Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2012, Article ID 927031, 18 pages
doi:10.1155/2012/927031

Research Article

Geometric Buildup Algorithms for Sensor Network Localization

Zhenzhen Zheng,[1] Xinlong Luo,[2] and Zhijun Wu[3]

[1] *Department of Mathematics, University of California, Irvine, CA 92697, USA*
[2] *School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[3] *Department of Mathematics, Iowa State University, Ames, IA 50011, USA*

Correspondence should be addressed to Zhijun Wu, zhijun@iastate.edu

Received 6 June 2011; Revised 22 August 2011; Accepted 22 August 2011

Academic Editor: Jinling Liang

We present a geometric buildup algorithm for solving the sensor network localization problem with either accurate or noisy distance data. The algorithm determines the locations of the sensors, one at a time, by using the distances between the determined sensors and the undetermined ones. Each time, only a small system of distance equations needs to be solved and therefore, in an ideal case when the required distances are available for every sensor to be determined, the computation can be completed in n steps if n sensors are to be determined. An algorithm with two buildup phases is also implemented to handle not only noisy but also sparse distance data with for example only a few distant anchors. We show our test results and compare them with other approaches.

1. Introduction

Ad hoc wireless sensor networks consist of a large number of spatially distributed sensor nodes that communicate with their nearby sensors within a radio range [1, 2]. The sensor data are relevant only if the sensors' locations are known. The expensive GPS (Global Positioning System) may locate only some of the sensors (called anchors). Most sensors can be located by means of some distance information obtained from the radio signals that the sensors receive [1]. The problem of finding the locations of the sensors given a few anchors and some local distance information among the sensors is called the sensor network localization problem.

Sensor network localization has been applied to many application fields, including environmental studies such as monitoring environmental conditions [3, 4], health cares such as patient tracking [5], and military applications such, as battlefield surveillance [6]. For example, [2, 7] mention that Southern California Edisons Nuclear Generating Station in San Onofre, Calif, USA has deployed wireless mesh-networked sensors from Dust Networks Inc.

to obtain real-time trend data. These data are used to predict which motors are about to fail, so they could be preemptively rebuilt or replaced during scheduled maintenance periods. Implementation of a sensor localization algorithm could provide a service that eliminates the need to record every sensor's location and its associated ID number in the network [2].

Mathematically, the sensor network localization problem can be described as follows. Assume that there are m anchors whose locations denoted by $a_k \in \mathcal{R}^2$, $k = 1, \ldots, m$, are known and $n - m$ sensors whose locations denoted by $x_j \in \mathcal{R}^2$, $j = m + 1, \ldots, n$ need to be decided. For a pair of sensors x_i and x_j, let their Euclidean distance be denoted by $d_{i,j}$. Similarly, let $d_{i,k}$ denote the Euclidean distance between an anchor a_k and a sensor x_i. Let N_x and N_a be two sets of node pairs,

$$\begin{aligned} N_x &= \{(i,j) : \|x_i - x_j\| = d_{i,j} < \mathrm{rd}\}, \\ N_a &= \{(i,k) : \|x_i - a_k\| = d_{i,k} < \mathrm{rd}\}, \end{aligned} \tag{1.1}$$

where $\|\cdot\|$ represents the Euclidean norm and rd is a fixed parameter called radio range. The sensor network localization problem can be formulated as a 2D distance geometry problem, to find the vectors $x_i \in \mathcal{R}^2$ for all $i = m + 1, \ldots, n$ such that

$$\begin{aligned} \|x_i - x_j\|^2 &= d_{i,j}^2 \quad \forall (i,j) \in N_x, \\ \|x_i - a_k\|^2 &= d_{i,k}^2 \quad \forall (i,k) \in N_a. \end{aligned} \tag{1.2}$$

In practice, the distance information is available only if the distance of two nodes is within a certain radio range. Therefore, the available distances are usually sparse, that is, only a small subset of all distances among the nodes is available. The distance data contains errors as well due to the accuracy of the measurements, the power of the sensors, and some other environmental factors. A distance geometry problem with sparse and inexact distances has been proved to be difficult to solve in general [8–10].

More specifically, if exact distances for all pairs of nodes are available, a distance geometry problem can be solved in polynomial time by using for example a singular value decomposition algorithm in $O(n^2)$ [9] or a geometric buildup algorithm in $O(n)$ [11], where n is the number of nodes to be determined and $O(\cdot)$ is the conventional expression for time complexity. If only a sparse set of distances is given, the problem is NP-hard in general [8], even if small distance errors are allowed [10].

Much work has been done on the sensor network localization problem. Biswas et al. [1, 12] proposed an SDP (semidefinite programming) approach to the problem. Wang et al. [13] made further SDP relaxations and developed NSDP (node-based SDP) and ESDP (edge-based SDP) algorithms for large-scale applications. Nie [14] presented an SOS (sum of squares) approach by formulating the problem as a minimization problem for a sum of squares. Tseng [15] developed an SOCP (second-order cone programming) relaxation method. Recent works also include Carter et al. [2], the SFSDP (sparse variant of FSDP [1]) approach by Kim et al. [16], Zhu et al. [17], the LPCGD (log-barrier penalty coordinate gradient descent) approach by Pong and Tseng [18], and the SNLSDPclique approach by Krislock and Wolkowicz [19, 20].

We investigate the solution of the sensor network localization problem within a geometric buildup framework. A basic geometric buildup algorithm [11] was proposed for the

solution of a 3D distance geometry problem with exact distances. The work was later extended to problems with sparse distances [21–23], inexact distances [24], and distance bounds [25]. All these works were applied to protein structure determination problems, which are to determine a set of points given a set of distances between the points. The idea of geometric buildup is to determine the points, one at a time, using the distances between the determined points and the undetermined ones. The algorithm can be applied to distance geometry problems in any finite dimensional Euclidean space including the sensor network localization in 2D.

In every buildup step, for determining a point, a small system of distance equations needs to be solved. However, the point may be overdetermined for there may be more equations than the coordinates for the point. On the other hand, the distances may have errors and the equations are most likely to be inconsistent. To overcome this difficulty, a least-squares approximation method has been developed for solving the distance equations [24]. The method employs a low-rank matrix approximation scheme [26], which requires the singular value decomposition for a small distance matrix. Here, our algorithm is different from [24] in the approximation scheme. We implement a low-rank positive semidefinite approximation scheme [27], which requires a spectral decomposition and guarantees a best-possible approximation to the solution of the distance equations in a least-squares sense.

