Journal of Global Optimization **15:** 219–234, 1999.

Distance Geometry Optimization for Protein Structures

JORGE J. MORÉ and ZHIJUN WU
Rice University, Houston, Texas, USA

(Accepted in original form 19 August 1999)

Abstract. We study the performance of the **dgsol** code for the solution of distance geometry problems with lower and upper bounds on distance constraints. The **dgsol** code uses only a sparse set of distance constraints, while other algorithms tend to work with a dense set of constraints either by imposing additional bounds or by deducing bounds from the given bounds. Our computational results show that protein structures can be determined by solving a distance geometry problem with **dgsol** and that the approach based on **dgsol** is significantly more reliable and efficient than multi-starts with an optimization code.

Key words: Distance geometry, Distance constraints, Protein structures

1. Introduction

Distance geometry problems for the determination of protein structures are specified by a subset $\mathcal{S}$ of all atom pairs and by the distances between atoms i and j for $(i, j) \in \mathcal{S}$. In practice, lower and upper bounds on the distances are given instead of precise values. The distance geometry problem with lower and upper bounds is to find a set of positions $x_1, \ldots, x_m$ in $\mathbb{R}^3$ such that

$$l_{i,j} \leqslant \| x_i - x_j \| \leqslant u_{i,j}, \qquad (i, j) \in \mathcal{S}, \tag{1.1}$$

where $l_{i,j}$ and $u_{i,j}$ are lower and upper bounds on the distances, respectively. Reviews and background on the application of distance geometry problems to protein structure determination can be found in Crippen and Havel [4], Havel [11,12], Torda and Van Gunsteren [30], Kuntz, Thomason and Oshiro [19], Brünger and Nilges [3], and Blaney and Dixon [2].

The distance geometry problem (1.1) can be formulated as a global optimization problem. The standard formulation, suggested by Crippen and Havel [4], is in terms of finding the global minimum of the function

$$f(x) = \sum_{i,j \in \mathcal{S}} p_{i,j}(x_i - x_j), \tag{1.2}$$

where the pairwise function $p_{i,j} : \mathbb{R}^n \mapsto \mathbb{R}$ is defined by

$$p_{i,j}(x) = \min^2 \left\{ \frac{\| x \|^2 - l_{i,j}^2}{l_{i,j}^2},\ 0 \right\} + \max^2 \left\{ \frac{\| x \|^2 - u_{i,j}^2}{u_{i,j}^2} \right\}. \tag{1.3}$$

Clearly, $x = \{x_1, \ldots, x_m\}$ solves the distance geometry problem if and only if x is a global minimizer of f and $f(x) = 0$.

In practice, distance geometry problems also impose chirality constraints on some of the atoms. These constraints can be handled by adding a term to the potential function (1.2) whenever chirality constraints are imposed on the atoms x_i, x_j, x_k, x_l. The chirality constraint function (for example, Havell [11,12]) is usually of the form

$$c_{i,j,k,l}(x) = (\mathrm{vol}(x_i, x_j, x_k, x_l) - v_{i,j,k,l})^2,$$

where vol is the oriented volume of the four atoms and $v_{i,j,k,l}$ is a target value. The approach in this paper can be extended to these chirality constraints, but since our aim is to develop reliable algorithms for the distance geometry problem (1.1), we consider only the potential function (1.2).

The **embed** algorithm [4,11,12] and the alternating projection algorithm [7,8] are the most promising techniques for the solution of the distance geometry problem (1.2). For related work, see [1–3,19]. General global optimization techniques (multi-starts with a local optimization algorithm, simulated annealing, genetic algorithms) and molecular dynamics algorithms could also be used, but they have not been shown to be suitable for distance geometry problems.

The algorithm that we propose in this paper works with the sparse set of distance constraints $\mathcal{S}$. In contrast, other algorithms for distance geometry tend to work with a dense set of constraints by either imposing additional bounds or by deducing bounds from the given bounds. For example, the first phase of the **embed** algorithm determines $l_{i,j}$ and $u_{i,j}$ by using the relationships

$$u_{i,j} = \min(u_{i,j}, u_{i,k} + u_{k,j}), \qquad l_{i,j} = \max(l_{i,j}, l_{i,k} - u_{k,j}, l_{j,k} - u_{k,i}),$$

which can be deduced from the triangle inequality. Given a full set of bounds, distances $\delta_{i,j} \in [l_{i,j}, u_{i,j}]$ are chosen, and an attempt is made to compute coordinates $x_1, \ldots, x_m$ by solving the special distance geometry problem

$$\| x_i - x_j \| = \delta_{i,j}, \quad (i, j) \in \mathcal{S}. \tag{1.4}$$

This attempt usually fails because the bounds $\delta_{i,j}$ tend to be inconsistent, but it can be used to generate an approximate solution. As a result, the **embed** algorithm may require many trial choices of $\delta_{i,j}$ in $[l_{i,j}, u_{i,j}]$ before a solution to problem (1.4) is found.

Other algorithms that work with a sparse set of distance constraints do not aim to solve the distance geometry problem (1.1), but to minimize a potential energy

function that incorporates distance constraints and other information to determine the protein structure. For work in this direction, see [15,28].

In our approach, we use Gaussian smoothing to transform f into a smoother function with fewer minimizers. An optimization algorithm (the limited-memory variable-metric code **vmlm** is then applied to the transformed function, and continuation techniques are used to trace the minimizers of the smooth function back to the original functon. An immediate advantage of our approach is that the work per iteration is proportional to $\mathcal{S}$, which for sparse distance data should be proportional to the number of atoms m.

