Exploiting Sparsity in SDP Relaxation
for Sensor Network Localization

Sunyoung Kim*, Masakazu Kojima†, and Hayato Waki‡
January 2008

Abstract.
A sensor network localization problem can be formulated as a quadratic optimization problem (QOP). For quadratic optimization problems, semidefinite programming (SDP) relaxation by Lasserre with relaxation order 1 for general polynomial optimization problems (POPs) is known to be equivalent to the sparse SDP relaxation by Waki *et al.* with relaxation order 1, except the size and sparsity of the resulting SDP relaxation problems. We show that the sparse SDP relaxation applied to the QOP is at least as strong as the Biswas-Ye SDP relaxation for the sensor network localization problem. A sparse variant of the Biswas-Ye SDP relaxation, which is equivalent to the original Biswas-Ye SDP relaxation, is also derived. Numerical results are compared with the Biswas-Ye SDP relaxation and the edge-based SDP relaxation by Wang *et al.*. We show that the proposed sparse SDP relaxation is faster than the Biswas-Ye SDP relaxation. In fact, the computational efficiency in solving the resulting SDP problems increases as the number of anchors and/or the radio range grow. The proposed sparse SDP relaxation also provides more accurate solutions than the edge-based SDP relaxation when exact distances are given between sensors and anchors and there are only a small number of anchors.

Key words. Sensor network localization problem, polynomial optimization problem, semidefinite relaxation, sparsity

* Department of Mathematics, Ewha W. University, 11-1 Dahyun-dong, Sudaemoon-gu, Seoul 120-750 Korea. The research was supported by Kosef R01-2005-000-10271-0 and KRF-2007-313-C00089.
`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 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 JSPS Fellows 18005736.
`waki9@is.titech.ac.jp`

1 Introduction

Sensor network localization problem arises in monitoring and controlling applications using wireless sensor networks such as inventory management and gathering environment data. Positioning sensors accurately in a wireless sensor network is an important problem for the efficiency of the applications where including GPS capability on every sensor in a network of inexpensive sensors is not an option. It is also closely related to distance geometry problems arising in predicting molecule structures and to graph rigidity.

The problem is to locate m sensors that fit the distances when a subset of distances and some sensors of known position (called anchors) are provided in a sensor network of n sensors, where $n > m$. Finding the solutions of this problem is a difficult problem. It is known NP-hard in general [20]. Various approaches thus have been proposed for the problem [1, 9, 10, 13, 14] to approximate the solutions.

Biswas and Ye [2] proposed a semidefinite programming (SDP) relaxation for the sensor network localization problem. A great deal of studies [3, 4, 5, 23, 28, 31] has followed in recent years. Compared with other methods for the problem, the SDP relaxation by [2] aimed to compute an accurate solution of the problem. Solving a large-scale SDP relaxation using software packages based on the primal-dual interior-point method [27, 32] is, however, known to be a computational challenge. As a result, the size of the sensor network localization problem that can be handled by the SDP relaxation is limited as mentioned in [3]. For the sensor network localization problem with a larger number of sensors, a distributed method in [3] 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 [9], and [28]. The solutions obtained by the SOCP relaxation are inaccurate compared to that by the SDP relaxation [28]. Edge-based SDP (ESDP) and node-based SDP (NSDP) relaxations were introduced in [31] to improve the computational efficiency of the original Biswas-Ye SDP relaxation in [2]. These SDP relaxations are weaker than the original SDP relaxation in theory, however, computational results show that the quality of the solution is comparable to that of the original Biswas-Ye SDP relaxation. It is also shown that much larger-sized problems can be handled.

In the sum-of-squares (SOS) method by Nie in [23], the sensor network localization problem was formulated as minimizing a polynomial with degree 4, and the solutions were found at the global minimizer. Sparsity of the polynomial objective function with degree 4 was utilized to reduce the size of the SOS relaxation. The advantage of this approach is that it provides highly accurate solutions. Numerical results for $n = 500$ in [23] showed that accurate solutions were found if exact distance information was given.

When solving a polynomial optimization problem (POP) as in [23], one of the deciding factors of the computational efficiency is the degree of the polynomials in the POP. The degree (and sparsity if exploited) decides the size of the SDP relaxation problem generated from the POP. Whether the global minimizer of a POP can be obtained computationally depends on the solvability of the SDP relaxation problem. It is thus imperative to have polynomials of lower degree in POPs to find the global minimizer of the POPs.

For general POPs, Lasserre [17] presented the hierarchical SDP relaxation whose convergence to a global minimizer is theoretically guaranteed. More accurate solutions can be computed if increasingly larger-sized SDP relaxations, whose size is decided by a positive number called the relaxation order, are solved. The size of POPs that can be solved by

Lasserre's SDP relaxation remains relatively small because the size of the SDP relaxation grows rapidly with the degree of the polynomials and the number of variables. A sparse SDP relaxation for POPs using the correlative sparsity of POPs was introduced to reduce the size of the SDP relaxation in [30]. We call Lasserre's relaxation the dense SDP relaxation as opposed to the sparse SDP relaxation in [30]. Although the theoretical convergence of the sparse SDP relaxation is shown in [18] for correlatively sparse POPs, the sparse SDP relaxation is theoretically weaker than the dense SDP relaxation in general.

For quadratic optimization problems (QOP), however, the sparse SDP relaxation with the relaxation order 1 and the dense SDP relaxation with the same relaxation order 1 are equivalent except the size and the sparsity of the resulting SDP relaxation problems as mentioned in Section 4.5 of [30]. Thus, the solution obtained using the sparse SDP relaxation is as accurate as the one by the dense SDP relaxation. Motivated by this observation, we study a QOP formulation for the sensor network localization problem. Our main objective is to improve the speed of solving the sensor network localization problem by exploiting the sparsity in the QOP formulation. The relationship between the Biswas-Ye SDP relaxation [2] and the dense SDP relaxation [17] of the proposed QOP formulation is examined. We show that the dense SDP relaxation with the relaxation order 1 (or the sparse SDP relaxation with the relaxation order 1) applied to the QOP formulation of the sensor network localization problem is at least as strong as the Biswas-Ye SDP relaxation in [2]. We also derive a sparse variant of the Biswas-Ye SDP relaxation and show that it is equivalent to the original Biswas-Ye SDP relaxation. We note that the dense SDP relaxation with the relaxation order 1 is a special case of the SDP relaxation by Shor [24, 25] for general quadratic optimization problems. See also the work of Fujie and Kojima in [11].

The QOP is solved numerically by SparsePOP [29], a matlab package for solving POPs using the correlative sparsity. A technique introduced in [16] is employed to improve the computational efficiency. Numerical results are compared with those with the Biswas-Ye SDP relaxation and the edge-based SDP relaxation in [31]. We show that (i) the proposed sparse SDP relaxation is faster than the Biswas-Ye SDP relaxation, (ii) less cpu time is required as the number of anchors and/or the radio range of sensors increase, (iii) more accurate solutions can be obtained using the sparse SDP relaxation than the edge-based SDP relaxation when exact distances are given between sensors and anchors and there are only a small number of anchors.

The sensor network localization problem is stated in detail in Section 2. A QOP formulation for the sensor network localization problem with exact and noisy distance measurements is presented. In Section 3, the dense and the sparse SDP relaxations are explained. A sparse variant of the Biswas-Ye SDP relaxation is described. We show that the dense SDP relaxation is at least as strong as the Biswas-Ye SDP relaxation. Additional techniques to reduce the size of the SDP relaxation, to refine solutions, and to choose the objective functions are shown in Section 4. Section 5 includes numerical results comparing with the results from the Biswas-Ye and the edge-based SDP relaxations. Section 6 contains concluding remarks and future directions.

2 Sensor network localization problems

Consider m sensors and m_a anchors, both located in the ℓ-dimensional Euclidean space $\mathbb{R}^\ell$, where ℓ is 2 or 3 in practice. Let $n = m + m_a$. The sensors are indexed with $p = 1, \ldots, m$ and the anchors with $r = m+1, \ldots, n$. We assume that the location $\boldsymbol{a}_r \in \mathbb{R}^\ell$ of anchor r is known for every $r = m+1, \ldots, n$, but the location $\boldsymbol{a}_p \in \mathbb{R}^\ell$ of sensor p is unknown for any $p = 1, \ldots, m$. We denote the exact distance $\|\boldsymbol{a}_p - \boldsymbol{a}_q\| > 0$ between sensors p and q by d_{pq} and the exact distance $\|\boldsymbol{a}_p - \boldsymbol{a}_r\| > 0$ between sensors p and r by d_{pr}. The exact values are not usually known in practice. Let $\mathcal{N}_x$ be a subset of $\{(p,q) : 1 \leq p < q \leq m\}$ (the set of pairs of sensors) and $\mathcal{N}_a$ be a subset of $\{(p,r) : 1 \leq p \leq m,\ m+1 \leq r \leq n\}$ (the set of pairs of sensors and anchors). Then, a sensor network localization problem is described as follows: Given distances (often containing noise) $\hat{d}_{pq} \approx d_{pq}$ between sensors p and q ($(p,q) \in \mathcal{N}_x$ and distances $\hat{d}_{pr} \approx d_{pr}$ between sensors p and r ($(p,r) \in \mathcal{N}_a$), compute or estimate locations $\boldsymbol{a}_p$ of sensor p ($p = 1, \ldots, m$). We consider the problem with exact distances in Section 2.1 and the problem with noisy distances in Section 2.2. Both problems are reduced to quadratic optimization problems (QOPs). The sparse SDP relaxation [30] with the relaxation order 1, which is equivalent to the dense SDP relaxation [18] as mentioned in Section 3.2, can be applied to the QOPs.

Naturally, we can represent a sensor network localization problem in terms of a geometrical network. Let $N = \{1, 2, \ldots, n\}$ denote the node set of sensors $p = 1, 2\ldots, m$ and anchors $r = m+1, m+2, \ldots, n$, and $\mathcal{N}_x \cup \mathcal{N}_a$ the set of undirected edges. We assume that all the nodes are located in the ℓ-dimensional space. To construct a geometrical network representation of the problem, we consider a graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ and add a positive number $\hat{d}_{pq}$ on each edge $(p,q) \in \mathcal{N}_x$ and a positive number $\hat{d}_{pr}$ on each edge $(p,r) \in \mathcal{N}_a$. Note that the node set N is partitioned into two subsets, the set of sensors $p = 1, 2, \ldots, m$ whose locations are to be approximated and the set of anchors $r = m+1, m+2, \ldots, n$ whose locations are known. Our main concern is to compute the locations of sensors with **accuracy** and **speed**. Thus, we focus on the methods of extracting a small-sized and sparse subgraph $G(N, E')$ from $G(N, E)$ in Section 4.1. We then replace the graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ by such a subgraph $G(N, E')$ in numerical computation. In this section, however, we formulate a sensor network localization problem with the graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ in a quadratic optimization problem (QOP).

2.1 Problems with exact distances

When all of the given distances $\hat{d}_{pq}$ ($(p,q) \in \mathcal{N}_x$) and $\hat{d}_{pr}$ ($(p,r) \in \mathcal{N}_a$) are exact, that is, $\hat{d}_{pq} = d_{pq}$ ($(p,q) \in \mathcal{N}_x$), $\hat{d}_{pr} = d_{pr}$ ($(p,r) \in \mathcal{N}_a$), the locations $\boldsymbol{x}_p = \boldsymbol{a}_p$ of sensors $p = 1, \ldots, m$ are characterized in terms of a system of nonlinear equations

$$d_{pq} = \|\boldsymbol{x}_p - \boldsymbol{x}_q\| \ (p,q) \in \mathcal{N}_x \ \text{ and } d_{pr} = \|\boldsymbol{x}_p - \boldsymbol{a}_r\| \ (p,r) \in \mathcal{N}_a.$$

To apply SDP relaxation, we transform this system into an equivalent system of quadratic equations

$$d_{pq}^2 = \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 \ (p,q) \in \mathcal{N}_x \ \text{ and } d_{pr}^2 = \|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 \ (p,r) \in \mathcal{N}_a. \quad (1)$$

In practice, a radio range $\rho > 0$ often determines $\mathcal{N}_x$ and $\mathcal{N}_a$.

$$\left.\begin{array}{rcl} \mathcal{N}_x &=& \{(p,q) : 1 \leq p < q \leq m,\ \|\boldsymbol{a}_p - \boldsymbol{a}_q\| \leq \rho\}, \\ \mathcal{N}_a &=& \{(p,r) : 1 \leq p \leq m,\ m+1 \leq r \leq n,\ \|\boldsymbol{a}_p - \boldsymbol{a}_r\| \leq \rho\}. \end{array}\right\} \tag{2}$$

If the radio range $\rho > 0$ is sufficiently large, the sets $\mathcal{N}_x$ and $\mathcal{N}_a$ coincide with the entire sets $\{(p,q) : 1 \leq p < q \leq m\}$ and $\{(p,r) : 1 \leq p \leq m,\ m+1 \leq r \leq n\}$, respectively. Decreasing the radio range $\rho > 0$ reduces the size of the sets $\mathcal{N}_x$ and $\mathcal{N}_a$ monotonically. For smaller size $\mathcal{N}_x$ and $\mathcal{N}_a$, the system of quadratic equations (1), which is rewritten as (3) below, is more likely to satisfy a structured sparsity called the correlative sparsity in the literature [15, 30]. This sparsity can be utilized to increase the effectiveness of the sparse SDP relaxation [30] when applied to the POPs (4) and (6).