The availability of an initial set of determined points can be important for a geometric buildup algorithm to start and succeed. Fortunately, for sensor network localization, there is usually a set of anchor nodes that can be used as an initial set of nodes. However, if there are only a few anchors scattered widely in space, the chance is that there may not be many sensors with (at least three) distances to these anchors and the algorithm may be able to determine only a small set of sensors. For this reason, we have also implemented a two-phase buildup algorithm. In the first phase, we determine as many sensors as possible starting with the initial anchors. In the second phase, from the undermined sensors, if there are any, we find a clique of sensors, that is, a subset of sensors with all distances among them available. We position the sensors in the clique in space (which is possible using their distances) and start from them to determine the remaining sensors. In this way, we can have more sensors determined than a single phase algorithm.

This paper is organized as follows. In Section 2, we present several possible versions of geometric buildup algorithms for sensor network localization. In Section 3, we present the numerical results for a set of simulated sensor network localization problems with exact and inexact sparse distances. We conclude the paper and make some remarks in Section 4.

2. The Geometric Buildup Algorithms

In this section, we present three versions of geometric buildup algorithms for sensor network localization. Three cases of problems are concerned (1) when exact distances are available, (2) when distances have errors, and (3) when there are only a few distant anchors. A geometric buildup algorithm, named as basic, extended, and two-phase geometric buildup algorithms, respectively, is described for each of the three cases.

2.1. The Basic Geometric Buildup Algorithm

When a set of exact distances is given, that is, the distances in (1.1) is accurate, a basic geometric buildup algorithm can be applied. The algorithm works as follows. It first takes the anchors as the initial set of determined sensors. Then, for any undetermined sensor j, it

Input: The positions of the anchors, the distances $d_{i,j}$, $(i,j) \in N_x \cup N_a$.
Output: The positions of a set of determined sensors.
Step 1: Let the anchors be the initial set of determined sensors.
Step 2: **Repeat:**
For each undetermined sensor j:
If a basis set of determined sensors is found for sensor j, determine sensor j by solving the linear system (2.2).
End
End
If no sensor can be determined in the loop, stop.
If all sensors are determined, stop.

Algorithm 1: The basic geometric buildup algorithm.

tries to find three determined sensors that are not collinear but have distances to sensor j. We call these three determined sensors a basis set of sensors for sensor j. Let $x_i = (x_{i,1}, x_{i,2})^T$ $(i = 1,2,3)$ be the coordinate vectors of the three determined sensors. Given the distances $d_{i,j}$ $(i = 1,2,3)$, the coordinate vector $x_j = (x_{j,1}, x_{j,2})^T$ of sensor j can be determined by using the following system of equations:

$$\|x_i\|^2 - 2x_i^T x_j + \|x_j\|^2 = d_{i,j}^2, \quad i = 1,2,3. \tag{2.1}$$

Subtracting equation i from equation $i+1$ $(i = 1,2)$ results in a linear system of equations

$$Ax_j = b, \tag{2.2}$$

where

$$A = -2\begin{bmatrix}(x_2 - x_1)^T\\ (x_3 - x_2)^T\end{bmatrix}, \qquad b = \begin{bmatrix}\left(d_{2,j}^2 - d_{1,j}^2\right) - \left(\|x_2\|^2 - \|x_1\|^2\right)\\ \left(d_{3,j}^2 - d_{2,j}^2\right) - \left(\|x_3\|^2 - \|x_2\|^2\right)\end{bmatrix}. \tag{2.3}$$

The points x_1, x_2, x_3 are not collinear, so the coefficient matrix A is nonsingular and the linear system (2.2) has a unique solution. An outline of the basic geometric buildup algorithm is given in Algorithm 1. Note that if there are at least three anchors in the network and the distances are exact, and if in every step, an undetermined sensor and a basis set of determined sensors associated with it can be found, then the basic geometric buildup algorithm can solve the problem in $O(n)$ computation time, where n is the total number of sensors to be determined.

2.2. The Extended Geometric Buildup Algorithm

Note that for every undetermined sensor, there may be more than three determined sensors that have distances to it. Therefore, there may be more than three distance equations in (2.1) for the sensor to satisfy. Of course, if the distances are accurate, or in other words, are exact, only three equations need to be solved as done in the basic algorithm, while all other

equations are satisfied automatically. However, in practice, the distances may have errors and therefore, the distance equations may not be consistent, and satisfying three of them does not necessarily satisfy all the equations. Besides, (2.1) cannot be reduced to (2.2) any more because there may not be a solution to the equations in (2.1) or, in other words, the equations may never hold and hence cannot add or subtract. The extended geometric buildup algorithm is developed to overcome these difficulties.

An extended geometric buildup algorithm works as follows. It again takes the anchors as the initial set of determined sensor. Then, for each undetermined sensor, it finds all the determined sensors that have distances to the undetermined sensor. If there are at least three such determined sensors and if they are not collinear, then the algorithm tries to find the coordinates for the undetermined sensor by solving a system of distance equations corresponding to all these determined sensors. In particular, since the equations may not be consistent, they are solved approximately in a least-squares sense as described in the following.

Let $x_{\ell+1} = (x_{\ell+1,1}, x_{\ell+1,2})^T$ be the coordinate vector of the sensor to be determined. Let $x_i = (x_{i,1}, x_{i,2})^T$ $(i = 1, \ldots, \ell)$ be the coordinate vectors of the determined sensors to be used for the determination of $x_{\ell+1}$. Let $d_{i,\ell+1}$ be the distances from x_i to $x_{\ell+1}$, $i = 1, \ldots, \ell$. Then the distance equations to be solved are

$$\|x_i\|^2 - 2x_i^T x_{\ell+1} + \|x_{\ell+1}\|^2 = d_{i,\ell+1}^2, \quad i = 1, \ldots, \ell. \tag{2.4}$$

The equations can be solved by using for example a standard nonlinear least-squares method, but we implement a method similar to that proposed in [24] and obtain a more direct solution to the equations. Instead of working on the system in (2.4), we expand it to the following system:

$$\|x_i\|^2 - 2x_i^T x_j + \|x_j\|^2 = d_{i,j}^2, \quad i, j = 1, \ldots, \ell + 1. \tag{2.5}$$