Gaussian smoothing was first used, by Scheraga and coworkers [16–18,25,26], in the diffusion equation method for protein conformation. In that application the Gaussian transform is usually evaluated by approximating the function and then transforming the approximation. On the other hand, Moré and Wu [22] showed that for distance geometry applications we can evaluate the Gaussian transform of (1.2) directly if the potential $p_{i,j}$ is a radial function, that is, a function of the form $p_{i,j}(x) = h_{i,j}(\| x \|)$.

The aim of this paper is to show that continuation algorithms, based on Gaussian smoothing, can be used to develop an efficient and reliable code for the solution of the distance geometry problem (1.1). The background needed to understand our code, **dgsol**, is presented in Sections 2 and 3. Section 2 outlines the smoothing properties of the Gaussian transform, while Section 3 presents our proposal to determine the Gaussian transform by using a discrete Gauss-Hermite transform.

We present an outline of **dgsol** in Section 4. Numerical results appear in Section 5. We pay special attention to the choice of continuation parameters because this is an important and unresolved issue in the use of Gaussian smoothing. Our numerical results, based on data drawn from the PDB data bank, show that **dgsol** can be used to determine the structure of protein fragments with up to 200 atoms.

We emphasize that the determination of protein structures from distance data requires appropriate data and an algorithm to determine solutions to (1.1). The issue of what distance data is needed has been addressed in several recent papers [1,15,20,28]. In this paper we do not address this issue, but concentrate on showing that **dgsol** can be used to obtain solutions to the distance geometry problem (1.1) for a wide range of distance data. To our knowledge, no other algorithm can make this claim. We plan to conduct additional testing with larger protein fragments and more realistic distance constraints.

2. Global smoothing

An appealing idea for finding the global minimizer of a function is to transform the function into a smoother function with fewer local minimizers, apply an optimization algorithm to the transformed function, and trace the minimizers back to the original function. A transformed function is a coarse approximation to the original function, with small and narrow minimizers being removed, while the overall struc-

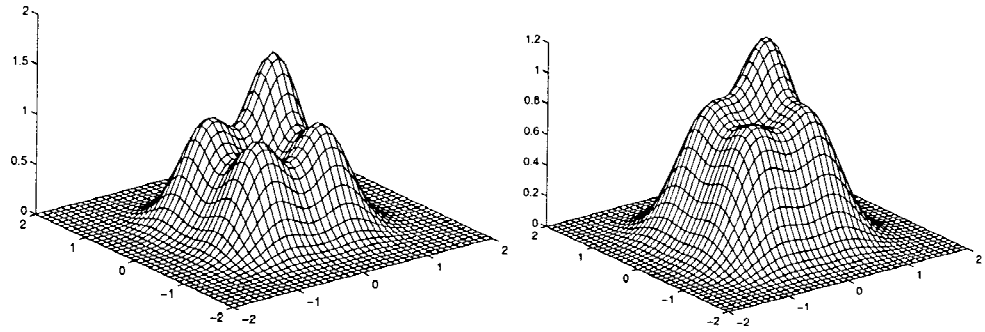

Figure 2.1. The Gaussian transform of a function. The original function ($\lambda = 0$) is on the left, while $\lambda = 0.3$ is on the right.

ture of the function is maintained. This property allows the optimization algorithm to skip less interesting local minimizers and to concentrate on regions with average low-function values where a global minimizer is most likely to be located.

The smoothing transform, called the Gaussian transform, depends on a parameter λ that controls the degree of smoothing. The original function is obtained if $\lambda = 0$, while smoother functions are obtained as λ increases. The Gaussian transform $\langle f \rangle_\lambda$ of a function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is

$$\langle f \rangle_\lambda(x) = \frac{1}{\pi^{n/2}\lambda^n} \int_{\mathbb{R}^n} f(y) \exp\left(-\frac{\| y - x \|^2}{\lambda^2}\right) \mathrm{d}y. \tag{2.1}$$

The value $\langle f \rangle_\lambda(x)$ is an average of f in a neighbourhood of x, with the relative size of this neighborhood controlled by the parameter λ. The size of the neighbourhood decreases as λ decreases, so that when $\lambda = 0$, the neighborhood is the center x. The Gaussian transform $\langle f \rangle_\lambda$ can also be viewed as the convolution of f with the Gaussian density function.

The Gaussian transform is a linear, isotone (order-preserving) operator that reduces the high-frequency components of f. Moreover, the Gaussian transform commutes with differentiation so that the Gaussian transform of the gradient (Hessian) is the gradient (Hessian) of the Gaussian transform. These properties of the Gaussian transform are not usually shared by other approaches to smoothing. For additional discussion of these properties, see Wu [31] and Moré and Wu [23,24].

We illustrate the transformation process in Figure 2.1 with a function that is the sum of four Gaussians. The original function ($\lambda = 0$) is on the left while $\lambda = 0.3$ is on the right. Note that the original function has four maximizers but that two of these maximizers have disappeared at $\lambda = 0.3$, and another minimizer is likely to disappear if λ is increased further. Figure 2.1 shows that the original function is gradually transformed into a smoother function with fewer local maximizers and that the smoothing increases as λ increases.

3. Computing the Gaussian transform of distance geometry functions

Computing the Gaussian transform requires the evaluation of n-dimensional integrals, but for many functions that arise in practice, it is possible to compute the Gaussian transform explicitly in terms of one-dimensional transforms. In particular, we now show that we can compute the Gaussian transform for distance geometry functions of the form (1.2) if the potential $p_{i,j}$ is a radial function, that is, a function of the form $p_{i,j}(x) = h_{i,j}(\| x \|)$.