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

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

Here, X_{ip} denotes the (i,p)th element of the matrix $\boldsymbol{X}$ or the ith element of $\boldsymbol{x}_p$. We call (3) *a system of sensor network localization equations*, and a matrix variable or a solution $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{\ell \times m}$ of the system (3) *a sensor location matrix*. By introducing an objective function that is identically zero, QOP for the sensor network localization without noise is obtained:

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

Let L_0 denote the set of solutions $\boldsymbol{X} \in \mathbb{R}^{\ell \times m}$ of the system of sensor network localization equations (3) (or the POP (4)).

2.2 Problems with noisy distances

When the given distances $\hat{d}_{pq} > 0$ $((p,q) \in \mathcal{N}_x)$ and $\hat{d}_{pr} > 0$ $((p,r) \in \mathcal{N}_a)$ contain noise, the system of sensor network localization equations (3) with $d_{pq} = \hat{d}_{pq}$ $((p,q) \in \mathcal{N}_x)$ and $d_{pq} = \hat{d}_{pr}$ $((p,r) \in \mathcal{N}_a)$ may not be feasible. In such a case, we can estimate the locations of sensors with a least square solution $\boldsymbol{X}$ of (3), *i.e.*, an optimal solution of the problem

$$\text{minimize} \quad \sum_{(p,q) \in \mathcal{N}_x} (\hat{d}_{pq}^2 - \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2)^2 + \sum_{(p,r) \in \mathcal{N}_a} (\hat{d}_{pr}^2 - \|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2)^2 \tag{5}$$

Notice that this is an unconstrained POP. Nie [23] applied the SDP relaxation [17] to the POP of this form. He also proposed a sparse SDP relaxation exploiting a special structure of the POP (5), and reported some numerical results. His sparse SDP relaxation possesses a nice theoretical property that if the system of sensor network localization equations (3) with $d_{pq} = \hat{d}_{pq}$ $((p,q) \in \mathcal{N}_x)$ and $d_{pr} = \hat{d}_{pr}$ $((p,r) \in \mathcal{N}_a)$ is feasible or if the POP (5) attains the optimal value 0, then so dose its sparse SDP relaxation. As a result, the sparse SDP relaxation is exact. A disadvantage of this formulation (5) lies in the high degree of the polynomial objective function, degree 4, in the unconstrained POP (5). Note that degree 4 is twice the degree of polynomials in the system of quadratic equations (3). This increases the size of the sparse SDP relaxation of the unconstrained POP (5).

We can reformulate the POP (5) as a quadratic optimization problem (QOP)

$$\left.\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(p,q)\in\mathcal{N}_x} \xi_{pq}^2 + \sum_{(p,r)\in\mathcal{N}_a} \xi_{pr}^2 \\ \text{subject to} & \displaystyle\hat{d}_{pq}^2 = \sum_{i=1}^{\ell} X_{ip}^2 - 2\sum_{i=1}^{\ell} X_{ip}X_{iq} + \sum_{i=1}^{\ell} X_{iq}^2 + \xi_{pq} \quad (p,q)\in\mathcal{N}_x, \\ & \displaystyle\hat{d}_{pr}^2 = \sum_{i=1}^{\ell} X_{ip}^2 - 2\sum_{i=1}^{\ell} X_{ip}a_{ir} + \|\boldsymbol{a}_r\|^2 + \xi_{pr} \quad (r,q)\in\mathcal{N}_a, \end{array}\right\} \quad (6)$$

where ξ_{pq} denotes an error variable. Now, the polynomials in the problem (6) are of degree 2, a half of the degree of the objective polynomial of the POP (5), which makes the size of the resulting dense SDP relaxation [17] smaller. In addition, the sparse SDP relaxation [30] with the relaxation order 1 is equivalent to the dense SDP relaxation with the same order.

Further discussions on the problems with noisy distances and a new QOP formulation are included in Section 4.5.

3 SDP relaxations

We describe SDP relaxations for the POP (4) derived from sensor network localization with exact distances. The SDP relaxation described for the QOP (4) in Section 3.1 is a special case of the SDP relaxation proposed by Lasserre [17] (the dense SDP relaxation) for general POPs in the sense that the relaxation order is fixed to 1 for the QOP (4). We also mention that the dense SDP relaxation described there is essentially a classical SDP relaxation proposed by Shor [24, 25] for QOPs. Instead of just referring to [11, 17, 24, 25], we describe the dense SDP relaxation in detail to compare with the Biswas-Ye SDP relaxation [2, 26] of the POP (4) in Section 3.2. In Section 3.3, we discuss sparse variants of the dense SDP relaxation given in Section 3.1 and the Biswas-Ye SDP relaxation [2]. Most of the discussions here are valid for the POP (6) for sensor network localization with noisy distances, but the details are omitted.

The following symbols are used to describe the dense and sparse SDP relaxations. Let

$$\begin{array}{rcl} \mathcal{I} & = & \{ip : 1 \leq i \leq \ell,\ 1 \leq p \leq m\}, \\ & & \text{the set of subscripts of the matrix variable } \boldsymbol{X}. \\ \#\mathcal{C} & = & \text{the number of elements in } \mathcal{C}\ (\mathcal{C} \subseteq \mathcal{I}). \end{array} \quad (7)$$

For every sensor location matrix variable $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{\ell\times m}$ and $\mathcal{C} \subseteq \mathcal{I}$, define

$(X_{ip} : ip \in \mathcal{C})$ = the row vector variable consisting of X_{ip} $(ip \in \mathcal{C})$, where the elements are arranged according to the lexicographical order of the subscripts $ip \in \mathcal{C}$; for example if $\mathcal{C} = \{11, 12, 21, 22\}$, then $(X_{ip} : ip \in \mathcal{C}) = (X_{11}, X_{12}, X_{21}, X_{22})$, and

$$(X_{ip} : ip \in \mathcal{C})^T (X_{ip} : ip \in \mathcal{C}) \;\; = \;\; \sum_{ip \in \mathcal{C}} \sum_{jq \in \mathcal{C}} \boldsymbol{E}(\mathcal{C})_{ipjq} X_{ip} X_{jq}, \tag{8}$$

where $\boldsymbol{E}(\mathcal{C})_{ipjq}$ denotes the $\#\mathcal{C} \times \#\mathcal{C}$ matrix whose (ip, jq)th element is 1 and all others 0. Specifically, we write $\boldsymbol{E}_{ipjq} = \boldsymbol{E}(\mathcal{I})_{ipjq}$ $((ip, jq) \in \mathcal{I} \times \mathcal{I})$;

$$(X_{ip} : ip \in \mathcal{I})^T (X_{ip} : ip \in \mathcal{I}) = \sum_{ip \in \mathcal{I}} \sum_{jq \in \mathcal{I}} \boldsymbol{E}_{ipjq} X_{ip} X_{jq}. \tag{9}$$

If $\mathcal{C} \subseteq \mathcal{I}$ and $ip, \; jq \in \mathcal{C}$, then each $\boldsymbol{E}(\mathcal{C})_{ipjq}$ forms a submatrix of $\boldsymbol{E}_{ipjq}$. Hence, the matrix on the right-hand side of (8) is a submatrix of that of (9). We also note that

$$\boldsymbol{E}(\mathcal{C})_{jqip} = \boldsymbol{E}(\mathcal{C})^T_{ipjq} \;\; (ip, \; jq \in \mathcal{C}),$$

as a result, the matrices in (8) and (9) are symmetric.

Replacing each $X_{ip} X_{jq}$ by a single variable U_{ipjq} in (9), we define a $\#\mathcal{I} \times \#\mathcal{I}$ matrix variable $\boldsymbol{U} = \sum_{ip \in \mathcal{I}} \sum_{jq \in \mathcal{I}} \boldsymbol{E}_{ipjq} U_{ipjq}$. From the identity $X_{ip} X_{jq} = X_{jq} X_{ip}$, $(ip, \; jq \in \mathcal{I})$, we impose $U_{ipjq} = U_{jqip}$, or that the matrix $\boldsymbol{U}$ is symmetric. The matrix variable $\boldsymbol{U}$ serves as a linearization of the polynomial (quadratic) matrix $(X_{ip} : ip \in \mathcal{I})(X_{ip} : ip \in \mathcal{I})^T$. We also use the notation $\boldsymbol{U}(\mathcal{C})$ for the submatrix variable of $\boldsymbol{U}$ consisting of elements U_{ipjq} $(ip, \; jq \in \mathcal{C})$: $\boldsymbol{U}(\mathcal{C}) = \sum_{ip \in \mathcal{C}} \sum_{jq \in \mathcal{C}} \boldsymbol{E}(\mathcal{C})_{ipjq} U_{ipjq}$.

3.1 The dense SDP relaxation

A polynomial (quadratic) matrix inequality

$$\boldsymbol{O} \preceq \begin{pmatrix} 1 & (X_{ip} : (ip \in \mathcal{I})) \\ (X_{ip} : (ip \in \mathcal{I}))^T & (X_{ip} : (ip \in \mathcal{I}))^T (X_{ip} : (ip \in \mathcal{I})) \end{pmatrix} \in \mathcal{S}^{\#\mathcal{I}+1} \tag{10}$$

is added to the system of sensor network localization equations (3) to derive the dense SDP relaxation. Since (10) holds for any sensor location matrix $\boldsymbol{X} \in \mathbb{R}^{\ell \times m}$, the solution set L_0 of (3) or the set of feasible solutions the POP (4) remains the same. Using (9), we rewrite the polynomial matrix inequality (10) as

$$\boldsymbol{O} \preceq \begin{pmatrix} 1 & (X_{ip} : (ip \in \mathcal{I})) \\ (X_{ip} : (ip \in \mathcal{I}))^T & \displaystyle\sum_{ip \in \mathcal{I}} \sum_{jq \in \mathcal{I}} \boldsymbol{E}_{ipjq} X_{ip} X_{jq} \end{pmatrix} \in \mathcal{S}^{\#\mathcal{I}+1}. \tag{11}$$

Now we linearize (3) and (11) by replacing every $X_{ip} X_{jq}$ with a single variable U_{ipjq} $(ip, \; jq \in \mathcal{I})$ and obtain SDP relaxations of the system (3) and the POP (4):

$$\left. \begin{array}{ll} d^2_{pq} = \displaystyle\sum_{i=1}^{\ell} U_{ipip} - 2 \sum_{i=1}^{\ell} U_{ipiq} + \sum_{i=1}^{\ell} U_{iqiq} & (p, q) \in \mathcal{N}_x, \\ d^2_{pr} = \displaystyle\sum_{i=1}^{\ell} U_{ipip} - 2 \sum_{i=1}^{\ell} X_{ip} a_{ir} + \|\boldsymbol{a}_r\|^2 & (p, r) \in \mathcal{N}_a, \\ \boldsymbol{O} \preceq \begin{pmatrix} 1 & (X_{ip} : (ip \in \mathcal{I})) \\ (X_{ip} : (ip \in \mathcal{I}))^T & \boldsymbol{U} \end{pmatrix}. & \end{array} \right\} \tag{12}$$

and

$$\text{minimize } 0 \text{ subject to the constraints (12).} \tag{13}$$

Let

$$L_{1d} = \left\{ \boldsymbol{X} \in \mathbb{R}^{\ell \times m} : \begin{array}{l} (\boldsymbol{X}, \boldsymbol{U}) \text{ is a solution of (12)} \\ \text{for some } \boldsymbol{U} \in \boldsymbol{S}^{\ell m \times \ell m} \end{array} \right\}$$

Proposition 3.1. $L_0 \subseteq L_{1d}$.

