ISSN 1342-2804

Research Reports on Mathematical and Computing Sciences

A Continuation Method for Large-sized Sensor Network Localization Problems

Sunyoung Kim and Masakazu Kojima

August 2011, B–466

Department of Mathematical and Computing Sciences
Tokyo Institute of Technology

SERIES B: Operations Research

A Continuation Method for Large-sized Sensor Network Localization Problems

Sunyoung Kim⋆ and Masakazu Kojima†

August, 2011.

Abstract.
The solution methods based on semidefinite programming (SDP) relaxations for sensor network localization (SNL) problems can not handle very large-sized SNL problems. We present a continuation method using the gradient descent method to efficiently solve large-sized SNL problems. We first formulate the problem as an unconstrained optimization problem and then apply the continuation on the distance information with the continuation parameter. We show numerically that the continuation method provides an approximate solution efficiently with comparable accuracy to that of SFSDP, a Matlab software package, which showed better performance than other SDP-based methods for solving various types of the problems. Numerical results are presented to illustrate the performance of the proposed method in comparison with SFSDP.

Key words. Sensor network localization problems, continuation methods, a first-order method, Matlab software package.

AMS Classification. 90C06, 90C52.

⋆ Department of Mathematics, Ewha W. University, 11-1 Dahyun-dong, Sudaemoon-gu, Seoul 120-750 Korea. The research was supported by KOSEF 2009-007-1314 and NRF 2010-000-8784.
`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.
`kojima@is.titech.ac.jp`

1 Introduction

Sensor network localization problems (SNL) has attracted considerable research interests for a broad spectrum of applications using wireless sensor networks. The SNL problem is to estimate the locations of m sensors of unknown positions using given distances and some sensors of known positions (called anchors) in a sensor network of n sensors, where $n > m$. Finding the solutions of this problem is known to be NP-hard in general [14]. Thus, approximating the solution of this problem has been dealt with from many angles [1, 6, 7, 10].

Among many approaches for SNL problems, the semidefinite programming (SDP) relaxation method proposed by Biswas and Ye in [2] has received a plenty of attention in the field of optimization. We will call this relaxation SDP relaxation method as the full SDP relaxation method and abbreviate it as FSDP. An advantages of FSDP is that it can provide approximate solutions with accuracy. It can solve small to medium-sized SNL problems. Recent studies [3, 4, 5, 15, 20, 22, 16] have been directed to improving the efficiency of solving larger-sized problems. The main difficulty of solving large-sized SNL problems by FSDP is from the fact that the SDP relaxation of the SNL problem is solved by one of the SDP solvers based on the primal-dual interior-point method [8, 19, 21]. Since handling large-scale SDPs by these software packages still remains a computational challenge, large-sized SDPs induced from the SNL problems can not be solved with SDP solvers. As a result, further relaxations of FSDP were proposed: the second-order cone programming (SOCP) relaxation proposed by [20], edge-based SDP (ESDP) and node-based SDP (NSDP) relaxations in [22]. Although the ESDP and NSDP relaxations attain better accuracy than the SOCP relaxation, the quality of the solutions by ESDP and NSDP is weaker than that of the original FSDP. They showed, however, computationally, the quality of the solution is comparable to that of FSDP. Recently, a further relaxation of ESDP was proposed in [16] and demonstrated to solve the SNL problems of increased size.

Another method based on the SDP relaxation was introduced by exploiting the sparsity of the SNL problem by Kim, Kojima and Waki [11, 12, 13]. This method, which we call a sparse version of the Biswas and Ye's SDP relaxation FSDP and abbreviate it as SFSDP, maintains the same theoretical property as FSDP, while providing the approximate solutions much faster. In fact, the performance of SFSDP is better than other methods proposed for the SNL problems in terms of the solution quality and the size of the SNL problems. It was shown in [13] that 2-dimensional problems with 20,000 sensors, and 3-dimensional problems with 5,000 sensors and 250 anchors could be solved efficiently using a machine with 16GB memory.

We note that the primal-dual interior-point method for the SDP relaxation employs a second-order method such as Newton's method and solving the Schur complement equation is the most time-consuming part of the primal-dual interior-point method. When much larger-sized SNL problems needs to be solved, numerical methods based on a first-order method can be considered for efficiency. This motivates us to study solving the SNL problems using the continuation method with the gradient method.

The conventional continuation method is used in general for tracing a solution $\boldsymbol{u}(t)$ of a system of nonlinear equation $F(\boldsymbol{u}, t) = 0$ with the parameter changing from 0 to 1. Here F is a mapping from $\mathbb{R}^n \times [0, 1]$ into $\mathbb{R}^n$. The SNL problem of finding the locations of sensors using the given distances can be expressed with the distance equations. The number of distance equations is usually larger than the number of unknowns, resulting an overdetermined system of nonlinear equations. Thus, we derive an unconstrained minimization problem with the parameter t changing 0 to 1 from the overdetermined system: For $F : R^n \times [0,1] \to R^r$ with $r > n$, we minimize $\sum_{i=1}^{r}(F_i(\boldsymbol{u},t))^2$ in $\boldsymbol{u} \in R^n$. With initial distances and locations of sensors simply computed at $t = 0$, we form a continuation from the initial distances to the given distances of the SNL problem. Our goal is to find the locations of sensors corresponding to the given distances by applying a first order method at each $t \in [0,1]$. In particular, the gradient method is applied to the minimization of $\sum_{i=1}^{r}(F_i(\boldsymbol{u},t))^2$ in $\boldsymbol{u} \in R^n$ as t varies from 0 to 1. Although there is no guarantee for the continuation method to attain a global minimizer of the SNL problem, the numerical solutions from the continuation method at $t = 1$ measured by the root mean squared distance (RMSD) show comparable accuracy to other methods for SNL problems. The continuation method can also be used to refine the approximate solution obtained by the other approaches.

The continuation method has computational advantages over SDP-based methods such as FSDP, SFSDP, and ESDP methods and the SOCP relaxation. First, it can efficiently solve much larger-sized problems. Second, the obtained approximate solution still have relatively good accuracy, although they can not be more accurate than the ones from the SDP-based methods in theory. Third, much less memory is required since the gradient method does not need to store large matrices.

We show in Section 5 that much larger-sized SNL problems can be solved with the continuation method than the methods based on the SDP relaxation. For instance, the approximate solutions of the SNL problems with 20,000 sensors in 2-dimensions and 10,000 sensors in 3-dimensions can be obtained with a PC with 4 GB. In [13], the largest size of the SNL problems that could be solved using a PC with 16GB is 20,000 in 2-dimensions and 5,000 in 3-dimensions. Numerical experiments demonstrate that the continuation method requires less the CPU time for most of test problems, except for the problems with a very small number of anchors.

This paper is organized as follows. In Section 2, preliminary materials of the SNL problem are presented. The continuation method is described in Section 3. In Section 4, we present methods for selecting initial locations of sensors. In addition, computational issues regarding step sizes and stopping conditions are discussed. Section 5 includes numerical results in comparison with SFSDP. We conclude in Section 6.

2 Preliminaries

We describe a SNL problem with m sensors and m_a $(= n - m)$ anchors. A radio range $\rho > 0$ determines the set $\mathcal{N}_x^\rho$ for pairs of sensors p and q and the set $\mathcal{N}_a^\rho$ for pairs of a sensor p and an anchor r. More precisely,

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

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.