Moré and Wu [22] showed that the Gaussian transform for the distance geometry function (1.2) can be expressed in the form

$$\langle f \rangle_\lambda(x) = \sum_{i,j \in \mathcal{S}} \frac{1}{\sqrt{2\pi} r_{i,j}} \int_{-\infty}^{+\infty} (r_{i,j} + \lambda s) h_{i,j}(r_{i,j} + \lambda s) \exp\left(-\frac{1}{2} s^2\right) \mathrm{d}s \tag{3.1}$$

where $r_{i,j} = \| x_i - x_j \|$. This expression is valid for all pairwise potentials of the form $p_{i,j}(x) = h_{i,j}(\| x \|)$. We are interested in the case where the pairwise potential $p_{i,j}$ is given by (1.2), so that the function $r \mapsto h_{i,j}(r)$ is defined by

$$h_{i,j}(r) = \overset{2}{\min}\left\{\frac{r^2 - l_{i,j}^2}{l_{i,j}^2}, 0\right\} + \overset{2}{\max}\left\{\frac{r^2 - u_{i,j}^2}{u_{i,j}^2}\right\}. \tag{3.2}$$

The one-dimensional integrals that appear in (3.1) can be evaluated explicitly in special cases. In particular, when $l_{i,j} = u_{i,j}$ for all $(i, j) \in \mathcal{S}$, the Gaussian transform can be expressed [23] in the form

$$\langle f \rangle_\lambda(x) = \sum_{i,j \in \mathcal{S}} \left[(\| x_i - x_j \|^2 - \delta_{i,j}^2)^2 + 10\lambda^2 \| x_i - x_j \|^2\right] + \gamma,$$

where γ is a constant that depends on λ.

An interesting property of the Gaussian transform is that the Gaussian transform (3.1) is infinitely differentiable whenever $\lambda > 0$. On the other hand, the original potential (1.2), with the pairwise potential $p_{i,j}$ defined by (1.3), is only piecewise twice differentiable. Moré and Wu [23,24] provide additional information on the properties of the Gaussian transform.

The one-dimensional integrals that appears in the Gaussian transform (3.1) can be approximated by Gaussian quadratures. If we use a Gaussian quadrature with q nodes, we obtain the Gauss-Hermite approximation

$$\langle f \rangle_{\lambda,q}(x) = \sum_{i,j \in \mathcal{S}} \frac{1}{r_{i,j}} \sum_{k=1}^{q} w_k (r_{i,j} + \lambda s_k) h_{i,j}(r_{i,j} + \lambda s_k), \tag{3.3}$$

where w_k and s_k are standard weights and nodes for the Gaussian quadrature for integrals of the form

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} g(s) \exp\left(-\frac{1}{2} s^2\right) \mathrm{d}s.$$

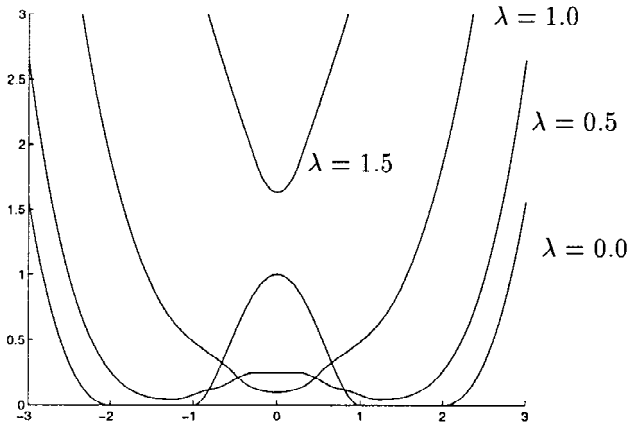

Figure 3.1. The function $[h]_{\lambda,q}$ for $\lambda = 0, 0.5, 1.0, 1.5$ and $q = 10$.

The weights and nodes can be found in the tables of Stroud and Secrest [29] or computed with the gauss subroutine in ORTHOPOL [6].

All of our numerical results are based on the Gauss–Hermite transform (3.3). We can gain insight into this transformation by noting that

$$\langle f \rangle_{\lambda,q}(x) = \sum_{i,j \in \mathcal{S}} [h_{i,j}]_{\lambda,q}(r_{i,j}),$$

where $[h]_{\lambda,q}$ is a function of the distance r defined by

$$[h]_{\lambda,q}(r) = \frac{1}{r} \sum_{k=1}^{q} w_k (r + \lambda s_k) h(r + \lambda s_k).$$

The function $[h]_{\lambda,q}$ agrees with the piecewise twice-differentiable function

$$h(r) = \min^2 \left\{ \frac{r^2 - l^2}{l^2}, 0 \right\} + \max^2 \left\{ \frac{r^2 - u^2}{u^2}, 0 \right\} \tag{3.4}$$

for $\lambda = 0$, but as λ increases, we obtain a smoother version of the function. This can be seen clearly in Figure 3.1, where we have plotted $[h]_{\lambda,q}$ for $\lambda = k/2$ with $0 \leqslant k \leqslant 3$, $q = 10$.

Figure 3.1 suggests that $[h]_{\lambda,q}$ is convex for $\lambda \geqslant \lambda_c$ for some $\lambda_c > 0$. This is not true in general, but holds when h is defined by (3.4). The value of λ_c depends on the bounds l and u, but we do not fully understand this relationship. For $l = u$, it is easy to show that $\lambda_c = u/\sqrt{5}$. Plots of $[h]_{\lambda,q}$ for $l < u$ suggest that $\lambda_c \leqslant \sqrt{2}u$, and therefore

$$\lambda_c \in \left[\frac{u}{\sqrt{5}}, \sqrt{2}u \right]$$

This is consistent with Figure 3.1, where $l = 1$, $u = 2$, and $[h]_{\lambda,q}$ is convex for $\lambda = 1.5$.

