J Optim Theory Appl (2011) 149: 580–598
DOI 10.1007/s10957-011-9806-6

Least-Squares Approximations in Geometric Buildup for Solving Distance Geometry Problems

Xin-long Luo · Zhi-jun Wu

Published online: 19 January 2011

Abstract In this article, we investigate some theoretical and computational issues of the geometric buildup algorithm proposed by Sit, Wu and Yuan (Bull. Math. Biol. 71:1914–1933, 2009) for the solution of the distance geometry problem with sparse and inexact distances. This algorithm repeatedly uses a least-squares approximation to determine the position of an undetermined atom, using the distances from this atom to a set of previously determined ones. The least-squares approximation, obtained from the singular value decomposition of a distance-induced matrix, can find the best possible position for the atom, even if the distances have small errors, as they usually do in practice, and therefore make the geometric buildup algorithm more stable than its previous versions that relied on linear system solvers. We estimate its numerical errors and prove some of its key mathematical properties. We also present some numerical results with varying some of the parameters in the algorithm and show how they may be used to improve its performance and computational accuracy.

Keywords Biomolecular modeling · Distance geometry · Nonlinear least squares · Singular value decomposition

Communicated by P.M. Pardalos.

The authors thank Prof. P.M. Pardalos for pointing out the references [23, 24].

X.-l. Luo (✉)
School of Information and Telecommunication Engineering, Beijing University of Posts and Telecommunications, P.O. Box 101, Beijing 100876, China
e-mail: luoxinlong@gmail.com

Z.-j. Wu
Department of Mathematics, Iowa State University, Ames, IA 50011, USA
e-mail: zhijun@iastate.edu

1 Introduction

A well-known problem in protein modeling is the determination of the structure of a protein with a given set of inter-atomic or inter-residue distances obtained from either physical experiments or theoretical estimates. A more general and abstract form of the problem is known as the distance geometry problem in mathematics [2–4], the graph embedding problem in computer science [5, 6], and the multidimensional scaling problem in statistics [7, 8]. Generally, the problem can be stated as to find the coordinates for a set of points in some topological space, given the distances for certain pairs of points. Therefore, in addition to protein modeling, where everything is discussed only in three-dimensional Euclidean space, the problem has applications in many other scientific and engineering fields as well, such as sensor network localization [9–11], image recognition [8], and protein classification [12], to name a few. In any case, the problem may or may not have a solution in a given topological space. Even if it does have a solution, the solution may not be easy to find, depending on the given distances.

Let n be the number of atoms in a given protein and $x_1, \ldots, x_n$ the coordinate vectors for the atoms $1, \ldots, n$, where $x_i = [x_{i1}, x_{i2}, x_{i3}]$ is a row vector and x_{i1}, x_{i2}, x_{i3} are the first, second, and third coordinates of atom i. Let d_{ij} be the distance between atoms i and j, i.e., $d_{ij} = \|x_i - x_j\|_2$, where $\|\cdot\|_2$ is the Euclidean norm. Then, the distance geometry problem for a given set of distances $\{d_{ij} : (i, j) \in S\}$ is to find the coordinates $x_1, \ldots, x_n$ for the atoms $1, \ldots, n$ so that the distance between atoms i and j equals the given distance d_{ij}, i.e., $\|x_i - x_j\|_2 = d_{ij}$, where (i, j) is in S. In practice, the distances may be affected by errors, and therefore, a more general yet practical form of the problem would be to find the coordinates $x_1, \ldots, x_n$ for the atoms such that their distances d_{ij} satisfy $l_{ij} \le d_{ij} \le u_{ij}$, where l_{ij} and u_{ij} are given lower and upper bounds of distances, and (i, j) is in S.

The distance geometry problem is polynomial time solvable, if the distances for all pairs of atoms are available. However, it has been proved to be NP-hard, when the distances for only a subset of all pairs of atoms are available [6]. If the errors of distances are small, the problem is still hard [13]. The existing approaches to the problem and their recent developments include, for example, the embed algorithm by Crippen and Havel [2], the alternating projection method by Glunt and Hayden et al. [14–16], the graph reduction approach by Hendrickson [5, 17], the global smoothing method by Moré and Wu [18], the stochastic/perturbation method by Zou, Byrd, and Schnabel [19], the multidimensional scaling method by Kearsly, Tapia, and Trosset [7], the dc programming method by Le Thi Hoai and Pham Dinh [20], the semidefinite programming approach by Biswas, Toh, and Ye [21], the stochastic search method by Grosso, Locatelli, and Schoen [22], the multivariate partition approach by Huang and Pardalos [23], the code partition algorithm by Huang, Pardalos and Shen [24], and the geometric buildup algorithm by Wu and co-workers [25–27].

Sit, Wu, and Yuan [1] recently proposed a new geometric buildup algorithm using a least-squares approximation for the solution of the distance geometry problem which arises from the determination of the structure of a protein. It repeatedly uses a least-squares approximation to determine the position of an undetermined atom, using the available distances from this atom to a set of previously determined ones.

The least-squares approximation, obtained from the singular value decomposition of a matrix induced by distances, can find the best possible position for the atom, even if the available distances have small errors, as they usually do in practice, and therefore, make the geometric buildup algorithm more stable than its previous versions that relied on linear system solves. In this article, we investigate some theoretical and computational issues of this algorithm. We estimate its numerical errors, verify its computational accuracy, and prove some of its key mathematical properties. We also present some numerical results with varying some of the parameters in the algorithm and show how they may be used to improve its performance.

2 Mathematical Background

Firstly, we give the concept of independent points as follows.

Definition 2.1 In a three-dimensional space, we say that four points are independent iff they are not coplanar, namely, their volume is not zero:

$$A = \begin{bmatrix} x_{11}, x_{12}, x_{13}, 1 \\ x_{21}, x_{22}, x_{23}, 1 \\ x_{31}, x_{32}, x_{33}, 1 \\ x_{41}, x_{42}, x_{43}, 1 \end{bmatrix}, \qquad \det(A) \neq 0, \tag{1}$$

where $\det(A)$ denotes the determinant of A and $x_i = [x_{i1}, x_{i2}, x_{i3}]$ denotes the coordinate vector of the ith point in a reference system, $i = 1:4$.

It is well known that the volume of the tetrahedron is the product between $\det(A)$ and $1/6$ (see [28]). Therefore, the volume of the tetrahedron is not zero iff $\det(A)$ does not equal zero in (1).

For a set of points, it is important to know how many points are required for uniquely determining an unknown point when all the distances from this unknown point to the given points are provided. The least number of points is four in a three-dimensional space. We state this fact in the following.

Theorem 2.1 *Let four independent points of* $\Re^3$ *and four positive reals be given. Then, there exists one and only one point of* $\Re^3$*, which has such reals as distances from the four points.*

Its proof can be found in many papers, such as [1, 25, 26]. For readers' convenience, we give its proof here again.