Note that $d_{i,j}, i, j = 1, \ldots, \ell$ in the added equations may or may not be available in the given distance data, but they can be computed if some of them are not because x_i, $i = 1, \ldots, \ell$ are already known. We then consider all x_i, $i = 1, \ldots, \ell + 1$ as unknowns and determine them all by solving the system of equations in (2.5). Since the relative positions of these sensors are invariant under any translation and orthogonal transformation, we can set a reference system so that the sensor to be determined is located at the origin or, in other words, $x_{\ell+1} = (0, 0)^T$. It follows that $\|x_i\| = d_{i,\ell+1}, \|x_j\| = d_{j,\ell+1}$ and

$$d_{i,\ell+1}^2 - 2x_i^T x_j + d_{j,\ell+1}^2 = d_{i,j}^2, \quad i, j = 1, \ldots, \ell. \tag{2.6}$$

Define a coordinate matrix X and an induced distance matrix D as follows:

$$\begin{aligned} X &= \{x_{i,k} : i = 1, \ldots, \ell, k = 1, 2\}, \\ D &= \left\{ \frac{\left(d_{i,\ell+1}^2 - d_{i,j}^2 + d_{j,\ell+1}^2\right)}{2} : i, j = 1, \ldots, \ell \right\}. \end{aligned} \tag{2.7}$$

It is easy to verify that $XX^T = D$, which has been widely studied in the classical multidimensional scaling or "MDS" [27–29]. If the distances have errors, the system $XX^T = D$ may not be consistent. It is natural to consider a least-squares problem

$$\min_{X \in \mathcal{R}^{\ell \times 2}} \left\| D - XX^T \right\|_F, \tag{2.8}$$

where $\| \cdot \|_F$ is the matrix Frobenius norm. [24] applied the best rank r matrix approximation coming from the classical Eckart-Young Theorem [26], which involves the singular value decomposition. Here, we apply a best positive semidefinite approximation which involves the spectral decomposition. The solution to the problem in (2.8) can be obtained from the following theorem.

Theorem 2.1 (see [19, 27]). *Let $M = \Sigma_{i=1}^{n} \lambda_i u_i u_i^T$ be the spectral decomposition of the symmetric matrix M, where $\lambda_1 \geq \cdots \geq \lambda_n$. Then $\widetilde{M} = \Sigma_{i=1}^{r} \lambda_i^+ u_i u_i^T$, where $\lambda_i^+ = \max\{0, \lambda_i\}$, is the best positive semidefinite approximation to the following problem,*

$$\begin{aligned} \min_{\widetilde{M}} \quad & \left\| M - \widetilde{M} \right\|_F, \\ \text{subj. to} \quad & \operatorname{rank}\left(\widetilde{M}\right) \leq r, \quad \widetilde{M} \succeq 0. \end{aligned} \tag{2.9}$$

Now suppose that the spectral decomposition of D is $U \Lambda U^T$, where the diagonal entries of Λ are in a decreasing order. Let $V = U(:, 1 : 2)$ and Σ be the diagonal matrix with $\Sigma_{ii} = \max\{0, \Lambda_{ii}\}$, where $i = 1, 2$. Then $X = V\Sigma^{1/2}$ solves the problem in (2.8), and the coordinates of all the sensors are obtained, with the sensor to be determined located at $(0, 0)^T$. To obtain the coordinates of this sensor in the original reference system, it can be transformed along with other ℓ sensors so that the recalculated coordinates of those sensors agree with their old ones as much as possible. The latter can be done by minimizing the so-called RMSD (root-mean-square deviation) of the coordinates (details at the end of this subsection).

It seems that the system of equations in (2.4) is simpler and easier to solve than that in (2.5), and the coordinates of x_i, $i = 1, \ldots, \ell$ are also recalculated in (2.5). It turns out that solving (2.5) instead of (2.4) is critical for the stability of the buildup algorithm. The solution to the system in (2.4) depends on previously calculated coordinates x_i, $i = 1, \ldots, \ell$ and, therefore, may inherit errors from previous calculations. If such errors are continuously passed down to later calculations, the buildup algorithm is most likely to end up with an incorrect set of coordinates for the sensors. In contrast, the solution to the system in (2.5) depends only on the distances among the sensors, most of which are given in the original distance data. The recalculation of the coordinates x_i, $i = 1, \ldots, \ell$ also "cutoffs" possible propagations of calculation errors, making the algorithm much more stable [24].

As we have mentioned above, the coordinates of x_i, $i = 1, \ldots, \ell + 1$ are determined in an independent reference system. In order to move the coordinates back to their original reference system, we need to make a proper translation and orthogonal transformation for the coordinates. Let $X \in \mathcal{R}^{\ell \times 2}$ and $Y \in \mathcal{R}^{\ell \times 2}$ be the previously calculated and recalculated

coordinate matrices of the ℓ determined sensors, respectively. We first calculate the geometric centers of X and Y,

$$X_c = \frac{1}{\ell}\sum_{i=1}^{\ell} X(i,:), \qquad Y_c = \frac{1}{\ell}\sum_{i=1}^{\ell} Y(i,:), \tag{2.10}$$

and then update X and Y:

$$\begin{aligned} X &:= X - I_{\ell,1}X_c, \\ Y &:= Y - I_{\ell,1}Y_c, \end{aligned} \tag{2.11}$$

where $I_{\ell,1}$ is an $\ell \times 1$ vector with all elements $1s$. After such a translation, the geometric centers of X and Y coincide at the origin. We then implement an orthogonal transformation on Y so that Y is aligned with X as much as possible. This can be done by choosing an appropriate orthogonal transform Q so that the root-mean-square deviation of X and YQ are minimized, that is,

$$\min_{Q:QQ^T=I} \mathrm{RMSD}(X,Y) = \frac{\|X - YQ\|_F}{\sqrt{\ell}}, \tag{2.12}$$

where $Q \in \mathcal{R}^{2\times 2}$ is an orthogonal matrix. Let $C = Y^T X$, and let $U\Sigma V^T$ be the singular value decomposition of C. It follows that $Q_{\mathrm{opt}} = UV^T$ is the optimal matrix of the above problem [30]. The coordinates of the ℓ determined sensors can then be obtained by setting

$$X := YQ_{\mathrm{opt}} + I_{\ell,1}X_c, \tag{2.13}$$

and the coordinates of the sensor $\ell + 1$ by

$$x_{\ell+1}^T := ((0,0) - Y_c) \times Q_{\mathrm{opt}} + X_c. \tag{2.14}$$

An outline of the extended geometric buildup algorithm is given in Algorithm 2.