The value of λ_c is important because if $[h_{i,j}]_{\lambda,q}$ is monotone and convex for $r > 0$ and $(i, j) \in \mathcal{S}$, then the Gauss–Hermite transform (3.3) is convex. This is usually undesirable; a preferable strategy is to choose λ so that only some of the functions $[h_{i,j}]_{\lambda,q}$ are convex. We will return to this point when we discuss numerical results.

4. Optimization algorithms

The algorithm that we use to solve the distance geometry problem (1.1) searches for a global minimizer of the function defined by (1.2) and (1.3) with a continuation algorithm based on the Gauss-Hermite transform $\langle f \rangle_{\lambda,q}$. Given a sequence of smoothing parameters

$$\lambda_0 > \lambda_1 > \cdots > \lambda_p = 0,$$

the continuation algorithm uses a local minimization algorithm to determine a minimizer x_{k+1} of $\langle f \rangle_{\lambda_k,q}$. The local minimization algorithm uses the previous minimizer x_k as the starting point for the search. In this manner a sequence of minimizers $x_1, \ldots, x_{p+1}$ is generated, with x_{p+1} a minimizer of f and the candidate for the global minimizer. Algorithm **dgsol** specifies the continuation algorithm.

Algorithm dgsol
Choose a random vector $x_0 \in \mathbb{R}^{m\times 3}$.
for $k = 0, 1, \ldots, p$
Determine $x_{k+1} =$ **locmin**$(\langle f \rangle_{\lambda_k,q}, x_k)$.
end do

In our notation, **locmin** $(\langle f \rangle_{\lambda_k,q}, x_k)$ is the minimizer generated by a local minimization algorithm with the starting point x_k. The local minimization algorithm has to be chosen with some care because $\langle f \rangle_{\lambda,q}$ is not twice continuously differentiable. The Hessian matrix is discontinuous at points where the argument of $h_{i,j}$ coincides with either $l_{i,j}$ or $u_{i,j}$. We cannot expect to avoid these discontinuities, in particular, if $l_{i,j}$ and $u_{i,j}$ are close.

For **locmin** we used a limited-memory variable metric algorithm of the form

$$x_{k+1} = x_k - \alpha_k H_k \nabla f(x_k),$$

where $\alpha_k > 0$ is the search parameter, and the approximation H_k to the inverse Hessian matrix is stored in a compact representation that requires the storage of only $2n_v$ vectors, where n_v is chosen by the user. The compact representation of H_k permits the efficient computation of $H_k \nabla f(x_k)$ in $(8n_v + 1)n$ flops, where $n = 3m$ is the number of variables; all other operations in an iteration of the algorithm require $11n$ flops.

We used the variable-metric limited-memory code **vmlm** in MINPACK-2. For additional information on this code, see

http://www.mcs.anl.gov/home/more/minpack2

The performance of the **vmlm** code depends on the amount of memory specified by n_v and on the tolerances τ_r and τ_a. We used $n_v = 10$. The tolerances τ_r and τ_a specify the accuracy of the minimizer; **vmlm** terminates with an iterate x if the code decides that either the relative convergence test

$$|f(x) - f(x^*)| \leqslant \tau_r |f(x^*)|$$

or the absolute convergence test

$$\max\{|f(x)|,\ |f(x^*)|\} \leqslant \tau_a$$

is satisfied for some minimizer x^* of f. In our numerical results we used $\tau_r = \tau_a = 10^{-8}$, which are not considered stringent values.

The random vector $x_0 \in \mathbb{R}^{m\times 3}$ used for algorithm **dgsol** depends on the distance data. In particular, we chose the coordinates $x_i \in \mathbb{R}^3$ of the starting point so that $\| x_i - x_j \| = \delta_{i,j}$ for some (i, j) in $\mathcal{S}$. Algorithm **struct** specifies the starting point

Algorithm struct
Set $\mathcal{L} = \{1, \ldots, m\}$.
do until $\mathcal{L}$ is empty
Choose $i \in \mathcal{L}$.
Set $\mathcal{M}_i = \{j : (i, j) \in \mathcal{S},\ j \in \mathcal{L}\}$.
For each $j \in \mathcal{M}_i$, generate $x_j \in \mathbb{R}^3$ such that $\| x_i - x_j \| = \delta_{i,j}$.
Remove i from $\mathcal{L}$.
end do

The starting point generated by this algorithm satisfies at least $m - 1$ distance constraints, where m is the number of atoms. Thus, the starting point is a solution to the distance geometry problem if $\mathcal{S}$ contains less than m constraints.

We also experimented with starting points that were chosen randomly, but since our results were not strongly dependent on the method used to generate the starting points, we present results for only the method specified above.

5. Computational experiments

In our computational experiments we studied the distance geometry problem (1.1) with the pairwise potential $p_{i,j}$ defined by (1.3). We used the **dgsol** algorithm as outlined in Section 4 and the Gauss–Hermite transform (3.3) with $q = 10$ nodes in the Gaussian quadrature.