Proof If we denote the coordinate vectors of the given points by $x_i = [x_{i1}, x_{i2}, x_{i3}]$ $(i = 1:4)$, and the coordinate vector of the unknown point by $x_5 = [x_{51}, x_{52}, x_{53}]$ in a reference system, we have

$$\begin{aligned} d_{5i}^2 &= \|x_5 - x_i\|_2^2 = \|(x_5 - x_1) + (x_1 - x_i)\|_2^2 \\ &= \|x_5 - x_1\|_2^2 + \|x_1 - x_i\|_2^2 - 2(x_5 - x_1)(x_i - x_1)^T \\ &= d_{51}^2 + d_{i1}^2 - 2(x_i - x_1)(x_5 - x_1)^T, \quad i = 2:4, \end{aligned} \tag{2}$$

where $d_{ij} = \|x_i - x_j\|_2$ $(i, j = 1:5)$. We denote

$$b_i := \frac{1}{2}(d_{51}^2 + d_{i1}^2 - d_{5i}^2), \quad i = 2:4;$$

$$B := \left[x_2 - x_1; x_3 - x_1; x_4 - x_1\right]; \quad \text{and} \quad z := x_5 - x_1.$$

From (2), we have

$$Bz^T = b. \tag{3}$$

The independence of such given points implies $\det(B) \neq 0$. Thus, (3) has a unique solution z, i.e., the coordinate vector of the fifth point can be uniquely determined by $x_5 = z + x_1$. □

In every step of the buildup algorithm, we need to transform the coordinates of some points from a reference system to another reference system and should not change their structure. This purpose can be attained by rigid transforms. Rigid transformations include three basic transformations, i.e., the translation transformation, the reflection transformation and the rotation transformation. The rotation and reflection transformations belong to the orthogonal transformation. In a two-dimensional space, the rotation transformation matrix has the form

$$Q := \begin{bmatrix} \cos\theta, \sin\theta \\ -\sin\theta, \cos\theta \end{bmatrix},$$

where θ is a rotation angle. The reflection transform matrix has the form

$$Q := \begin{bmatrix} \cos\theta, \sin\theta \\ \sin\theta, -\cos\theta \end{bmatrix},$$

and $S_\theta := [\cos(\theta/2), \sin(\theta/2)]^T$ is its reflection vector [29, p. 194]. Evidently, those two matrices are the orthogonal matrices. An orthogonal transformation does not change the Euclidean distances of pairwise points, so it does not change their structure, and consequently an orthogonal transformation is a rigid transformation.The translation transformation has the form $y = x + c$, where c is a constant vector. The translation transformation also does not change the distances of pairwise points, and consequently it also does not change their structure. Thus, the translation transformation is also a rigid transformation.

An interesting property is whether there exists a rigid transformation between two groups of points, if these two groups of points satisfy the same distance property. Under mild assumptions, the answer is true. We state this property as follows.

Theorem 2.2 *Assume that two sets of coordinate vectors* $X := [x_1; x_2; \ldots; x_n] \in \Re^{n\times 3}$ *and* $Y := [y_1; y_2; \ldots; y_n] \in \Re^{n\times 3}$ $(i = 1 : n,\ n \geq 3)$ *satisfy*

$$XX^T = C \quad \textit{and} \quad YY^T = C, \tag{4}$$

where $\operatorname{rank}(C) = 3,\ C \succeq 0,\ C = C^T$. *Then, there exists a unique orthogonal matrix* Q *to satisfy*

$$Y = XQ, \quad Q \in \Re^{3\times 3}. \tag{5}$$

Furthermore, their corresponding distances of those two sets X *and* Y *are equal, i.e.,*

$$\|x_i - x_j\|_2 = \|y_i - y_j\|_2, \quad i,\ j = 1 : n. \tag{6}$$

Proof Firstly, we prove that (5) is true. For matrices X and Y, there exist singular value decompositions [29]

$$\begin{aligned} &X = U_1\Sigma_1 V_1^T \quad \text{and} \quad Y = U_2\Sigma_2 V_2^T, \\ &U_i^T U_i = I, \qquad V_i^T V_i = I, \quad U_i \in \Re^{n\times 3},\ V_i \in \Re^{3\times 3},\ i = 1 : 2, \end{aligned} \tag{7}$$

where $\Sigma_i = \operatorname{diag}(\sigma_{i1}, \sigma_{i2}, \sigma_{i3})$, $\sigma_{i1} \geq \sigma_{i2} \geq \sigma_{i3} > 0$ $(i = 1 : 2)$. Then, from (4) and (7), we have

$$XX^T = U_1\Sigma_1^2 U_1^T = C = YY^T = U_2\Sigma_2^2 U_2^T. \tag{8}$$

From (7)–(8), we have

$$U_1\Sigma_1^2 = CU_1 \quad \text{and} \quad U_2\Sigma_2^2 = CU_2, \tag{9}$$

namely, the elements of Σ_1^2 and Σ_2^2 are the eigenvalues of matrix C. In addition, the diagonal elements of Σ_1 and Σ_2 have the same order. Therefore, Σ_1 equals Σ_2 and we denote

$$\Sigma := \Sigma_1 = \Sigma_2. \tag{10}$$

Therefore, the columns of U_1 and the corresponding columns of U_2 are the eigenvectors of matrix C, which have the same corresponding eigenvalues.

Evidently, the columns of U_i $(i = 1 : 2)$ belong to the range space of matrix C. Since $\operatorname{rank}(U_1) = \operatorname{rank}(U_2) = \operatorname{rank}(C) = 3$, and U_i $(i = 1 : 2)$ are both the basis matrices of the range space of matrix C, there exists a matrix P to satisfy

$$U_2 = U_1 P, \quad \text{i.e.,} \quad P = U_1^T U_2, \quad P \in \Re^{3\times 3}. \tag{11}$$

From (11) and $U_i^T U_i = I$ $(i = 1 : 2)$, we obtain $I = U_2^T U_2 = P^T U_1^T U_1 P = P^T P$, i.e., P is an orthogonal matrix. From (8), and (10)–(11), we have

$$\Sigma^2 P = P\Sigma^2. \tag{12}$$

Since Σ^2 is a diagonal matrix, from (12), we have

$$\sigma_i^2 p_{ij} = \sigma_j^2 p_{ij}, \quad \text{i.e.,} \quad (\sigma_i^2 - \sigma_j^2) p_{ij} = 0, \quad i, j = 1:3, \tag{13}$$

where σ_i $(i = 1:3)$ are the diagonal elements of Σ and p_{ij} $(i, j = 1:3)$ are the elements of matrix P. From $\sigma_i > 0$ $(i = 1:3)$ and (13), via distinguishing two cases between $p_{ij} = 0$ and $p_{ij} \neq 0$ $(i, j = 1:3)$, we have

$$\sigma_i p_{ij} = \sigma_j p_{ij}, \quad i,\ j = 1:3. \tag{14}$$

From (14), we get

$$\Sigma P = P \Sigma. \tag{15}$$

From (7) and (14), we construct a matrix

$$Q := V_1 P V_2^T = V_1 U_1^T U_2 V_2^T. \tag{16}$$

It is not difficult to verify that Q is an orthogonal matrix. Therefore, in order to prove (5), we only need to verify $Y = XQ$. From (7), (10), (15), and (16), we have

$$\begin{aligned} XQ &= U_1 \Sigma_1 V_1^T Q = U_1 \Sigma V_1^T V_1 P V_2^T = U_1 \Sigma P V_2^T = U_1 P \Sigma V_2^T \\ &= U_1 U_1^T U_2 \Sigma V_2^T = U_2 \Sigma_2 V_2^T = Y. \end{aligned}$$

Therefore, (5) is true.

Since $\operatorname{rank}(C) = 3$, from (4), it is not difficult to get $\operatorname{rank}(X) = \operatorname{rank}(Y) = 3$. Thus, the rank of $X^T X$ equals 3, i.e., $\det(X^T X) \neq 0$. Therefore, the orthogonal matrix which satisfies (5) is unique, and it is $Q = (X^T X)^{-1} X^T Y$.

