse SDP relaxation can be solved efficiently to find accurate locations of sensors.

In addition, SFSDP provides a regularization term [Leung and Toh 2009] in the objective function to obtain accurate locations of sensors for anchor-free problems.

The method for selecting $\mathcal{N}_a$ is presented in Section 4.1 and $\mathcal{N}_x$ in Section 4.2. Four types objective functions used in SFSDP are shown in Section 4.3. We use the notation $\deg(p, \overline{\mathcal{N}})$ for the number of edges incident to a sensor $p \in V_x$ for $\overline{\mathcal{N}}$.

4.1. Selecting edges from $\overline{\mathcal{N}}_a$ for $\mathcal{N}_a$

Given exact distances, we need to select at least $\ell + 1$ edges, incident to each sensor, from $\overline{\mathcal{N}}_x \cup \overline{\mathcal{N}}_a$ to locate all the sensors. In case of problems with noisy distances, more edges are needed to obtain the accurate location of sensors. We also note that the total

number of elements in $\mathcal{N}_x$ and $\mathcal{N}_a$ is equal to the number of equality constraints in the sparse SDP relaxations (11) and (12). In addition, the construction of k positive semidefinite constraints in (11) and (12) does not depend on $\mathcal{N}_a$ because the maximal cliques $C_1, C_2, \ldots, C_k$ of a chordal extension of the graph $G(V_x, \mathcal{N}_x)$ determine the positive semidefinite constraints as shown in Section 3.2. As the number of edges selected from $\overline{\mathcal{N}}_a$ for $\mathcal{N}_a$ increases, the number of edges that need to be selected from $\overline{\mathcal{N}}_x$ for $\mathcal{N}_x$ decreases. Thus, selecting edges from $\overline{\mathcal{N}}_a$ first leads to smaller maximal cliques than selecting edges from $\overline{\mathcal{N}}_x$ first. Since it is more efficient to solve the resulting sparse SDP relaxation with smaller maximal cliques, we give priority to the selection of edges of $\overline{\mathcal{N}}_a$ for the construction of the sparse SDP relaxations (11) and (12). Let $\nu = \ell + 1$ if all distances are exact and $\nu = 2 \times (\ell + 1)$ otherwise. SFSDP selects $\min\{\nu, \deg(p, \overline{\mathcal{N}}_a)\}$ edges from $\overline{\mathcal{N}}_a$ for each $p \in V_x$ to form $\mathcal{N}_a$.

4.2. Selecting edges from $\overline{\mathcal{N}}_x$ for $\mathcal{N}_x$

Assume that $\mathcal{N}_a$ has been constructed by the method described in Section 4.1 and fixed. For selection of edges for $\mathcal{N}_x$, we try to satisfy two conditions whenever possible: (i) at least κ edges from $\overline{\mathcal{N}}_x \cup \overline{\mathcal{N}}_a$ are incident to each sensor, where κ is taken to be at least $\ell + 1$ and (ii) at least one edge from $\overline{\mathcal{N}}_x$ needs to be selected to form $\mathcal{N}_x$ for each sensor. Define

$$\kappa_p = \kappa - \min\{\deg(p, \mathcal{N}_a),\ \ell\} \text{ for every sensor } p \in V_x.$$

We consider a family $\mathcal{G}_\kappa$ of subgraphs $G(V_x, \mathcal{N}_x)$ of $G(V_x, \overline{\mathcal{N}}_x)$ satisfying $\deg(p, \mathcal{N}_x) \geq \kappa_p$ for every sensor $p \in V_x$. SFSDP selects edges for $\mathcal{N}_x$, such that $G(V_x, \mathcal{N}_x)$ is a minimal graph in the class $\mathcal{G}_\kappa$, by applying the heuristic method

. Step 0: Let $V_x = \{1, 2, \ldots, m\}$, $p = 1$ and $\mathcal{N}_x = \emptyset$.
. Step 1: If $p = m$, stop.
. Step 2: For every edge $(p, q) \in \overline{\mathcal{N}}_x$ with $q \geq p + 1$, if either $\deg(p, \mathcal{N}_x) < \kappa_p$ or $\deg(q, \mathcal{N}_x) < \kappa_q$, then let $\mathcal{N}_x = \mathcal{N}_x \cup \{(p, q)\}$.
. Step 3: Let $p = p + 1$ and go to Step 1.

This heuristic is easy to implement and from the numerical experiments in Section 5, we have confirmed the effectiveness of the method. More precisely, this method constructs a graph $G(V_x, \mathcal{N}_x)$ that leads to a chordal extension with small maximal cliques $C_1, C_2, \ldots, C_k$, even when the original graph $G(V_x, \overline{\mathcal{N}}_x)$ is dense. We can verify that if $G(V_x, \overline{\mathcal{N}}_x)$ is a complete graph, as an extreme case, the method generates $\mathcal{N}_x = \{(p, q) : p < q \leq m, 1 \leq p \leq \kappa\}$. Thus, the graph $G(V_x, \mathcal{N}_x)$ induces a chordal extension $G(V_x, \widetilde{\mathcal{N}}_x)$ having maximal cliques $\{1, 2, \ldots, \kappa, q\}$ $(q = \kappa + 1, \ldots, m)$ of size $\kappa + 1$.

As an example, we considered a two-dimensional SNL problem of 1000 sensors and 4 anchors at the corners of the square $[0, 1] \times [0, 1]$ with exact distances and 0.1 radio range. Numerical results are reported for this case in Table 6.1. We used $\kappa = 4$. Before the selecting process, the number of cliques was 381, and the maximum, minimum and average size of cliques were 91, 8 and 31.7, respectively. After the selecting process, the number of cliques became 912, and the maximum, minimum and average of the cliques were 30, 4 and 6, respectively. Thus, we observe that the selecting process provides an increased number of cliques of smaller sizes, which can be dealt with more efficiently.