Let $\boldsymbol{D} \in R^{m\times n}$ be the distance matrix and its (p,q)th element d_{pq} denote the distance between the sensors $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ or the sensor $\boldsymbol{x}_p$ and the anchor $\boldsymbol{a}_q$:

$$d_{pq} = \begin{cases} \|\boldsymbol{x}_p - \boldsymbol{x}_q\| + \epsilon'_{pq} & \text{if } (p,q) \in \mathcal{N}_x^\rho \text{ and } p < q \\ \|\boldsymbol{x}_p - \boldsymbol{a}_q\| + \epsilon'_{pq} & \text{if } (p,q) \in \mathcal{N}_a^\rho \text{ and } p < q \\ 0 \quad \text{otherwise,} \end{cases} \tag{1}$$

where $\epsilon'_{pq} = 0$ for the problem with exact distances and ϵ'_{pq} means noise in the distances for the problem with noise. Note that $\boldsymbol{D}$ is upper triangular and the number of zero elements in $\boldsymbol{D}$ increases as ρ becomes smaller.

2.1 SNL problems with exact distances

The system of distance equations for the problem with exact distances is expressed as

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

where $\mathcal{N}_x$ is a subset of $\mathcal{N}_x^\rho$ and $\mathcal{N}_a$ a subset of $\mathcal{N}_a^\rho$.

Using the system of equations (2), we can formulate the SNL problem as an unconstrained optimization problem:

$$\text{minimize} \sum_{(p,q)\in\mathcal{N}_x} |\|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 - d_{pq}^2| + \sum_{(p,r)\in\mathcal{N}_a} |\|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 - d_{pr}^2|. \tag{3}$$

Note that the objective function of (3) is not smooth. This problem is reformulated as a minimization of a linear objective function subject to quadratic equality constraints to which we can apply an SDP relaxation [2].

The problem considered in [15] was

$$\text{minimize} \sum_{(p,q)\in\mathcal{N}_x} (\|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2 - d_{pq}^2)^2 + \sum_{(p,r)\in\mathcal{N}_a} (\|\boldsymbol{x}_p - \boldsymbol{a}_r\|^2 - d_{pr}^2)^2. \tag{4}$$

Since the objective function of (4) is smooth, a local method such as the gradient method can be applied. The degree of the objective function is 4, which requires more work than (3) if the methods based on SDP relaxation is used.

Alternatively, the SNL problem can be formulated as

$$\text{minimize } f(\boldsymbol{X}) := \sum_{(p,q)\in\mathcal{N}_x} (\|\boldsymbol{x}_p - \boldsymbol{x}_q\| - d_{pq})^2 + \sum_{(p,r)\in\mathcal{N}_a} (\|\boldsymbol{x}_p - \boldsymbol{a}_r\| - d_{pr})^2, \tag{5}$$

where we denote $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{\ell\times m}$. Note that this mapping f is continuously differentiable on the open dense subset

$$\Xi = \left\{ \boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{\ell\times m} : \begin{array}{l} \boldsymbol{x}_p \neq \boldsymbol{x}_q \ (1 \le p < q \le m) \\ \boldsymbol{x}_p \neq \boldsymbol{a}_r \ (1 \le p \le m < r \le n) \end{array} \right\}.$$

This model was used to refine the solutions obtained by the SDP relaxation of (3) in [2], with the gradient method. It was mentioned in [2] that it provided more accurate numerical solutions than (4). Based on this observation, we consider solving the SNL problem formulated as (5).

2.2 SNL problems with noisy distance data

For the problems with noise, an estimated distance d_{pq} in (1) contains noise ϵ'_{pq} between sensors p and q (or an estimated distance d_{pr} includes ϵ'_{pr} between sensor p and anchor r). Then, the same form of the problem (5) can be considered for noisy problems, only difference is that d_{pq} and d_{pr} contain noise. In the subsequent sections, we mainly discuss with the problem (5) with exact distances, however, the discussion can be applied to the SNL problems with noise similarly.

3 Continuation with the Gradient Method

The SNL problem is to find $\boldsymbol{X} \in \mathbb{R}^{\ell \times m}$ that minimizes (5) with the given distance matrix $\boldsymbol{D}$ described in (1). Let $\boldsymbol{D}^0$ be an initial distance matrix, with which we can compute the location of sensors easily. We employ a continuation method for the SNL problem with the give distance matrix $\boldsymbol{D}$ as follows: Let

$$\widehat{\boldsymbol{D}}(t) = (1-t)\boldsymbol{D}^0 + t\boldsymbol{D}, \tag{6}$$

where t is a continuation parameter $0 \leq t \leq 1$. Note that $\widehat{\boldsymbol{D}}(t)$ becomes the distance matrix while performing the continuation with t. More precisely, let $\widehat{d}_{pq}(t)$ and $\widehat{d}_{pr}(t)$ indicate the (p,q)th and (p,r)th element of $\widehat{\boldsymbol{D}}(t)$, and d^0_{pq} and d^0_{pr} the (p,q)th and (p,r)th element of $\boldsymbol{D}^0$, respectively. Then, (6) means

$$\widehat{d}_{pq}(t) = (1-t)d^0_{pq} + td_{pq} \quad (p,q) \in \mathcal{N}_x, \quad \widehat{d}_{pr}(t) = (1-t)d^0_{pr} + td_{pr} \quad (p,r) \in \mathcal{N}_a,$$

for $0 \leq t \leq 1$.

Since t changes from 0 to 1, we discretize the interval $[0,1]$ with some positive integer $\tau \geq 1$. Let $\Delta t = \frac{1}{\tau}$. Then, the interval $[0,1]$ can be discretized uniformly, i.e., $t_0 = 0,\ t_1 = t_0 + \Delta t,\ t_2 = t_1 + \Delta t, \ldots,\ t_\tau = 1$. For each value of t, we consider solving

$$\text{minimize } h(\boldsymbol{X}, t) = \sum_{(p,q)\in\mathcal{N}_x} (\|\boldsymbol{x}_p - \boldsymbol{x}_q\| - \widehat{d}_{pq}(t))^2 + \sum_{(p,r)\in\mathcal{N}_a} (\|\boldsymbol{x}_p - \boldsymbol{a}_r\| - \widehat{d}_{pr}(t))^2. \tag{7}$$

Obviously, when $t = 1$, $\widehat{\boldsymbol{D}}(t)$ becomes $\boldsymbol{D}$, thus, solving (7) returns to the original SNL problem (5).

For the brief description of the gradient method applied to the problem (7), we let $\boldsymbol{X}^k = [\boldsymbol{x}^k_1, \boldsymbol{x}^k_2, \ldots, \boldsymbol{x}^k_m] \in \mathbb{R}^{\ell \times m}$. The superscript k in $\boldsymbol{X}^k$ denotes kth iteration of the gradient method. We first choose an initial approximation $\boldsymbol{X}^0$ to $\boldsymbol{X}$. For each iteration k $(k = 1, 2, \ldots)$,

$$\boldsymbol{X}^{k+1} = [\boldsymbol{x}^k_1 - s_k \nabla_{\boldsymbol{x}_1} h(\boldsymbol{X}^k, t), \boldsymbol{x}^k_2 - s_k \nabla_{\boldsymbol{x}_2} h(\boldsymbol{X}^k, t), \ldots, \boldsymbol{x}^k_m - s_k \nabla_{\boldsymbol{x}_m} h(\boldsymbol{X}^k, t)]$$

is computed where s_k denotes a step length. The iteration continues until it satisfies one of the stopping conditions:

$$|h(\boldsymbol{X}^k, t) - h(\boldsymbol{X}^{k+1}, t))|/(1 + |h(\boldsymbol{X}^k, t))| < \epsilon \tag{8}$$

for a given tolerance ϵ, or the number of iterations exceeds the given maximum number of iterations.

