EUCLIDEAN DISTANCE MATRIX COMPLETION PROBLEMS

HAW-REN FANG* AND DIANNE P. O'LEARY†

June 6, 2010

Abstract. A Euclidean distance matrix is one in which the (i, j) entry specifies the squared distance between particle i and particle j. Given a partially-specified symmetric matrix A with zero diagonal, the Euclidean distance matrix completion problem (EDMCP) is to determine the unspecified entries to make A a Euclidean distance matrix.

We survey three different approaches to solving the EDMCP. We advocate expressing the EDMCP as a nonconvex optimization problem using the particle positions as variables and solving using a modified Newton or quasi-Newton method. To avoid local minima, we develop a randomized initialization technique that involves a nonlinear version of the classical multidimensional scaling, and a dimensionality relaxation scheme with optional weighting.

Our experiments show that the method easily solves the artificial problems introduced by Moré and Wu. It also solves the 12 much more difficult protein fragment problems introduced by Hendrickson, and the 6 larger protein problems introduced by Grooms, Lewis, and Trosset.

Key words. distance geometry, Euclidean distance matrices, global optimization, dimensionality relaxation, modified Cholesky factorizations, molecular conformation

AMS subject classifications. 49M15, 65K05, 90C26, 92E10

1. Introduction. Given the distances between each pair of n particles in $\mathbb{R}^r$, $n \geq r$, it is easy to determine the relative positions of the particles. In many applications, though, we are given only some of the distances and we would like to determine the missing distances and thus the particle positions. We focus in this paper on algorithms to solve this distance completion problem.

In this paper we use Euclidean distances $\|x\| = \sqrt{x^T x}$, where x is a column vector with r components. We call $D \in \mathbb{R}^{n\times n}$ a *Euclidean distance matrix* (EDM)[1] if there are points $p_1, p_2, \ldots, p_n$ such that $d_{ij} = \|p_i - p_j\|^2$ for $i, j = 1, 2, \ldots, n$. In particular, every EDM is a nonnegative symmetric matrix with zeros on its main diagonal, but not vice versa.

To define the *Euclidean distance matrix completion problem* (EDMCP), let $\boldsymbol{\mathcal{S}}_n$ denote the set of $n\times n$ real symmetric matrices and call a matrix A *symmetric partial* if some entries are unspecified but all specified entries occur in pairs $a_{ij} = a_{ji}$. The set of *completions* of a symmetric partial matrix A is defined as

$$\boldsymbol{\mathcal{C}}(A) := \{D \in \boldsymbol{\mathcal{S}}_n : d_{ij} = a_{ij} \text{ for all specified entries } a_{ij} \text{ in } A.\}.$$

A matrix $D \in \boldsymbol{\mathcal{C}}(A)$ is called an *EDM completion* of A if D is an EDM. The EDMCP is to find an EDM D that completes a given symmetric partial matrix A.

EDMs have been well-studied (e.g., [12, 13, 29, 35]), and some theoretical properties for EDMCPs have been developed (e.g., [1, 4, 16, 18]). Numerical optimization is widely used to solve EDMCPs (e.g., [2, 14, 17, 24, 33, 36]), but current approaches can be quite expensive and often fail to solve difficult problems. Applications of the EDMCP include determining the conformation of molecules (e.g., [11, 32]) and, in

*Department of Computer Science and Engineering, University of Minnesota, Minneapolis, Minnesota 55455 (hrfang@cs.umn.edu).

†Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland 20742 (oleary@cs.umd.edu).

[1]Some authors define an EDM Δ by $\delta_{ij} = \|p_i - p_j\|$, so our D is their $\Delta \circ \Delta$, where '$\circ$' denotes *Hadamard* (elementwise) product.

particular, proteins (e.g., [14, 15, 25]), using incomplete measurements of interatomic distances.

Following [17, 24, 36], we transform the EDMCP into a nonconvex optimization problem involving particle positions. Our contribution is to develop an efficient and reliable algorithm for solving this problem. We develop a randomized initialization scheme that reduces the iteration count and increases the chance of reaching the global minimum. We also develop a dimensionality relaxation scheme, related to those proposed by Crippen [6, 7], Havel [15], and Purisima and Scheraga [26], to continue the iteration in case a local minimizer is found. To deal with nonconvexity, our system is solved using the modified Newton methods from [9] as well as the BFGS quasi-Newton method [21]. The resulting method allows us to solve very challenging EDMCPs with a high number of unspecified distances.

The rest of this paper is organized as follows. Section 2 reviews the basic properties of EDMs. In Section 3 we present three different optimization problems equivalent to the EDMCP. Section 4 gives our randomized initialization scheme. Section 5 presents our dimensionality relaxation method. Experimental results are reported in Section 6. A conclusion is given in Section 7.

2. The geometry of EDMs. If matrix $D \in \boldsymbol{\mathcal{S}}_n$ has zero diagonal, we call it a *pre-EDM*. Clearly every EDM is a pre-EDM but not vice versa, even if all entries are nonnegative, since, for example, the distances might violate the triangle inequality. It is well-known [2, 13, 29, 35] that a pre-EDM $D \in \boldsymbol{\mathcal{S}}_n$ is an EDM if and only if D is negative semidefinite on the subspace $\{x \in \mathbb{R}^n : x^T e = 0\}$, where e is the column vector of ones. If $V \in \mathbb{R}^{n\times(n-1)}$ is a matrix whose columns form an orthonormal basis for this subspace, then the orthogonal projection matrix J onto this subspace is

$$J := VV^T = I - \frac{ee^T}{n} \in \boldsymbol{\mathcal{S}}_n.$$

Although the choice of V is not unique, J is unique and $J^2 = J$.

Now define the linear operator

$$\mathcal{T}(D) := -\frac{1}{2}JDJ \tag{2.1}$$

with domain $\boldsymbol{\mathcal{S}}_n$. Note that $D \in \boldsymbol{\mathcal{S}}_n$ is an EDM if and only if D is a pre-EDM and $\mathcal{T}(D)$ is positive semidefinite.

An $n\times r$ matrix P is called a *realization* of an EDM D if $PP^T = \mathcal{T}(D) =: B$. The ith row of P, denoted p_i^T, specifies the position of the ith particle. The *Gram matrix* B contains the inner products $b_{ij} = p_i^T p_j$ for $i, j = 1, \ldots, n$. Thus, determining positions from a given distance matrix is a matrix factorization problem. Three properties of an EDM D are of interest to us:

- If P is a realization of D, then since $Je = 0$, we see that $Be{=}0$ and $P^T e{=}0$. Thus the centroid of the points $p_1, p_2, \ldots, p_n$ is at the origin.
- If P is a realization of D, then so is PU for any orthogonal U.
- If $B = \mathcal{T}(D)$ has rank r, then there exists a realization $P \in \mathbb{R}^{n\times r}$ of D, and P must also have rank r. Thus there is no proper hyperplane in $\mathbb{R}^r$ that contains the points $p_1, p_2, \ldots, p_n$.

We use $X \succeq 0$ to indicate that matrix X is positive semidefinite. Denote the set of $n\times n$ symmetric positive semidefinite matrices by

$$\boldsymbol{\mathcal{S}}_n^+ := \{S \in \boldsymbol{\mathcal{S}}_n : S \succeq 0\}.$$

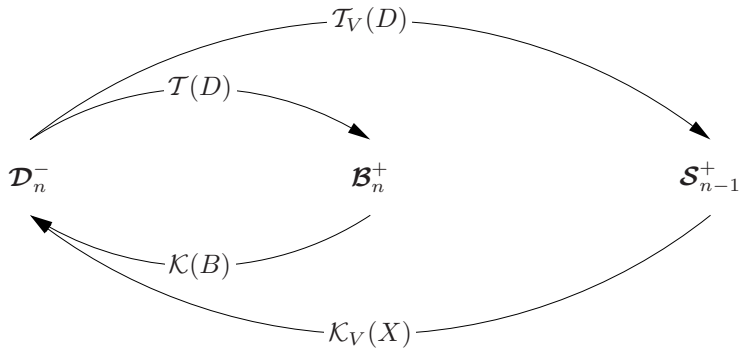

FIG. 2.1. *Relationships among the linear operators.*

We denote the column vector formed from the diagonal elements in D by $\mathrm{diag}(D)$. The two sets

$$\begin{aligned}\boldsymbol{\mathcal{B}}_n^+ &:= \{B \in \boldsymbol{\mathcal{S}}_n : Be = 0, B \succeq 0\}\\ \boldsymbol{\mathcal{D}}_n^- &:= \{D \in \boldsymbol{\mathcal{S}}_n : \mathrm{diag}(D) = 0 \text{ and } x^T Dx \le 0 \text{ for } x^T e = 0\}\end{aligned}$$

will be of particular interest to us. $\boldsymbol{\mathcal{S}}_n^+$, $\boldsymbol{\mathcal{B}}_n^+$, and $\boldsymbol{\mathcal{D}}_n^-$ are subspaces of $\boldsymbol{\mathcal{S}}_n$ and convex. Now define the linear operator

$$\mathcal{K}(B) := \mathrm{diag}(B)\, e^T + e\, \mathrm{diag}(B)^T - 2B, \tag{2.2}$$

where $B \in \boldsymbol{\mathcal{S}}_n$. The operators $\mathcal{T}$ defined in (2.1) and $\mathcal{K}$ have an interesting relationship, explained in the following lemma.

LEMMA 2.1. *The linear operators $\mathcal{T}$ and $\mathcal{K}$ satisfy*

$$\mathcal{T}(\boldsymbol{\mathcal{D}}_n^-) = \boldsymbol{\mathcal{B}}_n^+ \text{ and } \mathcal{K}(\boldsymbol{\mathcal{B}}_n^+) = \boldsymbol{\mathcal{D}}_n^-.$$

Moreover, $\mathcal{T}$ on $\boldsymbol{\mathcal{D}}_n^-$ is the inverse function of $\mathcal{K}$ on $\boldsymbol{\mathcal{B}}_n^+$.

Proof. Any matrix $D \in \boldsymbol{\mathcal{D}}_n^-$ is negative semidefinite on $\{x \in \mathbb{R}^n : x^T e = 0\} = \{Jy : y \in \mathbb{R}^n\}$. This implies that $\mathcal{T}(D) = -\frac{1}{2}JDJ^T$ is positive semidefinite and therefore in $\boldsymbol{\mathcal{B}}_n^+$, so $\mathcal{T}(\boldsymbol{\mathcal{D}}_n^-) \subseteq \boldsymbol{\mathcal{B}}_n^+$.