With an increasingly large value for κ, we can expect to obtain more accurate locations of sensors, although it takes longer to solve the sparse SDP relaxation problem. Some applications of the SNL problem (e.g., molecular conformation) do not have enough distance information to obtain accurate solutions. For these applications, κ can be set so as not to reduce the size of $\overline{\mathcal{N}}_x$. In SFSDP, a parameter named pars.minDegree

Functions	Description
SFSDPplus.m	A function that analyzes the given input data and calls SFSDP.m.
SFSDP.m	A solver that solves the problem and returns the locations of sensors. It may be called directly by users.
generateProblem.m	A function that generates an SNL problem with the given parameters.
test_SFSDP.m	A function that solves the problem generated by generateProblem.m using SFSDPplus.m.

Functions provided in SFSDP

can be specified for the value of κ. If users choose pars.minDegree $= \kappa \geq 100$, then $\mathcal{N}_x = \overline{\mathcal{N}}_x$ and $\mathcal{N}_a = \overline{\mathcal{N}}_a$ are used. The default value for pars.minDegree is ℓ + 2.

4.3. Selecting Objective Functions

SFSDP provides four objective functions to select depending on whether the problem contains noise and whether the number of anchors is small or there are no anchors. One of the four objective functions may be selected by setting a value of 0-3 for the parameter called pars.objSW.

Users can set pars.objSW = 0 for the zero objective function to solve the SNL problem (4) with exact distances, and pars.objSW = 1 for the sum of the 1-norm error to solve the SNL problem (5) with noisy distances. But these objective functions often fail to provide accurate locations of sensors if no or only a small number of anchors are available. In such cases, an objective function with the regularization term [Biswas et al. 2008; Leung and Toh 2009]

$$- \sum_{(p,q) \in \widetilde{\mathcal{N}}_x} \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 \tag{13}$$

gives more accurate locations. Recall that $\widetilde{\mathcal{N}}_x$ denotes the edge set of a chordal extension $G(V_x, \widetilde{\mathcal{N}}_x)$ of the graph of $G(V_x, \mathcal{N}_x)$, which was introduced for constructing the sparse SDP relaxations (11) and (12) in Section 3. Setting pars.objSW = 2 or 3 adds the regularization term described in (13) to the zero objective function or the sum of the 1-norm error, respectively.

5. NUMERICAL RESULTS

SFSDP includes four Matlab functions whose names and functionality are described in Table 5. See the user's guide [Kim et al. 2009b] for more details.

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

— ESDP using SDPA and SeDuMi
— FSDP using SDPA and SeDuMi
— SFSDP using SDPA and SeDuMi

applied to 2-dimensional SNL problems with 1000 to 5000 sensors. This comparison shows that SFSDP using SDPA is more efficient for computing the locations of sensors with higher accuracy than the other methods. We then compare the performance of SFSDP with SDPA on larger 2-dimensional problems with 20000 sensors, 3-dimensional problems with 3000 to 5000 sensors, and 3-dimensional anchor-free problems with 3000 to 5000 sensors. Numerical experiments (except for those reported in Section 5.2) were performed on 2.8GHz Quad-Core Intel Xeon with 4GB memory using SDPA 7.3.1 and SeDuMi 1.1R3. The problems with 20000 sensors in Section 5.2 were

solved on 2.8GHz Quad-Core Intel Core i7 with 16GB memory using SDPA 7.3.1 and SeDuMi 1.21. We used Matlab version 7.9. (R2009b).

Throughout our numerical experiments, we generated sensors $\boldsymbol{a}_p$ $(p = 1, 2, \ldots, m)$ randomly in the unit square $[0, 1]^2$ for 2-dimensional problems or in the unit cube $[0, 1]^3$ for 3-dimensional problems. The number of sensors ranged from 1000 to 20000 for 2-dimensional problems, and 3000 to 5000 for 3-dimensional problems. Two types of radio ranges were used. The first type is a constant radio range $\rho = 0.1$ for 2-dimensional problems (or $\rho = 0.250$ for 3-dimensional problems) which is independent of the number of sensors. The second type is $\rho = \sqrt{10/m}$ for 2-dimensional problems (or $\rho = (15/m)^{1/3}$ for 3-dimensional problems) where m denotes the number of sensors. The second type indicates that each square of size ρ in $[0, 1]^2$ contains 10 randomly generated sensors on average for 2-dimensional problems or each cube of size ρ in $[0, 1]^3$ contains 15 randomly generated sensors on average for 3-dimensional problems. Anchors were either randomly distributed in the unit square/cube or they were placed at the corners of the unit square/cube. In problems with randomly distributed anchors, the number of anchors was set to 10% or 5% of the number of sensors. The noisy factor was changed from 0.0 to 0.2. The distances were perturbed to create problems with noise

$$\begin{array}{l} \hat{d}_{pq} = \max\{(1 + \sigma\epsilon_{pq}),\ 0.1\}\|\boldsymbol{a}_p - \boldsymbol{a}_q\| \ ((p, q) \in \overline{\mathcal{N}}_x = \mathcal{N}_x^\rho), \\ \hat{d}_{pr} = \max\{(1 + \sigma\epsilon_{pr}),\ 0.1\}\|\boldsymbol{a}_p - \boldsymbol{a}_r\| \ ((p, r) \in \overline{\mathcal{N}}_a = \mathcal{N}_a^\rho), \end{array} \tag{14}$$

where $\sigma \geq 0$ denotes a noisy factor, ϵ_{pq} and ϵ_{pr} are chosen from the standard normal distribution $N(0, 1)$, and $\boldsymbol{a}_p$ denotes the true location of the pth sensor. Note that we set $\overline{\mathcal{N}}_x = \mathcal{N}_x^\rho$ and $\overline{\mathcal{N}}_a = \mathcal{N}_a^\rho$ when generating the test problems. As in [Biswas and Ye 2004; 2006; Biswas et al. 2006; Tseng 2007; Wang et al. 2008], the root mean square distance (RMSD)

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

where $\boldsymbol{x}_p$ denotes the computed locations of the pth sensor, 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.