We tested **dgsol** on data derived from protein fragments of a DNA-binding protein [10,27] available (ID code 1GPV) in the PDB data bank. We considered

protein fragments with 100 and 200 atoms. For each fragment, we generated a set of distances $\{\delta_{i,j}\}$ by using all distances between the atoms in the same residue as well as those in the neighbouring residues. Formally, if $\mathbb{R}_k$ is the k-th residue, then

$$\mathcal{S} = \{(i, j) : \ x_i \in \mathbb{R}_k, \ x_j \in (\mathbb{R}_k \cup \mathbb{R}_{k+1})\} \tag{5.1}$$

specifies the set of distances. This is not the only way to generate the sparse set $\mathcal{S}$. For example, Le Grand, Elofsson, and Eisenberg [20] generate $\mathcal{S}$ by setting

$$\mathcal{S} = \left\{(i, j) : \ \| x_i - x_j \| \leqslant c\right\} \tag{5.2}$$

for some cutoff $c < 0$.

The main aim of the computational experiments is to show that the **dgsol** code, which is based on Gaussian smoothing, provides a reliable and efficient approach to the solution of the distance geometry problem (1.1). In our computational results, a set of coordinates $x \in \mathbb{R}^{m\times 3}$ solves the distance geometry problem (1.1) if

$$(1 - \tau_d)l_{i,j} \leqslant \| x_i - x_j \| \leqslant u_{i,j}(1 + \tau_d), \quad (i, j) \in \mathcal{S}, \tag{5.3}$$

for some tolerance τ_d. We used $\tau_d = 10^{-2}$ since this tolerance reflects the accuracy available for bond lengths [5].

A secondary aim of the computational experiments is to study the dependence of the solution structures on variations on the bounds $l_{i,j}$ and $u_{i,j}$ by setting

$$l_{i,j} = (1 - \varepsilon)\delta_{i,j}, \qquad u_{i,j} = (1 + \varepsilon)\delta_{i,j}, \tag{5.4}$$

for some $\varepsilon \in (0, 1)$. With this formulation, we are able to study the behavior of the structures as ε varies over $(0, 1)$. We varied ε over [0.04, 0.16] since this translates into a 4–16% deviation from the expected value for the bond length. These variations seem to be typical [5].

In many of our numerical results we examine the performance of **dgsol** as ε and λ vary. In **dgsol** we use uniformly spaced smoothing parameters

$$\lambda_k = \lambda_0 \left(1 - \frac{k}{p}\right), \qquad 0 \leqslant k \leqslant p.$$

The number p of continuation steps was set to

$$p = \lceil 20\lambda_0 \rceil.$$

This choice implies that the separation $\lambda_{k+1} - \lambda_k$ between consecutive smoothing parameters is about 0.05.

The choice of λ_0 is important. If we start with λ_0 large, then all the information in the function is destroyed, and it is difficult to trace multiple paths. If we choose λ_0 small then $\langle f \rangle_{\lambda_0,q}$ will have many minimizers. Choosing λ_0 so that $\langle f \rangle_{\lambda_0,q}$ has a few minimizers allows us to trace multiple paths, and thus increases the chances of determining a global minimizer.

A reasonable λ_0 is obtained if half of the $[h_{i,j}]_{\lambda_0,q}$ are not convex. This provides an automatic choice for λ_0 that is not large and that works well. We can determine λ_0 by recalling that (see Section 3) for each function $h_{i,j}$ there is a $\lambda_{i,j}$ such that $[h_{i,j}]_{\lambda,q}$ is convex for $\lambda \geqslant \lambda_{i,j}$. We use

$$\lambda_{i,j} = \left(\frac{1}{\sqrt{5}}\, \rho_{i,j} + \sqrt{2}(1 - \rho_{i,j}) \right) u_{i,j}, \lambda_{i,j} = \frac{l_{i,j} - 1}{u_{i,j} - 1},$$

which specifies that $\lambda_{i,j}$ is a convex combination of $1/\sqrt{5}$ and $\sqrt{2}$. If $l_{i,j} = u_{i,j}$ then $\lambda_{i,j} = 1/\sqrt{5}$, which we know guarantees convexity of $[h_{i,j}]_{\lambda,q}$. This observation is important because our data $l_{i,j} \approx u_{i,j}$. We have verified, by plots of $[h_{i,j}]_{\lambda,q}$ similar to those in Figure 3.1, that for this choice of $\lambda_{i,j}$, the function $[h_{i,j}]_{\lambda,q}$ is convex for $\lambda \geqslant \lambda_{i,j}$. It would be interesting to obtain a formal proof of this result. λ_n future implementation, we will also use $\lambda_{i,j} = \frac{l_{i,j}}{u_{i,j}}$ to avoid dividing a zero in case $u_{i,j} = 1$.

Given $\lambda_{i,j}$ as defined above, we now choose λ_0 as the median of all the $\lambda_{i,j}$. With this choice, half of the pairwise functions $[h_{i,j}]_{\lambda_0,q}$ should not be convex. Hence, the initial function $\langle f \rangle_{\lambda_0,q}$ is smooth but not necessarily convex.

5.1. EXPERIMENT 1

In our first computational experiment we compare **dgsol** with **vmlm** from a set of 100 random starting points generated by algorithm **struct** of Section 4. We did this comparison because multi-starts with a local optimization code is a standard approach to solving global optimization problems. Comparisons with simulated annealing and genetic algorithms would also be of interest but are unlikely to perform better than multi-starts unless they also rely on optimization software to produce accurate structures.

We conducted two tests with $\varepsilon = 0.04$, one with **vmlm** and the other with **dgsol**. We compare the quality of the solutions obtained by **vmlm** and **dgsol** by computing the potential function (1.2) at the final iterate of the algorithm. These function values are then sorted and plotted in Figure 5.1.