Now suppose that $B \in \boldsymbol{\mathcal{B}}_n^+$. Observe that by the definition of $\mathcal{K}$ we see that $\mathcal{K}(B)$ is symmetric with zero diagonal. For any $x \in \mathbb{R}^n$ with $x^T e = 0$, $x^T \mathcal{K}(B) x = -2x^T Bx$. Therefore, $\mathcal{K}(B) \in \boldsymbol{\mathcal{D}}_n^-$, so $\mathcal{K}(\boldsymbol{\mathcal{B}}_n^+) \subseteq \boldsymbol{\mathcal{D}}_n^-$.

The fact that $\mathcal{T}$ on $\boldsymbol{\mathcal{D}}_n^-$ is the inverse function of $\mathcal{K}$ on $\boldsymbol{\mathcal{B}}_n^+$ follows from (2.1) and (2.2), and from this we conclude that $\mathcal{T}(\boldsymbol{\mathcal{D}}_n^-) = \boldsymbol{\mathcal{B}}_n^+$ and $\mathcal{K}(\boldsymbol{\mathcal{B}}_n^+) = \boldsymbol{\mathcal{D}}_n^-$. ∎

We now study the two composite linear operators

$$\begin{aligned}\mathcal{T}_V(D) &:= V^T \mathcal{T}(D) V = -\tfrac{1}{2} V^T D V\\ \mathcal{K}_V(X) &:= \mathcal{K}(VXV^T).\end{aligned}$$

LEMMA 2.2. *The linear operators $\mathcal{T}_V$ and $\mathcal{K}_V$ satisfy*

$$\mathcal{T}_V(\boldsymbol{\mathcal{D}}_n^-) = \boldsymbol{\mathcal{S}}_{n-1}^+ \text{ and } \mathcal{K}_V(\boldsymbol{\mathcal{S}}_{n-1}^+) = \boldsymbol{\mathcal{D}}_n^-.$$

Moreover, $\mathcal{T}_V$ on $\boldsymbol{\mathcal{D}}_n^-$ is the inverse function of $\mathcal{K}_V$ on $\boldsymbol{\mathcal{S}}_{n-1}^+$.

Proof. This follows from Lemma 2.1 and the fact that $V^T V = I$. ∎

We summarize our notation in Table 2.1 and summarize the relationships among the linear operators $\mathcal{T}$, $\mathcal{K}$, $\mathcal{T}_V$, and $\mathcal{K}_V$ in Figure 2.1. By Lemma 2.1 and Lemma 2.2, we conclude that for a given pre-EDM D, the following four conditions are equivalent:

1. D is an EDM.
2. $D \in \boldsymbol{\mathcal{D}}_n^-$.
3. $\mathcal{T}(D) \in \boldsymbol{\mathcal{B}}_n^+$.
4. $\mathcal{T}_V(D) \in \boldsymbol{\mathcal{S}}_{n-1}^+$.

These equivalences give us several approaches to the EDMCP, presented in the next section.

$n \times n$ real symmetric matrices	$\boldsymbol{\mathcal{S}}_n$
symmetric positive semidefinite matrices	$\boldsymbol{\mathcal{S}}_n^+ := \{S \in \boldsymbol{\mathcal{S}}_n : S \succeq 0\}$
completions of a symmetric partial matrix	$\boldsymbol{\mathcal{C}}(A) := \{D \in \boldsymbol{\mathcal{S}}_n : d_{ij} = a_{ij}$ for all specified entries a_{ij} in $A\}$
a subspace of centered matrices	$\boldsymbol{\mathcal{B}}_n^+ := \{B \in \boldsymbol{\mathcal{S}}_n : Be = 0, B \succeq 0\}$
a subspace of hollowed matrices	$\boldsymbol{\mathcal{D}}_n^- := \{D \in \boldsymbol{\mathcal{S}}_n : \mathrm{diag}(D) = 0$ and $x^T Dx \le 0$ when $x^T e = 0\}$
Orthog. projector onto $\{x \in \mathbb{R}^n : x^T e = 0\}$	$J := VV^T = I - \frac{ee^T}{n} \in \boldsymbol{\mathcal{S}}_n$, with $V^T V = I$
linear operators on $\boldsymbol{\mathcal{S}}_n$:	$\mathcal{T}(D) := -\frac{1}{2} JDJ$
	$\mathcal{T}_V(D) := V^T \mathcal{T}(D) V = -\frac{1}{2} V^T DV$
	$\mathcal{K}(B) := \mathrm{diag}(B)\, e^T + e\, \mathrm{diag}(B)^T - 2B$
	$\mathcal{K}_V(X) := \mathcal{K}(VXV^T)$
D is a pre-EDM if	$D \in \boldsymbol{\mathcal{S}}_n$ and $\mathrm{diag}(D) = 0$
D is an $n \times n$ EDM if	D is a pre-EDM and $B = \mathcal{T}(D) \in \boldsymbol{\mathcal{B}}_n^+$.
$n \times r$ position matrix	P
$n \times n$ Gram matrix	$B = PP^T = \mathcal{T}(D)$, where D is an EDM
$n \times n$ partial EDM	A
$n \times n$ weighting matrix	H
minimization function	$f_{A,H}(P) := \|H \circ (A - \mathcal{K}(PP^T))\|_F^2$

TABLE 2.1
Summary of notation.

3. Solving the EDMCP via numerical optimization. The EDMCP has been formulated in the literature as several different but related optimization problems, taking advantage of the relationships in Figure 2.1. We present three of these formulations in Sections 3.1–3.3 in order to give reasons for our own choice. In all cases, the given $n \times n$ symmetric partial matrix A has a distance matrix completion if and only if the global minimum of these problems is zero.

3.1. The dissimilarity parameterized formulation. Trosset [33] and Grooms, Lewis, and Trosset [14] started from the formulation

$$\begin{cases} \underset{B,D}{\text{minimize}} & \|B - \mathcal{T}(D)\|_F^2 \\ \text{subject to} & D \in \boldsymbol{\mathcal{C}}(A),\ B \in \boldsymbol{\mathcal{B}}_n^+,\ \mathrm{rank}(B) \le r. \end{cases} \tag{3.1}$$

This approach is related to the classical multidimensional scaling (MDS) described in [12, 31]. Both transform a distance matrix D to $\mathcal{T}(D)$ and find a Gram matrix $B \in \boldsymbol{\mathcal{B}}_n^+$ that best matches $\mathcal{T}(D)$.

Given a symmetric matrix $B \in \mathbb{R}^{n \times n}$, define

$$F_r(B) = \sum_{i=r+1}^{n} \lambda_i(B)^2 + \sum_{i=1}^{r} (\lambda_i(B) - \max\{\lambda_i(B), 0\})^2,$$

where $\lambda_i(B)$ is the ith largest eigenvalue of B. Then a pre-EDM D is an EDM with $\operatorname{rank}(\mathcal{T}(D)) \le r$ if and only if $F_r(\mathcal{T}(D)) = 0$. Hence problem (3.1) can be transformed into

$$\begin{cases} \underset{D}{\text{minimize}} & F_r(\mathcal{T}(D)) \\ \text{subject to} & D \in \boldsymbol{\mathcal{C}}(A). \end{cases} \tag{3.2}$$

The distance matrix D is indeed a dissimilarity matrix. Therefore, solving the EDMCP via the nonconvex formulation (3.2) is called the dissimilarity parameterized approach [14]. The number of variables in this formulation is proportional to the number of unspecified distances, so it is favored when this is small (i.e., less than $O(n^2)$). With efficient computational treatment [14], it can also be applied to solve problems with a large number of unspecified distances.

3.2. Convex semidefinite programming. Alfakih, Khandani and Wolkowicz [2] solved the EDMCP via semidefinite programming. They compute the solution as $D = \mathcal{K}_V(X)$, where X solves

$$\begin{cases} \underset{X}{\text{minimize}} & \|H \circ (A - \mathcal{K}_V(X))\|_F^2 \\ \text{subject to} & X \in \boldsymbol{\mathcal{S}}_{n-1}^+, \end{cases} \tag{3.3}$$

where '$\circ$' denotes the Hadamard (elementwise) product and H is a weight matrix such that $h_{ij} > 0$ if a_{ij} is specified, and otherwise $h_{ij} = 0$. If $h_{ij} := 1$ for all specified entries, then we minimize the Frobenius norm of the partial error matrix. Alternatively, if relative distance errors are the concern, set $h_{ij} := 1/a_{ij}$. The weights of the distances may also depend on their relative uncertainty.

This optimization problem is convex; hence any local minimizer will also be a global minimizer, a very desirable property. On the other hand, the domain is $\boldsymbol{\mathcal{S}}_{n-1}^+$, so the number of variables is $O(n^2)$, which can be quite large.

Note that the embedding dimension r is not specified in problem (3.3). As a result, the minimization algorithm might converge to a minimizer X to (3.3) with high rank. In this case, an algorithm in [3] can be applied to reduce the rank of X to some extent without leaving the minimum. This issue also occurs in our dimensionality relaxation scheme, which we will discuss in Section 5.

3.3. The nonconvex position formulation. Instead of solving the optimization problem (3.2) or (3.3), our work uses the formulation

$$\begin{cases} \underset{B}{\text{minimize}} & \|H \circ (A - \mathcal{K}(B))\|_F^2 \\ \text{subject to} & B \in \boldsymbol{\mathcal{B}}_n^+,\ Be = 0,\ \operatorname{rank}(B) = r, \end{cases} \tag{3.4}$$

where H is a weight matrix. Here we explicitly impose the embedding dimension r in the constraints.

We use the relation between an EDM D and its realization $P \in \mathbb{R}^{n\times r}$, with $B = PP^T$ and $Pe = 0$. We rewrite problem (3.4) as the following problem which we call the *position formulation*:

$$\begin{cases} \underset{P}{\text{minimize}} & \|H \circ (A - \mathcal{K}(PP^T))\|_F^2 =: f_{A,H}(P) \\ \text{subject to} & P \in \mathbb{R}^{n\times r}. \end{cases} \tag{3.5}$$

A careful reader may notice that the equality constraints $Pe = 0$ are not present in (3.5). Lemma 3.1 shows that removing these equality constraints does not change the global minimum, and therefore they are dispensable.