In the numerical experiments, each type of the test problems was generated 5 times and tested. The values of elapsed time in the tables below are the average of values from those 5 experiments, and the value of RMSD was computed by

$$\left(\frac{1}{5m}\sum_{k=1}^{5}\sum_{p=1}^{m}\|\boldsymbol{x}_p^k - \boldsymbol{a}_p^k\|^2\right)^{1/2},$$

where $\boldsymbol{a}_p^k$ and $\boldsymbol{x}_p^k$ denote the true and computed locations of the pth sensor of the kth instance of test problems.

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

In Table 5.1, the RMSD and elapsed time for ESDP, FSDP and SFSDP are compared for problems with 1000 sensors and $\rho = 0.1$. We let λ and κ denote upper and lower bounds respectively for the degree of any sensor node in ESDP described in Section 4.2 (see also Section 4.1 of [Kim et al. 2009a] for their precise definitions and the comparison). FSDP, ESDP and SFSDP were tested with SDPA and SeDuMi. After obtaining a solution by an SDP solver, the solution was refined using the gradient method. The two columns under RMSD indicate the values of RMSD before and after the refinement. In all tested problems, we observe that SDPA provides a solution much

faster than SeDuMi with comparable values of RMSD, and FSDP is much slower than ESDP and SFSDP. Notice that SFSDP attains solutions as accurately as FSDP. From these observations, we compare ESDP and SFSDP both using SDPA in the rest of this section.

Numerical tests with increasing values of the parameters λ and κ for ESDP and SFSDP, respectively, were performed and the results are shown in the lower part of Table 5.1. As the parameters increase, both ESDP and SFSDP with the gradient method showed an increase in execution time but produced more accurate values of RMSD. Notice that SFSDP(3) resulted in more accurate RMSD than ESDP(8).

Test problems				RMSD		Elapsed time		
$m,\ m_a,\ \rho$	σ	SDP(λ|κ)	Solver	SDP	w.Grad.	Solver	Grad.	Total
$m = 1000$,	0.0	FSDP(4)	SeDuMi	4.2e-5	7.1e-6	2907.2	0.1	2910.3
$m_a = 100$			SDPA	5.0e-4	1.0e-5	53.8	0.2	56.9
distributed		ESDP(5)	SeDuMi	1.2e-3	1.4e-4	56.3	0.6	78.5
randomly.			SDPA	1.0e-3	1.5e-4	19.4	0.6	43.2
$\rho = 0.100$		SFSDP(4)	SeDuMi	6.4e-6	2.1e-6	7.4	0.1	12.1
			SDPA	4.9e-5	6.0e-6	2.3	0.3	7.3
	0.1	FSDP(4)	SeDuMi	1.7e-2	7.1e-3	5285.7	1.5	5294.0
			SDPA	1.7e-2	7.1e-3	308.6	1.5	317.3
		ESDP(5)	SeDuMi	1.6e-2	7.9e-3	42.5	1.7	66.3
			SDPA	1.6e-2	7.7e-3	13.2	1.5	39.1
		SFSDP(4)	SeDuMi	1.7e-2	7.0e-3	20.2	6.1	34.9
			SDPA	1.7e-2	7.1e-3	5.1	5.0	18.9
$m = 1000$,	0.0	FSDP(4)	SeDuMi	3.0e-5	1.3e-5	2186.3	0.2	2189.9
$m_a = 4$			SDPA	5.5e-5	2.1e-5	45.4	0.3	48.8
at corners,		ESDP(5)	SeDuMi	3.9e-2	1.7e-2	47.9	13.6	80.9
$\rho = 0.100$			SDPA	3.7e-2	1.3e-2	16.7	13.6	49.9
		SFSDP(4)	SeDuMi	1.6e-5	9.0e-6	29.3	0.1	34.4
			SDPA	5.8e-5	2.5e-5	12.6	0.3	17.8
	0.1	FSDP(4)	SeDuMi	4.5e-2	1.0e-2	2921.6	14.6	2943.4
			SDPA	4.5e-2	1.0e-2	157.2	14.9	179.5
		ESDP(5)	SeDuMi	4.8e-2	2.0e-2	43.2	13.5	75.7
			SDPA	4.8e-2	1.9e-2	16.8	13.8	50.5
		SFSDP(4)	SeDuMi	4.5e-2	1.0e-2	45.6	12.2	66.9
			SDPA	4.5e-2	1.0e-2	18.3	12.5	39.8
		ESDP(4)	SDPA	4.7e-02	2.4e-02	13.6	8.9	40.1
		ESDP(6)	SDPA	4.9e-02	1.5e-02	20.7	10.9	52.9
		ESDP(8)	SDPA	5.2e-02	1.3e-02	31.6	12.6	70.0
		SFSDP(3)	SDPA	5.1e-02	1.2e-02	7.8	10.9	25.6
		SFSDP(5)	SDPA	4.3e-02	1.0e-02	39.9	8.8	56.5
		SFSDP(7)	SDPA	4.0e-02	7.8e-03	142.9	6.6	158.5

Comparing FSDP(4), ESDP(5) and SFSDP(5) with SeDuMi and SDPA to solve 2*-dimensional problems with 1000 sensors and* $\rho = 0.1$.