Now, we prove that the second part of Theorem 2.2 is also true. From (4), we have

$$x_i x_j^T = y_i y_j^T = c_{ij}, \quad i,\ j = 1:n, \tag{17}$$

where c_{ij} are the entries of matrix C. Then, from (17), it is easy to obtain

$$\|x_i - x_j\|_2^2 = \|x_i\|_2^2 - 2 x_i x_j^T + \|x_j\|_2^2 = c_{ii} - 2c_{ij} + c_{jj} = \|y_i - y_j\|_2^2,$$

i.e., (6) is also true. □

In both the embed algorithm [2] and the buildup algorithm, a nonlinear least squares approximation is used to obtain the coordinate vectors of some points. Specifically, they need to solve the following optimization problem

$$\min_Y \ \|YY^T - C\|_F^2, \quad \text{s.t.} \quad Y \in \Re^{n\times 3}, \quad C = C^T \in \Re^{n\times n}, \quad C \succeq 0, \tag{18}$$

where $\|\cdot\|_F$ is the Frobenius norm $\|A\|_F := (\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}^2)^{1/2}$, and the entries of matrix C are $c_{ij} := (d_{in}^2 + d_{jn}^2 - d_{ij}^2)/2,\ i,\ j = 1:n$.

The problem (18) and its general case have been investigated by many people. Its history can be traced back to Schmidt (1907) [30]. Subsequently, Weyl (1912) [31], Eckart and Young (1936) [32], Hoffman and Wielandt (1953) [33], Golub and Kahan (1965) [34], Havel (1998) [3] and other mathematicians had given their contributions. Its early history was reviewed by Stewart [35]. For readers' convenience, we give the best approximation solution of (18) and our proof. Our proof shares the same sprit with Hoffman and Wielandt [33], i.e., the proof comes down to solve a linear programming problem and depends on its geometric properties.

Theorem 2.3 *For the symmetric semi-positive definite matrix C in (18), we make a singular value decomposition $C = U\Sigma U^T$, where Σ is a diagonal matrix and its elements are the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$, and U is an orthogonal matrix. Then,*

$$Y^* = U(1:n, 1:3)\sqrt{\Sigma(1:3, 1:3)} \tag{19}$$

is the best approximation solution of (18) in a three-dimensional space.

Proof For any matrix $Y \in \Re^{n\times 3}$, we make a singular value decomposition

$$Y = Q\begin{bmatrix}\Sigma\\ 0\end{bmatrix}V^T, \qquad Q^TQ = I, \qquad V^TV = I, \quad Q \in \Re^{n\times n},\ V \in \Re^{3\times 3}, \tag{20}$$

where the diagonal elements of Σ have the descent order $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. Then, using the properties $\|A\|_F^2 = \mathrm{tr}(AA^T)$, $\mathrm{tr}(AB^T) = \mathrm{tr}(B^TA)$, and $C = U\Sigma U^T$, from (20), we have

$$\begin{aligned}
\|YY^T - C\|_F^2 &= \left\| Q\begin{bmatrix}\Sigma^2\\ 0\end{bmatrix}Q^T - U\Sigma U^T\right\|_F^2 = \mathrm{tr}\left(\left(Q\begin{bmatrix}\Sigma^2 & 0\\ 0 & 0\end{bmatrix}Q^T - U\Sigma U^T\right)^2\right)\\
&= \sum_{i=1}^{3}\lambda_i^4 + \sum_{i=1}^{n}\sigma_i^2 - 2\,\mathrm{tr}\left(\begin{bmatrix}\Sigma^2 & 0\\ 0 & 0\end{bmatrix}(Q^TU)\Sigma(U^TQ)\right)\\
&= \sum_{i=1}^{3}\lambda_i^4 + \sum_{i=1}^{n}\sigma_i^2 - 2\,\mathrm{tr}\left(\begin{bmatrix}\Sigma^2 & 0\\ 0 & 0\end{bmatrix}Z\Sigma Z^T\right)\\
&= \sum_{i=1}^{3}\lambda_i^4 + \sum_{i=1}^{n}\sigma_i^2 - 2\sum_{i=1}^{3}\sum_{j=1}^{n}\lambda_i^2\sigma_j z_{ij}^2,
\end{aligned} \tag{21}$$

where $Z := Q^TU$ and Z satisfies $Z^TZ = I$.

From (18) and (21), it needs to maximize the third term of the last equality of (21) in variables z_{ij} under the constraint $Z^T Z = I$. We let $p_{ij} = z_{ij}^2$, then the maximum of the following relaxation problem (22) is not less than the third term of the last equality in (21).

$$\begin{aligned} \max \quad & 2\sum_{i=1}^{3}\sum_{j=1}^{n}\lambda_i^2\sigma_j p_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{n} p_{ij} = 1, \quad i = 1:n; \\ & \sum_{i=1}^{n} p_{ij} = 1, \quad j = 1:n;\ p_{ij} \geq 0,\ i,\ j = 1:n. \end{aligned} \tag{22}$$

We denote $p_n := [p_{11}, \ldots, p_{1n}, p_{21}, \ldots, p_{2n}, \ldots, p_{n1}, \ldots, p_{nn}]$, and matrix $P_n := (p_{ij})$, $i, j = 1:n$. Then, (22) is a standard linear programming problem and it has an optimal solution. The linear programming problem (22) attains its maximum at one of its vertices, namely it can obtain its maximum at a basic optimal point p_n^* or P_n^* [36, pp. 369–370]. Therefore, in order to obtain an optimal solution of (22), we only need to consider its vertices.

Now, we investigate some properties of vertices of the linear programming problem (22). Since $\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij} = n = \sum_{j=1}^{n}\sum_{i=1}^{n} p_{ij}$, it is not difficult to see that those $2n$ linear constraint equations are dependent, Thus, (22) has at most $2n-1$ independent linear constraint equations. Therefore, its basic feasible point p_n^* has at least $n^2 - 2n + 1$ zero entries. It implies that matrix P_n^* has at least one row which has only one nonzero entry, where the elements of P_n^* equal the elements of a vertex p_n^*. We prove this property by contradiction. Assume that every row of P_n^* has at least two nonzero entries, then P_n^* has at most $n(n-2)$ zero entries, which contradicts that P_n^* has at least $n^2 - 2n + 1$ zero entries. Based on the same arguments, P_n^* has at least one column which has only one nonzero entry. We obtain that the nonzero entry of this column is 1.

We delete those special row and column of P_n^*, which have only one nonzero entry, respectively. Then, replacing n with $n-1$ in problem (22), we obtain the same structure problem as problem (22) in the $(n-1)^2$-dimensional space. Evidently, a vertex of problem (22) is also the vertex of this lower-dimensional linear programming problem. Therefore, matrix P_{n-1}^* has at least one row and one column which have only one nonzero entry based on the same arguments in the proof of the case of the n^2-dimensional space. By induction, we obtain that P_n^* is a permutation matrix.

Noticing $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ and using the just proof that P_n^* is a permutation matrix, the maximum of problem (22) has the form

$$\max_{(j_1, j_2, j_3)} 2\sum_{i=1}^{3}\lambda_i^2\sigma_{j_i}, \quad \sigma_{j_i} \in \{\sigma_1, \sigma_2, \sigma_3\}. \tag{23}$$

Now, we prove the following permutation inequality

$$\sum_{i=1}^{n} a_i b_i \geq \sum_{i=1}^{n} a_i b_{j_i},$$
$$\text{when } a_1 \geq a_2 \geq \cdots \geq a_n \text{ and } b_1 \geq b_2 \geq \cdots \geq b_n, \tag{24}$$

where the sequence $\{j_1, j_2, \ldots, j_n\}$ is another permutation of the sequence $\{1, 2, \ldots, n\}$. We prove the inequality (24) by induction. When $a_1 \geq a_2$ and $b_1 \geq b_2$, we have $(a_1 b_1 + a_2 b_2) \geq (a_1 b_2 + a_2 b_1)$. If we assume that the inequality (24) is true when $n = m - 1$, then we have

$$\begin{aligned}\sum_{i=1}^{m} a_i b_{j_i} &= a_1 b_l + a_k b_1 + \sum_{i=2, i\neq k}^{m} a_i b_{j_i} \\ &\leq a_1 b_1 + a_k b_l + \sum_{i=2, i\neq k}^{m} a_i b_{j_i} \\ &\leq a_1 b_1 + (a_2 b_2 + \cdots + a_m b_m),\end{aligned}$$

i.e., the inequality (24) is also true when $n = m$. Therefore, the inequality (24) is true.