Proof: Let $\boldsymbol{X} \in L_0$. Then, $\boldsymbol{X}$ satisfies (3) and (10). Let $\boldsymbol{U}$ be $\#\mathcal{I} \times \#\mathcal{I}$ symmetric matrix whose components are given by $U_{ipjq} = X_{ip}X_{jq}$ $(ip,\ jq \in \mathcal{I})$. Then, $(\boldsymbol{X}, \boldsymbol{U})$ satisfies (12). Therefore, $\boldsymbol{X} \in L_{1d}$. ∎

3.2 Comparison of the dense SDP relaxation with the Biswas-Ye SDP relaxation

The Biswas-Ye SDP relaxation [2] of the system of sensor network localization equations (3) is of the form

$$\left. \begin{array}{l} d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \quad (p, q) \in \mathcal{N}_x, \\ d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\sum_{i=1}^{\ell} X_{ip}a_{ir} + Y_{pp} \ (p, r) \in \mathcal{N}_a, \\ \boldsymbol{O} \preceq \begin{pmatrix} \boldsymbol{I}_\ell & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix}. \end{array} \right\} \tag{14}$$

Here $\boldsymbol{I}_\ell$ denotes the $\ell \times \ell$ identity matrix and

$$\boldsymbol{Y} = \begin{pmatrix} Y_{11} & \dots & Y_{1q} & \dots & Y_{1m} \\ \vdots & & \vdots & & \vdots \\ Y_{p1} & \dots & Y_{pq} & \dots & Y_{pm} \\ \vdots & & \vdots & & \vdots \\ Y_{m1} & \dots & Y_{mq} & \dots & Y_{mm} \end{pmatrix} \in \boldsymbol{S}^m \tag{15}$$

a matrix variable. Let

$$L_{2d} = \left\{ \boldsymbol{X} \in \mathbb{R}^{\ell \times m} : \begin{array}{l} (\boldsymbol{X}, \boldsymbol{Y}) \text{ is a solution of (14)} \\ \text{for some } \boldsymbol{Y} \in \boldsymbol{S}^{m \times m} \end{array} \right\}$$

Proposition 3.2. $L_{1d} \subseteq L_{2d}$.

Proof: Suppose that $\boldsymbol{X} \in L_{1d}$. Then, there exists a $\boldsymbol{U} \in \mathcal{S}^{\#\mathcal{I}}$ such that $(\boldsymbol{X}, \boldsymbol{U})$ satisfies (12). Let $\mathcal{C}_i = \{ip : 1 \leq p \leq m\}$ $(1 \leq i \leq \ell)$. Then, $\#\mathcal{C}_i = m$. Define an $m \times m$ symmetric matrix $\boldsymbol{Y}$ by $\boldsymbol{Y} = \sum_{i=1}^{\ell} \boldsymbol{U}(\mathcal{C}_i)$ or $Y_{pq} = \sum_{i=1}^{\ell} U_{ipiq}$ $(1 \leq p \leq m,\ 1 \leq q \leq m)$. We show that $(\boldsymbol{X}, \boldsymbol{Y})$ satisfies (14); then $\boldsymbol{X} \in L_{2d}$ follows. By definition, we observe that

$$Y_{pp} = \sum_{i=1}^{\ell} U_{ipip},\ Y_{pq} = \sum_{i=1}^{\ell} U_{ipiq} \quad \text{and} \quad Y_{qq} = \sum_{i=1}^{\ell} U_{iqiq}$$

for every $p = 1, \ldots, m$ and $q = 1, \ldots, m$. Thus, the first two relations of (14) follow from the first two relations of (12), respectively. Now, we consider the matrices

$$\begin{pmatrix} 1 & (X_{ip} : (i,p) \in \mathcal{C}_i) \\ (X_{ip} : (i,p) \in \mathcal{C}_i)^T & \boldsymbol{U}(\mathcal{C}_i) \end{pmatrix} \quad (1 \leq i \leq m). \tag{16}$$

Note that the matrices in (16) are positive semidefinite because they are submatrices of the positive semidefinite matrix

$$\begin{pmatrix} 1 & (X_{ip} : (ip \in \mathcal{I})) \\ (X_{ip} : (ip \in \mathcal{I}))^T & \boldsymbol{U} \end{pmatrix}$$

in (12). The positive semidefiniteness of the matrices in (16) implies that

$$\boldsymbol{O} \preceq \boldsymbol{U}(\mathcal{C}_i) - (X_{ip} : (i,p) \in \mathcal{C}_i)(X_{ip} : (i,p) \in \mathcal{C}_i)^T \quad (1 \leq i \leq \ell).$$

As a result,

$$\boldsymbol{O} \preceq \sum_{i=1}^{\ell} \left(\boldsymbol{U}(\mathcal{C}_i) - (X_{ip} : (i,p) \in \mathcal{C}_i)(X_{ip} : (i,p) \in \mathcal{C}_i)^T \right) = \boldsymbol{Y} - \boldsymbol{X}^T \boldsymbol{X}.$$

Finally, the relation $\boldsymbol{O} \preceq \boldsymbol{Y} - \boldsymbol{X}^T \boldsymbol{X}$ is equivalent to the last relation of (14). ∎

From the proof above, we can say that the Biswas-Ye SDP relaxation is an aggregation of the dense SDP relaxation described in Section 3.1. Propositions 3.2 makes it possible to apply some of the results on the Biswas-Ye SDP relaxation in [2] to the dense SDP relaxation given in Section 3.1. Among many results, it is worthy to mention that if the system of sensor network localization equations (3) is uniquely localizable, then $L_0 = L_{1d} = L_{2d}$. See Definition 1 and Theorem 2 of [26].

3.3 Exploiting sparsity in the SDP relaxation problems (13) and (14)

It is convenient to introduce an undirected graph $G(V, \mathcal{N}_x)$ associated with the sensor network localization problem to discuss sparsity exploitation in its SDP relaxations, where $V = \{1, \ldots, m\}$ denotes the set of sensors and $\mathcal{N}_x$ indicates the set $\{\{p, q\} : (p, q) \in \mathcal{N}_x\}$ of undirected edges of the graph. Let $G(V, \overline{E})$ be a chordal extension of $G(V, E)$, and $C_1, \ldots, C_k$ be the family of all maximal cliques of $G(V, \overline{E})$. A graph is called chordal if every cycle of length ≥ 4 has a chord (an edge joining two nonconsecutive vertices of the cycle). For the definition and basic properties of chordal graphs, we refer to [6]. We note that $k \leq m$ since $G(V, \overline{E})$ is chordal. We assume that $G(V, \overline{E})$ is sparse or that the size of each maximal clique of $G(V, \overline{E})$ is small. When the set $\mathcal{N}_x$ is determined by (2) for a small radio range $\rho > 0$, this assumption is expected to hold.

It should be noted that $G(V, \mathcal{N}_x)$ is obtained as a subgraph of the graph $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$, which has been introduced as a geometrical representation of a sensor network localization problem, by eliminating all anchor nodes $\{m+1, m+2, \ldots, n\}$ from N and all edges in $\mathcal{N}_a$. This means that anchors are not relevant at all to the discussion in this section on exploiting sparsity in the SDP relaxation problems. However, the edges in $\mathcal{N}_a$ play a crucial role in extracting a smaller and sparser subgraph $G(V, \mathcal{N}_x \cap \overline{E})$ of $G(V, \mathcal{N}_x)$, to which the method of this section can be applied for some $\overline{E} \subseteq \mathcal{N}_x \cup \mathcal{N}_a$. The main purpose of extracting a subgraph is that the sensor network localization problem with the reduced graph $G(N, \overline{E})$ can be solved more efficiently, resulting in highly accurate approximations of the locations of sensors. This will be discussed in Section 4.1.

A sparse SDP relaxation problem of the QOP (4) can be derived in two different ways. The one is an application of the sparse SDP relaxation by Waki, Kim, Kojima and Muramatsu [30] for solving a general sparse POPs to the QOP (4), and the other is an application of the positive definite matrix completion based on [12, 21] to the dense SDP relaxation problem (13). The correlative sparsity in POPs is an essential property in the former approach, while the aggregated sparsity plays a crucial role in the latter.

We derive a sparse relaxation simultaneously using the positive definite matrix completion from the dense SDP relaxation problems (13) and (14). The derivation of a sparse SDP relaxation problem of the QOP (4) via the positive definite matrix completion has not dealt previously. It is simple to describe, more importantly, it is consistent with the derivation of a sparse counterpart of the Biswas-Ye SDP relaxation problem (14).

First, we rewite the SDPs (13) and (14) in an equality standard primal SDP of the form

$$\text{minimize } \boldsymbol{A}_0 \bullet \boldsymbol{Z} \text{ subject to } \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t \ (t \in T) \ \boldsymbol{Z} \succeq \boldsymbol{O}, \tag{17}$$

where T denotes a finite index set. For (13), we define

$$\boldsymbol{Z} = \begin{pmatrix} X_{00} & (X_{ip} : (ip \in \mathcal{I})) \\ (X_{ip} : (ip \in \mathcal{I}))^T & \boldsymbol{U} \end{pmatrix}, \tag{18}$$

and for (14),

$$\boldsymbol{Z} = \begin{pmatrix} \boldsymbol{W} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{Y} \end{pmatrix}. \tag{19}$$

After the equality standard form SDP is obtained, the positive definite matrix completion method [12], or the conversion method using positive definite matrix completion [21] can be applied. Specifically, the application of the conversion method to (17) leads to a sparse SDP relaxation problem of (13) and a sparse variant of the Biswas-Ye SDP relaxation problem (14). The sparse SDP relaxation derived here is essentially the same as the sparse SDP relaxation [30] with the relaxation order 1 for the QOP (4). It should be emphasized that the resulting SDP problems are equivalent to the dense relaxation problems (13) and (14). We explain the derivation in detail below.

Let $\mathcal{V}$ denote the index set of rows (and columns) of the matrix variable $\boldsymbol{Z}$. We assume that the rows and columns of the matrix $\boldsymbol{Z}$ in (18) are indexed by 00 and ip $(i = 1, \ldots, \ell,\ p = 1, \ldots, m)$ in the lexicographical order and those of the matrix $\boldsymbol{Z}$ in (19) by $01, \ldots, 0\ell, *1, \ldots, *m$. Hence, $\mathcal{V} = \{00,\ ip\ (i = 1, \ldots, \ell,\ p = 1, \ldots, m)\}$ for (13), and $\mathcal{V} = \{10, \ldots, \ell 0, *1, \ldots, *m\}$ for (14), where $*$ denotes a fixed symbol or integer larger than ℓ so that each element of $\boldsymbol{A}_t$ can be written as $[\boldsymbol{A}_t]_{ipjq}$.

As in [12], we introduce the *aggregated sparsity pattern* $\mathcal{E}$ of the data matrices

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

where $[\boldsymbol{A}_t]_{ipjq}$ denotes the (ip, jq)th element of the matrix $\boldsymbol{A}_t$. The aggregated sparsity pattern can be represented geometrically with a graph $G(\mathcal{V}, \mathcal{E})$. Note that the edge set of the graph $\{\{ip, jq\} \in \mathcal{E} : ip \text{ is lexicographically smaller than } jq\}$ has been identified as $\mathcal{E}$ itself.