LEMMA 3.1. *Suppose we are given $P \in \mathbb{R}^{n\times r}$ with each row representing a point in Euclidean space. Let $P_1 = PU + eq^T$ where $U \in \mathbb{R}^{r\times r}$ is an orthogonal matrix and q is an arbitrary vector.. Then*

$$\mathcal{K}(P_1 P_1^T) = \mathcal{K}(PP^T),$$

where the linear operator $\mathcal{K}$ is defined in (2.2).

Proof. Using the fact that $UU^T = I$, we calculate

$$\begin{aligned}\mathcal{K}(P_1P_1^T) - \mathcal{K}(PP^T) &= \mathcal{K}(P_1P_1^T - PP^T)\\ &= \mathcal{K}(PUqe^T + eq^TU^TP^T + eq^Tqe^T)\\ &= \mathcal{K}(PUqe^T + eq^TU^TP^T) + q^Tq\mathcal{K}(ee^T)\\ &= 0.\end{aligned}$$

The last equality follows from the fact that $\mathcal{K}(pe^T + ep^T) = 0$ for all vectors p. ∎

By Lemma 3.1, the objective function in (3.5) is invariant under the rigid transformation of coordinates, i.e., applying an orthogonal transformation U^T to each point and then translating by a fixed vector q. Therefore, it is not necessary to have the centroid of the points in P at the origin, so the constraints $Pe = 0$ are dispensable.

Compared with (3.2) and (3.3), problem (3.5) has the key advantage of a relatively small number of variables, since $P \in \mathbb{R}^{n\times r}$. For real problems such as protein structure prediction, r is a small constant (e.g., $r = 3$), and therefore the problem size is $O(n)$. On the other hand, since the problem (3.5) is not convex, a minimization algorithm can converge to a local minimizer, whereas a global minimizer is required to solve the EDMCP.

3.4. Previous use of the position formulation. The objective function in (3.5) can be written as

$$f_{A,H}(P) = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}^2(a_{ij} - d_{ij})^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}^2(a_{ij} - \|p_i - p_j\|^2)^2. \tag{3.6}$$

The last term has been frequently used in the literature (e.g., [17, 24, 25, 36]). Some authors (e.g., [36]) also minimize the related function

$$\max_{i,j=1,\ldots,n} h_{ij}^2(a_{ij} - \|p_i - p_j\|^2)^2 \tag{3.7}$$

to solve the EDMCP. Another alternative (e.g., [6, 7, 11, 15]) is to minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}(\sqrt{a_{ij}} - \|p_i - p_j\|)^2. \tag{3.8}$$

This is sometimes called the *energy function* (e.g., [6, 7]).

Note that (3.6), (3.7), and (3.8) are three different distance constraint violation measurements. They are all bounded below by zero, and any zero value of them implies that the other two also have value zero.

Several methods have been proposed to solve the EDMCPs via the position formulation. For example, Crippen [6, 7] suggested a dimensionality relaxation scheme, which he called *energy embedding*. Glunt, Hayden, and Raydan [11] proposed the spectral gradient method and the data box algorithm. Havel [15] used simulated annealing

optimization. Hendrickson [17] presented a divide-and-conquer algorithm. Moré and Wu [24] used a smoothing and continuation method via a Gaussian transformation. Zou, Bird, and Schnabel [36] developed a stochastic/perturbation algorithm. The authors in [17, 24, 36] minimize (3.6), whereas the authors in [6, 7, 11, 15] minimize (3.8). Our method consists of multiple ingredients. Two of the most important ones are a randomized initialization technique and a dimensionality relaxation scheme that will be described in Sections 4 and 5, respectively. Our implementation minimizes (3.6) or (3.7) but can be adapted to minimize (3.8).

4. Randomized initialization. Consider the position formulation (3.5). A good initial estimate of the configuration can speed up the convergence and increase the chance of reaching the global minimum. Our initialization method is essentially a multistart algorithm. However, rather than randomly generating the coordinates in the sampling domain (e.g., [24, 36]), we use the existing information from the partial EDM for a better estimation of the configuration. More specifically, our approach is to fill the unspecified entries of A to form a pre-EDM F and then use metric multidimensional scaling to obtain an initial configuration $P \in \mathbb{R}^{n\times r}$. The details come next.

4.1. Nonlinear multidimensional scaling. Our observation is that each $n\times n$ partial EDM A is canonically associated with an undirected graph $G = (V, E)$ with vertices in $V = \{1, \ldots, n\}$ corresponding to the particles. For every specified entry d_{ij}, we place an edge between the ith and jth vertices $(i, j) \in E$ with weight equal to the distance $\sqrt{d_{ij}}$. This graph G is connected if and only if A_0, with zeros entered for missing distances, is irreducible. If A_0 is reducible, then this EDMCP can be partitioned into two or more independent EDMCPs. Hence we assume that G is connected.

Using this graph, we initialize each missing entry a_{ij} to be the squared length of the shortest path between vertices i and j to obtain a pre-EDM $F \in \mathbb{R}^{n\times n}$. This requires solving the all-pairs shortest path problem. In our implementation we use the Floyd-Warshall algorithm [10, 34].

Recall that a pre-EDM $F \in \mathbb{R}^{n\times n}$ is an EDM corresponding to a configuration $P \in \mathbb{R}^{n\times r}$ in the r-dimensional space if and only if $\mathcal{T}(F) \in \mathbb{R}^{n\times n}$ is positive semidefinite of rank r. Therefore, we consider the problem

$$\text{(4.1)} \qquad \begin{cases} \underset{B}{\text{minimize}} & \|\mathcal{T}(F) - B\|_F^2 \\ \text{subject to} & B \in \boldsymbol{\mathcal{S}}_n^+,\ \text{rank}(B) \le r. \end{cases}$$

The minimizer is the truncated spectral decomposition of $\mathcal{T}(F)$, formed from the r largest positive eigenvalues and the corresponding eigenvectors. If there are fewer than r positive eigenvalues, we take only the positive ones. The result is denoted by $B := U_r\Lambda_r U_r^T$, which yields the configuration $P := U_r\Lambda_r^{1/2}$. This discussion leads to Algorithm 1.

This initialization method is particularly relevant to molecular problems, where the specified entries are from the measurement of interatomic distances, mostly the shorter ones. In this case the lengths of the shortest paths approximate the geodesic distances which are upper bounds on the actual distances.

It is worth noting that algorithmically, the method just described is very similar to ISOMAP, a nonlinear version of multidimensional scaling applied to manifold learning [30]. The difference is that we use the graph from the partial EDM, whereas in [30]

function P=CONFIG_INIT(F, r)
{Purpose: obtain a configuration $P \in \mathbb{R}^{n\times r}$ from pre-EDM $F \in \mathbb{R}^{n\times n}$ by solving (4.1).}
Compute the lead eigenvalues $\lambda_1, \ldots, \lambda_r$ of $\mathcal{T}(F)$ and their eigenvectors $u_1, \ldots, u_r$;
$\lambda_i := \max\{0, \lambda_i\}$ for $i = 1, \ldots, r$;
$\Lambda_r := \mathrm{diag}(\lambda_1, \ldots, \lambda_r)$;
$U_r := [u_1, \ldots, u_r]$;
$P := U_r \Lambda_r^{1/2}$; ▷ $B := PP^T$ is the minimizer of (4.1)
end function

Algorithm 1: Configuration initialization.

the graph of the all-pairs shortest path problem is a k-nearest-neighbor graph of the high-dimensional input data.

4.2. Randomized geodesic distance scaling. Our numerical experience indicates that the initialization method in Section 4.1 is very effective for easier EDMCPs. For example, coupled with appropriate local minimization algorithms, it solves all the artificial problems from Moré and Wu [23, 24]; see Section 6.1 for details. However, for challenging EDMCPs, such as the difficult protein fragment problems presented by Hendrickson [17], this initialization method is far from sufficient.

The problem is that the all-pairs shortest path problem can greatly overestimate the missing distances, especially if the paths are long. We have used D to denote the EDM and F to denote the pre-EDM from solving the all-pair shortest path problem. In molecular problems, the measured distances are generally the shortest ones. In this case, if the shortest path between particles i and j in the graph has k segments, then

$$\frac{f_{ij}}{k^2} \le d_{ij} \le f_{ij}. \tag{4.2}$$

Let $s_{ij} := f_{ij}/d_{ij}$. We collect all scalars s_{ij} with f_{ij} consisting of k segments and compute the cumulative distribution. Figure 4.1 shows the result of the smallest (`20.unique`) and largest (`124.reduced`) EDMCPs from Hendrickson's test set [17] for $k = 2, 3, 4, 5$. We can see that most often $d_{ij} \ge f_{ij}/1.5$, demonstrating that the denominator value k^2 in (4.2) tends to be too pessimistic.

Motivated by these considerations, we propose forming a pre-EDM matrix $\widehat{F}$ with filled entries set to

$$\widehat{f}_{ij} = \frac{f_{ij}}{s_{ij}},$$

where f_{ij} is determined from the shortest path problem, $s_{ij} \in [1, k_{ij}]$ is normally distributed with mean 1.5 and standard deviation 0.3 (tails truncated), and k_{ij} is the number of segments in the shortest path between particle i and particle j.

Once the pre-EDM $\widehat{F}$ is obtained from scaling the filled entries in F, we can obtain an initial configuration $P :=$ CONFIG_INIT($\widehat{F}, r$) by Algorithm 1. The scaling is randomized so it naturally leads to a multistart algorithm for solving the position formulation.

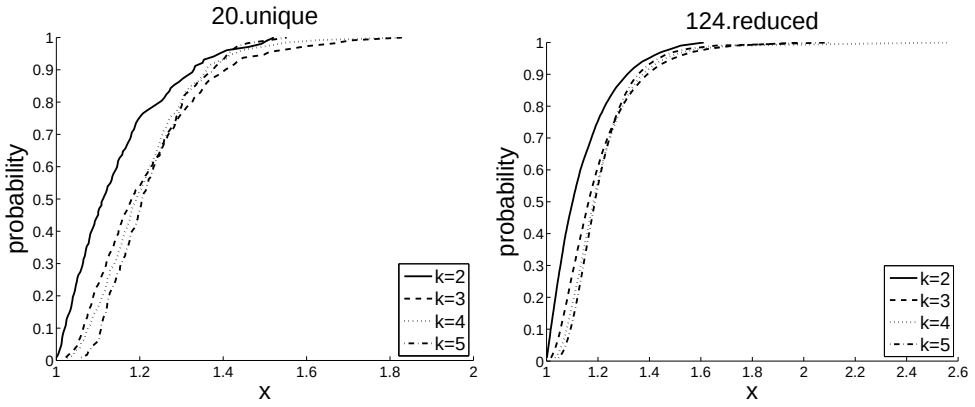

FIG. 4.1. *Probability that $s \leq x$ for two EDMCPs from Hendrickson's test set.*

4.3. Alternate initializations for our algorithm. The observations of Sections 4.1 and 4.2 lend insight into initializations that may improve the chance to solve difficult EDMCPs.