Numerical results from test problems with 3000 and 5000 sensors are shown in Table 5.1. The number of anchors was changed from 4 to 10% of m, the radio range was fixed to 0.1 or computed by $\sqrt{10/m}$, and the noisy factor was changed from 0.0 to 0.2. SDPA was used to solve the SDPs. We observe

(1) Total time spent by SFSDP and elapsed time by SDPA are shorter for the problems with exact distances ($\sigma = 0.0$) than noisy distances ($\sigma = 0.1$ or 0.2). As mentioned in Section 4.3, the objective function is set to zero for the problems with exact distances. This makes the size of SDP relaxations smaller.
(2) The timings obtained using SFSDP with SDPA decreased as the number of anchors increased. Recall that as the number of anchors decreases the candidates for $\mathcal{N}_a$ decreases and more edges need to be selected from $\mathcal{N}_x^\rho$ for $\mathcal{N}_x$ in the construction of the SDP relaxation. Thus, the graph $G(V_x, \mathcal{N}_x)$ becomes denser and the sizes of the positive semidefinite constraints in the resulting SDP relaxation grow (see Sections 4.1 and 4.2).
(3) A smaller value of the radio range ρ increases the total time used by SFSDP and the elapsed time required by SDPA. Note that, as ρ decreases, the number of edges in $\mathcal{N}_x^\rho$ decreases and selecting edges from $\mathcal{N}_x^\rho$ for $\mathcal{N}_x$ becomes more restricted. As a result, the chordal extension $G(V_x, \widetilde{\mathcal{N}}_x)$ of the graph $G(V_x, \mathcal{N}_x)$ becomes denser and the sizes of the positive semidefinite constraints in the resulting SDP relaxation increases. See also Sections 4.1 and 4.2.

Table 5.1 shows that SFSDP took longer to solve an SDP by SDPA than ESDP for the three problems: (a) $m = 3000$, $m_a = 4$, $\rho = \sqrt{10/m} \approx 0.058$ and $\sigma > 0$, (b) $m = 5000$, $m_a = 5\%$ of $m = 250$, $\rho = \sqrt{10/m} \approx 0.045$ and $\sigma > 0$, and (c) $m = 5000$, $m_a = 4$, $\rho = \sqrt{10/m} \approx 0.045$ and $\sigma > 0$. These are due to (1), (2) and (3).

The RMSD values reported in Table 5.1 and Figures 5.1 and 5.1 were obtained after refinement using the gradient method. We see that SFSDP provides more accurate values of RMSD than ESDP in Table 5.1. Figure 5.1 and 5.1 show the differences in the values of RMSD from SFSDP and ESDP for the problems with 1000 to 5000 sensors in the presence of noise. Figure 5.1 displays the results for the problems where the number of anchors is 10% of the number of sensors. For the experiments with the noisy factor 0.1 and 0.2, the values of RMSD from SFSDP are smaller than those from ESDP for all test problems. In Figure 5.1, we observe similar results for the problems with 4 anchors at the corner of the unit square.

5.2. Two-dimensional Larger-scale Problems

In Table 5.2, we show the numerical results for the problems with 20000 sensors. Anchors are randomly generated or placed at the corners of the unit square and the noisy factor σ, was changed from 0.0 to 0.2 when generating the problems. SFSDP solved the problems efficiently with accurate values of RMSD. Numerical results in Table 5.2 support remarks in (1), (2) and (3) in Section 5.1. We note that the test problems shown in [Pong and Tseng 2010] involve σ up to 0.01.

5.3. Three-dimensional Problems

Numerical results for three-dimensional problems are shown in Table 5.3. Test problems include 3000 to 5000 sensors with various anchor distributions. SFSDP efficiently solves three-dimensional problems with 3000 and 5000 sensors, providing accurate values of RMSD in all but two cases. Once again the results support remarks (1), (2) and (3) in Section 5.1.

Figure 1 displays the locations of sensors from SFSDP after refining with the gradient method for the 3-dimensional problems with 3000 sensors, 8 anchors at the corners, and a radio range of 0.25. The noisy factor of the problem on the left is 0.0 and on the right 0.1.