Using the inequality (24), and the assumptions $\sigma_1 \geq \sigma_2 \geq \sigma_3$ and $\lambda_1 \geq \lambda_2 \geq \lambda_3$, we obtain that the maximum of problem (23) equals $2\sum_{i=1}^{3} \lambda_i^2 \sigma_i$. Thus, we obtain the lower bound of the minimum of problem (21), namely,

$$\min_{\lambda_i} \left(\sum_{i=1}^{3} (\lambda_i^2 - \sigma_i)^2 + \sum_{i=4}^{n} \sigma_i^2 \right) = \sum_{i=4}^{n} \sigma_i^2.$$

It is not difficult to verify that this lower bound can be attained when we choose matrix Y^* as (19), namely, this lower bound is the minimum of problem (21). □

In the buildup algorithm, we want to find a rigid transformation which preserves the structure of points when their coordinates are transformed from one reference system to another reference system. If we denote the coordinate vectors of points as $X \in \Re^{n\times 3}$ in an original reference system and as $Y \in \Re^{n\times 3}$ in another reference system, in an ideal case, according to Theorem 2.2, there exists an orthogonal matrix $Q \in \Re^{3\times 3}$ and a translation vector $b \in \Re^{1\times 3}$ to satisfy $Y = XQ + e^T b$, $e = [1, 1, \ldots, 1]$. Since there are roundoff errors, we only expect to obtain an approximation transformation. We state this model as follows.

Theorem 2.4 *Assume that the coordinate vectors of two sets of points $X \in \Re^{n\times 3}$ and $Y \in \Re^{n\times 3}$ are known. We denote their center coordinates as*

$$\begin{aligned} x_c &:= (X(1,:) + X(2,:) + \cdots + X(n,:))/n, \\ y_c &:= (Y(1,:) + Y(2,:) + \cdots + Y(n,:))/n. \end{aligned} \tag{25}$$

Then, $Q := VU^T$ *and* $b := y_c - x_c Q$ *are the best approximation solution of*

$$\min_{Q,b} \|Y - (XQ + e^T b)\|_F^2, \quad s.t. \quad Q^T Q = I, \tag{26}$$

where $e := [1, \ldots, 1]$, *and orthogonal matrices* U *and* V *satisfy* [29]

$$\begin{aligned} &U\Sigma V^T = (Y - e^T y_c)^T (X - e^T x_c), \\ &\Sigma := \operatorname{diag}(\sigma_1, \sigma_2, \sigma_3), \quad \sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0. \end{aligned} \tag{27}$$

Proof Using the relations $\|A\|_F^2 = \operatorname{tr}(A^T A) = \operatorname{tr}(AA^T)$ and $Q^T Q = I$, from (26), we have

$$\begin{aligned} &\|Y - (XQ + e^T b)\|_F^2 \\ &\quad = \|(Y - XQ) - e^T b\|_F^2 = \operatorname{tr}(((Y - XQ) - e^T b)^T ((Y - XQ) - e^T b)) \\ &\quad = \operatorname{tr}((Y - XQ)^T (Y - XQ)) - 2b(eY - eXQ)^T + nbb^T \\ &\quad = \operatorname{tr}((Y - XQ)^T (Y - XQ)) + n\|b - (eY - eXQ)/n\|_2^2 \\ &\qquad - ((eY - eXQ)(eY - eXQ)^T)/n \\ &\quad \geq \operatorname{tr}((Y - XQ)^T (Y - XQ)) - ((eY - eXQ)(eY - eXQ)^T)/n. \end{aligned} \tag{28}$$

If Q and b are the optimal solution of (26), from (28), the vector b should satisfy

$$b := (eY - eXQ)/n = y_c - x_c Q, \tag{29}$$

where x_c and y_c are defined in (25). Thus, from (28)–(29), we convert the problem (26) to the following equivalent problem

$$\min_{Q} \|(Y - e^T y_c) - (X - e^T x_c)Q\|_F^2, \quad \text{s.t.} \quad Q^T Q = I. \tag{30}$$

Now, we solve the problem (30). From (30), we have

$$\begin{aligned} &\|(Y - e^T y_c) - (X - e^T x_c)Q\|_F^2 \\ &\quad = \operatorname{tr}((Y - e^T y_c)^T (Y - e^T y_c)) + \operatorname{tr}((X - e^T x_c)^T (X - e^T x_c)) \\ &\qquad - 2\operatorname{tr}((Y - e^T y_c)^T (X - e^T x_c)Q). \end{aligned} \tag{31}$$

In the above, we use the properties $Q^T Q = I$ and $\operatorname{tr}(A^T B) = \operatorname{tr}(BA^T)$. From (31), in order to obtain the minimum of (30), we only want to solve the following problem

$$\max_{Q} \operatorname{tr}((Y - e^T y_c)^T (X - e^T x_c)Q), \quad \text{s.t.} \quad Q^T Q = I. \tag{32}$$

For matrix $(Y - e^T y_c)^T (X - e^T x_c)$, we make a singular value decomposition as (27). Then, we have

$$\begin{aligned}&\operatorname{tr}((Y-e^T y_c)^T(X-e^T x_c)Q)\\&\quad=\operatorname{tr}(U\Sigma V^T Q)=\operatorname{tr}(\Sigma(V^T QU))=\operatorname{tr}(\Sigma Z)=\sum_{i=1}^{3}(\sigma_i z_{ii}),\end{aligned}\tag{33}$$

where $Z := V^T QU$ is an orthogonal matrix. Since $ZZ^T = I$, we have $\sum_{j=1}^{3} z_{ij}^2 = 1$, and consequently, $-1 \le z_{ii} \le 1\ (i = 1:3)$. Combining $\sigma_i \ge 0\ (i = 1:3)$, we obtain $\sum_{i=1}^{3}(\sigma_i z_{ii}) \le \sum_{i=1}^{3}\sigma_i$. Its upper bound $\sum_{i=1}^{3}\sigma_i$ can be attained when $z_{ii} = 1\ (i = 1:3)$. From $z_{ii} = 1\ (i = 1:3)$ and $ZZ^T = 1$, we get $Z = I$. Thus, we obtain an optimal solution of (30) when $Z = I$, namely, $Q = VZU^T = VU^T$. Consequently, $Q = VU^T$ and $b = y_c - x_c Q$ are the best approximation solution of (26). □

3 The Buildup Algorithm

In this section, we describe, in detail, the buildup algorithm using least squares approximations. The algorithm is similar to that given in [1], with some modifications. Assume that the structure of n points needs to be determined in a three-dimensional space, i.e., their coordinate vectors $X(i,:)\ (i = 1:n)$ need to be determined in a reference system. Matrix $D \in \Re^{n\times n}$ denotes the distance matrix of points, i.e., its elements $D(i,j)\ (i,j = 1:n)$ denote the distance between the ith point and the jth point. If $D(i,j) = 0$, we regard that the distance is unknown.

Algorithm 3.1 *Buildup Algorithm Using Least-Squares Approximations*

Step 0 Input the number of points n and the distance matrix $D \in \Re^{n\times n}$. Initialize the index set $\mathcal{S}_u$ of the undetermined points as $\mathcal{S}_u = [1, 2, \ldots, n]$ and the index set $\mathcal{S}_d$ of the determined points as $\mathcal{S}_d = [0, 0, \ldots, 0]$. We denote N_u as the number of undetermined points and N_d as the number of determined points, respectively. Initialize the parameter $K_u \ge 3$, such as $K_u = 10$, and let $N_u = n,\ N_d = 0$, $X(1:n,:) = 0$.