Firstly, to solve (3.5), we randomly generate a number of configurations P by the method described in Section 4.2, and pick the one with the smallest value $f_{A,H}(P)$. The selected configuration potentially has a better chance than the rest to reach the global minimum.

An alternative to our first heuristic is to keep generating configurations P until $f_{A,H}(P)$ is below a given cutoff; see, for example, [36]. For simplicity, in our experiments, we initialize each local minimization using the best configuration P (i.e., the one with least $f_{A,H}(P)$) among 100 randomly generated configurations.

Secondly, we may *stretch* the generated configuration P (i.e., multiply it by a scaling factor) and use the stretched configuration to initialize the minimization. Intuitively, the stretching would be inappropriate, since it results in serious violations of distance constraints. However, we found that in practice the stretching may increase the chance to lead to the global minimum of (3.5).

The idea of stretching in distance geometry problems is not new. For example, it has been applied in the local minimization phase of the algorithm in [36], where the stretching factor is randomly chosen from the range $[1.25, 2.5]$. In our experiments, we use either the stretching factor 2.0 or no stretching. A randomized stretching factor might improve our results.

4.4. The resulting algorithm. We summarize these ideas in Algorithm 2. We choose to minimize $f_{A,H}(P)$ in (3.6); alternatively, one may also use (3.7) or (3.8). A local minimization algorithm is required in (*) and (**) and there are many possible choices. Our implementation uses the optimization package `OPT++` [22], and we have included Newton's method implemented with various modified Cholesky algorithms for generating descent search directions [9], as well as the BFGS quasi-Newton method [21]. For difficult problems, this is not sufficient, so we consider an alternative, DIM_RELAX_LOCAL_MIN, in the next section.

5. Dimensionality relaxation. We present in this section a local minimization algorithm that uses dimensionality relaxation. This algorithm can improve the chance of reaching the global minimizer.

Input: an $n \times n$ partial EDM A; the embedding dimension r; and a weight matrix $H \in \mathbb{R}^{n \times n}$ associated with A.
Output: a configuration $P \in \mathbb{R}^{n \times r}$ such that $\mathcal{K}(PP^T)$ is an EDM completion to A.
Get pre-EDM F by solving the all-pairs shortest path problem from A; ▷ Sec. 4.1
$P :=$ CONFIG_INIT(F, r); ▷ Alg. 1
Optionally stretch P; ▷ Sec. 4.3
$P_{\text{sol}} =$ LOCAL_MIN$(f_{A,H}(P), P)$; ▷ (*)
or $P_{\text{sol}} =$ DIM_RELAX_LOCAL_MIN(A, H, F); ▷ Sec. 5
while $f_{A,H}(P_{\text{sol}}) > 0$ **do**
Generate 100 matrices $\widehat{F}$ by randomly scaling entries in F; ▷ Sec. 4.2
Compute $P :=$ CONFIG_INIT$(\widehat{F}, r)$ for all $\widehat{F}$; ▷ Algorithm 1
Pick the P with least $f_{A,H}(P)$; ▷ see (3.6)
{The corresponding pre-EDM is denoted by $\widetilde{F}$;}
Optionally stretch P; ▷ Sec. 4.3
$P_{\text{sol}} =$ LOCAL_MIN$(f_{A,H}(P), P)$; ▷ (**)
or $P_{\text{sol}} =$ DIM_RELAX_LOCAL_MIN$(A, H, \widetilde{F})$; ▷ Sec. 5
end while

Algorithm 2: A multistart algorithm to solve the position formulation.

5.1. The framework. When the embedding dimension is artificially large, it can be easier to reach the global minimum. For example, there may be no way to pass between two local minimizers for a set of particles in two dimensions without increasing the objective function; but if we allow the same set of particles to move in three dimensions, there may be such a path. This observation leads to the idea of artificially relaxing the embedding dimension when local minimization cannot solve the position formulation in the desired r-dimensional space.

With a pre-EDM $\widetilde{F}$, we can obtain an initial configuration by Algorithm 1 and then hopefully a solution by a local minimization program in the larger-dimension space. This solution may be utilized to initialize a guess in a reduced-dimension space, repeating until we reach the desired dimension. We are aided by the fact that our minimization function is 0 for a global minimizer, so we can easily recognize failure. If failure occurs, we could restart in an even higher-dimensional space.

There are three issues to be resolved: how much to increase the dimension, how fast to reduce the dimension, and how to use the global minimizer in a higher-dimensional space to initialize a minimization algorithm in a lower-dimensional space.

Regarding the first issue, to avoid extra work, we always first start with dimension r. If this fails to find a global minimizer, we restart the algorithm from dimension $r+1$, then from $r+2$, and repeat up to r_{max} (a user-defined parameter) if necessary.

Regarding the second issue, we again let the user define the dimension reduction parameter d. The extreme cases are to reduce directly from the increased dimension r' to r (i.e., $d := r' - r$) or to use the most gradual reduction (i.e., $d := 1$). As will be seen from Table 6.4, gradual reduction is more reliable but also more costly.

The remaining issue is how to extract lower-dimensional coordinates from the higher-dimensional ones. Alfakih and Wolkowicz [3] gave an iterative algorithm based on semidefinite programming. Crippen [6, 7] projected along the direction of the eigenvector corresponding to the smallest eigenvalue of the *weighted inertial tensor matrix* of the interatomic separation vectors. Havel [15] used four-dimensional relax-

ation and simulated annealing to find a rigid transformation to approximately project back to three-dimensional space. Purisima and Scheraga [26] used Cayley-Menger determinants to reduce the dimensionality. In addition, some standard dimensionality reduction techniques for data mining, such as principal component analysis (PCA), are also applicable here.

In our work, we use linear projection or nonlinear programming to extract the lower-dimensional coordinates. We discuss these approaches in the next two sections.

5.2. Dimensionality reduction by linear projection. We now present a linear projection method for dimensionality reduction and relate it to some methods in the literature. The problem is formulated as follows. We are given a higher-dimensional configuration $P \in \mathbb{R}^{n\times r'}$, and want to find a lower dimensional configuration $Q \in \mathbb{R}^{n\times r}$ by projection $Q = PU$, such that Q preserves the distance information as best as possible, where $U \in \mathbb{R}^{r'\times r}$ is the projection matrix.

In EDMCP, only distances specified in the $n \times n$ partial EDM A are of interest, and the distance information to preserve is $H \circ \mathcal{K}(PP^T)$ where H contains the given weights. We choose to limit ourselves to orthogonal projections, and determine Q by solving the optimization problem:

$$\left\{ \begin{array}{ll} \underset{Q}{\text{minimize}} & \|H \circ (\mathcal{K}(PP^T) - \mathcal{K}(QQ^T))\|_S \\ \text{subject to} & Q = PU,\ U^TU = I,\ U \in \mathbb{R}^{r'\times r}, \end{array} \right. \tag{5.1}$$

where $\|\cdot\|_S$ is the sum of the magnitudes of the elements in the matrix.

Denote the rows of P by p_i^T and those of Q by q_i^T. Since $\mathcal{K}(PP^T)$ and $\mathcal{K}(QQ^T)$ are the EDMs of the configurations P and Q, respectively, the objective function in (5.1) is

$$\sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij} \mid \|p_i - p_j\|_2^2 - \|q_i - q_j\|_2^2 \mid . \tag{5.2}$$

Let the columns of $\bar{U} \in \mathbb{R}^{r'\times(r'-r)}$ be an orthogonal basis for the orthogonal complement of the space spanned by the columns of $U \in \mathbb{R}^{r'\times r}$. Then for $i, j = 1, \ldots, n$,

$$\|p_i - p_j\|_2^2 - \|q_i - q_j\|_2^2 = \| \left[\begin{array}{c} U^T \\ \bar{U}^T \end{array} \right] (p_i - p_j)\|_2^2 - \|U^T(p_i - p_j)\|_2^2 = \|\bar{U}^T(p_i - p_j)\|_2^2.$$

Therefore, we can simplify (5.2) by observing that

$$\begin{aligned} & \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}(\|p_i - p_j\|_2^2 - \|q_i - q_j\|_2^2) \\ = & \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}\|\bar{U}^T(p_i - p_j)\|_2^2 \\ = & \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}(p_i - p_j)^T\bar{U}\bar{U}^T(p_i - p_j) \\ = & \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}\ \mathrm{trace}(\bar{U}^T(p_i - p_j)(p_i - p_j)^T\bar{U}) \\ = & \ \mathrm{trace}(\bar{U}^T(\sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}(p_i - p_j)(p_i - p_j)^T)\bar{U}). \end{aligned}$$

The minimum is obtained by choosing the columns of $\bar{U}$ to span the subspace defined by the eigenvectors of

$$\sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}(p_i - p_j)(p_i - p_j)^T \tag{5.3}$$

corresponding to the $r' - r$ smallest eigenvalues. Equivalently, we choose the columns of $U \in \mathbb{R}^{r' \times r}$ to span the subspace defined by the eigenvectors corresponding to the r largest eigenvalues of (5.3). Then $Q = PU$ minimizes (5.1).