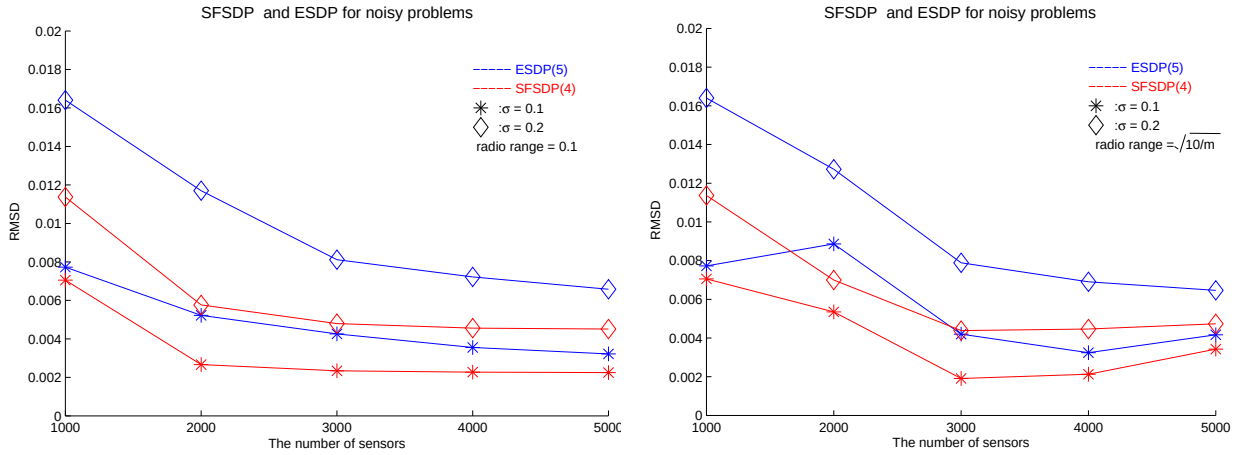
Test problems			ESDP(5)			SFSDP(4)		
			Elapsed time				Elapsed time	
$m,\ m_a$	ρ	σ	SDPA	Total	RMSD	RMSD	Total	SDPA
$m = 3000$,	0.100	0.0	68.0	359.2	1.7e-3	2.3e-7	36.9	6.2
$m_a = 10\%$ of m		0.1	71.3	351.3	4.3e-3	2.3e-3	95.8	15.9
$= 300$ distributed		0.2	70.1	348.0	8.1e-3	4.8e-3	98.8	15.1
randomly	$\sqrt{10/m}$	0.0	69.3	241.2	8.1e-4	4.0e-6	43.5	12.3
	≈ 0.058	0.1	47.5	216.9	4.2e-3	1.9e-3	88.2	22.4
		0.2	47.0	216.2	7.9e-3	4.4e-3	88.9	21.5
$m = 3000$,	0.100	0.0	61.8	251.6	2.1e-3	1.3e-6	37.9	7.0
$m_a = 5\%$ of m		0.1	60.3	251.1	5.6e-3	2.5e-3	88.6	15.3
$= 150$ distributed		0.2	58.8	248.9	1.1e-2	5.2e-3	91.8	14.4
randomly	$\sqrt{10/m}$	0.0	68.1	211.1	1.4e-3	9.5e-6	60.8	28.5
	≈ 0.058	0.1	44.9	194.7	7.3e-3	4.5e-3	112.0	39.9
		0.2	44.6	192.0	1.1e-2	5.8e-3	115.7	40.1
$m = 3000$,	0.100	0.0	84.6	274.0	1.2e-2	6.7e-6	74.2	36.8
$m_a = 4$ at corners		0.1	76.3	261.8	1.7e-2	5.6e-3	177.9	62.9
		0.2	76.3	263.4	3.3e-2	9.3e-3	176.7	63.3
	$\sqrt{10/m}$	0.0	71.1	258.3	7.2e-3	7.6e-5	206.6	167.5
	≈ 0.058	0.1	72.8	260.8	1.2e-2	7.1e-3	297.2	172.3
		0.2	77.2	265.4	3.3e-2	1.1e-2	312.4	176.4
$m = 5000$,	0.1	0.0	175.8	1791.1	1.1e-7	2.7e-7	94.7	11.2
$m_a = 10\%$ of m		0.1	160.2	1776.2	3.2e-3	2.2e-3	244.7	30.5
$= 500$ distributed		0.2	159.4	1779.1	6.6e-3	4.5e-3	241.4	28.9
randomly	$\sqrt{10/m}$	0.0	112.9	585.3	9.5e-4	4.4e-6	111.8	26.7
	≈ 0.045	0.1	88.0	562.3	4.2e-3	3.4e-3	219.1	28.5
		0.2	85.2	560.5	6.5e-3	4.7e-3	230.9	29.3
$m = 5000$,	0.100	0.0	140.1	916.6	6.6e-5	5.9e-7	93.6	11.5
$m_a = 5\%$ of m		0.1	123.1	899.6	4.2e-3	2.5e-3	229.8	31.3
$= 250$ distributed		0.2	121.1	911.4	8.9e-3	4.8e-3	231.3	27.5
randomly	$\sqrt{10/m}$	0.0	115.6	508.6	2.7e-3	1.5e-4	156.1	68.3
	≈ 0.045	0.1	83.3	481.0	6.8e-3	5.1e-3	283.0	98.9
		0.2	81.7	476.1	1.0e-2	6.3e-3	281.7	103.4
$m = 5000$,	0.100	0.0	164.9	668.5	1.3e-2	2.2e-5	153.6	56.7
$m_a = 4$ at corners		0.1	145.4	657.6	1.7e-2	5.4e-3	374.9	79.4
		0.2	145.3	658.0	3.1e-2	9.6e-3	328.6	79.5
	$\sqrt{10/m}$	0.0	145.3	644.8	8.1e-3	1.5e-4	604.6	502.2
	≈ 0.045	0.1	149.9	645.8	1.6e-2	6.0e-3	908.8	566.2
		0.2	155.4	659.7	4.2e-2	1.3e-2	1040.4	584.6

Comparison between ESDP(5) and SFSDP(4) with SDPA to solve 2-dimensional problems with 3000 and 5000 sensors.

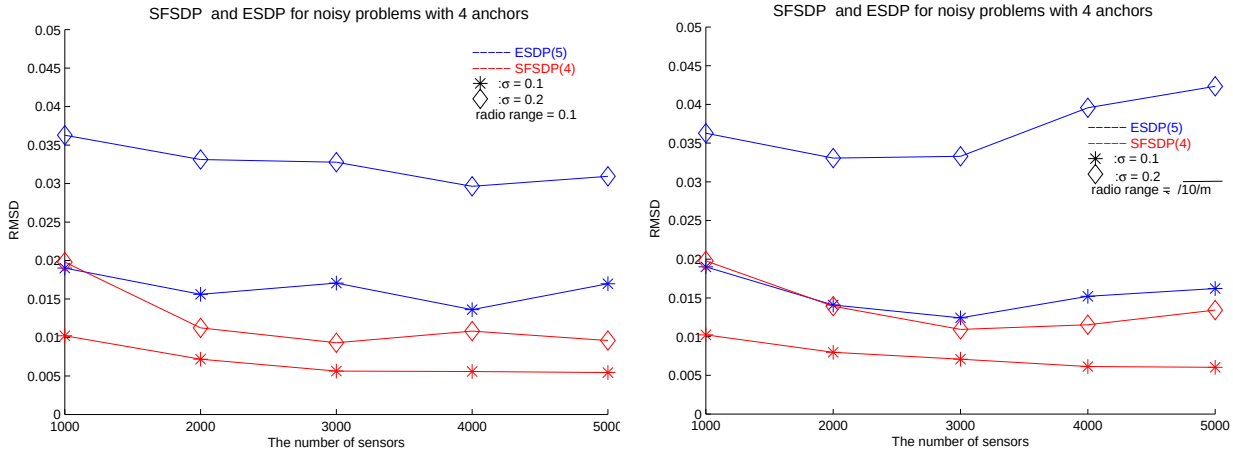
Figure 2 shows the locations of sensors before and after refining the solution with the gradient method for the 3-dimensional problems with 3000 sensors, 8 anchors at the corners, the radio range 0.25, and noisy factor 0.2.