Step 1 Find four independent points, i.e., the i_1th, i_2th, i_3th, i_4th point, whose distances $D(i_k, i_j)\ (k, j = 1:4)$ are all known. Determine their coordinates $X(i_k,:)$ in a reference system. Let

$$\begin{aligned}X(i_1,:) =&[0,0,0]; \qquad X(i_2,:) = [D(i_2,i_1),0,0];\\X(i_3,1) =&(D^2(i_3,i_1)+D^2(i_2,i_1)-D^2(i_3,i_2))/(2D(i_2,i_1)),\\X(i_3,2) =&\sqrt{D^2(i_3,i_1)-X^2(i_3,1)}, \qquad X(i_3,3) = 0;\\X(i_4,1) =&(D^2(i_4,i_1)+D^2(i_2,i_1)-D^2(i_4,i_2))/(2D(i_2,i_1)),\\X(i_4,2) =&(D^2(i_4,i_1)+D^2(i_3,i_1)-D^2(i_4,i_3)-2X(i_4,1)X(i_3,1))\\&/(2X(i_3,2)),\\X(i_4,3) =&\sqrt{D^2(i_4,i_1)-X^2(i_4,1)-X^2(i_4,2)}.\end{aligned}\tag{34}$$

It is not difficult to verify that $X(i_k,:)$ $(k=1:4)$ satisfy $\|X(i_k,:)-X(i_j,:)\|_2 = D(i_k,i_j)$.

Step 2 Let $\mathcal{S}_d=\{i_1,i_2,i_3,i_4,0,\ldots,0\}$ and

$$\mathcal{S}_u=\{1,2,\ldots,i_1-1,i_1+1,\ldots,i_2-1,i_2+1,\ldots,i_3-1,i_3+1,\ldots,i_4-1,\\ i_4+1,\ldots,n\},$$

i.e., we delete the indices i_1, i_2, i_3, i_4 from the undetermined index set $\mathcal{S}_u$. Let $N_d=N_d+4$ and $N_u=N_u-4$.

Step 3 Search for an undetermined point p from the index $\mathcal{S}_u$. For an undetermined point p, define a subset $\mathcal{S}_{su}$ of $\mathcal{S}_d$ as

$$\mathcal{S}_{su}=\{k|D(p,k)>0,k\in\mathcal{S}_d\}. \tag{35}$$

If the number of $\mathcal{S}_{su}$ is less than 3 for any undetermined point p, then stop, otherwise go to Step 4.

Step 4 Choose K points from the set $\mathcal{S}_{su}$, which have at least four independent points, where K satisfies $4\le K\le K_u$. Those K points compose a subset $\mathcal{S}_K$ of $\mathcal{S}_{su}$, i.e., $\mathcal{S}_K\subseteq\mathcal{S}_{su}$.

Step 5 Determine a substructure of points in $\mathcal{S}_K\cup\{p\}$. Using the distances of those $K+1$ points in $\mathcal{S}_K\cup\{p\}$, construct matrix $C\in\Re^{K\times K}$ as follows:

$$c_{ij}=\frac{1}{2}(D^2(\mathcal{S}_K(i),p)+D^2(\mathcal{S}_K(j),p)-D^2(\mathcal{S}_K(i),\mathcal{S}_K(j))),\quad i,\ j=1:K. \tag{36}$$

Let the coordinate vector of point p be 0 in another reference system. According to the same reasons with (18), we can obtain the new coordinate vectors of those K points in S_K in new reference system by solving the following nonlinear least-squares problem

$$\min_Y \|YY^T-C\|_F^2,\quad \text{s.t.}\quad Y\in\Re^{K\times 3},\quad C=C^T\in\Re^{K\times K}. \tag{37}$$

Make a singular decomposition value $C=U\Sigma U^T$, where the entries σ_i $(i=1:K)$ of the diagonal matrix Σ satisfy $\sigma_1\ge\cdots\ge\sigma_K\ge 0$, and U satisfies $U^TU=I$. From Theorem 2.3, we know that $Y=U(1:3,:)\sqrt{\Sigma(1:3,1:3)}$ is the best approximation solution of (37).

Step 6 Find a rigid transformation between two reference systems. Find an orthogonal matrix Q and a vector b to satisfy $X(\mathcal{S}_K(i),:)=Y(i,:)Q+b$, $i=1:K$. We convert it to the following constrained optimization problem

$$\min_{Q,b}\|X(\mathcal{S}_K(1:K),:)-(Y(1:K,:)Q+e^Tb)\|_F^2,\quad \text{s.t.}\quad QQ^T=I, \tag{38}$$

where $e=[1,\ldots,1]$. From Theorem 2.4, we obtain its solution as follows

$$Q=V_KU_K^T\quad\text{and}\quad b=x_c-y_cQ, \tag{39}$$

where

$$
\begin{aligned}
&x_c = (X(\mathcal{S}_K(1),:) + \cdots + X(\mathcal{S}_K(K),:))/K,\\
&y_c = (Y(1,:) + \cdots + Y(K,:))/K,\\
&(Y(1:K,:) - e^T y_c)^T (X(\mathcal{S}_K(1:K),:) - e^T x_c) = U_K \Sigma_K V_K^T, \qquad (40)\\
&U_K U_K^T = I, \qquad V_K V_K^T = I,\\
&\Sigma_K = \operatorname{diag}(\overline{\sigma}_1, \overline{\sigma}_2, \overline{\sigma}_3), \quad \overline{\sigma}_1 \geq \overline{\sigma}_2 \geq \overline{\sigma}_3 \geq 0.
\end{aligned}
$$

Step 7 Correct the coordinate vectors of points in $\mathcal{S}_K$ and determine point p. Using (39) and (40), correct the coordinate vectors of points in $\mathcal{S}_K$ and determine the coordinate vector of point p as $X(\mathcal{S}_K(i),:) = Y(i,:)Q + b$, $i = 1:K$, and $X(p,:) = b$.

Step 8 Add the index of point p into the determined index set $\mathcal{S}_d$. Delete the index of point p from the undetermined index set $\mathcal{S}_u$. The number of determined points N_d increases by one. The number of undetermined points N_u decreases by one. If $N_u = 0$, then stop, otherwise go to Step 3.

Remark 3.1 In Step 6 of Algorithm 3.1, we want to know whether there exists a rigid transformation between a new reference system and the original reference system for the same set of points when their distances are exact. It can be verified as follows.

Firstly, we verify that their corresponding distances in those two reference systems are equal. From (36), we have $c_{ij} = (X(\mathcal{S}_{\mathcal{K}}(i),:) - X(p,:))(X(\mathcal{S}_{\mathcal{K}}(j),:) - X(p,:))^T$. Consequently, from $X(\mathcal{S}_{\mathcal{K}}(i),:) \in \Re^{K\times 3}$, we obtain $\operatorname{rank}(C) \leq 3$. Therefore, from Theorem 2.3, we know that the best approximation solution $Y \in \Re^{K\times 3}$ of (37) satisfies $YY^T = C$. From $YY^T = C$ and (36), we have

$$
\begin{aligned}
&\|Y(i,:) - Y(j,:)\|_2^2\\
&\quad = \|Y(i,:)\|_2^2 + \|Y(j,:)\|_2^2 - 2Y(i,:)Y(j,:)^T = c_{ii} + c_{jj} - 2c_{ij}\\
&\quad = D^2(\mathcal{S}_K(i), p) + D^2(\mathcal{S}_K(j), p)\\
&\qquad - (D^2(\mathcal{S}_K(i), p) + D^2(\mathcal{S}_K(j), p) - D^2(\mathcal{S}_K(i), \mathcal{S}_K(j)))\\
&\quad = D^2(\mathcal{S}_K(i), \mathcal{S}_K(j)) = \|X(\mathcal{S}_K(i),:) - X(\mathcal{S}_K(i))\|_2^2, \quad i,\ j = 1:K,
\end{aligned}
$$

namely, their corresponding distances in those two reference systems are equal.