2.3. The Two-Phase Geometric Buildup Algorithm

The basic and extended geometric buildup algorithms both start building up from the anchors. If only a few anchors are available and if the distance data is also very sparse, the algorithms may not be able to determine any sensors since they may not be able to find an undetermined sensor that has at least three given distances to the anchors. Even if the algorithms can proceed, they may not be able to determine all the sensors if some undetermined sensors do not have enough required distances to the determined sensors. Here, we present a two-phase geometric buildup algorithm to deal with these situations.

In Phase 1, we proceed with the extended geometric buildup algorithm until no sensors can be determined. In Phase 2, we find, in the undetermined sensors, a subset of at least three sensors where the distances between every pair of sensors are given (details are in

Input: The positions of the anchors, the distances $d_{i,j}$, $(i,j) \in N_x \cup N_a$.
Output: The positions of a set of determined sensors.
Step 1: Set the anchors to be the initial set of determined sensors.
Step 2: **Repeat:**
For each undetermined sensor:
If the sensor has distances to ℓ ($\ell \geq 3$) required determined sensors, determine the positions of all $\ell + 1$ sensors by solving (2.8), and update the coordinates of all $\ell + 1$ sensors by proper translation and orthogonal transformation.
End
End
If no sensor can be determined in the loop, stop.
If all the sensors are determined, stop.

Algorithm 2: The extended geometric buildup algorithm.

Input: The set of undetermined sensors (I_2) in Phase 1.
Output: An initial set of sensors (I_1) with distances between each other known.
Step 1: Choose the first element in I_2 to be the first element in I_1.
Step 2: **Repeat:**
For each element in I_2:
If it has given distances to all the elements in I_1, add it into I_1.
End
End

Algorithm 3: Choose an initial set in Phase 2.

Algorithm 3 "choose an initial set in Phase 2"). More discussions on finding a clique can be found in [31, 32], and so forth. If such a set of sensors, say $\ell + 1$ sensors, are found, a system of distance equations as in (2.5) can be formed, and the positions of the sensors can be determined by solving these equations in the same way as we described in the previous subsection: we first set $x_{\ell+1} = (0,0)^T$. We then define X and D as in (2.7). Let the spectral decomposition of D be $U\Lambda U^T$, where the diagonal entries of Λ are in a decreasing order. Let $V = U(:, 1 : 2)$ and Σ be the diagonal matrix with $\Sigma_{ii} = \max\{0, \Lambda_{ii}\}$, where $i = 1, 2$. Then $X = V\Sigma^{1/2}$ gives the positions of the rest of the sensors. Once the positions of these sensors are determined, we can use them as an initial set of sensors to start the extended geometric buildup algorithm again. Upon finishing, another set of determined sensors is obtained.

Hopefully, the two sets of sensors determined in Phases 1 and 2 have an overlapping subset of at least three sensors, say k sensors. Let $X \in \mathcal{R}^{k\times 2}$ and $Y \in \mathcal{R}^{k\times 2}$ be the coordinate matrices of these sensors obtained in Phases 1 and 2, respectively. We can then make a proper translation and orthogonal transformation so that the root-mean-square deviation of X and Y are minimized. An outline of the two-phase buildup algorithm is given in Algorithm 4. Note that the parameter TH is a threshold used in the algorithm. If the percentage of the undetermined sensors in Phase 1 is greater than TH, the algorithm enters Phase 2, otherwise it stops, leaving a few sensors undecided. In principle, if there are a few sensors without

Input: The positions of the anchors, the distances $d_{i,j}$, $(i,j) \in N_x \cup N_a$.
Output: The positions of a set of determined sensors.
Step 1: **Phase 1:**
Set the anchors to be the initial set of sensors.
Apply the extended geometric buildup algorithm.
Step 2: **If** the percentage of the undetermined sensors is greater than TH.
Phase 2:
Find and determine an initial set of sensors.
Apply the extended geometric buildup algorithm.
Step 3: Align the sensors determined in the two phases.

Algorithm 4: The two-phase geometric buildup algorithm.

enough distance constraints (e.g., each with fewer than three distances), they are considered to be undecidable.

3. Numerical Results

In this section, we present some numerical results from applying the two-phase geometric buildup algorithm (abbreviated as BU) to a set of test problems for sensor network localization. The test problems were generated in a similar way as used in [1]. We randomly generate n points with a uniform distribution in a square of size 1×1 centered at the origin. Without loss of generality we choose the first m points to be the anchors. We compute the distances $\overline{d}_{i,j}$ between every pair of sensors, but select only those less than the given radio range rd. We also add a multiplicative random noise to every selected distance,

$$d_{i,j} = \overline{d}_{i,j}(1 + \mathrm{nf} \cdot \mathrm{randn}(1)), \tag{3.1}$$

where nf is a specified noisy factor, and randn(1) is a standard Gaussian random variable. We set the threshold TH = 0.1. The 10% threshold for starting Phase II is indeed arbitrary. It was used for our testing purposes. In real applications, it may be set to a practically acceptable value. That is, if the percentage is lower than that value, the algorithm can terminate.

The output includes three parameters and all of our outputs are the average results from five independent test problems. One parameter is T, the average CPU time in seconds over five cases except the time to generate the test problems. The buildup algorithm tries to determine all the points, but may terminate with only a subset of points as determined. In the latter case, we use another parameter NumUndet to report the average number of undetermined sensors. This is reasonable because there could be cases that some points are not determinable uniquely, for example, when a point has only one or two distances. If the undetermined points do have more than two distances, we would suggest using a general optimization algorithm to followup. However, in this paper, we have not included followup optimization, for we want to evaluate the performance of the buildup algorithm only. The last parameter is the RMSD value, measuring the average root-mean-square deviation of

Table 1: Input and output of the two-phase geometric buildup algorithm.

Input:
m :*number of anchors.*
n :*number of all the sensors, including anchors.*
$PP_{n\times 2}$: *original coordinates matrix of all sensors.*
rd: *radio range.*
nf :*noise factor*: $d_{i,j} = \overline{d}_{i,j}(1 + \text{nf} \cdot \text{randn}(1))$.
Output:
T: *average CPU time in seconds over five cases except the time to generate the test problems.*
NumUndet: *average number of undetermined sensors.*
RMSD: *average root-mean-square deviation defined in* (3.2).