This method of dimensionality reduction is related to some methods in the literature:

- The formula used here is mathematically the same as that used by Crippen [6, 7] in energy embedding, where the weighted inertial tensor matrix is our (5.3). Crippen [6] showed that when $r' - r = 1$ (i.e., the dimension is reduced by one), the formula minimizes (5.1). Our analysis is more general.
- If $h_{ij} = 1$ for $i, j = 1, \ldots, n$ (i.e., unit weights), then (5.3) simplifies to

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(p_i - p_j)(p_i - p_j)^T = \sum_{i=1}^{n}\sum_{j=1}^{n} p_i p_i^T - p_i p_j^T - p_j p_i^T + p_j p_j^T$$
$$= nP^TP - P^Tee^TP - P^Tee^TP + nP^TP = 2nP^T(I - \frac{1}{n}ee^T)P = 2n\bar{P}^T\bar{P},$$

  where $\bar{P} = (I - \frac{1}{n}ee^T)P$. The resulting method is the same as principal component analysis, which is known to be equivalent to classical multidimensional scaling. See, for example, [12].
- Now allow negative and unsymmetric weights. If the weights are normalized such that $\sum_j h_{ij} = 1$ for $i = 1, \ldots, n$, and if H^TH is diagonal, then the matrix in (5.3) is the one used in orthogonal neighborhood preserving projections (ONPP) [19, 20]. The difference is that here we compute the eigenvectors that correspond to the largest eigenvalues, whereas ONPP projects with eigenvectors associated with the smallest eigenvalues. The weights are also obtained differently.

5.3. Dimensionality reduction by nonlinear programming. We now give a new nonlinear dimensionality-reduction method, solving the following nonlinear programming problem:

$$\begin{cases} \underset{W}{\text{minimize}} & \|W_2\|_F^2 \\ \text{subject to} & f_{A,H}(W) \le 0,\ W = [W_1, W_2] \in \mathbb{R}^{n \times r'}, \end{cases} \tag{5.4}$$

where $W \in \mathbb{R}^{n \times r'}$ is a configuration in r'-dimensional space, W_1 has dimension $n \times r$, and $f_{A,H}(W)$ is the distance constraint violation measurement of the EDMCP defined in (3.6), which again can be replaced by (3.7) or (3.8). The objective function $\|W_2\|_F^2$ is the sum of squared coordinate values in the dimensions higher than r.

The constraint $f_{A,H}(W) \le 0$ forces $W \in \mathbb{R}^{n \times r'}$ to be a solution to the EDMCP in the r'-dimensional space. We use the configuration W_1 resulting from (5.4) as the estimated solution in the lower r-dimensional space. If the EDMCP has a solution in r-dimensional space, then the global minimum of (5.4) is zero and the corresponding W_1 is a solution. Since $f_{A,H}(W)$ is nonconvex, a local optimization algorithm may

fail to find a global minimizer, but the solution it computes still gives an estimate of the solution.

We can use the solution $P \in \mathbb{R}^{n\times r'}$ to the EDMCP in the higher r'-dimensional space as our initial guess in (5.4). It is a feasible initial guess since $f_{A,H}(P) = 0$. In our system, we use an interior point method implemented in `OPT++` [22] to solve (5.4). We have also incorporated the modified Cholesky algorithms listed in [9] to promote global convergence.

The linear dimensionality reduction method in Section 5.2, applied to the solution to the EDMCP in the higher r'-dimensional space, can be used to find a better initial feasible point $\widetilde{P} := P\widetilde{U}$ for solving (5.4). Two points deserve discussion. Firstly, since the orthogonal transformation $\widetilde{U}$ preserves distances between particles, $f_{A,H}(\tilde{P}) = 0$, so $\widetilde{P}$ is a feasible starting point for (5.4). Secondly, we can measure the quality of an estimate W by how closely W_1 satisfies $f_{A,H}(W_1) \leq 0$. The discussion in Section 5.2 indicates that the matrix formed from the first r columns $\widetilde{P}$ gives a global minimizer of (5.1). Therefore, this matrix nearly solves the constraint and is a good starting guess.

We used this guess in our experiments. As will be seen from Table 6.4, dimensionality reduction by nonlinear programming requires significantly more computation but is more robust than dimensionality reduction by linear projection.

Recall that the ultimate goal is to satisfy the distance constraints defined by the partial distance matrix A. Requiring W_2 to be zero does not differentiate the constrained distances from the zero-weighted ones. Therefore, a better alternative is to replace the objective function $\|W_2\|_F^2$ in (5.4) by

$$\|H \circ \mathcal{K}(W_2 W_2^T)\|_F^2, \tag{5.5}$$

where H is the weight matrix in (3.5). All the discussion above remains valid with this change.

5.4. Minimization using dimensionality relaxation. Algorithm 3, the routine DIM_RELAX_LOCAL_MIN, incorporates dimensionality reduction, using either the linear projection of Section 5.2 or the nonlinear programming approach of in Section 5.3. This algorithm can be used instead of LOCAL_MIN in Algorithm 2.

6. Experimental results. We now discuss our implementation and numerical results on two sets of test problems, those of Moré and Wu, those of Hendrickson, and those of Grooms Lewis, and Trosset.

Algorithm 2 and Algorithm 3 both make use of LOCAL_MIN, a local minimization algorithm. In addition, if we use nonlinear programming for dimensionality reduction (5.4) in Algorithm 3, then a (local) nonlinear programming method is needed. In all cases, we use the nonlinear optimization package `OPT++` [22], into which we have incorporated all the modified Cholesky algorithms listed in [9]. We minimize the objective function $f_{A,H}(P)$ from (3.6) and use binary weights in H (i.e., $h_{ij} = 1$ if a_{ij} is specified, and otherwise $h_{ij} = 0$, where A is the partial EDM that defines the EDMCP).

We measure the quality of the solution using the maximum relative-squared-distance-error

$$\epsilon_A(P) = \max_{a_{ij} \text{ specified}} |(d_{ij} - a_{ij})/a_{ij}|, \quad D = \mathcal{K}(PP^T). \tag{6.1}$$

function P=DIM_RELAX_LOCAL_MIN(A, H, F)
  Input: an $n \times n$ partial EDM A; a weight matrix $H \in \mathbb{R}^{n\times n}$, and a pre-EDM $F \in \mathbb{R}^{n\times n}$.
  Parameters: the embedding dimension r; the maximum embedding dimension r_{max}, and the dimension-reduction step d.
  Output: If successful: a solution $P \in \mathbb{R}^{n\times r}$ to the EDMCP.
  for $r_2 = r, r+1, \ldots, r_{max}$ **do**
    $r' := r_2$;
    Compute $P :=$ CONFIG_INIT(F, r'); ▷ Alg. 1
    Stretch P, if desired; ▷ Sec. 4.3
    Apply LOCAL_MIN to minimize $f_{A,H}(P)$, starting from $P \in \mathbb{R}^{n\times r'}$;
    while a global minimizer $P \in \mathbb{R}^{n\times r'}$ of $f_{A,H}(P)$ is found **do**
      if $r' = r$ **then**
        Return P as a solution to EDMCP;
      else
        Reduce the dimensionality of P to $r' := \max(r, r' - d)$; ▷ Sec. 5.2, 5.3
        Apply LOCAL_MIN to minimize $f_{A,H}(P)$, starting from P;
      end if
    end while
  end for
end function

Algorithm 3: A local minimization algorithm using dimensionality relaxation.

We say that the EDMCP defined by the partial EDM A is *solved* if $\epsilon_A(P) < 0.01$ [36] and *finely solved* if $\epsilon_A(P) < 10^{-5}$. We measure computational cost by counting function, gradient, and Hessian evaluations.

6.1. The Moré and Wu problems [23, 24]. These artificial problems concern $m = s^3$ particles placed on a three-dimensional lattice

$$\{(i_1, i_2, i_3) : i_1, i_2, i_3 = 0, \ldots, s-1\}.$$

The atom at (i_1, i_2, i_3) is indexed uniquely by

$$i = 1 + i_1 + i_2 s + i_3 s^2.$$

There are two sets of problems. In the Type-1 problems, the distance between atoms i and j is specified in the partial EDM A if $|i-j| \le s^2$. In the Type-2 problems, all distances greater than 2 are unspecified and distances less than or equal to 2 are specified.

These problems are relatively easy and can all be solved for $s = 3, 4, 5, 6, 7$ by a minimization program with the initialization technique in Section 4.1. The costs are summarized in Tables 6.1 and 6.2. We used various minimization algorithms: BFGS and the 10 modified Newton methods from [9] (our methods are named in boldface). In all cases the problems are finely solved.

Table 6.1 shows the well-known convergence difficulties of the SE90 algorithm [27]. This issue is resolved by the SE99 algorithm [28], a revision proposed by the same authors, Schnabel and Eskow. From Table 6.2 we see that our initialization scheme in Section 4.1 is more effective for the Type-2 problems. An explanation is that for these problems, the specified distances are the shortest ones. Therefore the

TABLE 6.1
Cost to solve the Moré and Wu Type-1 problems.

LOCAL_MIN method	s	3	4	5	6	7
	# specified distances	198 (56.41%)	888 (44.05%)	2800 (35.13%)	7110 (30.62%)	15582 (26.57%)
BFGS	# f eval.	48	100	182	292	424
	# g eval.	47	99	182	292	424
GMW81, **GMW-I**, **GMW-II**, SE99, **SE-I**	# f eval.	7	7	8	9	10
MS79, CH98, **LTL^T-MS79**, **LTL^T-CH98**	# g, H eval.	7	7	8	9	10
SE90	# f eval.	12	13	64	114	201
	# g, H eval.	12	13	64	114	201

TABLE 6.2
Cost to solve the Moré and Wu Type-2 problems.

LOCAL_MIN method	s	3	4	5	6	7
	# specified distances	185 (52.71%)	564 (27.98%)	1261 (16.27%)	2372 (10.22%)	3993 (6.81%)
BFGS	# f eval.	12	14	16	21	23
	# g eval.	11	13	16	21	22
GMW81, **GMW-I**, **GMW-II**, SE99, **SE-I**	# f eval.	5	5	5	5	5
MS79, CH98, **LTL^T-MS79**, **LTL^T-CH98**	# g, H eval.	5	5	5	5	5
SE90	# f eval.	11	5	5	5	5
	# g, H eval.	11	5	5	5	5

pre-EDM from solving the all-pairs shortest path problem is a very good estimate of the EDM. Tables 6.1 and 6.2 indicate that the 9 modified-Cholesky algorithms other than SE90 are comparably efficient, taking the same numbers of function, gradient, and Hessian evaluations to solve the test problems.

Our method is significantly more efficient than the global continuation method of Moré and Wu [24] and the stochastic/perturbation global optimization algorithm of Zou, Bird, and Schnabel [36] on these test problems.