To implement the continuation method for (7) with the gradient method, an initial guess for $\boldsymbol{X}$ at $t_0 = 0$ $(i = 0)$, denoted as $\boldsymbol{X}_0$, and an initial distance matrix $\boldsymbol{D}^0$ should be first decided. We discuss the computation of the initial guesses in Section 4. Then, the gradient method is applied to the problem

$$\text{minimize } h(\boldsymbol{X}, t_1)$$

with $\boldsymbol{X}_0$. If one of the stopping conditions described in (8) is satisfied, an approximate minimizer $\widetilde{\boldsymbol{X}}_1$ is obtained. Similarly, for each i ($i = 1, \ldots, \tau - 1$), the gradient method is applied to the problem

$$\text{minimize } h(\boldsymbol{X}, t_{i+1}) \tag{9}$$

using $\widetilde{\boldsymbol{X}}_i$ as the initial matrix. We then obtain the approximate minimizer $\widetilde{\boldsymbol{X}}_{i+1}$ from (9).

Since the continuation method does not include a predictor scheme, an approximate minimizer $\widetilde{\boldsymbol{X}}_i$ at t_i is directly used as the initial point for the corrector to compute $\boldsymbol{X}_{i+1}$ at t_{i+1}, the gradient method. The trajectory of the approximate local minimizers of $h(\boldsymbol{X}, t)$ for $0 \leq t \leq 1$ may not exist, and even when it exists, it may not move forward in the direction t, creating a jump in the values of $\widetilde{\boldsymbol{X}}_i$ and $\widetilde{\boldsymbol{X}}_{i+1}$ for some i ($0 \leq i < \tau$). However, in our numerical experiments shown in Section 5, we have not encountered this situation, and have successfully found $\boldsymbol{X}_\tau$.

4 The Algorithm and Computational Issues

The algorithm of the proposed continuation method for (7) consists of the following steps. An initial guess $\boldsymbol{X}_0^0$ for $\boldsymbol{X}$ denotes the initial matrix for the continuation method and the gradient method at the start of the continuation method. The algorithm is described in a way that the continuation from $t = 0$ to $t = 1$ can be applied more that once.

Algorithm 4.1.

Step 1. Take an initial guess $\boldsymbol{X}_0^0$ and $\boldsymbol{D}^0$ for $\boldsymbol{D}$ to start the continuation method. Set $t_0 = 0$. Choose $\tau \geq 1$ and compute the step size $\Delta t = 1/\tau$. Decide the maximum number of the outer iteration (maxIt), and a tolerance ϵ for the gradient method. Set OuterIteration = 1 and $t_1 = \Delta t$.

Step 2. Inner iteration:

For $i = 0, \ldots, \tau - 1$

a. Compute $\widehat{\boldsymbol{D}}$ by (6) at t_{i+1}.

b. Apply the gradient method to (9) with ϵ and $\boldsymbol{X}_i^0$ to obtain $\widetilde{\boldsymbol{X}}_{i+1}$.

c. $\boldsymbol{X}_{i+1}^0 \leftarrow \widetilde{\boldsymbol{X}}_{i+1}$.

d. $t_{i+2} \leftarrow t_{i+1} + \Delta t$.

end

Step 3. If OuterIteration $\geq$ maxIt, then stop.

Step 4. $\boldsymbol{X}_0^0 \leftarrow \widetilde{\boldsymbol{X}}_\tau$, and determine $\boldsymbol{D}^0$ for OuterIteration+1 and Δt. Set $t_0 = 0$ and $t_1 = \Delta t$.

OuterIteration $\leftarrow$ OuterIteration +1. Repeat from Step 2.

4.1 Iterative refinements

The continuation method described in Algorithm 4.1 can be used iteratively to refine the approximate solution by taking the outer iteration of Steps 2–4 until it satisfies a stopping condition, for instance, the difference in $\|\boldsymbol{X}\|$'s from two consecutive outer iterations is smaller than a given tolerance. We note that increasingly larger values of Δt, or even $\Delta t = 1$, can be used from the second outer iteration.

The efficiency of this iterative refinement greatly depends on the maximum number of iterations and the prescribed tolerance for the gradient method during the inner iteration since it is called repeatedly. In the early stage of iterations where initial $\boldsymbol{X}_i^0$ is a very rough approximation, a large value for the maximum number of iterations and a small tolerance for the gradient method can very much slow the whole process. In addition, those choices of the values do not always guarantee a very accurate minimizer at the final stage. In the numerical experiments presented in Section 5, we used a smaller number of maximum iteration, e.g., 200 - 300, for the gradient method in the early stage of the continuation method and a larger number, 3000, when t is close to 1. For tolerance, 1.e-4 was used for $0 < t < 1$ and the values of 1.e-8 to 1.e-12 were used when t is close to 1 in the experiments for Section 5. We compare two values of tolerance for $0 < t < 1$ in the subsequent section.

4.2 Construction of initial $\boldsymbol{X}_0^0$

Initial matrices for $\boldsymbol{D}^0$ and $\boldsymbol{X}_0^0$ can be determined in various ways for the Step 1 of the Algorithm 4.1. We describe three methods, among the methods experimented for the numerical experiments, and compare the results.

The simplest method to choose initial $\boldsymbol{X}_0^0$, not using any information from the given distance matrix $\boldsymbol{D}$, may be generating with random numbers. In this case, $\boldsymbol{D}^0$ can be computed using $\boldsymbol{X}_0^0$. We can not expect much accuracy in the early steps of the continuation method with randomly generated $\boldsymbol{X}_0^0$.

For the second method to compute an initial $\boldsymbol{X}_0^0$, we use the distance information given in $\boldsymbol{D}$. Let $\tilde{\boldsymbol{D}} \in R^{n \times n}$ and its element be defined by

$$\tilde{d}_{pq} \quad = \quad \begin{cases} d_{pq} & \text{if } (1 \leq p < q \leq m) \text{ or } (1 \leq p \leq m,\ m+1 \leq q \leq n) \\ \|\boldsymbol{a}_p - \boldsymbol{a}_q\| & \text{if } (m+1 \leq p,\ q \leq n) \end{cases}$$

Let

$$\boldsymbol{S} = \tilde{\boldsymbol{D}} + \tilde{\boldsymbol{D}}^T + (n+1)\boldsymbol{I}.$$