Table 2: Example 3.1: problems with exact distances.

Approach	n	m	rd	NumUndet	RMSD	T (sec.)
BU	500	50	0.1	1	$3.7e-16$	0.25
SFSDP	500	50	0.1	0	$4.7e-3$	13.1
SFSDP	500	50	0.3	0	$2.9e-7$	7.6
SNLSDPclique	500	50	0.1	0.4	$1e-14$	0.4
BU	1000	100	0.1	0	$4.6e-16$	0.6
SFSDP	1000	100	0.1	0	$3.4e-4$	22.7
SFSDP	1000	100	0.3	0	$2.2e-7$	16.0
SNLSDPclique	1000	100	0.1	0	$4e-15$	0.7
BU	2000	100	0.1	0	$1.3e-15$	7.0
SFSDP	2000	100	0.1	0	$7.7e-5$	42.3
SFSDP	2000	100	0.3	0	$5.2e-7$	34.9
SNLSDPclique	2000	100	0.1	0	$1e-14$	1.8
BU	4000	100	0.06	0	$7.5e-16$	9.4
SFSDP	4000	100	0.06	0	$2.1e-3$	317.8
SFSDP	4000	100	0.1	0	$1.9e-4$	109.2
SNLSDPclique	4000	100	0.06	0	$3e-14$	3.1

the calculated and actual locations of the determined sensors:

$$\text{RMSD} = \left(\frac{1}{p}\sum_{i=1}^{p}\|x_i - \overline{x}_i\|^2\right)^{1/2}, \tag{3.2}$$

where p is the number of the determined sensors, x_i and $\overline{x}_i$ are the true and calculated locations of the determined sensors, respectively. For convenience, we list all the input and output parameters in Table 1. Note that as an input the original coordinates matrix of all sensors (i.e., "$PP_{n\times 2}$") is used only to generate distances we need and evaluate the accuracy of the algorithms in numerical simulations.

All our calculations are done in MATLAB 7.9.0 (R2009b) on a Dell xps M1330 laptop with 2.00 GHz CPU and 3.00 GB memory.

Table 3: Large-scale problems with exact distances.

Approach	n	m	rd	NumUndet	RMSD	T (sec.)
BU	5000	100	0.06	0	$1.1e-15$	18.0
	6000	200	0.05	0	$7.3e-16$	17.9
	7000	200	0.05	0	$9.9e-16$	28.4
	8000	300	0.04	0	$7.2e-16$	32.1
	9000	300	0.04	0	$7.2e-16$	43.7
	10000	300	0.04	0	$8.1e-16$	51.9
SNLSDPclique	10000	300	0.04	0	$5e-14$	16.3

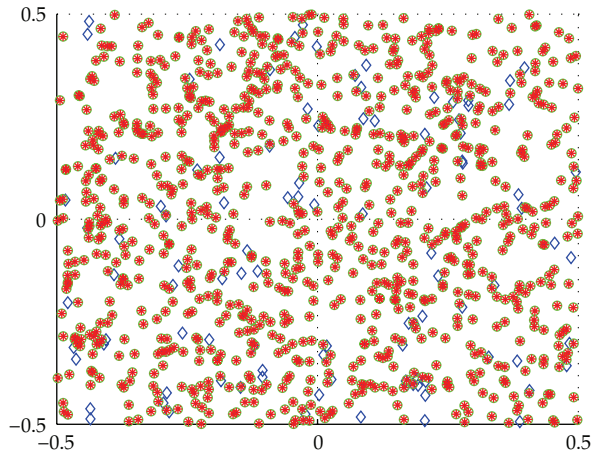

Figure 1: A network determined by the geometric buildup algorithm. $n = 1000$, $m = 100$, rd = 0.1, nf = 0. The (blue) diamonds refer to the positions of the anchors; the (green) circles to the original locations of the unknown sensors; the (red) asterisks to their estimated positions from the geometric buildup algorithm.

3.1. Problems with Exact Distances

Example 3.1. We have generated a set of sensor networks with 500, 1000, 2000, and 4000 nodes, respectively. We downloaded the code "SFSDP V111" of the SFSDP approach [16] from http://www.is.titech.ac.jp/~kojima/SFSDP/SFSDP.html. In Table 2, the "RMSD" of SFSDP are the average results over five cases without post-refinement of locations of sensors by the MATLAB function "lsqnonlin". The "T" of SFSDP is the average CPU time consumed by SeDuMi with the same accuracy parameter pars.eps = $1.0e-5$ as [16]. The numerical results in Table 2 show that for these problems with exact distances, the SFSDP approach performs well when rd is relatively larger, while the BU algorithm can find solutions to the problems with smaller RMSD values in shorter running time with fewer distance data (i.e., smaller rd). Figure 1 shows a graph of 1000 node network determined by the geometric buildup algorithm and all the sensors are accurately positioned.

We have also tested some larger-scale problems with exact distances. The results for each problem size are obtained and shown in Table 2. Note that in particular, a sensor network of 10000 nodes was solved by the geometric buildup algorithm in less than 1 minute.

Table 4: Example 3.2.

n	m	rd	nf	NumUndet	RMSD	T (sec.)
50	3	0.3	0	7.4	$6.0e-16$	0.04
50	3	0.35	0	3.8	$4.5e-16$	0.05

Table 5: Example 3.3.

Approach	n	m	rd	nf	NumUndet	RMSD	Time[a]
BU[b]					12.2	$8.0e-16$	1 sec
SOS[c]	500	4	0.3	0	0	$2.9e-6$	85 min
ESDP[d]					0	$1e-6$	30 sec

[a]This time indicates all the running time including that to generate problems.
[b]BU was implemented on a laptop with 3.00 GB memory and 2.00 GHz CPU.
[c]SOS [14] was implemented on a Linux machine with 0.98 GB RAM and 1.46 GHz CPU. The "RMSD" and "Time" of SOS come from [14].
[d]ESDP [13] was implemented on a laptop with 1.99 GB RAM and 1.06 GHz CPU. The "RMSD" and "Time" of ESDP come from [13].