An immediate observation that can be made from Figure 5.1 is that the potential function (1.2) has at least 100 distinct minimizers. We justify this observation by noting that all the minimizers obtained by the **vmlm** algorithm have distinct function value. This observation is of interest because it is usually difficult to find the global minimizer when the optimization problem has many minimizers.

The results in Figure 5.1 show that the **vmlm** algorithm fails to find the global minimizer in all cases. This is perhaps not surprising because the **vmlm** code is a local minimization algorithm. Nevertheless, we expected to find the global solution in at least a few cases. However, the results in Figure 5.1 show that **vmlm** is able to find only local minimizers with relatively high function values; in all cases the potential function value is at least 0.5.

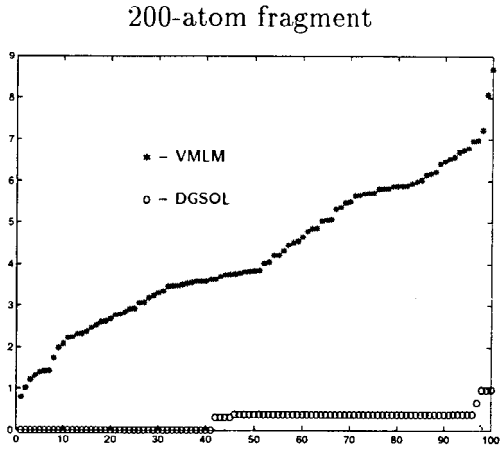

Figure 5.1. Potential function values for multi-start **vmlm** and **dgsol** for $\varepsilon = 0.04$.

The results in Figure 5.1 also show that the smoothing approach of **dgsol** works quite well for this problem and is able to find the global solution in 41 cases. Also note that in all cases **dgsol** finds a global minimizer or a local minimizer with low function value.

5.2. EXPERIMENT 2

In our second experiment we compare the performance of the multi-start **vmlm** with **dgsol** for problems with $\varepsilon > 0$ and for both the 100-atom and 200-atom fragments. In each case we used the 100 random starting points generated by algorithm **struct** of Section 4 and counted the number of (global) solutions found by each algorithm. Recall that for these results we count a set of coordinates $x \in \mathbb{R}^{m\times 3}$ as a solution to the distance geometry problem (1.1) if (5.3) is satisfied with $\tau_d = 10^{-2}$. Results for this experiment appear in Table 5.1.

Table 5.1. Distance geometry solutions obtained by **vmlm** and **dgsol**

100-atom fragment			200-atom fragment		
ε	**vmlm**	**dgsol**	ε	**vmlm**	**dgsol**
0.04	1	80	0.04	0	41
0.08	1	74	0.08	0	66
0.12	8	100	0.12	2	97
0.16	47	100	0.16	9	100

The results in Table 5.1 show that **dgsol** is significantly more reliable than the multi-started **vmlm** for both the 100-atom fragment and the 200-atom fragment. For both algorithms the reliability increases with ε. This result is to be expected because as ε increases, the measure of the solution set also increases. In other words, if $x \in \mathbb{R}^n$ satisfies (1.1) for $l_{i,j}$ and $u_{i,j}$ specified by (5.4), then x also satisfies (1.1) for all larger ε.

Note that the reliability of both algorithms decreases as we go from the 100-atom fragment to the 200-atom fragment. This result is to be expected because the number of minimizers of the distance geometry problem also increases as the number of atoms increases.

We emphasize that we have been using **dgsol** with 100 starting points to test the reliability of **dgsol**. In practice we can expect to find a global minimizer after at most *six* starting points. This rule of thumb is justified by the results in Table 5.1, which show that in all cases we have 40% reliability, and thus a standard calculation shows that after six trials we have a 95% chance of finding a global minimum.

5.3. EXPERIMENT 3

In general, the distance geometry problem (1.1) can have many solutions, so there is no reason to expect that the structures generated by **dgsol** will agree with the structure that was used to generate the data. In this experiment we study the relationship between the structures obtained for various ε and the original data.

We compare structures by measuring the deviation between the coordinates and the distances for the generated structure and the original structure. A standard measure for comparing structures is the coordinate RMSD (root-mean-square-deviation)

$$E_C = \min \left\{ \left(\frac{1}{m} \sum_{i=1}^{m} \| y_i - Q x_i \|^2 \right)^{1/2} : Q \in \mathbb{R}^{3\times 3}, \text{ orthogonal} \right\}, \quad (5.5)$$

where m is the number of atoms in the structure. Optimal superposition by translation is assured if the structures $\{x_i\}$ and $\{y_i\}$ are translated so their center of gravity is at the origin. In Table 5.2 we present the results of computing E_C for the global solutions found by **dgsol**.

The computation of the coordinate error E_C is known as the orthogonal Procrustes problem in the numerical analysis literature; E_C can be computed accurately and efficiently from the singular value decomposition of the 3×3 matrix $X^T Y$, where $X = [x_1, \ldots, x_m]$ and $Y = [y_1, \ldots, y_m]$. For details see, for example, Golub and VanLoan [9, page 582].

The coordinate error E_C is commonly used to measure the deviation between structures. In particular, many researchers require that structures have an E_C of 1–2 Å to be considered similar, while others only require an E_C of 2–3 Å. These criteria are not universally accepted since E_C has a number of deficiencies. In

Table 5.2. Coordinate error E_C for 100-atom (left) and 200-atom (right) fragments

	E_C (RMSD)			E_C (RMSD)	
ε	Min	Ave	ε	Min	Ave
0.04	0.063	0.067	0.04	1.5	1.7
0.08	0.11	0.12	0.08	1.5	1.9
0.12	0.27	0.60	0.12	1.4	2.2
0.16	0.37	1.0	0.16	0.7	2.9

particular, E_C is dependent on the scaling of the coordinates. For a discussion of these deficiencies, see Mairov and Crippen [21].