5.4. Anchor-free Problems in Three Dimensions

SFSDP handles anchor-free problems in ℓ dimensions, $\ell = 2$ or 3, by first fixing $\ell + 1$ sensors as anchors, which form a clique, and then applying the sparse SDP relaxation to the resulting problem. More precisely, if an anchor-free SNL problem with n sensors



The number of anchors is 10 % of the number of sensors. The anchors are distributed randomly. The radio range used for the left figure is 0.1, and for the right $\sqrt{10/m}$.



Four anchors are placed at the corner of $[0,1]^2$. *The radio range used for the left figure is 0.1, and for the right* $\sqrt{10/m}$.

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

We can measure the accuracy of computed solutions when the true locations $\boldsymbol{a}_p$ $(p = 1, 2, \ldots, n)$ of all sensors are known. In the numerical experiments whose re-

Test problems			RMSD		Elapsed time		
$m_a,\ m$	ρ	σ	SDPA	w.Grad.	SDPA	Grad.	Total
$m = 20000$,	0.100	0.0	1.44e-06	3.09e-07	93.4	0.7	326.9
$m_a = 10\%$ of m		0.1	9.54e-03	2.21e-03	159.2	16.8	877.3
$= 2000$ distributed		0.2	1.97e-02	4.43e-03	148.2	23.5	882.1
randomly	$\sqrt{10/m}$	0.0	1.10e-04	4.07e-06	466.2	4.4	708.0
	≈ 0.022	0.1	3.19e-03	7.86e-04	301.5	35.9	823.0
		0.2	6.27e-03	1.50e-03	237.6	41.4	773.1
$m = 20000$,	0.100	0.0	1.44e-06	3.09e-07	88.0	0.7	320.2
$m_a = 5\%$ of m		0.1	9.54e-03	2.21e-03	158.5	14.6	877.8
$= 1000$ distributed		0.2	1.97e-02	4.44e-03	150.1	17.1	880.0
randomly	$\sqrt{10/m}$	0.0	1.96e-04	8.56e-06	1487.6	6.4	1752.8
	≈ 0.022	0.1	3.69e-03	8.76e-04	1879.2	45.5	2388.4
		0.2	7.28e-03	1.96e-03	1577.9	46.6	2096.3
$m = 20000$,	0.100	0.0	4.01e-05	6.92e-06	182.9	2.0	469.2
$m_a = 4$ at corners		0.1	4.18e-02	7.57e-03	403.0	137.7	1140.0
		0.2	6.63e-02	1.08e-02	402.6	146.0	1150.5
	$\sqrt{10/m}$ ≈ 0.022	Requires more than 16Gb of memory					

Numerical results on SFSDP(4) with SDPA applied to 2*-dimensional problems with* 20000 *sensors.*

sults are shown in Table 5.4, the values of RMSD for the computed locations of sensors $\boldsymbol{x}_p$ $(p = 1, 2, \ldots, n)$ are evaluated after applying a linear transformation (translation, reflection, orthogonal rotation, and scaling) T, provided by the Matlab program *p*rocrustes.m [Biswas et al. 2008; Leung and Toh 2009]. This function minimizes the total squared errors $\sum_{p=1}^{n} \|T(\boldsymbol{x}_p) - \boldsymbol{a}_p\|^2$ between the true and transformed approximate locations of sensors. We also observe that the location of sensors, as shown by the values of RMSD is found accurately using SFSDP.

6. CONCLUDING REMARKS

We have described the Matlab package SFSDP. It is designed to solve larger SNL problems than other available software. SFSDP can be used for problems with various anchor locations and anchor-free problems in both two and three dimensions.

SFSDP demonstrates the computational advantages over other methods in solving large SNL problems as shown in Section 5. These come from utilizing the aggregated and correlative sparsity of the problem, which reduces the size of FSDP relaxation. One advantage of SFSDP is that it is equipped with both SDPA and SeDuMi. Incorporating SFSDP with SDPA allows faster solutions that using SeDuMi. However, in our experience, the accuracy of the computed solution is better with SeDuMi for some problems.

The SNL problem has a number of applications where computational efficiency is an important issue. The SDP approach has been known to be effective in locating sensors, however, solving large problems with this approach has been a challenge. We hope to improve the performance of the sparse SDP relaxation implemented in SFSDP, in particular, when the original problem does not provide enough distance information between sensors and/or no or few anchors are available.