We apply the symmetric reverse Cuthill-McKee permutation (SYMRCM) to $\boldsymbol{S}$. As a result, an array of indices $\mathcal{I}$ is obtained. $\mathcal{I}(j)$ contains a value from 1 to n for $1 \leq j < n$. We denote $\boldsymbol{X}_0^0(\mathcal{I}(j))$ to indicate the $\mathcal{I}(j)$th sensor in $\boldsymbol{X}_0^0$. The idea of generating $\boldsymbol{X}$ using the SYMRCM is based on that the array $\mathcal{I}$ indicates the sensors located nearby. That is, $\boldsymbol{X}_0^0(\mathcal{I}(j))$ and $\boldsymbol{X}_0^0(\mathcal{I}(j+1))$ for $(1 \leq j < n-1)$ are located nearby. We generate initial points for the locations of sensors $\boldsymbol{X}_0^0$ using random numbers $\boldsymbol{r} = (r_1, \ldots, r_\ell) \in R^\ell$, where each element of $\boldsymbol{r}$ is in the interval $(0, 1)$, and then, arrange the initial points according to $\mathcal{I}$. More precisely, let η be a integer from 0 to $n-1$ and $\boldsymbol{\eta} = (\eta, \ldots, \eta) \in R^{\ell}$, and δ a small number. The elements of $\boldsymbol{X}_0^0$ is computed by

$$\boldsymbol{X}_0^0(\mathcal{I}(k)) = \boldsymbol{\eta}/m + \delta \boldsymbol{r} \quad (\boldsymbol{\eta} = (\eta, \ldots, \eta) \in R^{\ell}, \quad \boldsymbol{\eta} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{m} - \mathbf{1}),$$

and $\boldsymbol{D}^0$ is computed using $\boldsymbol{X}_0^0$.

The third method to obtain an initial $\boldsymbol{X}_0^0$ is applying a local method such as the gradient method to (7). In particular, we set all elements of $\boldsymbol{X}$ to the center point of all anchors, and then apply the gradient method once with this initial $\boldsymbol{X}$. Then, we obtain an approximate minimizer from the gradient method and use the approximate minimizer as the initial guess $\boldsymbol{X}_0^0$, and $\boldsymbol{D}^0$ is computed from $\boldsymbol{X}_0^0$ for the Algorithm 4.1.

The performance of the continuation method may depend on the methods to generate initial matrices $\boldsymbol{X}_0^0$. Table 1 compares the initial matrices generated randomly, by the SYMRCM, and by one application of the gradient method using an initial guess of the center point of the anchors. We used the continuation method for the 2-dimensional test problems with $n = 3,000$, $m_a = 300, 150$, $\rho = 0.1, \sqrt{10/m}$, and noisy factor $\sigma = 0.0, 0.1$, and 0.2. We used $\epsilon = 1.e\text{-}12$ and the maximum number of iterations 250 as the stopping conditions for the gradient method. Details on generating test problems are described in Section 5. Numerical experiments were performed on 2.8GHz Quad-Core Intel Xeon with 4GB memory.

The root mean square distance (RMSD) is computed by

$$\left(\frac{1}{m} \sum_{p=1}^{m} \|\boldsymbol{x}_p - \boldsymbol{a}_p\|^2 \right)^{1/2}, \tag{10}$$

where $\boldsymbol{x}_p$ denotes the computed location of the pth sensor and $\boldsymbol{a}_p$ true location of the pth sensor, to measure the accuracy of computed locations of m sensors.

We observe that the initial matrices obtained by the gradient method lead to slightly more accurate solutions for most of the problems with shorter elapsed time than the initial matrices randomly generated or by SYMCRM. The problems with 5000 sensors are also tested and similar results were obtained. Based on these results, the gradient method was used to generate $\boldsymbol{X}_0^0$ in the subsequent numerical experiments.

4.3 Step size Δt

The continuation method starts with rough estimations of $\boldsymbol{X}$ and $\boldsymbol{D}$ as described in the previous section. Although a small-sized step is necessary to gradually refine the approximations to $\boldsymbol{X}$ during the continuation process, it would be time-consuming as the gradient method needs to be applied whenever t is updated. We have tested whether it is more efficient to solve the problem (7) with a step size $\Delta t < 1$ and $\Delta t = 1$.

The 2-dimensional SNL problems are solved by the continuation method with the initial matrices computed by the third method in Section 4.2 with $\Delta t = 0.01, 0.1$ and 1. Table 2 and Figure 1 show that the continuation method with $\Delta t = 1$ on average results in larger RMSDs than the continuation method with $\Delta t = 0.1$. Similar results are obtained for other test problems shown in Section 5. We also observe that a smaller step size $\Delta t = 0.01$ does not greatly improve the quality of the obtained solution for most of problems, although it takes much longer elapsed time.

Test Problems			Random X_0^0		SYMRCM		Initial Gradient	
			E.Time		E.Time		E.Time	
$m,\ m_a$	ρ	σ	Total	RMSD	Total	RMSD	Total	RMSD
$m = 3000$, m_a of m $= 300$	0.100	0.0	18.3	4.5e-08	22.8	4.6e-08	6.5	1.7e-07
		0.1	18.7	2.2e-03	24.0	2.2e-03	6.1	2.1e-03
		0.2	21.1	3.5e-03	18.7	3.5e-03	5.4	4.2e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	9.5	1.4e-08	9.2	1.9e-08	8.0	1.9e-07
		0.1	8.8	1.7e-03	10.8	1.7e-03	8.4	1.7e-03
		0.2	8.7	3.5e-03	8.9	3.5e-03	8.2	3.5e-03
$m = 3000$, $m_a = 150$	0.100	0.0	23.8	7.5e-03	19.6	7.5e-03	9.0	1.7e-07
		0.1	22.8	8.3e-03	18.1	7.6e-03	8.6	2.2e-03
		0.2	21.6	8.7e-03	25.5	3.8e-03	8.1	4.4e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	29.0	1.0e-06	26.9	4.2e-07	19.5	4.0e-07
		0.1	13.2	1.3e-02	15.0	2.9e-03	11.6	2.7e-03
		0.2	12.0	9.8e-03	14.7	4.8e-03	10.8	4.3e-03

Table 1: Comparison on initial matrices $\boldsymbol{X}_0^0$ generated randomly, by the SYMCRM, and by the gradient method for the continuation method to solve 2-dimensional problems with 3000 sensors. The continuation step 0.1 is used, a tolerance for the gradient is $1.e$-4 during the continuation, $1.e$-12 at the final step $t = 1$. E.Time means "elapsed time".

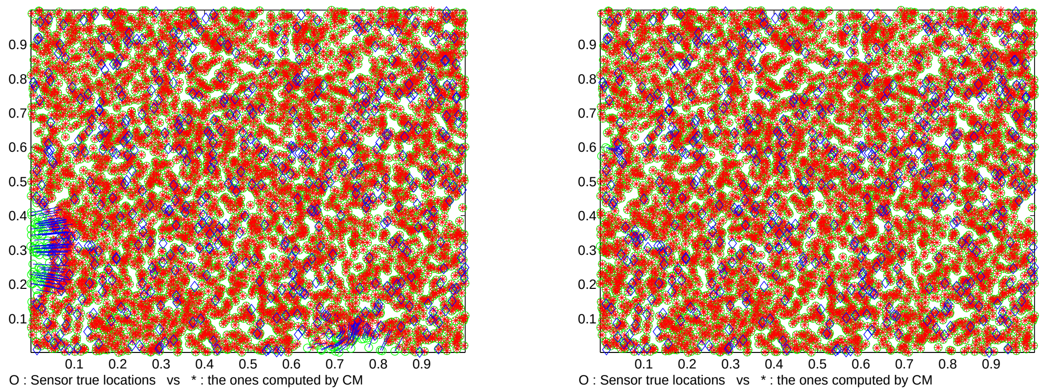

Figure 1: The number of anchors is 5 % of 5000 sensors. The anchors are distributed randomly. The radio range is 0.045, and the noisy factor 0.0. The locations of the sensors obtained with $\Delta t = 1$ on the left and $\Delta t = 0.1$ on the right. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment error between the true and computed location.