If we accept the view that proteins with E_C of 2–3 Å are similar, then the results in Table 5.2 show that, on the average, **dgsol** is able to find structures that are similar to the original structure. If we adopt the more stringent criterion that structures with E_C of 1–2 Å are similar, then our results show that **dgsol** finds structures that are similar if $\varepsilon \leqslant 0.08$, that is, if the lower and upper bounds differ by about 16%. If we increase ε past 0.08 then the average E_C becomes larger than 2 Å, but, as shown by the smallest E_C, we are still able to find similar structures.

We did not expect to find small values for E_C since our data does not include all the distances, but only the distances between successive residues in the sequence. Moreover, note that we are not including all the distances within a given cutoff, as when the sparsity set $\mathcal{S}$ is specified by (5.2).

5.4. EXPERIMENT 4

In the last experiment we did not consider the performance of **dgsol**. Instead, we wanted to verify, computationally, that the number of minimizers of the Gauss–Hermite transform $\langle f \rangle_{\lambda,q}$ decreases as λ increases. This experiment is interesting from a theoretical viewpoint because it provides insight into the smoothing approach. We used **vmlm** with the 100 random staring points generated by algorithm **struct** on the 200-atom fragment.

The number of distinct minimizers found by **vmlm** is plotted in Figure 5.2. For these results, minimizers x_1 and x_2 of $\langle f \rangle_{\lambda,q}$ are declared to be the same if

$$|\langle f \rangle_{\lambda,q}(x_1) - \langle f \rangle_{\lambda,q}(x_2)| \leqslant \tau_r \ \max\{|\langle f \rangle_{\lambda,q}(x_1)|, \ |\langle f \rangle_{\lambda,q}(x_2)|\},$$

where $\tau_r = 10^{-6}$, or if

$$\max\{|\langle f \rangle_{\lambda,q}(x_1)|, \ |\langle f \rangle_{\lambda,q}(x_2)|\} \leqslant \tau_a,$$

where $\tau_a = 10^{-2}$. In other words, the minimizers are declared to be equal if they are smaller than τ_a, or if they are larger than τ_a and their relative error is at most τ_r.

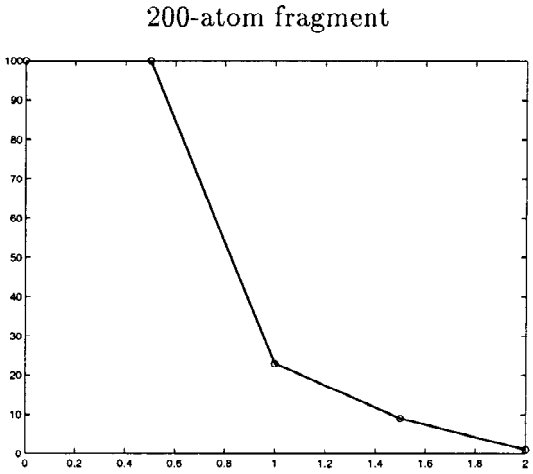

Figure 5.2. Number of minimizers of $\langle f \rangle_{\lambda,q}$ as a function of λ for $\varepsilon = 0.04$.

The number of minimizers is sensitive to the choice of τ_r and τ_a, but the general trend is clear. The results in Figure 5.2 show that, as predicted by the theory, the number of minimizers of $\langle f \rangle_{\lambda,q}$ decreases as λ increases. Also note that the initial drop in the number of minima is dramatic as λ varies in $(0, 1)$.

6. Concluding remarks

Our computational results suggest that protein structures can be determined by solving a distance geometry problem with **dgsol** and that the approach based on **dgsol** is significantly more reliable and efficient than multi-starts with an optimization code. Our results also raise a number of interesting issues that we plan to address in future work. In particular, we wish to expand our testing to larger protein fragments (possibly a complete protein) and to distance data generated from NMR experiments. Another interesting issue is the dependence of the structures on the distance dat a. From a mathematical viewpoint, we do not know when structures can be determined uniquely with exact, but incomplete distance data. For some results in this direction, see Hendrickson [13,14].

Acknowledgments

Our work has been influenced, in particular, by conversations with Paul Bash, Gordon Crippen, and Teresa Head-Gordon. Gail Pieper, as usual, deserves special thanks for her comments on the manuscript.

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, US Department of Energy, under Contract W-31-109-Eng-38 and by the Argonne Director's Individual Investigator Program.