6.2. The Hendrickson problems [17]. These 12 problems were derived from a typical small protein, bovine pancreatic ribonuclease A, which consists of 124 amino acids, and after discarding end chains, 1849 atoms. The data set consists of all distances between pairs of atoms in the same amino acid, together with 1167 additional distances between hydrogen atoms that are closer than 3.5 Å. The problems were generated by taking fragments consisting of the first 20, 40, 60, 80, and 100 amino acids, as well as the complete protein with 124 amino acids. After graph reduction, there are at least 63 and at most 777 atoms in each problem. See [17] for details. The numbers of atoms and specified distances for these problems are listed in Table 6.3. Note that specifying only a small percentage distances makes the problem more difficult, so we can foresee that the Hendrickson test problems are significantly more difficult than the Moré and Wu test problems. We used these problems to test the use of Algorithm 3 in Algorithm 2.

We now specify some implementation details. We use the deterministic initialization from Section 4.1 and 100 random initializations from Section 4.2 for each setting and count how many times the solution is found. To do this, we replace the **while** loop in Algorithm 2 by a **for** loop for 100 random initializations. Also, in Algorithm 3, we continue the **for** and **while** loops even if a coarse but not fine solution is found.

The small yet still interesting problem `20.unique` provides a good test bed for exploring various parameter settings in our algorithms. We use the SE-I algorithm as

TABLE 6.3
Sizes and specified distances for the Hendrickson problems.

problem	# atoms	# specified dist.	problem	# atoms	# specified dist.
`20.unique`	63	236 (12.1%)	`20.reduced`	102	336 (6.52%)
`40.unique`	174	786 (5.22%)	`40.reduced`	236	957 (3.45%)
`60.unique`	287	1319 (3.21%)	`60.reduced`	362	1526 (2.34%)
`80.unique`	377	1719 (2.43%)	`80.reduced`	480	2006 (1.74%)
`100.unique`	472	2169 (1.95%)	`100.reduced`	599	2532 (1.41%)
`124.unique`	695	3283 (1.36%)	`124.reduced`	777	3504 (1.16%)

TABLE 6.4
Number of successful runs for `20.unique` *using 100+1 initializations.*

max. dim. r_{max}	dim. reduct. rate	dim. reduct. method	no stretching # coarse sol.	no stretching # fine sol.	stretching by 2.0 # coarse sol.	stretching by 2.0 # fine sol.
3	N/A		7	7	15	15
4	$d = 1$	linear	39	36	31	28
		nlp	54	53	51	50
5	$d = 2$	linear	43	39	35	31
		nlp	70	68	63	62
	$d = 1$	linear	46	41	37	33
		nlp	76	76	62	60

LOCAL_MIN, since it was very effective in our early tests on distance geometry problems (e.g., [8, chapter 6]). We use either the stretching factor 2.0 or no stretching. We set the maximum dimensionality relaxation $r_{max} = 3$ (i.e., no relaxation), 4, or 5. For $r_{max} = 5$, we use $d = 1$ to reduce the dimension to 4 and then 3, or $d = 2$ to reduce directly to 3-dimensions. In reducing the dimension, we use the linear projection from Section 5.2 or the nonlinear programming (nlp) method of Section 5.3.

The results on `20.unique` are summarized in Table 6.4. Stretching improves the result, and it is clear that dimensionality relaxation is a powerful technique to increase the number of successful trials. Using nonlinear programming to reduce the dimension outperforms the linear projection method, albeit at a higher computational cost.

Note that dimensionality relaxation is an expensive scheme, increasing not only the number of function, gradient, and Hessian evaluations, but also the cost of each evaluation, due to the increased dimension. In subsequent experiments, we limit the dimensionality relaxation to $r_{max} = 4$ and we use the nonlinear programming method to reduce the dimension.

For 9 of the 12 Hendrickson test problems, we used the deterministic initialization plus 100 random initializations. For the two largest problems, `124.unique` and `124.reduced`, costs are quite high. We found the solution within the first 20 random initializations, so we did not continue. We had difficulty solving the problem `80.unique`, so we used 1000 random initializations on this problem.

Tables 6.5 and 6.6 show the results of successful runs without stretching and with stretching factor 2.0, respectively.[2] We separate the numbers of evaluations into those

[2]Note that the reported function evaluations count only those used in the `OPT++` solvers [22] and do not include the 100 function evaluations for each initialization to select the pre-EDM in Algorithm 2. The cost of the eigencomputation in Algorithm 1, invoked by Algorithm 2, is also omitted in the tables. Neither of these neglected costs is significant, compared to the computational cost of the optimization algorithms.

in the three-dimensional space, those in the four-dimensional space, and those used in solving the nonlinear programming problem (5.4) that projects from one to the other. We set the maximum number of nonlinear programming iterations to 200, using up to 201 gradient/Hessian evaluations for each minimization. The nonlinear programming algorithm often goes the maximum number of iterations without terminating, but the quality of the resulting projection is usually sufficient to obtain a coarse solution. If all 100+1 runs reach 200 iterations, then the total number of gradient/Hessian evaluations is $(100+1) \times 201 = 20301$, as seen in a few problems.

From Tables 6.5 and 6.6, we see that the multistart algorithm can solve the three smallest problems `20.unique`, `20.reduced`, and `40.unique`. If we include the dimensionality relaxation scheme from Algorithm 3, we also solve the next three problems `40.reduced`, `60.unique`, and `60.reduced`. The stretching heuristic was particularly useful for the larger problems. Coupled with dimensionality relaxation, it solved all of those problems plus `100.unique`, `100.reduced`, `124.unique`, and `124.reduced`. The remaining two unsolved problems are `80.unique` and `80.reduced`; see Table 6.7.

The best configuration P obtained for `80.reduced` has $\epsilon_A(P) = 0.01211$, slightly bigger than our tolerance. To refine this solution, we can apply local minimization to minimize the square of $\epsilon_A(P)$, defined by (6.1), using the solution from Algorithm 2 as a starting guess. Note that the squared $\epsilon_A(P)$ is indeed (3.7) with weights $h_{ij} = 1/a_{ij}$. This scheme may not always succeed in obtaining significant reduction of $\epsilon_A(P)$. However, it is inexpensive and therefore we can apply various local minimizers and choose the best result. In this case, the best result is $\epsilon_A(P) = 0.00651$, obtained with the SE90 algorithm.

For `80.unique`, the best configuration obtained has $\epsilon_A(P) = 0.03187$, more than three times our tolerance. Therefore we adopted another method to refine the solution. The pre-EDM F leading to the best configuration P is potentially useful, but so far we used only the SE-I algorithm for the local minimizer with stretching factor 1.0 (i.e., no stretching) or 2.0. To improve this solution, we applied Algorithm 3 with this pre-EDM F, with stretching factors $1.0, 1.5, 2.0, 2.5, 3.0$, using the 10 modified Cholesky algorithms local minimization algorithms from [9], as well as the BFGS quasi-Newton method. The maximum dimensionality relaxation was set to $r_{max} = 4$. Two of the runs were successful. We obtained $\epsilon_A(P) = 0.00248$ with the MS79 algorithm and stretching factor 3.0, and $\epsilon_A(P) = 0.00786$ with the LTL^{T}-MS79 algorithm and stretching factor 2.5.

It is interesting to see the effectiveness of the dimensionality reduction, the stretching, and the combination of them. Recall that we used 20+1 initializations for `124.unique` and `124.reduced` each. Figure 6.1 shows the resulting $\epsilon_A(P)$ of each run. The runs without random scaling of distances are indexed by 0. It is clear that dimensionality relaxation is more effective than stretching to approach the global minimum, albeit at a higher computational cost. The combination of the two schemes further improves the result.

We also used the BFGS quasi-Newton method instead of SE-I for local minimizations. Since it is significantly more time-consuming, we tested with only the four smallest Hendrickson problems. Comparing Tables 6.5 and 6.8, we see that without stretching, this algorithm succeeds more often than SE-I. However, it never succeeded when using stretching.

In comparison, Moré and Wu [23] applied their global continuation method to the smallest problem `20.unique`. Using 100 random runs, they reached $\epsilon_A(P) = 0.02$

TABLE 6.5

Successful results with no stretching using LOCAL_MIN *(SE-I) for $r_{max} = 3$ and Algorithm 3 with SE-I for $r_{max} = 4$. In each case, 100+1 initializations were used.*

problem	max. dim. r_{max}	dim. extra	total # f eval.	total # g, H eval.	# coarse sol.	# fine sol.	best $\epsilon_A(P)$
20.unique	3	0	6225	4410	7	7	< 1e-6
	4	0	8472	6156	54	53	< 1e-6
		nlp	79995	16007			
		1	27828	18345			
20.reduced	3	0	8863	5611	1	0	0.00022
	4	0	15976	10222	6	1	< 1e-6
		nlp	93656	17672,17676			
		1	14496	9746			
40.unique	3	0	5536	3723	8	3	< 1e-6
	4	0	8507	5764	20	11	< 1e-6
		nlp	78871	16072			
		1	31352	19698			
40.reduced	3	0	11961	7273	0	0	0.01907
	4	0	16995	10576	11	3	< 1e-6
		nlp	95648	18447,18448			
		1	31928	20301			
60.unique	3	0	5827	3820	0	0	0.01047
	4	0	8711	5811	56	18	< 1e-6
		nlp	94354	17219			
		1	33607	20301			
60.reduced	3	0	16492	10123	1	0	0.00512
	4	0	29243	17996	38	2	< 1e-6
		nlp	98590	17811,17812			
		1	33378	20301			

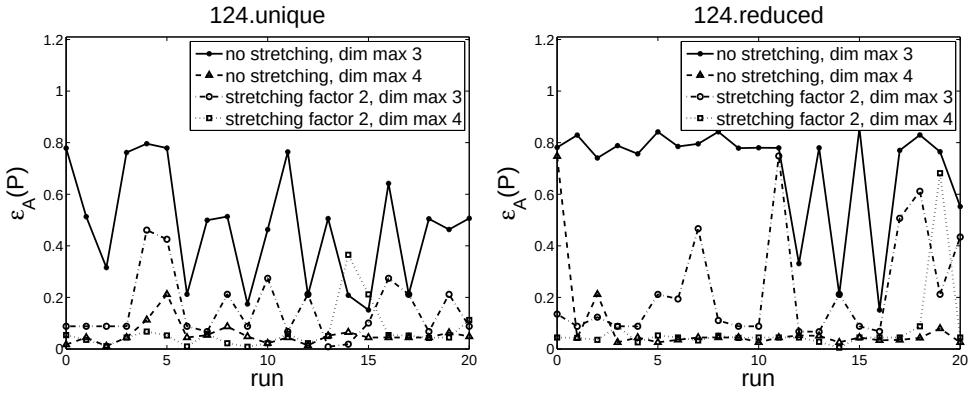

FIG. 6.1. *Probability that $s \leq x$ for two EDMCPs from Hendrickson's test set.*

but were not able to solve or to finely solve the problem. We were consistently able to finely solve this problem even without dimensional relaxation (Table 6.4). Zou, Bird, and Schnabel [36] solved the smallest 7 of the 12 problems, using their stochastic/perturbation global optimization algorithm applied to a position formulation. We solved all 12 problems, at lower cost. Zou *et al.* [36] used the BFGS quasi-Newton method as the local minimization algorithm. For the smallest problem `20.unique`, they used 692,448 function evaluations. We were able to solve this problem without