Now, construct a chordal extension $G(\mathcal{V}, \overline{\mathcal{E}})$ of $G(\mathcal{V}, \mathcal{E})$ by simulating the chordal extension from $G(V, E)$ to $G(V, \overline{E})$. For (13), define

$$\begin{aligned}
\overline{\mathcal{E}} &= \{\{00, ip\} \ (1 \leq i \leq \ell, \ 1 \leq p \leq m)\} \\
&\quad \cup \{\{ip, jq\} \ ((p, q) \in \overline{E}, \ 1 \leq i \leq \ell, \ 1 \leq j \leq \ell)\} \\
&\quad \cup \{\{ip, jp\} \ (1 \leq p \leq m, \ 1 \leq i < j \leq \ell)\}, \\
\overline{\mathcal{C}}_h &= \{00, \ 1p, \ldots, \ell p \ (p \in C_h)\} \ (1 \leq h \leq k),
\end{aligned}$$

and for (14),

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

Then, we can verify that $G(\mathcal{V}, \overline{\mathcal{E}})$ forms a chordal extension of $G(\mathcal{V}, \mathcal{E})$, and that $\overline{\mathcal{C}}_1, \ldots, \overline{\mathcal{C}}_k$ are its maximal cliques in both cases.

If we apply the conversion method [21] to (17) using the information on the chordal graph $G(\mathcal{V}, \overline{\mathcal{E}})$ and its maximal cliques $\overline{\mathcal{C}}_1, \ldots, \overline{\mathcal{C}}_k$, then we obtain an SDP problem

$$\left.\begin{array}{ll}
\text{minimize} & \boldsymbol{A}_0 \bullet \boldsymbol{Z} \\
\text{subject to} & \boldsymbol{A}_t \bullet \boldsymbol{Z} = b_t \ (t \in T), \ \boldsymbol{Z}_{\overline{\mathcal{C}}_h, \overline{\mathcal{C}}_h} \succeq \boldsymbol{O} \ (1 \leq h \leq k).
\end{array}\right\} \tag{20}$$

Here $\boldsymbol{Z}_{\overline{\mathcal{C}}_i, \overline{\mathcal{C}}_h}$ denotes a submatrix of $\boldsymbol{Z}$ consisting of the elements $\boldsymbol{Z}_{ipqj}$ $((i, p) \in \overline{\mathcal{C}}_h, \ (j, p) \in \overline{\mathcal{C}}_h)$. If the size of every $\overline{\mathcal{C}}_h$ is small, we can solve the SDP (20) more efficiently than the original SDP (17)

For (13), the SDP (20) is rewriiten as

$$\left.\begin{array}{ll}
\text{minimize} & 0 \\
& d_{pq}^2 = \displaystyle\sum_{i=1}^{\ell} U_{ipip} - 2\sum_{i=1}^{\ell} U_{ipiq} + \sum_{i=1}^{\ell} U_{iqiq} \ \ (p, q) \in \mathcal{N}_x, \\
& d_{pr}^2 = \displaystyle\sum_{i=1}^{\ell} U_{ipip} - 2\sum_{i=1}^{\ell} X_{ip} a_{ir} + \|\boldsymbol{a}_r\|^2 \ \ (p, r) \in \mathcal{N}_a, \\
& \boldsymbol{O} \preceq \begin{pmatrix} 1 & (X_{ip} : (ip \in \widetilde{\mathcal{C}}_h)) \\ (X_{ip} : (ip \in \widetilde{\mathcal{C}}_h))^T & \boldsymbol{U}(\widetilde{\mathcal{C}}_h) \end{pmatrix} \ (1 \leq h \leq k),
\end{array}\right\} \tag{21}$$

where $\widetilde{\mathcal{C}}_h = \overline{\mathcal{C}}_h \backslash \{00\} = \{1p, \ldots, \ell p \ (p \in C_h)\}$. For (14), the (20) is rewritten as

$$\left.\begin{array}{ll}
\text{minimize} & 0 \\
& d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \ \ (p, q) \in \mathcal{N}_x, \\
& d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\displaystyle\sum_{i=1}^{\ell} X_{ip} a_{ir} + Y_{pp} \ (p, r) \in \mathcal{N}_a, \\
& \boldsymbol{O} \preceq \begin{pmatrix} \boldsymbol{I}_\ell & (\boldsymbol{x}_p : p \in C_h) \\ (\boldsymbol{x}_p : p \in C_h)^T & \boldsymbol{Y}_{C_h, C_h} \end{pmatrix} \ (1 \leq h \leq k),
\end{array}\right\} \tag{22}$$

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

$$
\begin{array}{rcl}
L_{1s} & = & \left\{ \boldsymbol{X} \in \mathbb{R}^{\ell \times m} : \begin{array}{l} (\boldsymbol{X}, \boldsymbol{U}) \text{ is a solution of (21)} \\ \text{for some } \boldsymbol{U} \in \boldsymbol{S}^{\ell m \times \ell m} \end{array} \right\}, \\
L_{2s} & = & \left\{ \boldsymbol{X} \in \mathbb{R}^{\ell \times m} : \begin{array}{l} (\boldsymbol{X}, \boldsymbol{Y}) \text{ is a solution of (22)} \\ \text{for some } \boldsymbol{Y} \in \boldsymbol{S}^{m \times m} \end{array} \right\}.
\end{array}
$$

Proposition 3.3. $L_{1s} = L_{1d} \subseteq L_{2s} = L_{2d}$

Proof: By Proposition 3.2, we know that $L_{1d} \subseteq L_{2d}$. The equivalence of the SDPs (17) and (20) is established in [12]. Hence $L_{1s} = L_{1d}$ and $L_{2s} = L_{2d}$ follow. ∎

We briefly mention a generalization of the sparse variant (22) of the Biswas-Ye SDP relaxation, which includes the node-based and edge-based SDP relaxations of sensor network localization problems proposed in [31]. They are regarded as further relaxations of the Biswas-Ye SDP relaxation. We call them NSDP and ESDP, respectively, as in [31]. Our sparse SDP relaxation described above is compared with ESDP with numerical results in Section 5.

Let Γ be a family of nonempty subsets of the set $\{1, \ldots, m\}$ of sensors. Then, a relaxation of the Biswas-Ye SDP relaxation (14) is

$$
\left.
\begin{array}{ll}
\text{minimize} & 0 \\
& d_{pq}^2 = Y_{pp} + Y_{qq} - 2Y_{pq} \quad (p,q) \in \mathcal{N}_x, \\
& d_{pr}^2 = \|\boldsymbol{a}_r\|^2 - 2\sum_{i=1}^{\ell} X_{ip} a_{ir} + Y_{pp} \ (p,r) \in \mathcal{N}_a, \\
& \boldsymbol{O} \preceq \begin{pmatrix} \boldsymbol{I}_\ell & \boldsymbol{X}(C) \\ \boldsymbol{X}(C)^T & \boldsymbol{Y}(C) \end{pmatrix} \quad (C \in \Gamma).
\end{array}
\right\} \tag{23}
$$

Obviously, if we take the family of maximum cliques $C_1, \ldots, C_k$ of the chordal extension $G(V, \overline{E})$ of the aggregated sparsity pattern graph $G(V, \overline{E})$ for Γ, the SDP (23) coincides with the SDP (22). On the other hand, if $\Gamma = \{\{q \in V : (p,q) \in \mathcal{N}_x\} : p \in V\}$ in the SDP (23), it becomes the NSDP relaxation, and if $\Gamma = \{\{p,q\} : (p,q) \in \mathcal{N}_x\}$, the ESDP relaxation is obtained. In case of the ESDP relaxation, each member of the family Γ consists of two elements, and the matrix in the positive semidefinite condition of (23) is $(\ell+2) \times (\ell+2)$. Let L_n and L_e denote the solution sets of the NSDP and ESDP relaxation, respectively. By construction, we know that $L_{2d} \subseteq L_n \subseteq L_e$. It was shown in [31] that if the underlying graph $G(V, \mathcal{E})$ is chordal, then $L_{2d} = L_n$. In this case, we know that $G(V, \mathcal{E}) = G(V, \overline{\mathcal{E}})$ and that $L_{2d} = L_{2s} = L_n$ also follows from Proposition 3.3.

4 Additional techniques

We present two methods for reducing the size of the sparse SDP relaxation problem (21) to increase the computational efficiency. In the method described in Section 4.1, the size of the system of sensor network localization equations (3) is reduced before applying the sparse SDP relaxation. The method in Section 4.2 is to decrease the size of the SDP relaxation problem by eliminating free variables [16]. We also address the issues of strengthening the sparse SDP relaxation by adding inequalities for the search region, refining solutions of the SDP relaxation using a nonlinear least square method, and modifying the objective function of the QOP (6) in Sections 4.3, 4.4 and 4.5, respectively. Except for the last section Section 4.5, only the sparse SDP relaxation (21) of the QOP (4) from the sensor network localization problem with exact distances is considered. All the discussions there, however, can be applied to the sparse SDP relaxation of the POP(6) for the sensor network localization problem with noisy distance.

4.1 Reducing the size of the system of sensor network localization equations (3)

Recall that $G(N, \mathcal{N}_x \cup \mathcal{N}_a)$ denotes 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 $\deg(p, E')$ denote 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 necessary (but not sufficient) to determine their locations when they are located in generic positions. Therefore, 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 $\min\{\deg(p, \mathcal{N}_x \cup \mathcal{N}_a), \kappa\}$ for every sensor node p, where κ is a positive integer not less than $\ell + 1$. We choose a minimal subgraph $G(N, \overline{E})$ from the family $\mathcal{G}_\kappa$, and replace $\mathcal{N}_x$ and $\mathcal{N}_a$ by $\mathcal{N}_x \cap \overline{E}$ and $\mathcal{N}_a \cap \overline{E}$, respectively, in the system of sensor network localization equations (3).

When choosing edges from $\mathcal{N}_x$ and $\mathcal{N}_a$ for a reduced edge set $\overline{E}$, we give the priority to the edges in $\mathcal{N}_a$ over ones in $\mathcal{N}_x$. More precisely, for every sensor $p = 1, 2, \ldots, m$, we first choose at most $s + 1$ edges $(p, r) \in \mathcal{N}_a$, and then choose edges from $\mathcal{N}_x$ to produce a minimal subgraph $G(N, \overline{E})$ satisfying the desired property. As $\mathcal{N}_a$ involves more edges, the resulting subgraph $G(V, \mathcal{N}_x \cap \overline{E})$ becomes sparser and small. Thus, the sparse SDP relaxation in Section 3.3 for the reduced problem with the graph $G(V, \mathcal{N}_x \cap \overline{E})$ can be solved more efficiently. This will be confirmed through numerical results in Section 5.

With an increasingly larger value for κ, we can expect to obtain more accurate locations of the sensors though it takes longer to solve the sparse SDP relaxation. In the numerical experiments in Section 5, we took $\kappa = \ell + 2$.

A different way of reducing the size of the system of sensor network localization equations (3) was proposed by Wang *et al.* [31]. They restricted the degree of each sensor node to a small positive integer λ. Let $\mathcal{G}^\lambda$ denote the family of subgraphs $G(N, E')$ of $G(N, E)$ such that $\deg(p, E') \leq \lambda$ for every sensor p. It was suggested to take $\lambda \in \{7, 8, 9, 10\}$ in their MATLAB program ESDP of an SDP relaxation method proposed for the 2-dimensional sensor network localization problem in [31].

We tested these two methods, the one choosing a minimal subgraph from the family $\mathcal{G}_\kappa$, and the other choosing a maximal subgraph from the family $\mathcal{G}^\lambda$, and found that the former method is more effective when it is combined with the sparse SDP relaxation (21) described in Section 3.3 and also with the Biswas-Ye SDP relaxation (14) described in Section 3.2.