Secondly, we construct a new induced matrix $\overline{C}$ from the distances of those K points as

$$
\begin{aligned}
\overline{c}_{ij} &= (Y(i,:) - Y(1,:))(Y(j,:) - Y(1,:))^T\\
&= (X(\mathcal{S}_K(i),:) - X(\mathcal{S}_K(1),:))(X(\mathcal{S}_K(i),:) - X(\mathcal{S}_K(1),:))^T\\
&= \frac{1}{2}(D^2(\mathcal{S}_K(i),1) + D^2(\mathcal{S}_K(j),1) - D^2(\mathcal{S}_K(i),\mathcal{S}_K(j))), \quad i,\ j = 2:K. \qquad (41)
\end{aligned}
$$

From Theorem 2.2 and (41), there exists an orthogonal matrix $\overline{Q}$ to satisfy

$$X(\mathcal{S}_K(i),:) - X(\mathcal{S}_K(1),:) = (Y(i,:) - Y(1,:))\overline{Q}, \quad i = 2:K. \tag{42}$$

If we let $\overline{b} = X(\mathcal{S}_K(1),:) - Y(1,:)\overline{Q}$, then (42) gives $X(\mathcal{S}_K(i),:) = Y(i,:)\overline{Q} + \overline{b}$, $i = 1:K$. Therefore, there exists a rigid transformation between those two reference systems.

Remark 3.2 In Step 4 of Algorithm 3.1, there are two reasons for restricting the upper bound of the number of $\mathcal{S}_K$, i.e., according to computational experiments, we choose $K_u = 10$, however, K_u around 10 is not sensitive to Algorithm 3.1 from Table 1 of Sect. 4.1.

One reason is that four known independent points are sufficient to determine an unknown point if their distances are known and the small number of $\mathcal{S}_K$ saves the computing time.

The other reason is as follows. We want to make a singular value decomposition to obtain the coordinate vectors $Y(i,:)$ of K points in (37) of Step 5. If we let $\overline{C} = C + E$, where matrix C is defined by (36) and E is a perturbed matrix, we have "small singular values tend to increase under perturbation, and the increment is proportional to $\sqrt{K}$" (see [37]). It can be illustrated by letting the special perturbed matrix $E := \epsilon e^T e$, where $e := [1, \ldots, 1]$. Specifically, $\|E\|_2 = \sqrt{K}\epsilon$ is the maximum of its absolute eigenvalues. Using the Wely theorem [31]:

$$|\overline{\sigma}_i - \sigma_i| \le \|E\|_2, \quad i = 1:K,$$

where $\overline{\sigma}_i$ and σ_i are the singular values $\overline{C}$ and C, respectively, we have $\overline{\sigma}_i \le \sigma_i + \|E\|_2 = \sigma_i + \sqrt{K}\epsilon$. Therefore, in order to control the error propagation, the dimension of matrix C is not too large.

4 Numerical Experiments

In this section, we test Algorithm 3.1 for molecular conformation problems. All problems are extracted from PDB (Protein Data Bank [38]) files. We measure the computed results by the minimum root mean-square deviation (RMSD) between the computed structure and the original structure. RMSD is defined by

$$\text{RMSD} = \min_{Q \in \Re^{3\times 3}} \left\{ \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|(x_i - x_c) - (\bar{x}_i - \bar{x}_c)Q\|_2^2} \right\}, \tag{43}$$

where x_i and $\bar{x}_i$ denote the computed coordinate vectors and actual coordinate vectors of the ith atom, respectively, and N is the number of determined atoms, and $x_c = (\sum_{i=1}^{N} x_i)/N$, $\bar{x}_c = (\sum_{i=1}^{N} \bar{x}_i)/N$ are the corresponding centers of the computed structure and the original structure, respectively, and $Q\Re^{3\times 3}$ is an orthogonal matrix.

Another practical measurement is the given distance mean error (GDME), which is defined by

$$\mathrm{GDME} = \sqrt{\frac{1}{M_g} \sum_{(i,j)\in\mathcal{S}_g} (\|X(i,:) - X(j,:)\|_2 - d_{ij})^2}, \tag{44}$$

$$\mathcal{S}_g = \{(i,j) | d_{ij} > 0,\ X(i,:) \text{ and } X(j,:) \text{ are determined}\},$$

where d_{ij} is the given distance, and M_g is the number of the set $\mathcal{S}_g$.

All test problems are executed by an IBM R32 laptop with an Intel Pentium processor 1.60 GHz and 768M RAM. They are solved in MATLAB 6.5 environment [39]. We give the meanings of some abbreviated phrases. TA denotes the total number of atoms of a test protein. DA denotes the total number of determined atoms of a test protein. CPU (in seconds) denotes the computing time of Algorithm 3.1 for determining a structure of a test protein, which does not include the computing time of distances generated from original coordinate vectors of a test protein and the computed coordinate vectors written into a output file.

4.1 Numerical Experiments for Choosing the Parameter K_u

In this subsection, we give some suggestions to choose the number of $\mathcal{S}_K$, which is a subset of the determined points whose interatomic distances are known between them and the current undetermined point in Step 4 of Algorithm 3.1. We choose stochastically 1AX8.pdb which includes 1003 atoms, and 1F39.pdb which includes 1534 atoms as the test proteins. Since the test protein needs to have enough distances to observe the impact of different K_u for Algorithm 3.1, we choose the big cutoff distance, for example, 9 Å. When the cutoff distance equals 9 Å, for those two test proteins, known distances over all distances of atoms are about 9.861% and 6.77% for 1AX8.pdb and 1F39.pdb, respectively.

The numerical results are reported in Table 1. From Table 1, we find the upper bound K_u of the number of $\mathcal{S}_K$ should not be too big and Algorithm 3.1 is not sensitive to K_u around 10.

4.2 Problems with Exact Distances

In order to evaluate the performance of Algorithm 3.1, we test different cutoff distances of a test protein. The Distance is regarded as the unknown distance when the distance is bigger than the cutoff distance. The numerical results are reported in Table 2. From Table 2, we find that Algorithm 3.1 is efficient for those test problems. There are few atoms can not be determined when the cutoff distance is 5 angstroms. All atoms can be determined when the cutoff distance is 6 or 7 angstroms. For problem 1HMV.pdb, the computing time of the 5 angstroms cutoff distance is bigger than the computing time of the 6 or 7 angstroms cutoff distance. The reason is that it needs to compute many unknown distances $D(\mathcal{S}_K(i), \mathcal{S}_K(j)) = \|X(\mathcal{S}_K(i),:) - X(\mathcal{S}_K(j),:)\|_2$ in Step 5 of Algorithm 3.1 and they occupy much time, when the given distances are not much redundance.