Recently Krislock and Wolkowicz [19, 20] proposed an SNLSDPclique approach which is very efficient for noiseless problems. We also ran "SNLSDPclique-0.2" downloaded from http://orion.math.uwaterloo.ca/~hwolkowicz/henry/software/EDM.shtml and present the results in Tables 2 and 3. We find that in these cases when the sensor network is relatively small (e.g., $n = 500, 1000$), the two algorithms perform very closely and the BU algorithm is a little more accurate than the SNLSDPclique algorithm (note that these results of SNLSDPclique have been very accurate); when the sensor network is relatively big (e.g., $n \geq 2000$), the SNLSDPclique algorithm runs faster than the BU algorithm and the BU algorithm is still a little more accurate. We will further demonstrate the performance behaviors of these two algorithms on the noisy problems in Table 7.

3.2. Problems with a Few Anchors

Example 3.2. We have also tested a special network generated by [1, 12]. This network consists of 50 sensors, including 3 anchors. Exact distances are assumed and therefore nf = 0. The radio range rd takes values from 0.2 to 0.35. The average performance results for rd = 0.3, 0.35 are listed in Table 4. We can see that the problems were solved in less than 1 second. Note that for each rd in these tests, in two runs only Phase 1 was executed, while in the other three runs Phase 2 was also invoked. It showed that sometimes Phase 2 was necessary for sparse distance data. However, when we reduced rd to 0.25 or 0.2, the distances became very sparse, and even in Phase 2, only a few sensors could be determined.

Example 3.3. Another problem we have tested comes from [14]. We have randomly generated 500 sensors $x_1^*, x_2^*, \ldots, x_{500}^*$ and the anchors were chosen to be the four points at $(\pm 0.45, \pm 0.45)$. The distance set A was generated as follows. Initially, set $A = \emptyset$. Then for each i from 1 to 500, compute the set $I_i = \{j : \|x_i^* - x_j^*\|_2 \leq 0.3,\ j \geq i\}$; if $|I_i| \geq 10$, let A_i be the subset of I_i consisting of the 10 smallest integers; otherwise, let $A_i = I_i$; then let $A = A \cup \{(i, j) : j \in A_i\}$. The distance set B is chosen such that $B = \{(i, k) : \|x_i^* - a_k^*\|_2 \leq 0.3\}$, that is, every anchor is connected to all the sensors that are within distance 0.3. The whole distance set is $A \cup B$. Since there are no noises, nf = 0.

Table 6: Example 3.4.

n	m	rd	nf	NumUndet	RMSD of BU[a]	T (sec.)	RMSD of ESDP[a]
1000	100	0.06	0	38.6	$3.2e-16$	2.2	$2e-3$
1000	100	0.06	0.001	26.2	$4.7e-3$	2.1	$3e-3$
1000	100	0.06	0.01	36.8	$3.3e-2$	2.5	$2e-2$
4000	400	0.035	0	5.6	$3.2e-16$	41.3	$1e-3$
4000	400	0.035	0.001	4	$3.9e-3$	40.7	$8e-4$
4000	400	0.035	0.01	3.2	$1.2e-2$	40.8	$3e-2$

[a]RMSD of ESDP is obtained by implementing the steepest descent local search refinement for noisy sensor network problems. RMSD of BU is obtained without postrefinement. The RMSD values of ESDP come from [13].

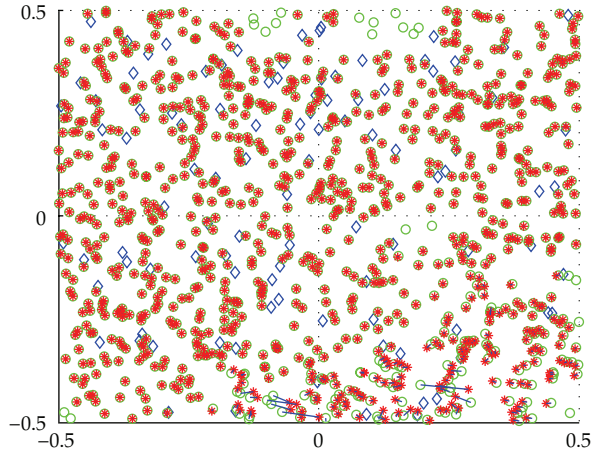

Figure 2: Graphical display of a sensor network with $n = 1000$, $m = 100$, rd = 0.06, nf = 0.001. The (blue) diamonds refer to the positions of the anchors; the (green) circles to the original locations of the unknown sensors; the (red) asterisks to their estimated positions by BU. The discrepancies between the original and the estimated points are indicated by the solid lines. RMSD = $5.9e-3$.

In this problem, the four anchors are far away from each other. If one unknown sensor has distances to at least three of them, the radio range has to be at least the distance from one of them to the origin ($\simeq 0.6364$), which is not possible. Therefore, the geometric buildup algorithm has to go to Phase 2 to solve the problem. The average performance results are listed in Table 5. We also list the results of the SOS approach [14] and the ESDP approach [13] in Table 5. Since the SOS code is not published online and the published ESDP code implements the steepest descent local search refinement, while the BU algorithm does not implement any postrefinement, we decided to directly cite their original results. From Table 5 we see that on this test problem the BU algorithm outperforms the SOS approach in the accuracy and running time and it outperforms the ESDP approach in the accuracy at least.

3.3. Problems with Noisy Distances

Example 3.4. We have also tested a set of problems with a large fraction of anchors but low distance noises (e.g., $m = 0.1n$ and nf ≤ 0.01). These problems are mid- to large-scale, tested by SOCP [15], ESDP [13], and LPCGD [18]. The parameters are set as follows:

nf = 0, 0.001, 0.01, rd = 0.06 for $n = 1000$, rd = 0.035 for $n = 4000$. For each set of parameters, the average results are in Table 6. We see that the numbers of undetermined sensors are relatively small (less than 5% of the total number of sensors). In particular, for the problem with 4000 sensors, there are only a few sensors undetermined. The average RMSD values of the BU algorithm without postrefinement are close to those of ESDP with postrefinement on noisy problems. Figure 2 shows one example of 1000 sensors with nf = 0.001 in which the locations of sensors are fairly accurate. Note that in the five cases of 4000 nodes with nf = 0.001, although the average RMSD is $3.9e-3$, the best RMSD value is $9.1e-5$ actually without any postrefinement.