4.2 Reducing the size of the SDP relaxation problem (21)

We rewrite the sparse SDP relaxation problem (21) as a dual form SDP with equalities

$$\left.\begin{array}{ll} \text{maximize} & \sum_{i=1}^{k} b_i y_i \\ \text{subject to} & \boldsymbol{a}_0 - \sum_{i=1}^{k} \boldsymbol{a}_i y_i = \boldsymbol{0},\ \boldsymbol{A}_0 - \sum_{i=1}^{k} \boldsymbol{A}_i y_i \succeq \boldsymbol{O}. \end{array}\right\} \tag{24}$$

for some vectors $\boldsymbol{a}_i$ $(i = 0, 1, \ldots, k)$, symmetric matrices $\boldsymbol{A}_i$ $(i = 0, 1, \ldots, k)$ and real numbers $b_i = 0$ $(i = 1, \ldots, k)$. The corresponding primal SDP is of the form

$$\left.\begin{array}{ll} \text{minimize} & \boldsymbol{a}_0^T \boldsymbol{z} + \boldsymbol{A}_0 \bullet \boldsymbol{Z} \\ \text{subject to} & \boldsymbol{a}_i^T \boldsymbol{z} + \boldsymbol{A}_i \bullet \boldsymbol{Z} = b_i\ (i = 1, \ldots, k),\ \boldsymbol{Z} \succeq \boldsymbol{O}, \end{array}\right\} \tag{25}$$

which contains free vector variable $\boldsymbol{z}$. In [16], Kobayashi, Nakata and Kojima proposed a method to reduce the size of the primal-dual pair SDPs of this form by eliminating the free variable vector $\boldsymbol{z}$ from the primal SDP (25) (and the equality constraint $\boldsymbol{a}_0 - \sum_{i=1}^{k} \boldsymbol{a}_i y_i = \boldsymbol{0}$ from the dual (24)). We used an improved version of their method before applying SeDuMi [27] to solve primal-dual pair SDPs (25) and (24) in the numerical experiments in Section 5. The method worked very effectively to reduce the computational time for solving the SDPs (24) and (25). The improved method will be reported in a forthcoming paper, and the details are omitted here.

4.3 Adding linear or quadratic inequalities describing the search region

Suppose that unknown locations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ of the sensors satisfy linear or quadratic inequalities $g_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \leq 0$ $(i = 1, \ldots, k)$. Then, these inequality constraints can be added to the POPs (4) to strengthen the SDP relaxation. If $g_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \leq 0$ $(i = 1, \ldots, k)$ are convex, then any feasible solution $(\boldsymbol{X}, \boldsymbol{U}) = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{U})$ of the sparse SDP relaxation problem (21) satisfies the inequalities. For example, if the unit square $[0, 1] \times [0, 1]$ or the unit circle $\{(x, y) : x^2 + y^2 \leq 1\}$ is the region for searching sensors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$, we can add $0 \leq x_{ip} \leq 1$ $(p = 1, \ldots, m,\ i = 1, 2)$ or $1 - (\boldsymbol{x}_p)^T \boldsymbol{x}_p \geq 0$ $(p = 1, \ldots, m)$, respectively to the POP (4).

4.4 Refining solutions by a nonlinear least square method

A nonlinear optimization method can be used to refine the solution obtained by an SDP relaxation of the sensor network localization problem as suggested in [5], where the gradient method is used. In the numerical experiments in Section 5, the MATLAB function "lsqnonlin" an implementation of the trust-region reflective Newton method [7, 8] for nonlinear least square problems with bound constraints is used.

4.5 Modification of the objective function for the sensor network localization problem with noisy distances

When the given distances $\hat{d}_{pq}$ $((p,q) \in \mathcal{N}_x \cup \mathcal{N}_a)$ contain noise, the system of sensor network localization equations (3) may not have any solution. To deal with noise, an objective function should be introduced. If the true sensor locations $\boldsymbol{a}_p$ $(p = 1, \ldots, m)$ were known in advance, *the root mean square distance (rmsd)*

$$\frac{1}{m}\left(\sum_{p=1}^{m} \|\boldsymbol{x}_p - \boldsymbol{a}_p\|^2\right)^{1/2} \tag{26}$$

could be an objective function. In fact, the rmsd has been used to evaluate the quality of computed locations $\boldsymbol{x}_p$ of sensors $p = 1, \ldots, m$ generated by numerical methods in [2, 3, 4, 28, 31]. Thus, it is reasonable to choose an objective function that can decrease the value of the rmsd.

In Section 2.2, we have introduced the objective function (5), which was employed in Nie [23]. The objective function used in [31] is

$$\sum_{(p,q) \in \mathcal{N}_x} \left| \hat{d}_{pq}^2 - \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 \right| + \sum_{(p,r) \in \mathcal{N}_a} \left| \hat{d}_{pr}^2 - \|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 \right|. \tag{27}$$

Both of the objective functions involve the deviation $\|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2$ from $\hat{d}_{pq}^2$ (or $\|\boldsymbol{x}_p - \boldsymbol{x}_r\|^2$ from $\hat{d}_{pr}^2$). We also tested the objective function

$$\sum_{(p,q) \in \mathcal{N}_x} (\hat{d}_{pq} - \|\boldsymbol{x}_p - \boldsymbol{x}_q\|)^2 + \sum_{(p,r) \in \mathcal{N}_a} (\hat{d}_{pr} - \|\boldsymbol{x}_p - \boldsymbol{a}_r\|)^2. \tag{28}$$

This may be a better choice than the objective function (5) because deviations between $\hat{d}_{pq}$ and $\|\boldsymbol{x}_p - \boldsymbol{x}_q\|$ $((p,q) \in \mathcal{N}_x \cup \mathcal{N}_a)$ are involved more directly than the deviation between their squares in (5) and (27). The minimization of this objective function can be formulated as a QOP

$$\left.\begin{array}{ll}
\text{minimize} & \displaystyle\sum_{(p,q) \in \mathcal{N}_x} (\hat{d}_{pq} - v_{pq})^2 + \sum_{(p,r) \in \mathcal{N}_a} (\hat{d}_{pr} - v_{pr})^2 \\
\text{subject to} & v_{pq}^2 - \|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 = 0,\ v_{pq} \geq 0 \ ((p,q) \in \mathcal{N}_x), \\
& v_{pr}^2 - \|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 = 0,\ v_{pr} \geq 0 \ ((p,r) \in \mathcal{N}_a), \\
& (1-\phi)\hat{d}_{pq} \leq v_{pq} \leq (1+\psi)\hat{d}_{pq} \ ((p,q) \in \mathcal{N}_x), \\
& (1-\phi)\hat{d}_{pr} \leq v_{pr} \leq (1+\psi)\hat{d}_{pr} \ ((p,r) \in \mathcal{N}_a),
\end{array}\right\} \tag{29}$$

where $\phi \in [0,1]$ and $\psi \geq 0$ are fixed real numbers. We note that the last two lines of upper and lower bound constraints in (29) are added to strengthen the SDP relaxation (see Section 5.6 of [30]), and that if we take $\phi = \psi = 0$, then the QOP (29) coincides with the QOP (4). The sparse SDP relaxation of the QOP (29) provided smaller rmsd values and better solution quality than the sparse SDP relaxation of the QOP (6) in the numerical experiments although it required slightly more cpu time since the objective function in (29) is not as simple as that in (6).

5 Numerical results

We compare the sparse SDP relaxation (SSDP) with an implementation of the Biswas-Ye SDP relaxation [3] (FSDP) and the edge-based SDP relaxation [31] (ESDP). Recall that SSDP is an application of the sparse SDP relaxation by Waki, Kim, Kojima and Muramatsu [30] for a general sparse polynomial optimization problem to the QOP (4) for sensor network localization problems with exact distances, and to the QOP (29) for problems with noisy distances. For problems with exact distances, see also (21) for SSDP, (14) for FSDP and (23) with $\Gamma = \{\{p, q\} : (p, q) \in \mathcal{N}_x\}$ for ESDP, respectively. ESDP is shown to be more efficient than FSDP, the POP method in [23], and the SOCP relaxation in [28]. MATLAB codes for ESDP and FSDP are downloaded from the website [33].

SDP relaxation problems are constructed by applying SparsePOP [30] to the QOPs (4) and (29), and solved by SeDuMi [27] in SparsePOP. Both FSDP and ESDP use SeDuMi. For refining the obtained solutions, ESDP utilizes the gradient method while SSDP uses a nonlinear least square method provided by the MATLAB function lsqnonlin, which is an implementation of the trust region Newton method [7, 8] for nonlinear least square problems with bound constraints. As mentioned in Section 4.2, an improved version of the method in [16] for handling equality constraints to reduce the size of the SSDP relaxation problem is employed before applying SeDuMi for the numerical experiments.

Numerical test problems are generated as follows: m sensors $\boldsymbol{a}_p$ $(p = 1, 2, \ldots, m)$ are distributed randomly in the 2-dimemensional unit square $[0, 1]\times[0, 1]$ or the 3-dimemensional unit cube $[0, 1]^3$, where $m = 500$ and 1000 are used in the 2-dimensional problems and $m = 250$ in the 3-dimensional problems. Anchors are placed as follows:

$$
\begin{aligned}
\text{center3} &: \text{3 anchors at } (0.5, 0.5), (0.6., 0.5), (0.5, 0.6),\\
\text{bd3} &: \text{3 anchors on the boundary points } (0, 0), (0.5, 0), (0, 0.5),\\
\text{corner4} &: \text{4 anchors on the corners } (a_1, a_2),\ a_i \in \{0, 1\},\\
5 \times 5 &: \text{25 anchors on the grid } (a_1, a_2),\ a_i \in \{0, 1/4, 2/4, 3/4, 1\},\\
\text{rand50} &: \text{50 randomly placed anchors in } [0, 1] \times [0, 1],\\
\text{rand100} &: \text{100 randomly placed anchors in } [0, 1] \times [0, 1]
\end{aligned}
$$

for 2-dimensional problems, and

$$
\begin{aligned}
\text{center4} &: \text{4 anchors at } (0.5, 0.5, 0.5), (0.5, 0.6., 0.5), (0.5, 0.5, 0.6),\\
&\quad\ (0.6, 0.5, 0.5),\\
\text{bd4} &: \text{4 anchors on the boundary } (0, 0, 0), (0.5, 0, 0), (0, 0.5, 0),\\
&\quad\ (0, 0, 0.5),\\
\text{corner8} &: \text{8 anchors on the corners } (a_1, a_2, a_2),\ a_i \in \{0, 1\},\\
3 \times 3 \times 3 &: \text{27 anchors on the grid } (a_1, a_2, a_3),\ a_i \in \{0, 1/2, 1\},\\
\text{rand25} &: \text{25 randomly placed anchors.}
\end{aligned}
$$

for the 3-dimensional problems. A radio range ρ chosen from $\{0.1,\ 0.2,\ 0.3\}$ for the 2-dimensional problems or from $\{0.3,\ 0.4,\ 0.5\}$ for the 3-dimensional problems determines the sets $\mathcal{N}_x$ and $\mathcal{N}_a$ by (2). The exact distances

$$
d_{pq} = \|\boldsymbol{a}_p - \boldsymbol{a}_q\| \ ((p, q) \in \mathcal{N}_x) \text{ and } d_{pr} = \|\boldsymbol{a}_p - \boldsymbol{a}_r\| \ ((p, r) \in \mathcal{N}_a)
$$

are computed. For numerical problems with noisy distances, we further perturb the distances as

$$\hat{d}_{pq} = (1 + \sigma\epsilon_{pq})d_{pq} \ ((p, q) \in \mathcal{N}_x) \text{ and } \hat{d}_{pr} = (1 + \sigma\epsilon_{pr})d_{pr} \ ((p, r) \in \mathcal{N}_a). \tag{30}$$

Here σ denotes a nonnegative constant, and ϵ_{pq} and ϵ_{pr} are chosen from the standard normal distribution $N(0, 1)$. We call σ a noisy factor, and we take $\sigma = 0.01$ and $\sigma = 0.1$ for noisy problems in Sections 5.2 and 5.3.

Throughout Section 5, ρ denotes radio range, λ an upper bound for the degree of any sensor node described in Section 4.1 for ESDP, κ a lower bound for the degree of any sensor node described in Section 4.1 for FSDP and SSDP, "cpu" cpu time in seconds consumed by SeDuMi with the accuracy parameter pars.eps = 1.0e-5. The root mean square distance (rmsd) (see (26)) is used to measure the accuracy of the locations of sensor $p = 1, 2, \ldots, m$ computed by SeDuMi and to measure the accuracy of their refinement by the gradient method in ESDP or the MATLAB function lsqnonlin in FSDP and SSDP. The values of rmsd after the refinement are included in the parentheses. We note that ESDP provides only rmsd after refining the locations of sensors. Numerical experiments were performed on PowerPC 1.88GHz with 2GB memory.