TABLE 6.6

Successful results with stretching by 2.0 *using* LOCAL_MIN *(SE-I) for* $r_{max} = 3$ *and Algorithm 3 with SE-I for* $r_{max} = 4$. *In each case, 100+1 initializations were used except for* `124.unique` *and* `124.reduced` *(20+1).*

problem	max. dim. r_{max}	dim. extra	total # f eval.	total # g, H eval.	# coarse sol.	# fine sol.	best $\epsilon_A(P)$
`20.unique`	3	0	5416	3964	15	15	0.00022
	4	0	8280	6015	51	50	< 1e-6
		nlp	61860	13185,13187			
		1	25493	16807			
`20.reduced`	3	0	8330	5292	1	0	0.00022
	4	0	15756	10114	6	2	< 1e-6
		nlp	96280	18503,18504			
		1	11893	8080			
`40.unique`	3	0	5251	3541	18	9	< 1e-6
	4	0	8712	5882	33	18	< 1e-6
		nlp	71522	15006			
		1	29600	18492			
`40.reduced`	3	0	10444	6319	0	0	0.05034
	4	0	16447	10149	15	2	< 1e-6
		nlp	94451	18453,18454			
		1	32283	20301			
`60.unique`	3	0	5318	3617	15	2	< 1e-6
	4	0	8151	5511	47	11	< 1e-6
		nlp	91169	17369			
		1	33097	19899			
`60.reduced`	3	0	13620	8544	5	1	< 1e-6
	4	0	24329	15221	29	1	< 1e-6
		nlp	95295	17342,17345			
		1	33016	20100			
`100.unique`	3	0	7260	4667	0	0	0.21854
	4	0	13240	8497	1	1	< 1e-6
		nlp	40656	7442			
		1	33778	20301			
`100.reduced`	3	0	14095	8368	0	0	0.11454
	4	0	26570	15861	1	1	3.01e-6
		nlp	32974	7727,7728			
		1	33741	20301			
`124.unique`	3	0	1175	771	1	0	0.00859
	4	0	1931	1271	3	0	0.00859
		nlp	19611	3624			
		1	7138	4174			
`124.reduced`	3	0	2511	1564	0	0	0.06708
	4	0	3774	2342	1	0	0.00508
		nlp	18731	3605			
		1	7190	4221			

dimensional relaxation using BFGS with 47,830 function evaluations, and using SE-I with 6,225 function evaluations. For the next smallest problem `20.reduced`, Zou *et al.* used 498,500 function evaluations, and we used 170,492 for BFGS and 15,976 for SE-I.

6.3. The Grooms, Lewis, and Trosset problems [14]. This synthetic problem set was created from 6 molecules from the Protein Data Bank [5] by dropping distances larger than 6 Å. For each molecule, when multiple structures are known, the first is used.

TABLE 6.7
Best results with stretching by 2.0 *using* LOCAL_MIN *(SE-I) for* $r_{max} = 3$ *and Algorithm 3 for* $r_{max} = 4$ *with SE-I for problems* `80.unique` *(1000+1 initializations) and* `80.reduced` *(100+1). Postprocessing (see text) yielded coarse solutions for both problems.*

problem	max. dim. r_{max}	dim. extra	total # f eval.	total # g, H eval.	# coarse sol.	# fine sol.	best $\epsilon_A(P)$
`80.unique`	3	0	92845	56643	0	0	0.19871
	4	0	171388	105227	0	0	0.03187
		nlp	450238	99089			
		1	336040	201182			
`80.reduced`	3	0	13923	8323	0	0	0.01211
	4	0	27463	16322	0	0	0.01211
		nlp	56479	12373			
		1	33465	20248			

TABLE 6.8
Results with no stretching using LOCAL_MIN *(BFGS) for* $r_{max} = 3$ *and Algorithm 3 with BFGS for* $r_{max} = 4$*. In each case, 100+1 initializations were used.*

problem	max. dim. r_{max}	dim. extra	total # f eval.	total # g eval.	# coarse sol.	# fine sol.	best $\epsilon_A(P)$
`20.unique`	3	0	47830	47135	18	18	< 1e-6
	4	0	97454	67746	67	65	< 1e-6
		nlp	220626	27196			
		1	95485	92492			
`20.reduced`	3	0	102350	101930	2	2	< 1e-6
	4	0	170492	156113	13	6	< 1e-6
		nlp	297566	22909			
		1	148407	148172			
`40.unique`	3	0	83352	82539	19	7	< 1e-6
	4	0	166341	131493	31	14	< 1e-6
		nlp	282704	19509			
		1	141159	141094			
`40.reduced`	3	0	144270	143226	0	0	0.03409
	4	0	246489	222376	12	5	2.10e-6
		nlp	303789	20936			
		1	152073	151601			

TABLE 6.9
Results using LOCAL_MIN *(SE-I) for* $r_{max} = 3$*.*

molecule	# atoms	# specified dist.	stretch. factor	total # f eval.	total # g, H eval.	$\epsilon_A(P)$
`2IGG`	973	31287 (6.62%)	1.0	12	11	< 1e-6
			2.0	10	10	< 1e-6
`1RML`	2064	76830 (3.61%)	1.0	23	15	< 1e-6
			2.0	9	9	< 1e-6
`1AK6`	2738	112284 (3.00%)	1.0	35	26	2.53
			2.0	18	16	< 1e-6
`1A24`	2952	106182 (2.44%)	1.0	13	13	< 1e-6
			2.0	11	11	< 1e-6
`3MSP`	3980	131438 (1.66%)	1.0	35	24	< 1e-6
			2.0	18	17	< 1e-6
`3EZA`	5147	178272 (1.35%)	1.0	26	20	< 1e-6
			2.0	14	13	< 1e-6

Our results are summarized in Table 6.9. Using the SE-I algorithm as LOCAL_MIN and the initialization technique in Section 4.1, we solved all problems except `1AK6` with no stretching (stretching factor 1.0). Using stretching factor 2.0, all 6 problems were solved. Compared with the Hendrickson problems, these problems are larger but easier.

Figure 6.2 gives the plots of the maximum error in the computed distances for each iteration. It is clear that stretching greatly speeds up the convergence. Compared to the work of Grooms, Lewis, and Trosset [14], we obtained solutions with higher accuracy in fewer iterations. Note, however, that the computation cost per iteration is different.

7. Conclusion. We surveyed three different approaches to solving the EDMCP and developed algorithms using the position formulation. We used a randomized initialization scheme and a dimensionality relaxation scheme to enhance convergence to the global optimizer. Using our algorithms, we were able to rather efficiently solve all problems in the Moré and Wu test set, the Hendrickson test set, and the Grooms, Lewis, and Trosset test set.

Acknowledgement. The Department of Computer Science and Engineering and the Minnesota Supercomputing Institute, University of Minnesota provided the computer resources for experiments. We are grateful to Yousef Saad for helpful discussions on dimensionality reduction techniques, to Che-Rung Lee for help with `OPT++`, and to David Fushman for discussions on molecular biology. This work was supported in part by NSF Grant CCF 0514213 and DOE Grant DESC0002218.

REFERENCES

[1] A. Y. Alfakih, *On the uniqueness of Euclidean distance matrix completions*, Linear Algebra Appl., 370 (2003), pp. 1–14.
[2] A. Y. Alfakih, A. Khandani, and H. Wolkowicz, *Solving Euclidean distance matrix completion problems via semidefinite programming*, Comput. Optim. Appl., 12 (1999), pp. 13–30.
[3] A. Y. Alfakih and H. Wolkowicz, *On the embeddability of weighted graphs in Euclidean spaces*, Tech. Report CORR 98-12, Computer Science Department, University of Waterloo, 1998.
[4] M. Bakonyi and C. R. Johnson, *The Euclidean distance matrix completion problem*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 646–654.
[5] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, *The protein data bank*, Nucleic Acids Res., (2000), pp. 235–242.
[6] G. M. Crippen, *Conformational analysis by energy embedding*, J. Comp. Chem., 3 (1982), pp. 471–476.
[7] ———, *Conformational analysis by scaled energy embedding*, J. Comp. Chem., 5 (1984), pp. 548–554.
[8] H.-r. Fang, *Matrix Factorization, Triadic Matrices, and Modified Cholesky Factorizations for Optimization*, PhD thesis, Univ. of Maryland, 2006.
[9] H.-r. Fang and D. P. O'Leary, *Modified Cholesky factorizations: A catalog with new approaches*, Math. Programming A, 115 (2008), pp. 319–349.
[10] R. W. Floyd, *Algorithm 97: Shortest path*, Commun. ACM, 5 (1962), p. 345.
[11] W. Glunt, T. L. Hayden, and M. Raydan, *Molecular conformation from distance matrices*, J. Comp. Chem., 14 (1993), pp. 114–120.
[12] J. C. Gower, *Some distance properties of latent root and vector methods used in multivariate analysis*, Biometrika, 53 (1966), pp. 325–338.
[13] ———, *Properties of Euclidean and non-Euclidean distance matrices*, Linear Algebra Appl., 67 (1985), pp. 81–97.
[14] I. G. Grooms, R. M. Lewis, and M. W. Trosset, *Molecular embedding via a second order dissimilarity parameterized approach*, SIAM J. Sci. Comput., 31 (2009), pp. 2733–2756.