We also compared the running time of the BU algorithm with the ESDP [13], SOCP1(SeDuMi) [15], SOCP2(SCGD) [15], and LPCGD [18] approaches. Based on their published results and our running results, we think on these problems the BU algorithm is most likely to run faster than ESDP, SOCP1(Se DuMi), SOCP2(SCGD), and LPCGD may run faster than BU. Since the running environments are quite different and only the ESDP code with a post-refinement is published online, it is hard to compare these approaches exactly in terms of running time, which makes us not list their running time here. Interested readers can refer to the above papers.

We have also tested some problems with larger noises, for example, nf = 0.1. The average results are shown in Table 7. As in Example 3.1, the running time of SFSDP is the average CPU time in seconds over five cases consumed by SeDuMi with the accuracy parameter pars.eps = $1.0e-5$. Kim et al. [16] used three different values of rd for each problem and we chose the rd with the best result. We also ran "SNLSDPclique-0.2" [19, 20] and show the average results over five cases in Table 7. We can see that although the SFSDP approach is a little more accurate than the BU algorithm, BU obtains similar accuracy with SFSDP (i.e., their RMSD values have the same orders) in shorter running time with fewer distance data. The SNLSDPclique approach runs faster than the BU algorithm, but BU needs fewer distance data with large noises than SNLSDPclique to obtain same orders of RMSD values. We also see that the RMSD values of the BU algorithm become larger than those of problems with low noises. This may be due to the fact that the buildup algorithm is an iterative algorithm and large noises affect the accuracy of calculations. However, we see from Table 7 that the errors of the BU algorithm may be improved by increasing the number of anchors.

Note that for SFSDP, larger rd values result in smaller RMSDs, but for BU, larger rd values may result in larger RMSD, as shown in Table 8. We can see that for the network of 2000 nodes with nf = 0.1, when rd varies from 0.05 to 0.11, RMSD increases from $6.9e-2$ to $3.7e+1$. For the network of 1000 nodes with nf = 0.001, as rd varies from 0.06 to 0.22, RMSD decreases first, then increases. The reason is that for an undetermined sensor, larger rd may result in more neighboring determined sensors and thus a larger-size noisy least-squares problem, the solution to which may involve larger errors.

3.4. Impact of Noise, the Number of Anchors, and the Radio Range

We now summarize in the following on how the distance noise, the number of anchors, and the radio range affect the performance of the geometric buildup algorithm.

First, as shown in Table 6, the RMSD value of a network determined by the geometric buildup algorithm increases as nf increases. For example, for a network of 4000 nodes, when the noise factor nf increases from 0.001 to 0.01, the RMSD value increases from around $1e-3$ to $1e-2$.

Table 7: Examples with nf = 0.1.

Approach	n	m	rd	NumUndet	RMSD	T (sec.)
BU	1000	100	0.08	0	$4.1e-2$	0.53
SFSDP	1000	100	0.2	0	$2.6e-2$	44.0
SNLSDPclique	1000	100	0.2	0	$2e-2$	1
SNLSDPclique	1000	100	0.08	0.2	$5e-1$	0.7
BU	2000	100	0.05	3.4	$7.9e-2$	8.9
SFSDP	2000	100	0.2	0	$2.6e-2$	134.6
SNLSDPclique	2000	100	0.2	0	$4e-2$	2.7
SNLSDPclique	2000	100	0.05	1.8	$2e+1$	1.4
BU	2000	200	0.05	3	$3.6e-2$	6.0
BU	4000	100	0.035	6	$9.99e-2$	75.7
SFSDP	4000	100	0.1	0	$1.6e-2$	269.4
SNLSDPclique	4000	100	0.1	0	$2e-1$	5.6
SNLSDPclique	4000	100	0.035	11.4	$9e+2$	3.9
BU	4000	200	0.035	8.8	$4.7e-2$	65.6
BU	4000	300	0.035	5.8	$3.2e-2$	52.4
BU	4000	400	0.035	5.8	$2.6e-2$	43.0

Table 8: Effect of varying radio ranges.

n	m	rd	nf	NumUndet	RMSD
1000	100	0.06	0.001	20.8	$4.2e-3$
1000	100	0.08	0.001	0.6	$1.3e-3$
1000	100	0.12	0.001	0	$2.5e-4$
1000	100	0.15	0.001	0	$8.9e-4$
1000	100	0.18	0.001	0	$6.7e-2$
1000	100	0.2	0.001	0	$1.3e-1$
1000	100	0.22	0.001	0	$1.3e+1$
2000	100	0.05	0.1	5.4	$6.9e-2$
2000	100	0.07	0.1	0	$7.0e-2$
2000	100	0.08	0.1	0	$1.2e-1$
2000	100	0.1	0.1	0	$7.3e-1$
2000	100	0.11	0.1	0	$3.7e+1$

Second, as shown in Table 7, increasing the number of anchors increases the accuracy of localization. It may reduce the time to find the required determined sensors for an undetermined sensor as well. For example, for a network of 4000 nodes, when m increases from 100 to 400, the RMSD value decreases from $9.99e-2$ to $2.6e-2$ and the running time decreases from 75.7 s to 43.0 s.

Third, as further demonstrated in Table 8, the impact of the radio range depends on the noise. As mentioned in Example 3.4, for an undetermined sensor, larger rd values may result in more neighboring determined sensors and thus a larger-size noisy least-squares problem whose optimal solution may involve larger errors. For example, if nf is small, increasing rd in an appropriate range can improve the accuracy of localization, while if nf is large, increasing rd may increase the RMSD value.

4. Conclusion

In this paper, we have investigated a geometric buildup approach to the sensor network localization problem. We follow the buildup scheme [24] which was applied to protein structure determination problems. The main difference between our algorithm and [24] is that we employ a low-rank positive semidefinite approximation scheme, which requires a spectral decomposition for a small distance matrix and guarantees a best-possible approximation to the solution of the distance equations in a least-squares sense. We have also implemented a two-phase buildup algorithm to handle problems with a few anchors and sparse distances. In principle, the initial clique in Phase II may find only several points while there are still larger cliques in the graph. We would like to consider some efficient clique searching algorithms in our future efforts.

We have tested the geometric buildup algorithm on a set of simulated sensor network localization problems with sparse and exact or inexact distances. The results showed that the algorithm runs fast on the test problems with acceptable accuracy. The algorithm is easy to follow and implement, and if further developed, may particularly be suitable for large-scale applications. The algorithm still needs to be improved to handle problems with large distance noises and problems with extremely sparse distances, which we will work on in our future efforts.