5.1 Problems with exact distances

Tables 1 shows numerical results on the problems with 500 sensors randomly generated in $[0, 1] \times [0, 1]$ and exact distances. If we compare the values of rmsd, we see that

(a) FSDP(4) and SSDP(4) attain similar quality of rmsd in all test problems, however, cpu time consumed by FSDP(4) to solve SDP relaxation problems using SeDuMi is larger than SSDP(4). In particular, the difference in cpu time becomes larger as ρ increases to 0.2 and 0.3.

Based on this observation, ESDP and SSDP are compared on larger-scale problems with or without noise in 2-dimensional and 3-dimensional space in the following discussion. Table 2 shows numerical results on the problems with 1000 sensors randomly generated in $[0, 1] \times [0, 1]$ and exact distances. From Tables 1 and 2, we notice the following.

(b) For center3 and bd3 with $\rho = 0.1$, very large values of rmsd are obtained with ESDP(7), ESDP(10) and SSDP(4). See Figures 1 for center3 with $\rho = 0.1$ in Table 1.

(c) In cases of center3 with $\rho = 0.2$ and bd3 with $\rho = 0.2$ and 0.3, SSDP(4) attained rmsd small enough to locate the sensors accurately while ESDP(7) and ESDP(10) did not. See Figure 2.

(d) As the number of sensors is increased from 500 in Table 1 to $1,000$ in Tablel 2, the values of rmsd become smaller in many cases including center3 with $\rho = 0.3$ and corner4 with $\rho = 0.1$.

(e) Except the cases mentioned in (b), (c) and (d), ESDP(7), ESDP(10) and SSDP(4) attain small values of rmsd.

(f) SSDP(4) spend more cpu time to solve SDP relaxation problems by SeDuMi for corner4, 5×5, bd3 and center3 with $\rho = 0.1$, and less cpu time in all the other problems than ESDP(7) and/or ESDP(10).

(g) In all test problems, cpu time consumed by SSDP(4) to solve SDP relaxation problems using SeDuMi decreases as the radio range ρ increases. Note that as the radio range ρ increases, $\mathcal{N}_a$ involves more edges, and the reduced problem after applying the method in Section 4.1 becomes sparse and small. As a result, its SDP relaxation can be solved faster. This can be observed in Table 3 where information on the sparsity of the SDP relaxation problems constructed by SSDP is exhibited. It is known that the size of the Schur complement matrix and the number of nonzeros of its sparse Cholesky factor, which are denoted as sizeR and #nnzL, respectively, are key factors to measure the size and the sparsity of the SDP relaxation problems. As ρ changes from 0.1 to 0.3, the considerable decrease in sizeR and #nnzL leads to shorter cpu time by SeDuMi.

	anchor	ρ=0.1		ρ=0.2		ρ=0.3	
SDP($\lambda\|\kappa$)	location	rmsd	cpu	rmsd	cpu	rmsd	cpu
ESDP(7)	center3	(1.4e-02)	107.9	(6.1e-03)	107.9	(5.2e-04)	98.7
ESDP(10)	center3	(1.3e-02)	117.0	(4.6e-03)	193.0	(9.0e-06)	161.0
FSDP(4)	center3	1.3e-02(1.2e-02)	393.1	3.7e-04(4.6e-08)	519.7	(5.5e-09)	508.3
SSDP(4)	center3	8.4e-03(7.7e-03)	164.1	1.7e-04(1.2e-07)	56.2	(8.3e-09)	30.9
ESDP(7)	bd3	(2.8e-02)	30.0	(2.1e-02)	163.2	(1.3e-02)	100.1
ESDP(10)	bd3	(2.9e-02)	47.6	(2.1e-02)	262.0	(1.0e-02)	169.3
FSDP(4)	bd3	1.3e-02(1.9e-03)	461.6	5.2e-06(1.4e-07)	399.7	(4.9e-09)	548.9
SSDP(4)	bd3	3.2e-02(2.8e-02)	172.6	2.6e-03(3.2e-08)	60.3	(2.6e-08)	34.1
ESDP(7)	corner4	(1.8e-03)	39.4	(7.1e-06)	217.8	(4.1e-07)	106.9
ESDP(10)	corner4	(1.1e-03)	82.2	(1.8e-07)	237.5	(8.4e-09)	106.6
FSDP(4)	corner4	7.8e-04(3.0e-04)	515.8	4.5e-06(8.3e-09)	474.2	(4.7e-10)	399.5
SSDP(4)	corner4	1.1e-03(3.0e-04)	243.8	4.7e-06(4.8e-09)	49.2	(7.6e-10)	26.8
ESDP(7)	5×5	(8.9e-06)	98.0	(1.9e-09)	147.1	(6.6e-10)	46.5
ESDP(10)	5×5	(1.1e-07)	68.3	(2.7e-09)	93.9	(5.3e-10)	133.4
FSDP(4)	5×5	2.3e-05(6.4e-09)	340.6	1.2e-06(3.1e-10)	317.2	(1.9e-09)	367.4
SSDP(4)	5×5	1.8e-05(3.4e-09)	83.0	2.6e-06(1.9e-09)	7.6	(2.9e-09)	5.5
ESDP(7)	rand50	(2.1e-05)	87.4	(5.0e-09)	75.8	(6.7e-10)	70.2
ESDP(10)	rand50	(9.0e-07)	144.1	(9.3e-10)	104.9	(9.8e-10)	196.1
FSDP(4)	rand50	2.5e-04(3.4e-09)	352.0	1.3e-07(7.2e-09)	386.9	(4.3e-10)	410.2
SSDP(4)	rand50	5.4e-05(1.6e-08)	27.9	6.2e-08(3.2e-09)	6.9	(2.2e-15)	5.4

Table 1: Numerical results on a 2-dimensional problem with randomly generated 500 sensors in $[0, 1] \times [0, 1]$ and exact distances.

Tables 1 and 2 show that SSDP(4) does not attain small values of rmsd for the problems of center3 and bd3 with $\rho = 0.1$, and Table 4 displays how rmsd is improved as λ increases to 5, 6 and 8. See also Figure 3.

	anchor	ρ=0.1		ρ=0.2		ρ=0.3	
SDP($\lambda\|\kappa$)	location	rmsd	cpu	rmsd	cpu	rmsd	cpu
ESDP(7)	center3	(8.5e-03)	256.7	(4.9e-03)	296.9	(2.1e-05)	280.4
ESDP(10)	center3	(9.0e-03)	480.2	(3.9e-03)	635.1	(5.0e-06)	514.2
SSDP(4)	center3	4.7e-03(4.3e-03)	574.4	1.4e-04(5.6e-10)	119.7	(3.4e-09)	71.6
ESDP(7)	bd3	(2.0e-02)	126.4	(1.5e-02)	321.0	(8.2e-03)	202.6
ESDP(10)	bd3	(2.0e-02)	283.1	(1.4e-02)	560.4	(6.3e-03)	370.0
SSDP(4)	bd3	1.5e-02(1.4e-02)	557.9	9.6e-04(2.3e-08)	107.0	(1.0e-09)	84.2
ESDP(7)	corner4	(2.4e-04)	193.6	(4.3e-06)	349.7	(4.1e-08)	165.0
ESDP(10)	corner4	(9.7e-05)	351.8	(1.8e-07)	593.8	(2.0e-08)	278.7
SSDP(4)	corner4	3.5e-05(3.9e-08)	892.7	3.5e-06(1.1e-08)	101.3	(9.0e-10)	55.2
ESDP(7)	5×5	(4.5e-08)	147.6	(1.3e-09)	205.9	(2.5e-09)	183.8
ESDP(10)	5×5	(6.2e-08)	283.5	(1.2e-09)	428.1	(2.9e-10)	466.4
SSDP(4)	5×5	2.0e-05(4.3e-09)	239.7	1.0e-06(9.6e-10)	19.3	(1.8e-10)	14.2
ESDP(7)	rand100	(4.4e-06)	179.7	(7.0e-10)	210.1	(8.4e-11)	225.0
ESDP(10)	rand100	(3.4e-08)	297.9	(1.2e-09)	445.1	(3.7e-10)	663.0
SSDP(4)	rand100	3.7e-05(1.1e-09)	35.8	5.0e-08(1.1e-09)	15.4	(9.4e-17)	15.2

Table 2: A randomly generated 2-dimensional problem with 1000 sensors and noisyFac = 0.0. par.eps = 1.0e-5.

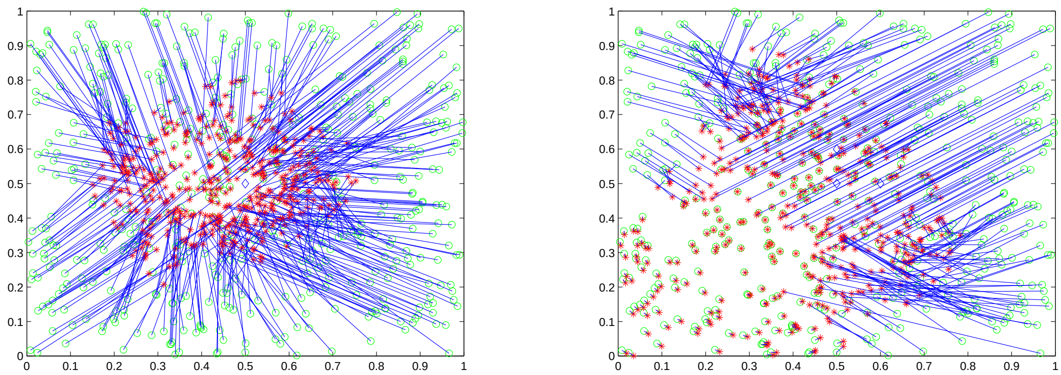

Figure 1: ESDP(10) after applying the gradient method and SSDP(4) after applying lsqnonlin for 500 sensors, 3 anchors near the center and ρ = 0.1. Here a circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment a deviation from true and computed locations.

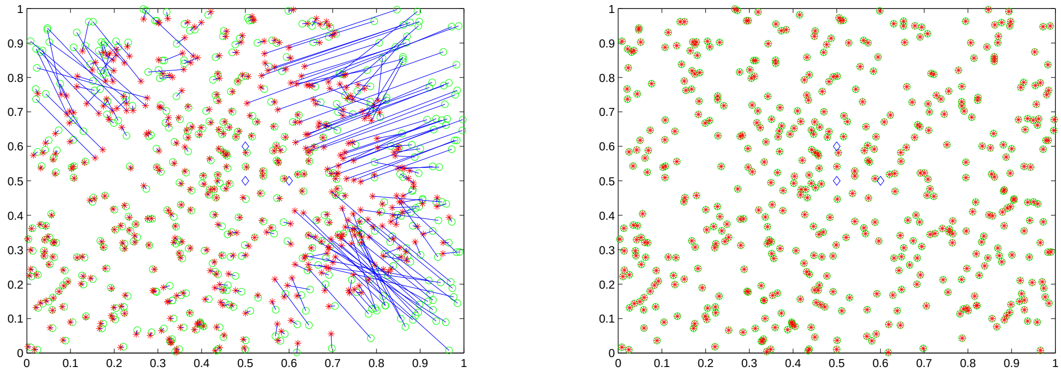

Figure 2: ESDP(10) after applying the gradient method and SSDP(4) after applying lsqnonlin for 500 sensors, 3 anchors near the center and $\rho = 0.2$.

	anchor	ρ=0.1		ρ=0.2		ρ=0.3	
	location	sizeR	#nnzL	sizeR	#nnzL	sizeR	#nnzL
ESDP(7)	corner4	22,722	1,039,584	25,227	1,945,155	25,784	1,647,843
ESDP(10)	corner4	29,113	1,945,365	32,872	3,776,211	33,748	3,250,629
FSDP(4)	corner4	1,690	1,428,895	1,911	1,826,916	1,966	1,933,916
SSDP(4)	corner4	13,137	3,632,361	9,592	1,096,530	8,481	575,600
ESDP(7)	5×5	23,172	1,081,915	27,145	2,042,682	29,804	1,836,950
ESDP(10)	5×5	29,563	1,925,326	34,790	4,139,700	37,768	3,771,251
FSDP(4)	5×5	1,754	1,539,135	2,051	2,104,326	2,493	3,108,771
SSDP(4)	5×5	10,737	1,618,590	5,245	145,918	4,206	61,440

Table 3: The size of the Schur complement matrix (sizeR) and the number of nonzeros in its sparse Cholesky factorization (#nnzL) when executing SeDuMi.

anchor	$\lambda = 5$		$\lambda = 6$		$\lambda = 8$	
location	rmsd	cpu	rmsd	cpu	rmsd	cpu
center3	7.2e-03(6.8e-03)	235.2	1.0e-03(9.6e-07)	965.7	1.6e-03(1.4e-07)	1235.7
bd3	2.2e-02(2.0e-02)	381.7	1.4e-02(5.0e-03)	658.5	5.7e-03(3.2e-04)	1767.5

Table 4: Numerical results on SSDP with $\lambda = 5$, 6 and 8 applied to randomly generated 2-dimensional problems with 500 sensors and noisyFac = 0.0. par.eps = 1.0e-5.