Test problems			RMSD		Elapsed time		
m, m_a	ρ	σ	SDPA	w.Grad.	SDPA	Grad.	Total
$m = 3000$,	0.250	0.0	6.2e-06	9.8e-07	12.1	0.4	48.1
$m_a = 10\%$ of m		0.1	3.6e-02	9.5e-03	20.9	19.5	138.2
= 300		0.2	6.1e-02	1.7e-02	20.3	26.3	144.7
distributed	$(15/m)^{1/3}$	0.0	1.0e-04	2.9e-06	49.3	1.2	87.3
randomly	≈ 0.171	0.1	3.2e-02	7.0e-03	51.3	27.8	164.4
		0.2	4.9e-02	1.6e-02	51.6	25.3	162.0
$m = 3000$,	0.250	0.0	2.6e-05	1.2e-06	13.5	0.7	51.1
$m_a = 5\%$ of m		0.1	5.0e-02	1.0e-02	20.6	17.9	134.2
=150		0.2	7.5e-02	1.9e-02	20.4	25.6	142.0
distributed	$(15/m)^{1/3}$	0.0	1.4e-04	6.3e-06	260.4	2.4	301.0
randomly	≈ 0.171	0.1	4.8e-02	1.8e-02	190.3	40.9	312.5
		0.2	6.7e-02	2.7e-02	194.0	41.5	315.8
$m = 3000$,	0.250	0.0	1.0e-04	7.5e-06	368.4	1.2	413.0
$m_a = 8$ at corners		0.1	9.1e-02	1.4e-02	428.4	64.8	589.5
		0.2	1.4e-01	2.6e-02	422.2	45.9	563.6
	$(15/m)^{1/3}$	Requires more than 16Gb of memory					
	≈ 0.171						
$m = 5000$,	0.250	0.0	1.4e-06	4.2e-07	17.5	0.5	112.2
$m_a = 10\%$ of m		0.1	3.2e-02	7.9e-03	41.9	30.1	337.8
=500		0.2	6.1e-02	1.6e-02	37.5	31.8	331.8
distributed	$(15/m)^{1/3}$	0.0	8.8e-05	2.1e-06	194.5	2.8	295.4
randomly	≈ 0.144	0.1	2.8e-02	5.8e-03	166.2	51.4	445.8
		0.2	4.2e-02	1.2e-02	170.1	54.4	452.2
$m = 5000$,	0.250	0.0	1.8e-05	7.7e-07	18.7	1.2	117.3
$m_a = 5\%$ of m		0.1	3.7e-02	8.3e-03	40.8	34.7	337.8
= 250		0.2	6.3e-02	1.7e-02	39.3	42.6	348.9
distributed	$(15/m)^{1/3}$	Requires more than 16Gb of memory					
randomly	≈ 0.144						

Numerical results on SFSDP(5) with SDPA applied to 3-dimensional problems.

Test problems			RMSD		Elapsed time		
m, m_a	ρ	σ	SDPA	w.Grad.	SDPA	Grad.	Total
$m = 3000$,	0.250	0.0	5.0e-05	5.1e-06	441.8	0.6	490.3
$m_a = 0$		0.1	1.6e-01	9.4e-03	504.6	11.4	612.3
		0.2	1.8e-01	2.5e-02	497.0	13.4	607.4
$m = 3000$,	$(15/m)^{1/3}$	Requires more than 16Gb of memory					
$m_a = 0$	≈ 0.171						
$m = 5000$,	0.250	0.0	5.2e-05	7.6e-06	723.6	0.9	852.0
$m_a = 0$		0.1	8.1e-02	9.3e-03	790.3	16.2	1060.3
		0.2	1.3e-01	1.9e-02	776.3	16.4	1046.5

Numerical results on SFSDP(5) with SDPA applied to 3-dimensional anchor-free problems.

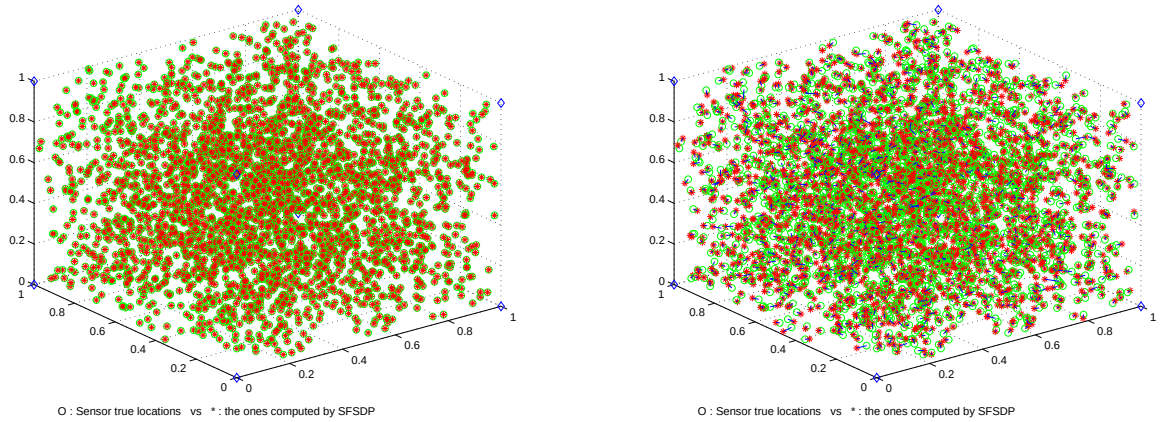


Fig. 1. A 3-dimensional problem with 3000 sensors, 8 anchors at the corners, $\rho = 0.25$, and $\sigma = 0.0$ on the left and $\sigma = 0.1$ on the right. The locations of sensors from SFSDP($\kappa = 4$) after the refinement. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a difference between true and computed location.

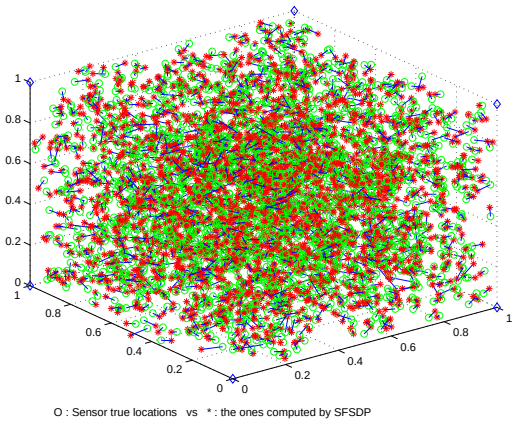


Fig. 2. A 3-dimensional problem with 3000 sensors, 8 anchors at the corners, $\rho = 0.25$, and $\sigma = 0.2$. The locations of sensors from SFSDP($\kappa = 4$) after the refinement using the gradient method. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a difference between true and computed location.

ACKNOWLEDGMENTS

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

REFERENCES

ALFAKIH, A. Y., KHANDANI, A., AND WOLKOWICZ, H. 1999. Solving euclidean matrix completion problem via semidefinite programming. *Comput. Opt. Appl. 12*, 13–30.