Step size			0.01		1		0.1	
Test problems			E.Time		E.Time		E.Time	
$m,\ m_a$	ρ	σ	Total	RMSD	Total	RMSD	Total	RMSD
$m = 3000$, m_a of m $= 300$	0.100	0.0	34.9	7.8e-08	4.5	2.1e-02	6.5	1.7e-07
		0.1	32.5	2.1e-03	5.5	2.0e-02	6.1	2.1e-03
		0.2	29.8	4.6e-03	3.9	2.0e-02	5.4	4.2e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	37.4	1.8e-07	17.8	1.6e-02	8.0	1.9e-07
		0.1	38.4	1.7e-03	15.9	1.4e-02	8.4	1.7e-03
		0.2	37.5	3.4e-03	11.5	1.6e-02	8.2	3.5e-03
$m = 3000$, $m_a = 150$	0.100	0.0	41.5	1.6e-07	7.1	2.6e-02	9.0	1.7e-07
		0.1	37.4	2.2e-03	10.1	3.1e-02	8.6	2.2e-03
		0.2	36.4	4.4e-03	6.4	3.0e-02	8.1	4.4e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	122.7	8.4e-07	16.3	2.8e-02	19.5	4.0e-07
		0.1	52.5	2.7e-03	25.1	2.9e-02	11.6	2.7e-03
		0.2	56.6	4.3e-03	17.9	3.0e-02	10.8	4.3e-03
$m = 5000$, $m_a = 500$ randomly distributed	0.100	0.0	65.8	1.7e-08	5.8	2.0e-08	10.9	1.6e-08
		0.1	58.9	2.1e-03	3.5	2.1e-03	7.8	2.1e-03
		0.2	51.7	4.2e-03	4.5	4.2e-03	8.2	4.2e-03
	$\sqrt{10/m}$ ≈ 0.045	0.0	94.5	9.9e-08	18.6	1.1e-02	14.5	1.1e-07
		0.1	83.3	1.2e-03	31.0	1.2e-02	14.4	1.4e-03
		0.2	81.0	2.5e-03	19.8	1.1e-02	15.4	2.6e-03
$m = 5000$, $m_a = 5\%$ of m $= 250$ randomly distributed	0.100	0.0	68.0	2.6e-08	9.5	8.3e-03	12.9	3.9e-08
		0.1	51.7	2.1e-03	16.1	1.8e-02	12.0	2.1e-03
		0.2	57.9	4.3e-03	19.4	9.2e-03	15.6	4.3e-03
	$\sqrt{10/m}$ ≈ 0.045	0.0	79.6	4.5e-07	32.3	2.1e-02	18.5	4.8e-07
		0.1	82.4	1.4e-03	40.1	2.1e-02	17.6	1.4e-03
		0.2	82.3	2.7e-03	28.5	2.3e-02	18.2	3.0e-03

Table 2: Comparison of the step size to solve 2-dimensional problems. A tolerance for the gradient is $1.e$-4 during the continuation, $1.e$-12 at the final step $t = 1$. E.Time means "elapsed time".

4.4 Stopping conditions for the gradient method

The continuation method presented in this section is based on the repeated use of the gradient method. Thus, it is important to choose appropriate stopping conditions for the gradient method to enhance the overall performance. Recall that the gradient method stops when the number of iterations reaches the maximum number of iterations or the difference between the two objective values of the recent iterations satisfy the condition (8). In the numerical experiments, different values of the maximum number of iterations, if they were larger than, for instance, 500 for $t < 1$, did not yield much difference in the numerical results. However, values for the tolerance did affect the results as shown in Table 3, which compares two choices of the tolerance, 1.e-4 and 1.e-6, for the gradient method in the continuation process for $0 < t < 1$. The test problems are 2-dimensional SNL problems with 3000 and 5000 sensors. At $t = 1$, we set the value of the tolerance to be 1.e-12 for the both cases. We observe that choosing 1.e-6 for the tolerance takes much longer elapsed time than 1.e-4 for similar accuracy.

In the gradient method, a simple step size control is included as described in [5].

4.5 Selecting edges

The minimum number of edges incident to a sensor to determine the locations of all sensors with exact distances is $\ell + 1$. If there exist more edges than $\ell + 1$ for a sensor, some of the edges can be eliminated for computational efficiency. This technique was implemented in SFSDP [12] using the parameter "minDegree" and described in [13]. Consider input sets $\overline{\mathcal{N}}_x \subset \mathcal{N}_x^\rho$ and $\overline{\mathcal{N}}_a \subset \mathcal{N}_a^\rho$. These input sets $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$ can be directly used as $\mathcal{N}_x$ and $\mathcal{N}_a$ in (2). Then, the number of elements of $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$ is the number of distance equations in (2). Alternatively, if subsets of $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$ are used in (2), the number of distance equations is reduced. Choosing subsets of $\overline{\mathcal{N}}_x$ and $\overline{\mathcal{N}}_a$ has been implemented using the minimum degree denoted by "minDegree". In general, more accurate locations of sensors are obtained in longer computational time as the sizes of $\mathcal{N}_x \subset \overline{\mathcal{N}}_x$ and $\mathcal{N}_a \subset \overline{\mathcal{N}}_a$ increase. For details, we refer to [13].

5 Numerical Experiments

We compare the continuation method with SFSDP using SDPA [18] that was shown to be more efficient than ESDP in [13] and SFSDP using SeDuMi [17]. SNL problems in 2- and 3-dimensions are tested with initial matrices generated by the third method in Section 4. We show that the continuation method works efficiently than SFSDP for most of test problems, except for the problems with a very small number of anchors. The accuracy is comparable to that of SFSDP. We show that one of the advantages of the continuation method is that it can handle larger-sized problems than the methods based on the SDP relaxation.

Numerical experiments were performed on 2.8GHz Quad-Core Intel Xeon with 4GB memory. We note that the experiments on very large-sized problems in [13] were performed on 2.8GHz Quad-Core Intel Core i7 with 16GB memory. In this paper, we only use the machine with 4GB memory. All programs were implemented in Matlab and matrix multiplications in the gradient method were executed by C++ routines.