Acknowledgments

Zheng's work was done during her visit to Iowa State University. Her work was supported by the postdoctoral fellowship provided by the Laurence H. Baker Center for Bioinformatics and Biological Statistics and the Department of Mathematics, Iowa State University. Luo's work was done during his visit to Iowa State University. His work was supported by the Grant 2007CB310604 from National Basic Research Program of China and Chinese Scholar Council no. [2007]3091. Wu's work was supported partially by the NIH/NIGMS Grant R01GM081680 from the National Institutes of Health, and the NSF/DMS Grant DMS0914354 from the National Science Foundation, United States. The first author appreciates Professor Yinyu Ye for the SDP code of the sensor network localization given in a workshop (Beijing, 2006) and her advisor Professor Yaxiang Yuan for his encouragement and help all the time. The authors would like to thank Atilla Sit for sharing his initial SVD build-up code and Vladimir Sukhoy and the referees for many helpful discussions and suggestions.

References

[1] 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, (IPSN '04)*, pp. 46–54, New York, NY, USA, April 2004.

[2] M. W. Carter, H. H. Jin, M. A. Saunders, and Y. Ye, "SpaseLoc: an adaptive subproblem algorithm for scalable wireless sensor network localization," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 1102–1128, 2006.

[3] D. Culler and W. Hong, "Wireless sensor networks," *Communications of the ACM*, vol. 47, pp. 30–33, 2004.

[4] T.-H. Yi, H.-N. Li, and M. Gu, "Optimal sensor placement for health monitoring of high-rise structure based on genetic algorithm," *Mathematical Problems in Engineering*, vol. 2011, Article ID 395101, 12 pages, 2011.

[5] T. Gao, D. Greenspan, M. Welsh, R. R. Juang, and A. Alm, "Vital signs monitoring and patient tracking over a wireless network," in *Proceedings of the 27th Annual International Conference of the*

IEEE Engineering in Medicine and Biology Society, (EMBS '05), pp. 102–105, Shanghai, China, September 2005.

[6] M. Lawlor, "Small systems, big business," *Signal Magazine*, 2005.

[7] A. Dragoon, "Small wonders," CIO Magazine, 2005.

[8] J. B. Saxe, "Embeddability of weighted graphs in k-space is strongly NP-hard," in *Proceedings of the 17th Allerton Conference in Communication, Control and Computing*, pp. 480–489, Monticello, Ill, USA, 1979.

[9] G. M. Crippen and T. F. Havel, *Distance Geometry and Molecular Conformation*, Research Studies Press, Chichester, UK, 1988.

[10] J. J. More and Z. Wu, "ε-optimal solutions to distance geometry problems via global continuation," in *Global Minimization of Non-Convex Energy Functions*, P. M. Pardalos, D. Shalloway, and G. Xue, Eds., pp. 151–168, American Mathematical Society, Providence, RI, USA, 1996.

[11] Q. Dong and Z. Wu, "A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances," *Journal of Global Optimization*, vol. 22, no. 1-4, pp. 365–375, 2002.

[12] P. Biswas, T.-C. Lian, T.-C. Wang, and Y. Ye, "Semidefinite programming based algorithms for sensor network localization," *ACM Transactions on Sensor Networks*, vol. 2, no. 2, pp. 188–220, 2006.

[13] Z. Wang, S. Zheng, Y. Ye, and S. Boyd, "Further relaxations of the semidefinite programming approach to sensor network localization," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 655–673, 2008.

[14] J. Nie, "Sum of squares method for sensor network localization," *Computational Optimization and Applications*, vol. 43, no. 2, pp. 151–179, 2009.

[15] P. Tseng, "Second-order cone programming relaxation of sensor network localization," *SIAM Journal on Optimization*, vol. 18, no. 1, pp. 156–185, 2007.

[16] S. Kim, M. Kojima, and H. Waki, "Exploiting sparsity in SDP relaxation for sensor network localization," *SIAM Journal on Optimization*, vol. 20, no. 1, pp. 192–215, 2009.

[17] Z. Zhu, A. M.-C. So, and Y. Ye, "Universal rigidity and edge sparsification for sensor network localization," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 3059–3081, 2010.

[18] T. K. Pong and P. Tseng, "(Robust) edge-based semidefinite programming relaxation of sensor network localization," *Mathematical Programming*. In press.

[19] N. Krislock and H. Wolkowicz, "Explicit sensor network localization using semidefinite representations and facial reductions," *SIAM Journal on Optimization*, vol. 20, no. 5, pp. 2679–2708, 2010.

[20] N. Krislock, *Semidefinite facial reduction for low-rank euclidean distance matrix completion*, Ph.D. thesis, University of Waterloo, Waterloo, Canada, 2010.

[21] Q. Dong and Z. Wu, "A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data," *Journal of Global Optimization*, vol. 26, no. 3, pp. 321–333, 2003.

[22] D. Wu and Z. Wu, "An updated geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data," *Journal of Global Optimization*, vol. 37, no. 4, pp. 661–673, 2007.

[23] D. Wu, Z. Wu, and Y. Yuan, "Rigid versus unique determination of protein structures with geometric buildup," *Optimization Letters*, vol. 2, no. 3, pp. 319–331, 2008.

[24] A. Sit, Z. Wu, and Y. Yuan, "A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation," *Bulletin of Mathematical Biology*, vol. 71, no. 8, pp. 1914–1933, 2009.

[25] A. Sit and Z. Wu, "Solving a generalized distance geometry problem for protein structure determination," *Bulletin of Mathematical Biology*. In press.

[26] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[27] R. Mathar and R. Meyer, "Preorderings, monotone functions, and best rank r approximations with applications to classical MDS," *Journal of Statistical Planning and Inference*, vol. 37, no. 3, pp. 291–305, 1993.

[28] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling*, Chapman and Hall, London, UK, 2en edition, 1994.

[29] W. S. Torgerson, "Multidimensional scaling. I. Theory and method," *Psychometrika*, vol. 17, pp. 401–419, 1952.

[30] G. H. Golub and C. F. V. Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Md, USA, 1989.
[31] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds., pp. 85–103, Plenum Press, New York, NY, USA, 1972.
[32] J. M. Robson, "Finding a maximum independent set in time $O(2^{n/4})$," Technical Report 1251-01, Université Bordeaux I, Bordeaux, France, 2001.