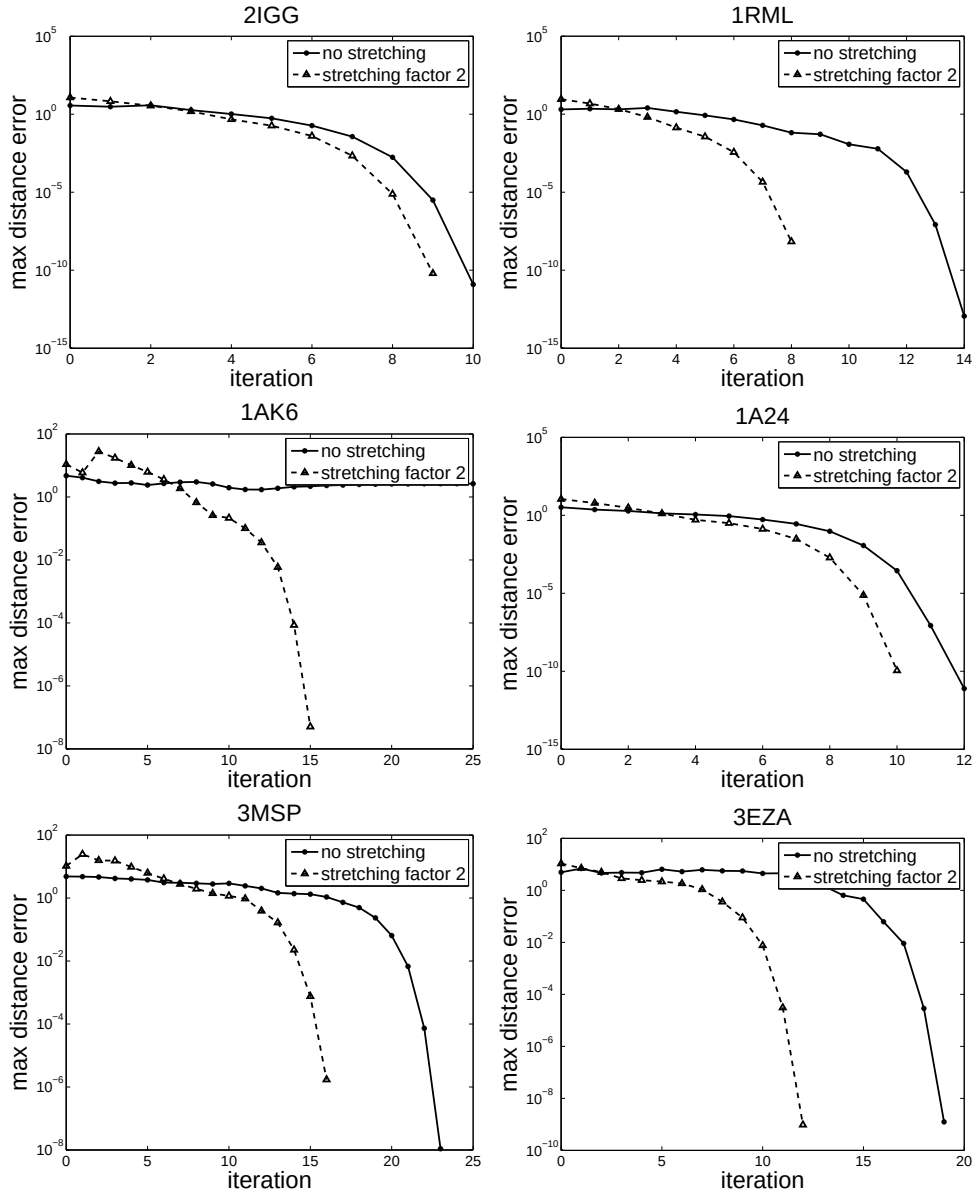

FIG. 6.2. *Maximum error (Å) vs. iteration number for the Grooms, Lewis, and Trosset problems.*

[15] T. F. HAVEL, *An evaluation of computational strategies for use in the determination of protein structure from distance geometry constraints obtained by nuclear magnetic resonance*, Prog. Biophys. Mol. Biol., 56 (1991), pp. 43–78.

[16] B. HENDRICKSON, *Conditions for unique graph realizations*, SIAM J. Sci. Comput., 21 (1992), pp. 65–84.

[17] ———, *The molecule problem: exploiting structure in global optimization*, SIAM J. Optim., 5 (1995), pp. 835–857.

[18] C. R. JOHNSON, *Connections between the real positive semidefinite and distance matrix completion problems*, Linear Algebra Appl., 223–224 (1995), pp. 375–391.

[19] E. KOKIOPOULOU AND Y. SAAD, *Orthogonal neighborhood preserving projections*, in Proc. the

5th IEEE International Conference on Data Mining, 2005, pp. 234–241.
[20] ———, *Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique*, IEEE Trans. Pattern Analysis and Machine Intelligence, 29 (2007), pp. 2143–2156.
[21] T. G. Kolda, D. P. O'Leary, and L. Nazareth, *BFGS with update skipping and varying memory*, SIAM Journal on Optimization, 8 (1998), pp. 1060–1083.
[22] J. C. Meza, R. A. Oliva, P. D. Hough, and P. J. Williams, *OPT++: An object-oriented toolkit for nonlinear optimization*, ACM Trans. Math. Softw., 33 (2007), p. 12.
[23] J. J. Moré and Z. Wu, *ϵ-optimal solutions to distance geometry problems via global continuation*, Tech. Report MCS-P520-0595, Mathematics and Computer Science Division, Argonne National Laboratory, 1995.
[24] ———, *Global continuation for distance geometry problems*, SIAM J. Optim., 7 (1997), pp. 814–836.
[25] ———, *Distance geometry optimization for protein structures*, J. Global Optim., 15 (1999), pp. 219–234.
[26] E. O. Purisima and H. A. Scheraga, *An approach to the multiple-minima problems by relaxing dimensionality*, Proc. Natn. Acad. Sci. USA, 83 (1986), pp. 2782–2786.
[27] R. B. Schnabel and E. Eskow, *A new modified Cholesky factorization*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 1136–1158.
[28] ———, *A revised modified Cholesky factorization algorithm*, SIAM J. Optim., 9 (1999), pp. 1135–1148.
[29] I. J. Schoenberg, *Remarks to Maurice Frechet's article: Sur la definition axiomatique d'une classe d'espaces vectoriels distancies applicables vectoriellement sur l'espace de Hilbert*, Ann. Math., 36 (1935), pp. 724–732.
[30] J. B. Tenenbaum, V. de Silva, and J. C. Langford, *A global geometric framework for nonlinear dimensionality reduction*, Science, 290 (2000).
[31] W. S. Torgerson, *Multidimensional scaling: I. theory and method*, Psychometrika, 30 (1952), pp. 333–367.
[32] M. W. Trosset, *Applications of multidimensional scaling to molecular conformation*, Computing Science and Statistics, 29 (1998), pp. 148–152.
[33] ———, *Distance matrix completion by numerical optimization*, Comput. Optim. Appl., 17 (2000), pp. 11–22.
[34] S. Warshall, *A theorem on boolean matrices*, J. ACM, 9 (1962), pp. 11–12.
[35] G. Young and A. S. Householder, *Discussion of a set of points in terms of their mutual distances*, Psychometrika, 3 (1938), pp. 19–22.
[36] Z. Zou, R. H. Bird, and R. B. Schnabel, *A stochastic/perturbation global optimization algorithm for distance geometry problems*, J. Global Optim., 11 (1997), pp. 91–105.

Appendix: Derivatives for the Position Formulation. We note that the gradient and the Hessian matrix of $f_{A,H}(P)$ in (3.6) are easily computed. By (2.2) and the fact that $B = PP^T$,

$$d_{ij} = b_{ii} + b_{jj} - 2b_{ij} = \sum_{k=1}^{r} p_{ik}^2 + \sum_{k=1}^{r} p_{jk}^2 - 2\sum_{k=1}^{r} p_{ik}p_{jk}$$

for $i, j = 1, 2, \ldots, n$. Therefore,

$$\frac{\partial d_{ij}}{\partial p_{st}} = \begin{cases} 2(p_{st} - p_{jt}) & \text{if } s = i \neq j, \\ 2(p_{st} - p_{it}) & \text{if } s = j \neq i, \\ 0 & \text{otherwise.} \end{cases}$$

and

$$\frac{\partial^2 d_{ij}}{\partial p_{st}\partial p_{uv}} = \begin{cases} 2 & \text{if } ((s = u = i \neq j) \vee (s = u = j \neq i)) \wedge (t = v), \\ -2 & \text{if } ((s = i \neq u = j) \vee (s = j \neq u = i)) \wedge (t = v), \\ 0 & \text{otherwise.} \end{cases}$$

By (3.6),

$$\begin{aligned} \frac{\partial f}{\partial p_{st}} &= 2\sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}^2(d_{ij} - a_{ij})\frac{\partial d_{ij}}{\partial p_{st}} \\ &= 4\sum_{j=1}^{n} h_{sj}^2(d_{sj} - a_{sj})(p_{st} - p_{jt}) + 4\sum_{i=1}^{n} h_{is}^2(d_{is} - a_{is})(p_{st} - p_{it}) \\ &= 8\sum_{i=1}^{n} h_{is}^2(d_{is} - a_{is})(p_{st} - p_{it}). \end{aligned}$$

If $u = s$,

$$\begin{aligned} \frac{\partial^2 f}{\partial p_{st}\partial p_{uv}} &= 8\sum_{i=1}^{n} h_{is}^2(2(p_{sv} - p_{iv})(p_{st} - p_{it}) + (d_{is} - a_{is})\frac{\partial}{\partial p_{uv}}(p_{st} - p_{it})) \\ &= \begin{cases} 16\sum_{i=1}^{n} h_{is}^2(p_{sv} - p_{iv})(p_{st} - p_{it}) & \text{if } v \neq t, \\ 8\sum_{i=1}^{n} h_{is}^2(2(p_{st} - p_{it})^2 + (d_{is} - a_{is})) & \text{if } v = t. \end{cases} \end{aligned}$$

If $u \neq s$,

$$\begin{aligned} \frac{\partial^2 f}{\partial p_{st}\partial p_{uv}} &= 8h_{us}^2(2(p_{uv} - p_{sv})(p_{st} - p_{ut}) + (d_{us} - a_{us})\frac{\partial}{\partial p_{uv}}(p_{st} - p_{ut})) \\ &= \begin{cases} 16h_{us}^2(p_{uv} - p_{sv})(p_{st} - p_{ut}) & \text{if } v \neq t, \\ -8h_{us}^2(2(p_{st} - p_{ut})^2 + (d_{us} - a_{us})) & \text{if } v = t. \end{cases} \end{aligned}$$

Using these formulas, problem (3.5) can be solved using variants of Newton's method for minimization.