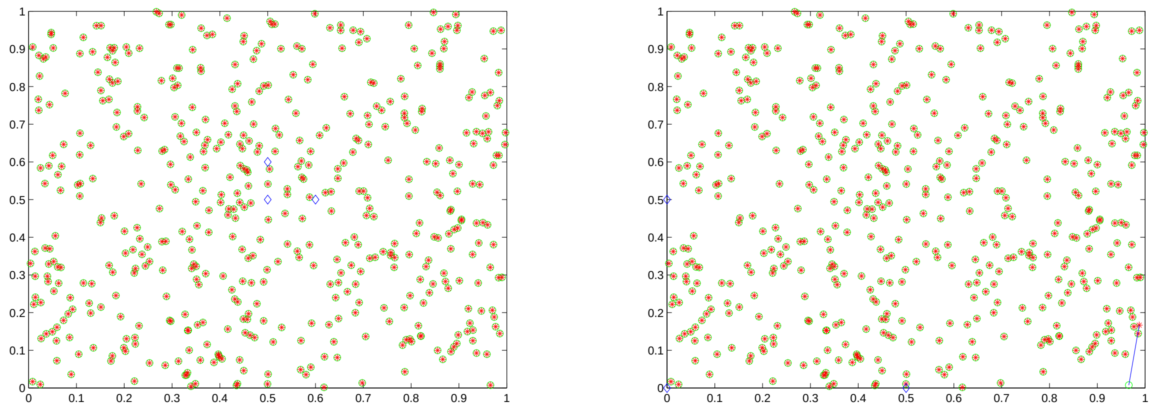

Figure 3: SSDP(8) after applying nsqnonlin for 500 sensors, 3 anchors near the center and $\rho = 0.1$, and SSDP(8) after applying nsqnonlin for 500 sensors, 3 anchors near the boundary and $\rho = 0.1$.

5.2 Problems with noisy distances

For numerical experiments in this subsection, the noisy distances are generated by (30), where $\sigma = 0.01$ is used in Table 5 and $\sigma = 0.1$ in Table 6. SSDP for problems with noisy distances is obtained by applying SparsePOP to the QOP (29), where $\phi = \psi = 4\sigma$. In most problems in Table 5 and Table 6, we observe the following.

(h) ESDP(7) and/or ESDP(10) spend less cpu time to solve SDP relaxation problems using SeDuMi than SSDP(4).

(i) For the problems of center3 with $\rho = 0.2,\ 0.3$ and bd3 with $\rho = 0.2,\ 0.3$ of Table 5, neither ESDP(7) nor ESDP(10) attains rmsd small enough to locate sensors accurately, while SSDP(4) attains small rmsd. See Figures 4 and 5 for the comparison between ESDP(10) and SSDP(4) applied to center3 with $\rho = 0.2$. If we compare the figures on the left with the ones on the right in Figures 4 and 5, we see the effectiveness of the gradient method and lsqnonlin for refining the solutions obtained from the SDP relaxations, respectively. Except these two cases, the values for rmsd obtained by both methods are comparable.

From the numerical results in Sections 5.1 and 5.2, we observe the computational advantage of SSDP(4) in getting accurate solutions faster over ESDP(7) or ESDP(10) for the problems with exact distances. For the problems with noisy distances, the computational performance of ESDP(4) is comparable to that of ESDP(7) and ESDP(10).

5.3 Three-dimensional problems

Solving 3-dimensional problems are far more difficult than the 2-dimensional problems. Thus, we only show numerical results on problems with 250 sensors in Table 7. In the column of $\rho = 0.3$ of Table 7, out-of-memory error, denoted by -, often occurred for $\rho = 0.3$. From Table 7, we notice the following.

	anchor	ρ=0.1		ρ=0.2		ρ=0.3	
SDP($\lambda\|\kappa$)	location	rmsd	cpu	rmsd	cpu	rmsd	cpu
ESDP(7)	center3	(1.0e-02)	245.5	(8.0e-03)	267.2	(4.8e-03)	144.8
ESDP(10)	center3	(1.0e-02)	529.0	(7.6e-03)	547.6	(4.1e-03)	337.3
SSDP(4)	center3	1.0e-02(8.2e-03)	1816.9	3.0e-03(5.6e-05)	535.4	(7.0e-05)	393.2
ESDP(7)	bd3	(2.0e-02)	162.7	(1.9e-02)	196.3	(1.7e-02)	135.3
ESDP(10)	bd3	(1.9e-02)	471.5	(1.7e-02)	433.6	(1.5e-02)	295.3
SSDP(4)	bd3	1.2e-02(1.1e-02)	1610.6	4.6e-03(4.7e-05)	463.7	(7.0e-05)	413.1
ESDP(7)	corner4	(3.3e-04)	177.1	(5.0e-05)	226.2	(6.4e-05)	139.4
ESDP(10)	corner4	(9.7e-05)	514.1	(4.0e-05)	670.4	(4.8e-05)	274.9
SSDP(4)	corner4	3.3e-04(4.1e-05)	1758.6	2.4e-04(5.1e-05)	540.1	(7.2e-05)	364.5
ESDP(7)	5×5	(1.3e-04)	207.4	(3.2e-05)	198.3	(3.9e-05)	140.1
ESDP(10)	5×5	(9.5e-05)	384.8	(2.8e-05)	641.6	(3.5e-05)	285.3
SSDP(4)	5×5	1.7e-04(2.3e-05)	620.5	1.1e-04(4.5e-05)	189.5	(6.2e-05)	192.0
ESDP(7)	rand100	(1.3e-04)	180.9	(2.1e-05)	235.6	(2.4e-05)	275.4
ESDP(10)	rand100	(1.5e-05)	311.3	(2.0e-05)	648.8	(2.4e-05)	515.3
SSDP(4)	rand100	1.4e-04(2.1e-05)	262.6	1.8e-04(4.4e-05)	205.8	(6.3e-05)	238.0

Table 5: Numerical results on a 2-dimensional problem with randomly generated 1000 sensors in $[0,1] \times [0,1]$ and noisy distances (the noise factor $\sigma = 0.01$).

	anchor	ρ=0.1		ρ=0.2		ρ=0.3	
SDP($\lambda\|\kappa$)	location	rmsd	cpu	rmsd	cpu	rmsd	cpu
ESDP(7)	center3	(1.1e-02)	201.7	(9.7e-03)	166.8	(7.8e-03)	114.7
ESDP(10)	center3	(1.1e-02)	369.0	(8.7e-03)	408.4	(5.3e-03)	235.3
SSDP(4)	center3	1.2e-02(1.0e-02)	1450.5	8.8e-03(7.4e-03)	413.5	(3.0e-03)	358.2
ESDP(7)	bd3	(1.9e-02)	161.8	(1.9e-02)	137.4	(1.9e-02)	97.6
ESD(10)	bd3	(2.0e-02)	308.6	(2.1e-02)	297.2	(1.9e-02)	181.7
SSDP(4)	bd3	1.4e-02(1.1e-02)	1354.6	1.3e-02(1.1e-02)	420.3	(1.4e-02)	389.8
ESDP(7)	corner4	(4.8e-04)	229.3	(5.4e-04)	149.4	(6.5e-04)	98.1
ESDP(10)	corner4	(4.3e-04)	347.1	(4.1e-04)	381.4	(4.8e-04)	192.7
SSDP(4)	corner4	1.3e-03(3.8e-04)	1611.2	1.4e-03(5.0e-04)	509.3	(7.6e-04)	305.3
ESDP(7)	5×5	(2.2e-04)	155.2	(3.1e-04)	116.5	(3.9e-04)	97.6
ESDP(10)	5×5	(2.0e-04)	295.5	(2.8e-04)	337.1	(3.6e-04)	192.0
SSDP(4)	5×5	5.4e-04(2.3e-04)	666.9	6.6e-04(4.5e-04)	182.1	(6.2e-04)	197.4
ESDP(7)	rand100	(2.8e-04)	146.5	(2.1e-04)	154.9	(2.4e-04)	169.1
ESDP(10)	rand100	(2.6e-04)	200.6	(2.0e-04)	406.2	(2.4e-04)	277.1
SSDP(4)	rand100	5.1e-04(2.1e-04)	231.0	9.5e-04(4.6e-04)	208.0	(7.9e-04)	223.0

Table 6: Numerical results on a 2-dimensional problem with randomly generated 1000 sensors in $[0,1] \times [0,1]$ and noisy distances (the noise factor $\sigma = 0.1$).

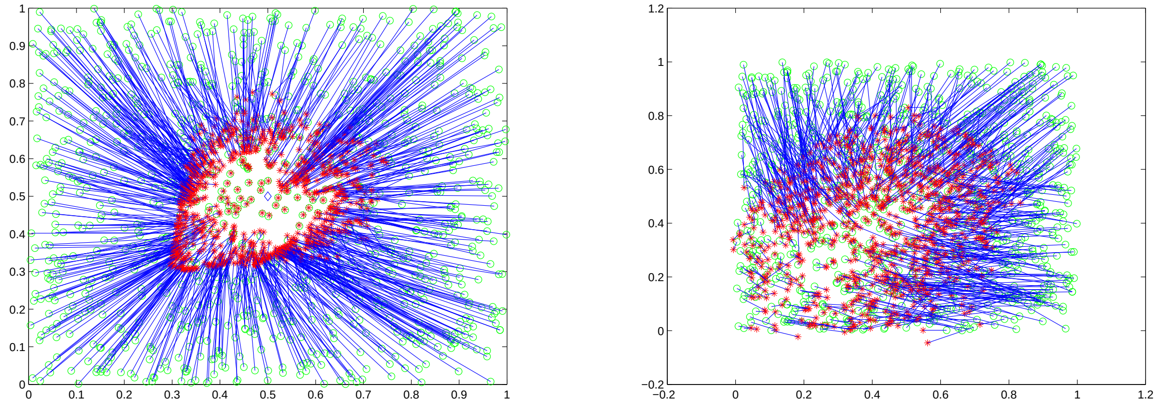

Figure 4: ESDP(10) before and after applying the gradient method for 1000 sensors , 3 anchors near the center and $\rho = 0.2$.

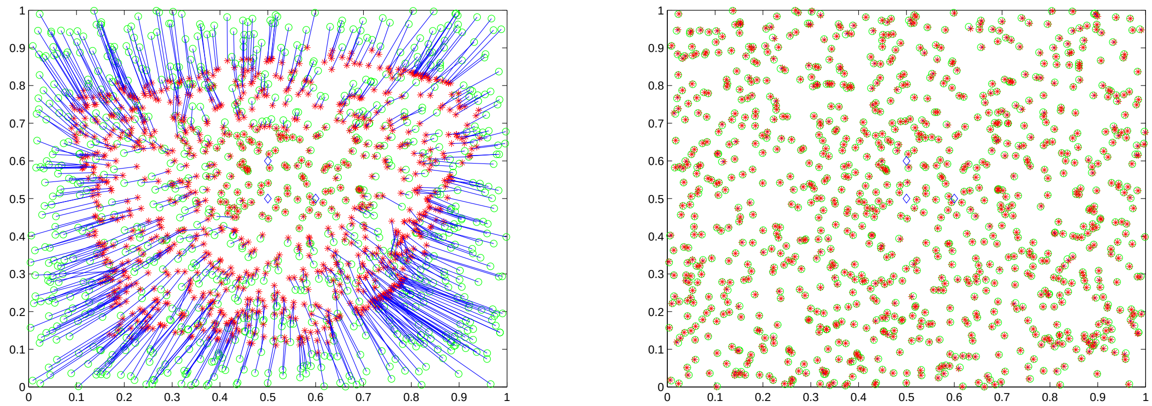

Figure 5: SSDP(4) before and after applying the matlab function lsqnonlin for 1000 sensors, 3 anchors near the center and $\rho = 0.2$.