References

1. Aszódi, A., Gradwell, M.J. and Taylor, W.R. (1996), Protein fold determination using a small number of distance restraints, in: H. Bohr and S. Brunak (eds), *Protein Folds, A Distance Based Approach* (pp. 85–97).
2. Blaney, J.M. and Dixon, J.S. (1994), Distance geometry in molecular modeling, in: CRC Press. K.B. Lipkowitz and D.B. Boyd (eds), *Reviews in Computational Chemistry* (pp. 299–335), vol. 5, VCH Publishers.
3. Brünger, A.T. and Nilges, M. (1993), Computational challenges for macromolecular structure determination by X-ray crystallography and solution NMR-spectroscopy, *Q. Rev. Biophys.* 26: 49–125.
4. Crippen, G.M. and Havel, T.F. (1988), *Distance Geometry and Molecular Conformation*, John Wiley & Sons.
5. Engh, R.A. and Huber, R. (1991), Accurate bond and angle parameters for X-ray protein structure refinement, *Acta Cryst.* 47: 392–400.
6. Gautschi, W. (1994), Algorithm 726: ORTHOPOL – A package of routines for generating orthogonal polynomials and Gauss-type quadrature rules, *ACM Trans. Math. Software* 20: 21–62.
7. Glunt, W., Hayden, T.L. and Raydan, M. (1993), Molecular conformation from distance matrices, *J. Comp. Chem* 14: 114–120.
8. Glunt, W., Hayden, T.L. and Raydan, M. (1994), Preconditioners for distance matrix algorithms, *J. Comp. Chem* 15: 227–232.
9. Golub, G.H. and van Loan, C.F. (1989), *Matrix Computations*, The Johns Hopkins University Press.
10. Guan, Y., Zhang, H., Konings, R.N.H., Hilbers, C.W., Terwilliger, T.C. and Wang, A.H.-J. (1994), Crystal structure of Y41H and Y41F mutants of gene V suggest possible protein-protein interactions in the GVP-SSDNA complex, *Biochemistry* 33: 7768.
11. Havel, T.F. (1991), 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: 43–78.
12. Havel, T.F. (1995), *Distance geometry*, in Encyclopedia of Nuclear Magnetic Resonance, D.M. Grant and R.K. Harris, eds., John Wiley & Sons, 1995, pp. 1701–1710.
13. Hendrickson, B.A. (1991), The molecule problem: Determining conformation from pairwise distances, Ithaca, NY: Cornell University, Ph.D. thesis.
14. Hendrickson, B.A. (1995), The molecule problem: Exploiting structure in global optimization, *SIAM J. Optimization* 5: 835–857.
15. Hoch, J.C. and Stern, A.S. (1992), A method for determining overall protein fold from nmr distance restraints, *J. Biomolecular NMR* 2: 535–543.
16. Kostrowicki, J. and Piela, L. (1991), Diffusion equation method of global minimization: Performance for standard functions, *J. Optim. Theory Appl.* 69: 269–284.
17. Kostrowicki, J., Piela, L., Cherayil, B.J. and Scheraga, H.A. (1991), Performance of the diffusion equation method in searches for optimum structures of clusters of Lennard-Jones atoms, *J. Phys. Chem.* 95: 4113–4119.
18. Kostrowicki, J. and Scheraga, H.A. (1992), Application of the diffusion equation method for global optimization to oligopeptides, *J. Phys. Chem.* 96: 7442–7449.
19. Kuntz, I.-D., Thomason, J.F. and Oshiro, C.M. (1993), Distance geometry, in: N.J. Oppenheimer and T.L. James (eds), *Methods in Enzymology*, vol. 177, pp. 159–204, Academic Press.
20. Le Grand, S., Elofsson, A. and Eisenberg, D. (1996), The effect of distance-cutoff on the performance of the distance matrix error when used as a potential function to drive conforma-

tional search, in: H. Bohr and S. Brunak (eds), *Protein Folds, A Distance Based Approach*, pp. 105–113, CRC Press, Inc.

21. Mairov, V.N. and Crippen, G.M. (1995), Size-independent comparison of protein three-dimensional structures, *Proteins: Struct. Func. Genetics* 22: 273–283.
22. Moré, J.J. and Wu, Z. (1995), ε-optimal solutions to distance geometry problems via global continuation, in: P.M. Pardalos, D. Shalloway and G. Xue (eds), *Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding*, pp. 151–168, American Mathematical Society.
23. Moré, J.J. and Wu, Z. (1995), *Global continuation for distance geometry problems*, Preprint MCS-P505-0395, Argonne National Laboratory, Argonne, Illinois. Accepted for publication in the SIAM Journal on Optimization.
24. Moré, J.J. and Wu, Z. (1996), Smoothing techniques for macromolecular global optimization, in: G.D. Pillo and F. Giannessi (eds.), *Nonlinear Optimization and Applications*, pp. 297–312, Plenum Press.
25. Piela, L., Kostrowicki, J. and Scheraga, H.A. (1989), The multiple-minima problem in the conformational analysis of molecules: Deformation of the protein energy hypersurface by the diffusion equation method, *J. Phys. Chem.* 93: 3339–3346.
26. Scheraga, H.A. (1992), Predicting three-dimensionall structures of oligopeptides, in: K.B. Lipkowitz and D.B. Boyd (eds), *Reviews in Computational Chemistry*, vol. 3, pp. 73–142, VCH Publishers.
27. Skinner, M.M., Zhang, H., Leschnitzer, D.H., Guan, Y., Bellamy, H., Sweet, R.M., Gray, C.W., Konings, R.N.H., Wang, A.H.-J. and Terwilliger, T.C. (1994), Structure of the gene V protein of bacteriophage F1 determined by multi-wavelength X-ray diffraction on the selenomethionyl protein, *Proc. Nat. Acad. Sci. USA* 91: 2071.
28. Smith-Brown, M.J., Kominos, D. and Levy, R.M. (1993), Global folding of proteins using a limited number of distance constraints, *Protein Engng.* 6: 605–614.
29. Stroud, A.H. and Secrest, D. (1996), *Gaussian Quadrature Formulas*, Prentice-Hall, Inc.
30. Torda, A.E. and van Gunsteren, W.F. (1992), Molecular modeling using nuclear magnetic resonance data, in: K.B. Lipkowitz and D.B. Boyd (eds), *Reviews in Computational Chemistry*, vol. 3, pp. 143–172, VCH Publishers
31. Wu, Z. (1996), The effective energy transformation scheme as a special continuation approach to global optimization with application to molecular conformation, *SIAM J. Optimization* 6: 748–768.