Test problems $\setminus$ ϵ			1.e-6		1.e-4	
$m,\ m_a$	ρ	σ	E.Time	RMSD	E.Time	RMSD
$m = 3000$,	0.100	0.0	12.4	6.1e-08	6.5	1.7e-07
m_a of m		0.1	11.4	2.1e-03	6.1	2.1e-03
$= 300$		0.2	11.1	4.2e-03	5.4	4.2e-03
randomly	$\sqrt{10/m}$	0.0	15.7	1.0e-07	8.0	1.9e-07
distributed	≈ 0.058	0.1	14.9	1.7e-03	8.4	1.7e-03
		0.2	14.6	3.4e-03	8.2	3.5e-03
$m = 3000$,	0.100	0.0	17.0	1.7e-07	9.0	1.7e-07
$m_a = 5\%$ of m		0.1	17.4	2.2e-03	8.6	2.2e-03
$= 150$ distributed		0.2	15.6	4.4e-03	8.1	4.4e-03
randomly	$\sqrt{10/m}$	0.0	28.5	9.9e-07	19.5	4.0e-07
	≈ 0.058	0.1	17.5	2.1e-03	11.6	2.7e-03
		0.2	18.2	4.2e-03	10.8	4.3e-03
$m = 5000$,	0.100	0.0	21.7	1.4e-08	10.9	1.6e-08
$m_a = 500$		0.1	17.5	2.1e-03	7.8	2.1e-03
		0.2	18.2	4.2e-03	8.2	4.2e-03
randomly	$\sqrt{10/m}$	0.0	32.7	1.3e-07	14.5	1.1e-07
distributed	≈ 0.045	0.1	32.8	1.4e-03	14.4	1.4e-03
		0.2	34.4	2.5e-03	15.4	2.6e-03
$m = 5000$,	0.100	0.0	24.7	1.1e-07	12.9	3.9e-08
$m_a = 5\%$ of m		0.1	27.7	2.1e-03	12.0	2.1e-03
$= 250$ distributed		0.2	30.5	4.3e-03	15.6	4.3e-03
randomly	$\sqrt{10/m}$	0.0	44.5	5.1e-07	18.5	4.8e-07
	≈ 0.045	0.1	41.1	1.4e-03	17.6	1.4e-03
		0.2	29.4	2.7e-03	18.2	3.0e-03

Table 3: Comparison of tolerances for the gradient method in the continuation method to solve 2-dimensional problems with 3000 and 5000 sensors and anchors distributed randomly.

The 2-dimensional test problems were generated with sensors and anchors distributed randomly in $[0,1]^2$. The radio range was varied from 0.058 to 0.2 and the noisy factor from 0.0 to 0.2. Two types of the values for the radio ranges were chosen as in [13]. More precisely, the first type is to choose the values of the radio range independent of the number of the sensors. The second type is to choose the value of the radio range so that each square of size ρ in $[0,1]^2$ contains on average 10 randomly generated sensors.

The 3-dimensional test problems were generated randomly with 3000 to 10000 sensors in $[0,1]^3$ with two kinds of radio ranges, independent and dependent on the number of the sensors so that each cube of size ρ in $[0,1]^3$ contains on average 15 randomly generated sensors.

With the noisy factor changing from 0.0 to 0.2, the distances were perturbed to create problems with noise:

$$\begin{array}{l} \bar{d}_{pq} = \max\{(1+\sigma\delta_{pq}),\ 0.1\}d_{pq}\ ((p,q) \in \mathcal{N}_x^{\rho}), \\ \bar{d}_{pr} = \max\{(1+\sigma\delta_{pr}),\ 0.1\}d_{pr}\ ((p,r) \in \mathcal{N}_a^{\rho}), \end{array} \tag{11}$$

where $\sigma \geq 0$ denotes noisy factor, and δ_{pq} and δ_{pr} are chosen from the standard normal distribution $N(0,1)$, and d_{pq} and d_{pr} indicate the exact distances in (1), i.e., $\epsilon'_{pq} = \epsilon'_{pr} = 0$. As in [2, 3, 4, 20, 22], the root mean square distance (RMSD) defined in (10) is used to measure the accuracy of locations of m sensors computed by the continuation method and SFSDP.

In the description of the numerical results, "CM" means the continuation method, "SYMRCM" the symmetric reverse Cuthill-McKee permutation, and "E.Time" elapsed time. For SFSDP, total E.Time indicates the elapsed time for generating SDP relaxation, solving SDP using SDPA, and refining the approximated solution from SDPA by the gradient method. When SFSDP was implemented, the value 4 and 5 were used for the minimum degree of a sensor node of 2-dimensional and 3-dimensional problems, respectively. For details on the minimum degree, we refer to [13].

5.1 Two-dimensional Problems

To show how the continuation method gradually finds the solution, we first display the computed solution by the continuation method when $t = 0.1$, $t = 0.7$ in Figure 2, and $t = 1$ in Figure 3 for the problem with 5000 sensors, 250 anchors, noisy factor 0.1, and radio range 0,1. We observe that the approximate solution is computed with accuracy as t approaches to 1.

In Tables 4, we compare the numerical results by the continuation method with SFSDP. It shows that the continuation method obtains more accurate solutions than SFSDP for all test problems, taking shorter elapsed time except for some problems with exact distances.

The continuation method can solve the SNL problems with 20,000 sensors as shown in Table 5, which could not be handled by SFSDP due to out-of-memory error using the same machine with 4GB memory.

As discussed in Section 4, the problems with a small number of anchors are sometimes difficult to deal with for lack of distance information. Table 6 displays the numerical results for the 2-dimensional problems with 4 anchors placed at the corners. We observe that the continuation method does not perform well for the problems with a very small number of anchors. This is because the distance information between the anchors and the sensors is scarce to proceed with the continuation method.

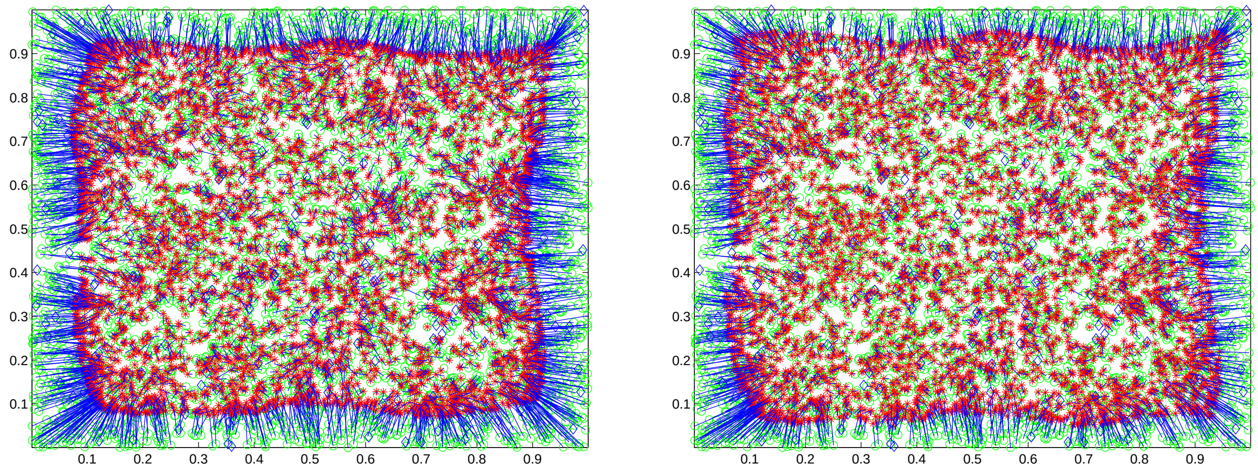

Figure 2: The number of anchors is 5 % of the number of sensors. The anchors are distributed randomly. The radio range is 0.1 and the noisy factor is 0.1. The locations of the sensors at $t = 0.1$ on the left and $t = 0.7$ on the right. A green circle denotes the true location of a sensor, a red $\star$ the computed location of a sensor, a blue diamond an anchor, and a blue line segment error between true and computed location.