Table 1 Various K_u in Algorithm 3.1 for problems with exact distances

ID	TA	≤ 9 Å, $K_u = 4$		≤ 9 Å, $K_u = 5$		≤ 9 Å, $K_u = 6$		≤ 9 Å, $K_u = 7$	
		CPU	RMSD	CPU	RMSD	CPU	RMSD	CPU	RMSD
1AX8	1003	0.841	6.6e+00	0.881	5.4e+00	0.931	9.0e−14	1.001	7.3e−14
1F39	1534	1.5930	7.1e+00	1.582	1.4e−11	1.742	1.4e−12	1.763	3.8e−13

ID	TA	≤ 9 Å, $K_u = 8$		≤ 9 Å, $K_u = 9$		≤ 9 Å, $K_u = 10$		≤ 9 Å, $K_u = 11$	
		CPU	RMSD	CPU	RMSD	CPU	RMSD	CPU	RMSD
1AX8	1003	1.172	6.1e−14	1.15	4.2e−14	1.24	4.4e−14	1.402	4.2e−14
1F39	1534	1.793	1.3e−13	1.883	5.8e−14	1.863	6.2e−14	2.063	1.1e−13

ID	TA	≤ 9 Å, $K_u = 15$		≤ 9 Å, $K_u = 20$		≤ 9 Å, $K_u = 25$		≤ 9 Å, $K_u = 30$	
		CPU	RMSD	CPU	RMSD	CPU	RMSD	CPU	RMSD
1AX8	1003	1.79	4.7e−14	2.914	1.4e−13	3.405	5.6e−12	4.437	8.4e−12
1F39	1534	2.563	5.8e−14	3.425	7.1e−14	4.517	2.0e−13	5.868	8.6e−12

ID	TA	≤ 9 Å, $K_u = 35$		≤ 9 Å, $K_u = 40$		≤ 9 Å, $K_u = 45$		≤ 9 Å, $K_u = 50$	
		CPU	RMSD	CPU	RMSD	CPU	RMSD	CPU	RMSD
1AX8	1003	5.447	3.2e−10	6.479	1.1e−06	7.721	1.6e−05	9.952	9.6e−04
1F39	1534	8.822	1.7e−10	10.656	6.2e−07	11.867	2.7e−03	13.419	2.1e+08

ID	TA	≤ 9 Å, $K_u = 51$		≤ 9 Å, $K_u = 52$		≤ 9 Å, $K_u = 53$		≤ 9 Å, $K_u = 54$	
		CPU	RMSD	CPU	RMSD	CPU	RMSD	CPU	RMSD
1AX8	1003	9.194	3.1e−03	9.464	5.5e−02	9.754	7.7e+01	9.914	4.6e+02

Table 2 Problems with exact distances

ID	TA	≤ 5 Å			≤ 6 Å			≤ 7 Å		
		DA	CPU	RMSD	DA	CPU	RMSD	DA	CPU	RMSD
1PTQ	402	402	0.411	6.5e−14	402	0.421	2.9e−14	402	0.431	1.0e−14
1HOE	558	558	0.571	2.5e−14	558	0.581	5.2e−14	558	0.591	2.3e−14
1LFB	641	641	0.711	6.6e−14	641	0.691	2.1e−14	641	0.691	2.6e−14
1PHT	814	809	1.082	5.0e−14	814	0.901	4.3e−14	814	0.881	4.3e−14
1POA	914	914	0.911	2.4e−13	914	1.021	5.9e−14	914	1.021	2.9e−14
1AX8	1003	1003	1.312	7.5e−14	1003	1.302	1.3e−13	1003	1.242	5.8e−13
4MBA	1086	1083	1.442	1.9e−13	1086	1.272	7.8e−14	1086	1.271	1.4e−13
1F39	1534	1534	2.174	2.2e−13	1534	2.053	1.3e−13	1534	2.013	1.1e−13
1RGS	2015	2010	4.456	7.4e−13	2015	2.744	1.5e−13	2015	2.694	1.1e−13
1BPM	3672	3669	12.067	4.7e−13	3672	6.569	1.2e−13	3672	6.219	6.8e−14
1HMV	7398	7389	80.636	4.9e−09	7398	20.740	2.4e−12	7398	23.524	5.1e−13

Table 3 Problems with disturbed distances (cutoff = 5 Å)

ID	TA	RE: 1.0e−6			RE: 1.0e−4			RE: 1.0e−3		
		CPU	RMSD	GDME	CPU	RMSD	GDME	CPU	RMSD	GDME
1PTQ	402	0.451	1.4e−04	1.3e−04	0.450	8.7e−03	6.5e−03	0.451	3.3e−02	2.9e−02
1HOE	558	0.621	6.6e−05	5.4e−05	0.640	4.4e−03	3.8e−03	0.681	1.4e−02	1.2e−02
1LFB	641	0.771	2.3e−05	1.5e−05	0.771	1.4e−03	1.0e−03	0.871	2.6e−02	2.0e−02
1PHT	814	3.836	4.9e−05	4.0e−05	3.705	2.3e−02	2.1e−02	3.726	7.1e−02	6.2e−02
1POA	914	1.122	5.8e−04	4.0e−04	1.181	2.9e−03	2.4e−03	1.091	4.2e−02	3.0e−02
1AX8	1003	1.512	1.7e−04	1.0e−04	1.462	4.7e−02	3.1e−02	1.453	1.1e+01	1.8e+01
4MBA	1086	4.036	4.5e−03	2.6e−04	3.485	3.6e−02	3.0e−02	3.526	1.5e+00	2.0e+00
1F39	1534	2.143	2.6e−04	1.6e−04	2.163	2.0e−02	1.0e−02	2.323	3.4e+00	2.5e+00
1RGS	2015	15.453	2.4e−02	5.5e−04	17.084	4.2e+00	4.3e+00	17.285	9.8e+00	9.8e+00
1BPM	3672	39.106	1.0e−02	8.4e−04	37.764	3.3e−02	2.6e−02	37.143	8.8e+00	1.3e+01

Table 4 Problems with disturbed distances (cutoff = 8 Å)

ID	TA	RE: 1.0e−4			RE: 1.0e−3			RE: 1.0e−2		
		CPU	RMSD	GDME	CPU	RMSD	GDME	CPU	RMSD	GDME
1PTQ	402	0.461	5.5e−04	5.8e−04	0.461	6.6e−03	6.8e−03	0.471	5.2e−02	5.8e−02
1HOE	558	0.641	5.9e−04	6.1e−04	0.641	8.1e−03	7.6e−03	0.661	9.3e−02	1.1e−01
1LFB	641	0.791	9.0e−04	8.2e−04	0.761	6.6e−03	6.2e−03	0.822	7.6e−02	6.6e−02
1PHT	814	0.951	9.0e−04	8.9e−04	1.022	8.0e−03	7.4e−03	0.982	1.4e−01	1.6e−01
1POA	914	1.091	7.2e−04	7.6e−04	1.152	6.0e−03	5.6e−03	1.132	7.6e−02	7.5e−02
1AX8	1003	1.272	9.7e−04	8.6e−04	1.262	9.7e−03	9.1e−03	1.402	3.7e+00	6.7e+00
4MBA	1086	1.362	9.4e−04	9.0e−04	1.352	6.7e−03	6.2e−03	1.382	7.7e−02	7.4e−02
1F39	1534	2.063	6.3e−03	3.6e−03	2.213	6.3e+00	7.0e+00	2.313	7.9e+00	6.2e+00
1RGS	2015	3.124	2.0e−03	1.4e−03	3.685	2.9e−02	1.8e−02	3.915	3.6e+00	3.6e+00
1BPM	3672	8.442	2.4e−03	2.2e−03	7.521	1.3e−02	9.2e−03	7.321	6.0e−01	6.2e−01
1HMV	7398	20.86	1.1e+00	2.1e+00	20.179	1.8e+01	2.9e+01	22.373	5.2e+01	8.9e+01

4.3 Problems with Inexact Distances

Generally, measurement data have errors, so we test Algorithm 3.1 for problems with small relative errors. We let $\overline{d}_{ij}(\varepsilon_{ij}) = d_{ij} + \varepsilon_{ij}(\theta)$ as an approximation distance between the ith atom and the jth atom ($i,\ j = 1:n$), where $\varepsilon_{ij}(\theta)$ is a random variable, which satisfies the uniform distribution on $[-\mathrm{RE}, \mathrm{RE}]$. We report the numerical results in Table 3 and Table 4. We find that Algorithm 3.1 determines the structure of a test protein well except for problem 1HMV which has 7398 atoms, when the relative distance error is small, such as the maximum absolute relative error being less than 0.001.