(j) When the number of anchors are small (center4, bd4 and conrner8) and/or the radio range ρ is small ($\rho = 0.3$), it takes very long for SeDuMi to solve SDP relaxation problems.

(k) For problems with no noise ($\sigma = 0$) and $\rho = 0.4,\ 0.5$, SSDP(5) attains very small rmsd values, however, as the noisy factor σ grows, the values of rmsd increase. See Figure 6 and Figure 7. In Figure 6, we also observe the effectiveness of lsqnonlin in improving rmsd.

	anchors'	ρ=0.3		ρ=0.4		ρ=0.5	
σ	location	rmsd	cpu	rmsd	cpu	rmsd	cpu
0.00	center4	8.9e-03(9.5e-03)	4448.5	1.0e-03(7.9e-09)	2013.5	(1.2e-08)	268.6
0.00	bd4	-		8.0e-03(4.5e-08)	2937.9	(6.4e-09)	606.4
0.00	corner8	-		6.6e-06(3.8e-09)	1856.3	(5.3e-09)	194.0
0.00	$3 \times 3 \times 3$	1.7e-04(3.0e-04)	2206.4	8.3e-06(7.3e-10)	56.4	(6.2e-10)	8.4
0.00	rand25	2.2e-04(3.0e-09)	501.0	2.3e-05(4.3e-09)	43.3	(4.4e-10)	11.1
0.01	center4	-		7.6e-03(3.0e-04)	3128.4	(3.1e-04)	433.4
0.01	bd4	-		2.3e-02(2.7e-04)	3105.7	(2.0e-03)	748.6
0.01	corner8	-		1.2e-03(2.7e-04)	2285.9	(2.9e-04)	390.0
0.01	$3 \times 3 \times 3$	-		7.8e-04(2.5e-04)	123.3	(2.9e-04)	45.9
0.01	rand25	3.1e-03(8.3e-04)	1031.8	2.6e-03(2.3e-04)	124.6	(2.8e-04)	56.1
0.10	center4	-		2.1e-02(1.8e-02)	2160.7	(6.1e-03)	366.9
0.10	bd4	-		3.4e-02(2.3e-02)	3106.8	(6.1e-03)	622.5
0.10	corner8	-		6.0e-03(2.6e-03)	2152.9	(3.3e-03)	350.0
0.10	$3 \times 3 \times 3$	-		3.5e-03(2.5e-03)	139.4	(2.8e-03)	46.2
0.10	rand25	8.3e-03(6.8e-03)	646.9	9.1e-03(7.4e-03)	99.9	(4.7e-03)	55.3

Table 7: Numerical results on a 3-dimensional problem with randomly generated 250 sensors in $[0, 1]^3$.

6 Concluding remarks

We have formulated the sensor network localization problem with exact distances as a QOP (4) and the problem with noisy distances as a QOP (29), and proposed to apply the sparse SDP relaxation [30] with the relaxation order 1 to the QOPs. We have shown that the sparse SDP relaxation is equivalent to the dense relaxation [17] with the same relaxation order 1, and that it is at least as strong as the Biswas-Ye SDP relaxation [2] theoretically. We have also derived a sparse variant of the Biswas-Ye SDP relaxation. For the solutions of QOPs derived from the problems with exact distances and with noisy distances, a matlab package SparsePOP is applied and a matlab function "lsqnonlin" is used for refining the obtained solution. Numerical results demonstrate that exploiting sparsity in our method is very effective in computing accurate solutions in less cpu time, especially for the problems with exact distances. In particular, (i) the sparse SDP relaxation is faster than the Biswas-Ye

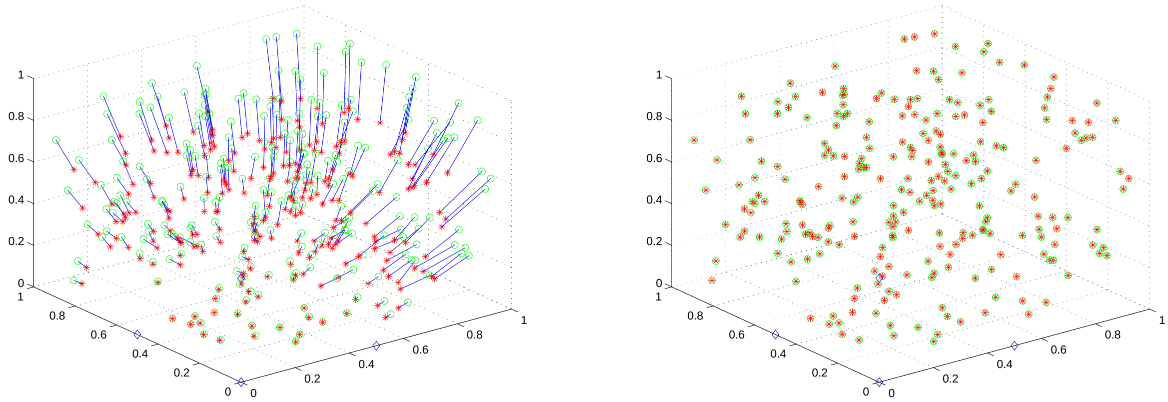

Figure 6: SSDP(5) before and after applying lsqnonlin for 250 sensors, 3 anchors on the boundary, $\sigma = 0.0$ and $\rho = 0.4$.

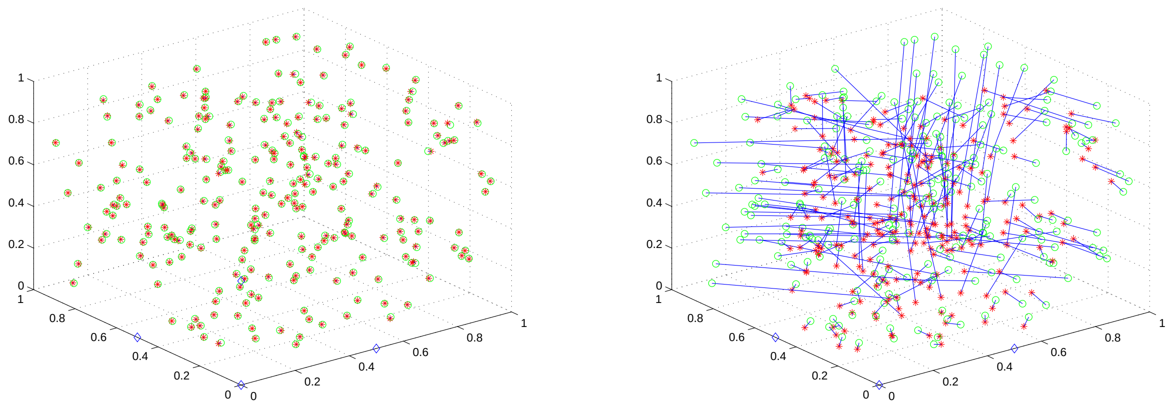

Figure 7: The figure on the left: SSDP(5) after applying lsqnonlin for 250 sensors, 4 anchors on the boundary, $\sigma = 0.01$ and $\rho = 0.4$. The figure on the right: SSDP(5) after applying lsqnonlin for 250 sensors, 4 anchors on the boundary, $\sigma = 0.1$ and $\rho = 0.4$

SDP relaxation, (ii) it requires less cpu time as the number of anchors and/or the radio range increase, and (iii) it provides more accurate solutions than the edge-based SDP relaxation [31] for the problems with exact distances and a small number of anchors.

We have applied SparsePOP to the QOPs (4) and (29) to construct their sparse SDP relaxation problems with the relaxation order 1. We should, however, note that SparsePOP is designed for solving general sparse POPs. Thus, its application to the QOPs (4) and (29) is not very efficient. For better computational efficiency, it is necessary to develop codes specialized for generating sparse SDP relaxation problems with the relaxation order 1 from the QOPs (4) and (29). It is also interesting to apply the positive definite matrix completion method [12, 21] and its parallel implementation SDPARA-C [22] to the (dense) SDP relaxation (13) and the Biswas-Ye SDP relaxation as discussed in Section 3.3.

References

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

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

[3] 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.

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

[5] 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.

[6] J. R. S. Blair and B. Peyton (1993) "An introduction to chordal graphs and lieque trees," In: A. George, J. R. Gilbert and J. W. H. Liu des, *Graph Theory and Sparse Matrix Computation*, Springer, New York, pp.1-29.

[7] T. F. Coleman and Y. Li (1994) "On the Convergence of Reflective Newton Methods for Large-Scale Nonlinear Minimization Subject to Bounds," *Mathematical Programming,* **67,** 2, 189-224.

[8] T. F. Coleman and Y. Li (1996) "An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds," *SIAM Journal on Optimization,* **6**, 418-445.

[9] 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.

[10] 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.*

[11] Fujie, T., Kojima, M. (1997): "Semidefinite relaxation for nonconvex programs," *Journal of Global Optimization* **10**, 367–380

[12] M. Fukuda, M. Kojima, K. Murota and K. Nakata (2000) "Exploiting sparsity in semidefinite programming via matrix completion I: General framework," *SIAM Journal on Optimization*, **11**, 647-674.

[13] 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," March.

[14] 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.

[15] K. Kobayashi, S. Kim and M. Kojima, Correlative sparsity in primal-dual interior-point methods for LP, SDP and SOCP, to appear in *Applied Mathematics and Optimization.*

[16] K. Kobayashi, K. Nakata, and M. Kojima (2007) "A conversion of an SDP having free variables into the standard form SDP," *Computational Optimization and Applications*, **36**, 289-307.

[17] J. B. Lasserre (2001) "Global optimization with polynomials and the problems of moments," *SIAM Journal on Optimization*, **11**, 796–817.

[18] J. B. Lasserre (2006) "Convergent SDP-relaxations in polynomial optimization with sparsity," *SIAM Journal on Optimization*, **17**, 3, 822-843.

[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] J. J. Moré, Z. Wu (1997) "Global continuation for distance geometry problems," *SIAM Journal on Optimization*, **7**, 814-836.

[21] 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," *Mathematical Programming*, **95**, 303-327.

[22] K. Nakata, M. Yamashita, K. Fujisawa and M. Kojima (2006) "A parallel primal-dual interior-point method for semidefinite programs using positive definite matrix completion," *Parallel Computing*, **32**, 24-43.

[23] J. Nie (2006) "Sum of sqaures method for sensor network localization," preprint.

[24] N. Z. Shor (1987) “Quadratic optimization problems,” *Soviset journal of Computer and Systems Sciences*, **25**, 1-11.

[25] N. Z. Shor (1990) “dual quadratic estimates in polynomial and boolean programming,” *Annals of Operations Research*, **25**, 163-168.

[26] A. M. So and Y. Ye (2007) “Theory of semidefinite programming for sensor network localization,” *Mathematical Programming,* Ser. B, **109**, 367-384.

[27] J. F. Strum, “SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones,” *Optimization Methods and Software*, **11 & 12** (1999) 625-653.

[28] P. Tseng, (2007) “Second order cone programming relaxation of sensor network localization,” to appear in *SIAM Journal on Optimization.*

[29] H. Waki, S. Kim, M. Kojima and M. Muramatsu (2007) “SparsePOP : a Sparse Semidefinite Programming Relaxation of Polynomial Optimization Problems,” Research report B-414, Dept. of Math. & Computing Sciences, Tokyo Institute of Technology, August.

[30] H. Waki, S. Kim, M. Kojima, M. Muramatsu and H. Sugimoto (2006) “Sums of Squares and Semidefinite Programming Relaxations for Polynomial Optimization Problems with Structured Sparsity,” *SIAM Journal on Optimization,* **17**, 218–242.

[31] Z. Wang, S. Zheng, S. Boyd, and Y. Ye (2007) “Further relaxations of the SDP approach to sensor network localization,” preprint.

[32] M. Yamashita, K. Fujisawa and M. Kojima (2003) “Implementation and evaluation of SDPA 6.0 (SemiDefinite Programming Algorithm 6.0),” *Optimization Methods and Software,* **18**, 491-505.

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