BISWAS, P., LIANG, T.-C., TOH, K.-C., WANG, T.-C., AND YE, Y. 2006. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering 3*, 360–371.

BISWAS, P., LIANG, T.-C., WANG, T.-C., AND YE, Y. 2006. Semidefinite programming based algorithms for sensor network localization. *ACM Tran. Sensor Networks 2*, 188–220.

BISWAS, P., TOH, K., AND YE, Y. 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.

BISWAS, P. AND YE, Y. 2004. Semidefinite programming for ad hoc wireless sensor network localization. In *Proceedings of the third international symposium on information processing in sensor networks*. ACM, Berkeley, California.

BISWAS, P. AND YE, Y. 2006. 'a distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization. In *Multiscale Optimization Methods and Applications*. Springer.

BLAIR, J. R. S. AND PEYTON, B. 1993. An introduction to chordal graphs and clique trees. In *A. George, J. R. Gilbert and J. W. H. Liu des, Graph Theory and Sparse Matrix Computation*. Springer, New York, 1–29.

CARTER, M. W., JIN, H. H., SAUNDERS, M. A., AND YE, Y. 2006. Spaseloc: an adaptive subproblem algorithm for scalable wireless sensor network localization. *SIAM J. Optim. 17,* 4, 1102–1128.

DOHERTY, L., PISTER, K. S. J., AND GHAOUI, L. E. 2001. Convex position estimation in wireless sensor networks. In *Proceedings of 20th INFOCOM*. Vol. 3. 1655–1663.

EREN, T., GOLDENBERG, D. K., WHITELEY, W., WANG, Y. R., MORSE, A. S., ANDERSON, B. D. O., AND BELHUMEUR, P. N. 2004. Rigidity, computation, and randomization in network localization. In *in Proceedings of IEEE Infocom*.

FUJISAWA, K., FUKUDA, M., KOBAYASHI, K., KOJIMA, M., NAKATA, K., NAKATA, M., AND YAMASHITA, M. 2008. Sdpa (semidefinite programming algorithm) user's manual — version 7.0.5. Tech. Rep. Research Report B-448, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

FUKUDA, M., KOJIMA, M., MUROTA, K., AND NAKATA, K. 2000. Exploiting sparsity in semidefinite programming via matrix completion i: General framework. *SIAM J. Optim. 11*, 647–674.

GANESAN, D., KRISHNAMACHARI, B., WOO, A., CULLER, D., ESTRIN, D., AND S.WICKER. 2002. An empirical study of epidemic algorithms in large scale multihop wireless network. Tech. Rep. IRB-TR-02-003, Intel Corporation.

GOLUMBIC, M. C. 1980. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York.

HOWARD, A., MATARIĆ, M., AND SUKHATME, G. 2001. Relaxation on a mesh: a formalism for generalized localization. In *IEEE/RSJ International conference on intelligent robots and systems*. Wailea, Hawaii, 1055–1060.

KIM, S., KOJIMA, M., MEVISSEN, M., AND YAMASHITA, M. 2009. User's manual for sparsecolo: Conversion methods for sparse conic-form linear optimization problems. Tech. Rep. Research Report B-453, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

KIM, S., KOJIMA, M., MEVISSEN, M., AND YAMASHITA, M. 2011. Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. *Math. Program. 129,* 1, 33–68.

KIM, S., KOJIMA, M., AND WAKI, H. 2009a. Exploiting sparsity in sdp relaxation for sensor network localization. *SIAM J. Optim. 20,* 1, 192–215.

KIM, S., KOJIMA, M., AND WAKI, H. 2009b. User's manual for sfsdp: a sparse version of full semidefinite programming relaxation for sensor network localization problems. Tech. Rep. Research Report B-449, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan.

KOBAYASHI, K., KIM, S., AND KOJIMA, M. 2008. Correlative sparsity in primal-dual interior-point methods for lp, sdp and socp. *Appl. Math. Opt. 58,* 1, 69–88.

LEUNG, N.-H. Z. AND TOH, K.-C. 2009. An sdp-based divide-and-conquer algorithm for large scale noisy anchor-free graph realization. *SIAM J. Sci. Comput. 31*, 4351–4372.

LIAN, T.-C., WANG, T.-C., AND YE, Y. 2004. A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization. Tech. Rep. Technical report, Dept. of Management Science and Engineering, Stanford University.

NAKATA, K., FUJISAWA, K., FUKUDA, M., KOJIMA, M., AND MUROTA, K. 2003. Exploiting sparsity in semidefinite programming via matrix completion ii: Implementation and numerical results. *Math. Program. 95*, 303–327.

NIE, J. 2009. Sum of squares method for sensor network localization. *Comput. Opt. Appl. 43*, 151–179.

PONG, T. K. AND TSENG, P. 2010. (robust) edge-based semidefinite programming relaxation of sensor network localization. *To appear in Math. Program.*.

SDPA Homepage 2009. SDPA 7.3.1, http://sdpa.sourceforge.net/.

SeDuMi Homepage. http://sedumi.mcmaster.ca.

STRUM, J. F. 1999. Sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optim. Methods Soft. 11*, 625–653.

TSENG, P. 2007. Second order cone programming relaxation of sensor network localization. *SIAM J. Optim. 18*, 156–185.

TÜTÜNCÜ, R. H., TOH, K. C., AND TODD, M. J. 2003. Solving semidefinite-quadratic-linear programs using sdpt3. *Math. Program. 95*, 189–217.

WAKI, H., S. KIM, M. K., AND MURAMATSU, M. 2006. Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim 17*, 218–242.

WANG, Z., ZHENG, S., BOYD, S., AND YE, Y. 2008. Further relaxations of the sdp approach to sensor network localization. *SIAM J. Optim. 19,* 2, 655–673.