5 Conclusions

In this article, we give a detailed description and implementation of a stable geometric buildup algorithm for the distance geometry problem, following the work by Sit, Wu, and Yuan [1]. The buildup algorithm described here is different its previous versions that it only uses partial distance information of determined points when it determines the coordinate vector of the undetermined point in every buildup step.

From the numerical results in Sect. 4, we find that Algorithm 3.1 is effective for the large molecule structure problem, when its distances are exact, even if its relative distance errors are small. However, there are at least two future works that can be done. One is to solve the problem when its relative errors of known distances are large. The other is to solve the problem when the points fluctuate, which occurs in real proteins. We expect that the buildup algorithm can be extended to these two cases as well.

Acknowledgements The first author would like to thank Iowa State University for providing a good research environment when he visited it in the fall of 2008. This research was supported in part by Grant 2009CB320401 from National Basic Research Program of China, and Grant YBWL2010152 from Huawei Technologies Co., Ltd., and Grant BUPT2009RC0118 from the Fundamental Research Funds for the Central Universities,and Grant 2007111533 from Chinese Scholar Council for his visiting Iowa State University, and grant RO1GM081680 from the National Institutes of Health, United States.

References

1. Sit, A., Wu, Z., Yuan, Y.: A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation. Bull. Math. Biol. **71**, 1914–1933 (2009)
2. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
3. Havel, T.F.: Distance geometry: theory, algorithms, and chemical applications. In: Encyclopedia of Computational Chemistry. Wiley, New York (1998)
4. Huang, H.-X., Liang, Z.-A., Pardalos, P.: Some properties for the Euclidean distance matrix and positive semi-definite matrix completion problems. J. Glob. Optim. **25**, 3–21 (2003)
5. Hendrickson, B.: Conditions for unique graph realizations. SIAM J. Comput. **21**, 65–84 (1992)
6. Saxe, J.B.: Embeddability of weighted graphs in k-space is strongly NP-hard. In: Proceedings 17th Annual Allerton Conference on Communication, Control and Computing, pp. 480–489 (1979)
7. Kearsley, A., Tapia, R., Trosset, M.: The solution of the metric STRESS and SSTRESS problems in multidimensional scaling using Newton's method. Comput. Comput. Stat. **13**, 369–396 (1998)
8. Klock, H., Buhmann, J.M.: Multidimensional scaling by deterministic annealing. In: Pilillo, M., Hancock, E.R. (eds.) Energy Minimization Methods in Computer Vision and Pattern Recognition. Lecture Notes in Computer Science, vol. 1223, pp. 246–260. Springer, Berlin (1997)
9. So, A., Ye, Y.: Theory of semidefinite programming for sensor network localization. Math. Program. **109**, 367–384 (2007)
10. Wang, Z., Zheng, S., Boyd, S., Ye, Y.: Further relaxations of the SDP approach to sensor network localization. SIAM J. Optim. **19**, 655–673 (2008)
11. Zheng, Z.-Z., Luo, X.-L., Wu, Z.: A geometric buildup algorithm for the solution of the sensor network localization problem. Comput. Optim. Appl. (2010, accepted)
12. Hou, J.T., Sims, G.E., Zhang, C., Kim, S.H.: A global representation of the protein fold space. Proc. Natl. Acad. Sci. USA **100**, 2386–2390 (2003)
13. Moré, J.J., Wu, Z.: ε-optimal solutions to distance geometry problems via global continuation. In: Pardalos, P.M., Shalloway, D., Xue, G. (eds.) Global Minimization of Non-convex Energy Functions: Molecular Conformation and Protein Folding, pp. 151–168. American Mathematical Society, Providence (1996)

14. Glunt, W., Hayden, T.L.: Improved convergence and speed for the distance geometry program APA to determine protein structure. Comput. Chem. **25**, 223–230 (2001)
15. Glunt, W., Hayden, T.L., Hong, S., Wells, J.: An alternating projection algorithm for computing the nearest Euclidean distance matrix. SIAM J. Matrix Anal. Appl. **11**, 589–600 (1990)
16. Glunt, W., Hayden, T.L., Raydan, M.: Molecular conformations from distance matrices. J. Comput. Chem. **14**, 114–120 (1993)
17. Hendrickson, B.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
18. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. J. Glob. Optim. **15**, 219–234 (1999)
19. Zou, Z.H., Bird, R.H., Schnabel, R.B.: A stochastic/perturbation global optimization algorithm for distance geometry problems. J. Glob. Optim. **11**, 91–105 (1997)
20. Le Thi, A.H., Pham Dinh, T.: Large scale molecular optimization from distance matrices by a d. c. optimization approach. SIAM J. Optim. **14**, 77–114 (2003)
21. Biswas, P., Toh, K.C., Ye, Y.: A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. SIAM J. Sci. Comput. **30**, 1251–1277 (2008)
22. Grosso, A., Locatelli, M., Schoen, F.: Solving molecular distance geometry problems by global optimization algorithms. Comput. Optim. Appl. **43**, 23–37 (2009)
23. Huang, H.-X., Pardalos, P.M.: A multivariate partition approach to optimization problems. Cybern. Syst. Anal. **38**, 265–275 (2002)
24. Huang, H.-X., Pardalos, P.M., Shen, Z.-J.: Equivalent formulations and necessary optimality conditions for the Lenard-Jones problem. J. Glob. Optim. **22**, 97–118 (2002)
25. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. J. Glob. Optim. **22**, 365–375 (2002)
26. Dong, Q., Wu, Z.: A geometric buildup algorithm for solving the molecular distance geometry problem with sparse distance data. J. Glob. Optim. **26**, 321–333 (2003)
27. Wu, D., Wu, Z.: An updated geometric buildup algorithm for solving the molecular distance geometry problem with sparse distance data. J. Glob. Optim. **37**, 661–673 (2007)
28. Strang, G.: Linear Algebra and Its Applications, 3rd edn. Thomson Learning, Andover (1988)
29. Golub, G.H., Van Loan, C.F.: Matrix Computations. Johns Hopkins University Press, Baltimore (1989)
30. Schmidt, E.: Zur Theorie der linearen und nichtlinearen Integralgleichungen. I Teil. Entwicklung willkürlichen Funktionen nach System vorgeschriebener. Math. Ann. **63**, 433–476 (1907)
31. Weyl, H.: Das asymptotische Verteilungsgesetz der Eigenwert linearer partieller Differentialgleichungen (mit einer Anwendung aufder Theorie der Hohlraumstrahlung). Math. Ann. **71**, 441–479 (1912)
32. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika **1**, 211–218 (1936)
33. Hoffman, A.J., Wielandt, H.W.: The variation of the spectrum of a normal matrix. Duke Math. J. **20**, 37–39 (1953)
34. Golub, G.H., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. SIAM J. Numer. Anal. **2**, 205–224 (1965)
35. Stewart, G.W.: On the early history of the singular value decomposition. SIAM Rev. **35**, 551–566 (1993)
36. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, Berlin (1999)
37. Stewart, G.W.: Perturbation theory for the singular value decomposition. In: Vacarro, R.J. (ed.) SVD and Signal Processing, II. Elsevier, Amsterdam (1991)
38. http://www.pdb.org
39. MATLAB 6.5, The MathWorks Inc. (2003). http://www.mathworks.com