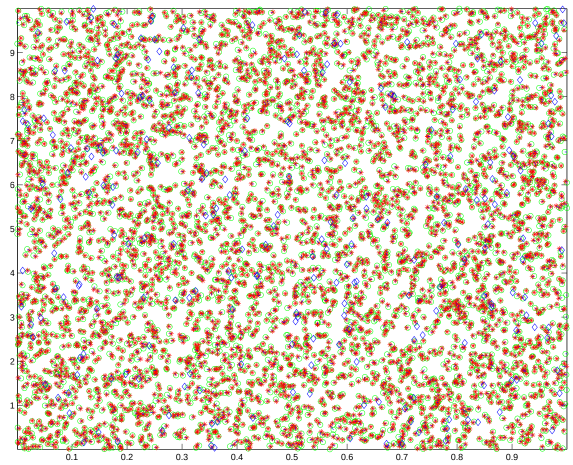

Figure 3: The locations of the sensors at $t = 1$.

Test problems			RMSD	E.time		RMSD
$m,\ m_a$	ρ	σ	SFSDP		CM	
$m = 3000$, m_a of m $= 300$ randomly distributed	0.100	0.0	2.3e-7	36.9	6.5	1.7e-07
		0.1	2.3e-3	95.8	6.1	2.1e-03
		0.2	4.8e-3	98.8	5.4	4.2e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	4.0e-6	43.5	8.0	1.9e-07
		0.1	1.9e-3	88.2	8.4	1.7e-03
		0.2	4.4e-3	88.9	8.2	3.5e-03
$m = 3000$, $m_a = 5\%$ of m $= 150$ distributed randomly	0.100	0.0	1.3e-6	37.9	9.0	1.7e-07
		0.1	2.5e-3	88.6	8.6	2.2e-03
		0.2	5.2e-3	91.8	8.1	4.4e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	9.5e-6	60.8	19.5	4.0e-07
		0.1	4.5e-3	112.0	11.6	2.7e-03
		0.2	5.8e-3	115.7	10.8	4.3e-03
$m = 5000$, $m_a = 500$ randomly distributed	0.100	0.0	2.7e-7	94.7	10.9	1.6e-08
		0.1	2.2e-3	244.7	7.8	2.1e-03
		0.2	4.5e-3	241.4	8.2	4.2e-03
	$\sqrt{10/m}$ ≈ 0.045	0.0	4.4e-6	111.8	14.5	1.1e-07
		0.1	3.4e-3	219.1	14.4	1.4e-03
		0.2	4.7e-3	230.9	15.4	2.6e-03
$m = 5000$, $m_a = 5\%$ of m $= 250$ distributed randomly	0.100	0.0	5.9e-7	93.6	12.9	3.9e-08
		0.1	2.5e-3	229.8	12.0	2.1e-03
		0.2	4.8e-3	231.3	15.6	4.3e-03
	$\sqrt{10/m}$ ≈ 0.045	0.0	1.5e-4	156.1	18.5	4.8e-07
		0.1	5.1e-3	283.0	17.6	1.4e-03
		0.2	6.3e-3	281.7	18.2	3.0e-03

Table 4: Comparison between the continuation method and SFSDP with SDPA to solve 2-dimensional problems with 3000 and 5000 sensors and anchors distributed randomly. Initial $\boldsymbol{X}^0$ is generated by applying the gradient method once. The continuation step is 0.1, the tolerance for the gradient is 1.e-4 during the continuation, and 1.e-12 at $t = 1$. The value of minDegree is 4 for SFSDP and 40 for CM.

Test problems			CM	
$m,\ m_a$	ρ	σ	E.Time	RMSD
$m_a = 10\%$ of m $= 2000$ distributed randomly	0.1	0.0	161.6	5.8e-08
		0.1	75.4	1.9e-03
		0.2	53.5	3.8e-03
	0.022	0.0	25.5	1.9e-07
		0.1	81.5	6.3e-04
		0.2	86.0	1.3e-03
$m_a = 5\%$ of m $= 1000$ distributed randomly	0.1	0.0	125.7	5.8e-08
		0.1	53.7	1.9e-03
		0.2	50.7	3.8e-03
	0.022	0.0	92.6	2.1e-03
		0.1	94.4	2.4e-03
		0.2	96.1	2.4e-03

Table 5: The continuation method to solve 2-dimensional problems with 20000 sensors. Initial $\boldsymbol{X}^0$ is generated by applying the gradient method once. The continuation step is 0.1, the tolerance for the gradient is 1.e-4 during the continuation, and 1.e-12 at $t = 1$. The value of minDegree is 40.

Test problems			RMSD	E.time		RMSD
$m,\ m_a$	ρ	σ	SFSDP		CM	
$m = 3000$, $m_a = 4$ at corners	0.100	0.0	6.7e-6	74.2	286.3	7.5e-04
		0.1	5.6e-3	177.9	266.4	3.7e-03
		0.2	9.3e-3	176.7	237.8	7.1e-03
	$\sqrt{10/m}$ ≈ 0.058	0.0	7.6e-5	206.6	97.3	6.7e-02
		0.1	7.1e-3	297.2	85.4	5.7e-02
		0.2	1.1e-2	312.4	98.3	7.5e-02

Table 6: Comparison between the continuation method and SFSDP with SDPA to solve 2-dimensional problems with 3000 sensors and 4 anchors. Initial $\boldsymbol{X}^0$ is generated by applying the gradient method once. The continuation step is 0.1, the tolerance for the gradient is 1.e-4 during the continuation, and 1.e-12 at $t = 1$. The value for “minDegree” for CM is 70.

5.2 Three-dimensional Problems

We solved the 3-dimensional test problems in [13] to compare the performance of the continuation method with SFSDP. The numerical results in [13] are shown in Table 7 for the problems with randomly distributed anchors. We see that CM obtains the solutions much faster than SFSDP and the accuracy of the approximate solutions obtained by the continuation method is higher than that by SFSDP in all test problems. Note that the continuation method could solve problems with 5000 sensors, 0.144 radio range while SFSDP failed to solve those due to out-of-memory error. We confirm that one of the advantages of the continuation method is less memory requirement.

Test problems			RMSD	E.time		RMSD
$m,\ m_a$	ρ	σ	SFSDP		CM	
$m = 3000$,	0.250	0.0	9.8e-07	48.1	52.6	3.2e-07
$m_a = 10\%$ of m		0.1	9.5e-03	138.2	12.5	7.5e-03
$= 300$		0.2	1.7e-02	144.7	12.6	1.5e-02
distributed	$(15/m)^{1/3}$	0.0	2.9e-06	87.3	25.8	6.2e-07
randomly	≈ 0.171	0.1	7.0e-03	164.4	18.3	6.2e-03
		0.2	1.6e-02	162.0	17.9	1.2e-02
$m = 3000$,	0.250	0.0	1.2e-06	51.1	18.1	7.6e-08
$m_a = 5\%$ of m		0.1	1.0e-02	134.2	16.2	7.6e-03
$=150$		0.2	1.9e-02	142.0	16.3	1.5e-02
distributed	$(15/m)^{1/3}$	0.0	6.3e-06	301.0	40.5	9.9e-07
randomly	≈ 0.171	0.1	1.8e-02	312.5	29.8	6.5e-03
		0.2	2.7e-02	315.8	36.7	1.4e-02
$m = 5000$,	0.250	0.0	4.2e-07	112.2	31.6	1.0e-07
$m_a = 10\%$ of m		0.1	7.9e-03	337.8	15.3	7.5e-03
$=500$		0.2	1.6e-02	331.8	13.8	1.5e-02
distributed	$(15/m)^{1/3}$	0.0	2.1e-06	295.4	34.1	9.2e-08
randomly	≈ 0.144	0.1	5.8e-03	445.8	24.5	5.2e-03
		0.2	1.2e-02	452.2	23.8	1.0e-02
$m = 5000$,	0.250	0.0	7.7e-07	117.3	52.0	5.7e-08
$m_a = 5\%$ of m		0.1	8.3e-03	337.8	22.7	7.4e-03
$= 250$		0.2	1.7e-02	348.9	23.8	1.5e-02
distributed	$(15/m)^{1/3}$	0.0			34.5	1.2e-07
randomly	≈ 0.144	0.1	Out-of-memory		27.8	5.3e-03
		0.2			28.1	1.1e-02

Table 7: Numerical comparison between the continuation method and SFSDP with SDPA to solve 3-dimensional problems. The value of minDegree for SFSDP and CM is 5 and 50, respectively.

Test results for the problems with 10000 sensors are displayed in Table 8. It shows that the continuation method is successful in solving 3-dimensional large-sized problems without taking long elapsed time.

Figures 4 and 5 exhibit how the continuation method works as t varies from 0 to 1 for 3-

Test problems			CM	
			E.time	
$m,\ m_a$	ρ	σ	Total	RMSD
$m_a = 10\%$ of m	0.25	0.0	48.6	1.6e-07
$= 1000$ distributed		0.1	33.9	7.4e-03
randomly		0.2	24.7	1.5e-02
$m_a = 5\%$ of m	0.25	0.0	75.2	1.1e-07
$= 500$ distributed		0.1	69.4	7.6e-03
randomly		0.2	57.1	1.5e-02
$m_a = 1\%$ of m	0.25	0.0	37.8	1.1e-06
$= 100$ distributed		0.1	55.4	7.4e-03
randomly		0.2	31.3	1.5e-02
$m_a = 10\%$ of m	0.171	0.0	50.7	2.7e-08
$= 1000$ distributed		0.1	34.5	5.1e-03
randomly		0.2	25.5	1.0e-02
$m_a = 5\%$ of m	0.171	0.0	101.2	3.7e-07
$= 500$ distributed		0.1	96.9	5.4e-03
randomly		0.2	88.6	1.1e-02
$m_a = 1\%$ of m	0.171	0.0	66.2	5.2e-08
$= 100$ distributed		0.1	49.3	5.2e-03
randomly		0.2	65.2	1.0e-02

Table 8: The continuation method to solve 3-dimensional problems with 10000 sensors. The prescribed tolerance for the gradient method is 1.e-4 during the continuation, and 1.e-12 at $t = 1$. The value of minDegree is 50.

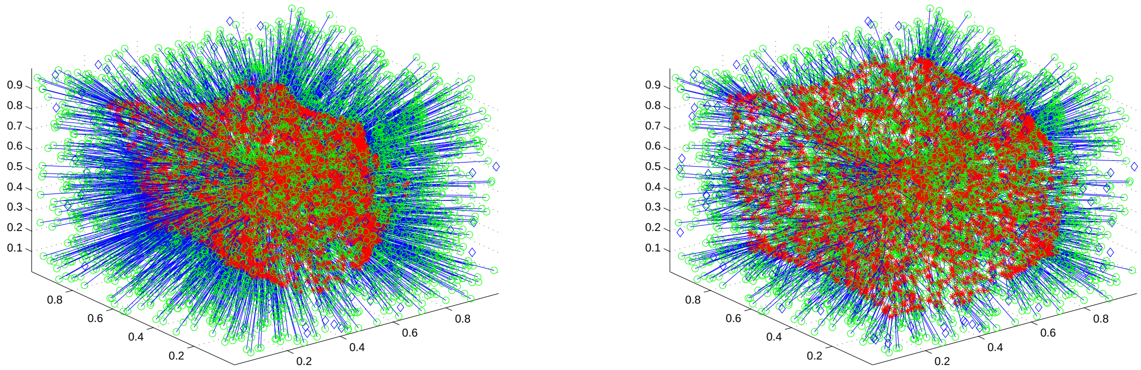

Figure 4: A 3-dimensional problem with 5000 sensors, $\rho = 0.25$, and $\sigma = 0.1$. The locations of sensors at $t = 0.1$ on the left and $t = 0.7$ on the right. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment error between the true and computed location.

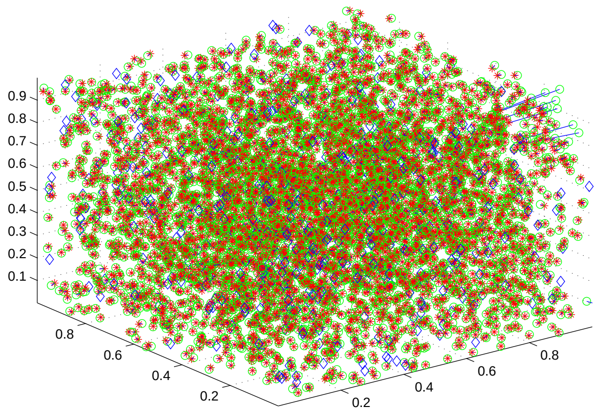

Figure 5: A 3-dimensional problem with 5000 sensors, $\rho = 0.25$, and $\sigma = 0.1$ at $t = 1$. A circle denotes the true location of a sensor, $\star$ the computed location of a sensor, and a line segment between the true and computed location.

dimensional problems.

6 Concluding Remarks

We have proposed the continuation method for the SNL problems to efficiently solve larger-sized SNL problems than the methods based on the SDP relaxation. The SNL problem has been formulated as an unconstrained problem and solved by the continuation method. In this framework of the continuation method, the gradient method is repeatedly applied by changing the distance matrix using the continuation parameter t from 0 to 1.

For the SNL problems, the continuation method performs more efficiently than SFSDP with SDPA, which was shown to be faster than available methods, as shown in Section 5, except for the problems with a very few anchors. The accuracy obtained by the continuation method is higher than that by SFSDP for the most of the tested problems with more anchors than 4. In particular, the memory requirement is smaller than SFSDP, as a result, much larger-sized problems can be solved.

If the SNL problem has a very few anchors, then the accuracy of the solutions obtained by the continuation method was not as good as that by SFSDP. For this kind of the SNL problems, the locations of the sensors that satisfy the distance equations with high accuracy can be used as additional anchors. More specifically, we regard those sensors as anchors, change the number of sensors and anchors, and modify the distance matrix $\boldsymbol{D}$. This will be studied in the future.

Acknowledgments

The authors would like to thank Professor Kim Chuan Toh for Matlab programs procruste.m and gradient.m, and Profession Makoto Yamashita for the c++ rountines for matrix mulplication.

References

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

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

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

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

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

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

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

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

[9] T. Gunji, S. Kim, M. Kojima, A. Takeda, K. Fujisawa and T. Mizutani. PHoM – a Polyhedral homotopy continuation method for polynomial systems, *Computing,* **73** (2004) 55–77.

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

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

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

[13] S. Kim, M. Kojima, H. Waki, and M. Yamashita. SFSDP: a Sparse Version of Full SemiDefinite Programming Relaxataion for Sensor Network Localization Problems , Research Report B-457, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, Japan, 2009.

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

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

[16] T. K. Pong and P. Tseng. (Robust) Edge-Based Semidefinite Programming Relaxation of Sensor Network Localization, to appear in Math. Program. 2011.

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

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

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

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

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

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