BIOINFORMATICS

Vol. 00 no. 00 2010
Pages 1–7

A Convex Programming Model for Protein Structure Prediction

Chris Kauffman* and George Karypis

Department of Computer Science, University of Minnesota
117 Pleasant St SE, Room 464, Minneapolis, MN 55455

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: Distance geometry methods provide a mechanism to produce atom coordinates based on distance measurements taken from such sources as protein nuclear magnetic resonance spectroscopy experiments. Traditional methods suffice when accurate distance measurements are available but it is unclear effectively well they handle noisy distance measurements.

Results: We introduce the concept of convex prediction and convex design which are combined in Domo, a novel method to solve the coordinate realization problem. On a benchmark of 124 proteins, Domo produces coordinate estimates which are statistically superior to standard distance geometry techniques with a very competitive computational cost. One may interpret convex prediction as a rough potential energy model for protein structures. We investigate the robustness of convex prediction for its applications to full ab initio protein structure estimation.

Availability: Software developed for this study is available at `http://bioinfo.cs.umn.edu/supplements/eccb2010` along with supplementary material.

Contact: kauffman@cs.umn.edu, karypis@cs.umn.edu

1 INTRODUCTION

In this work work we study how to estimate atom coordinates from noisy distance information, alternately called the coordinate realization or embedding problem. The problem arises during nuclear magnetic resonance (NMR) spectroscopy experiments in which the distances between proximal atoms in a molecule can be determined but a computational method must be used to infer coordinates for the structure under study. With the assumption of low noise in the input distances, standard methods provide reasonable solutions for this coordinate realization problem. Here we examine the tolerance of two standard methods to small and large degrees of noise and propose a novel approach to the problem.

The primary tool we employ for solving distance geometry problems is convex optimization, particularly semidefinite programming (SDP). SDPs are similar to linear programs in that a linear objective function is optimized with affine constraints. The central difference is that the optimization variable in a SDP is a positive semidefinite matrix which is capable of representing pairwise distances. Constraints in the SDP become constraints on the distances between atoms in the system.

The central idea of our method is to derive an objective for a SDP which targets the desired pairwise distances. This process we call *convex design*. After such an objective has been found it is minimized, a process referred to as *convex prediction*. The result of convex prediction is readily converted into atom coordinates for a final structure prediction. The whole process is referred to as the *design of minimization objective* method, abbreviated Domo.

The value of Domo is apparent when the quality of its coordinate predictions is compared to standard distance geometry approaches on a large set of proteins which are represented in a reduced model. Pairwise distances between bonded atoms in the protein are assumed to be known exactly. The distances between unbonded atoms in the proteins are perturbed and provided to the distance geometry methods to test how well they estimate the original coordinates at various levels of noise. Our experiments on show Domo to be statistically superior than two standard distance geometry methods, producing mean RMSDs which are half that of competitors at some noise levels. In most cases, Domo is able to effectively satisfy distance constraints with very low comparatively computational cost.

Convex prediction may be interpreted as a crude potential energy model for protein structures. Domo determines the energy function to be minimized from a distance matrix by using convex design to derive an appropriate objective. The quality of structure predictions made by Domo indicates that these objectives are worthy of study in their own right. Combined with a sequence-based method to predict the objectives, pure convex prediction would constitute a full system for ab initio protein structure estimation. To gain insight into how tolerant such a system might be to misprediction of the objectives, we compared pure convex predictions of protein structure based on noisy objectives to the corresponding true structures. Our results suggest potential for objective-based approaches to quickly and accurately predict protein structure.

2 RELATED WORK

Our model functions in the space of distance matrices making it natural to examine how other distance geometry methods fare in comparison. Distance geometry approaches roughly amount to using a pairwise distance matrix as input and producing d-dimensional coordinates which adhere as closely as possible to

*to whom correspondence should be addressed

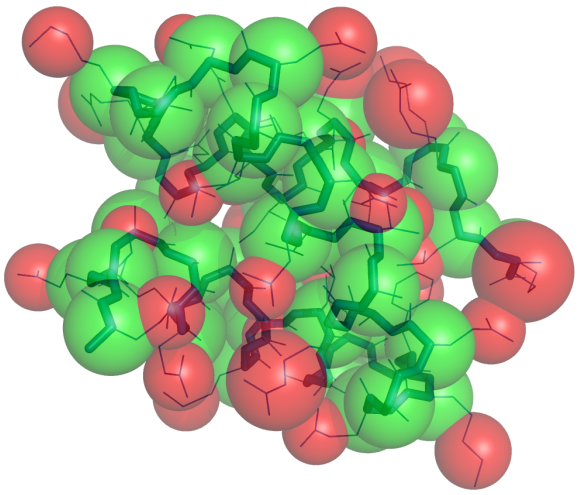

Fig. 1. The hard sphere model used for protein d1ixsa_ (ASTRAL ID) in the study. Main chain spheres are colored green and side chain spheres are colored red.

the available proximity information. In our case $d = 3$ which corresponds to three-dimensional molecules.

A number of disciplines use distance geometry methods. In statistics and psychology, multidimensional scaling has been studied as a way to visualize data using only similarity or dissimilarity information [2]. Nuclear magnetic resonance spectroscopy (NMR) can generate distance-based information about molecules in solution, particularly proteins. Converting this distance information into atom coordinates allows the structure of the protein to be ascertained [13, 6]. Distance geometry methods have also been explored in their own right for ab initio protein structure prediction [10]. Finally, there has been renewed interest in distance geometry through semidefinite programming due to its applications to wireless sensor location [4] and to graph embedding [11]. In both these problems, the central idea is to solve a SDP involving distance information and convert the solution into coordinates for the final problem answer. The convex prediction described in this work is close to structure preserving embedding in [11]. The primary difference is that we use convex design to obtain objectives specific to each problem rather than use the same heuristic objective for all problem instances. We also focus on geometry constraints and explicitly attempt to produce low-dimensional solutions which are not done in structure preserving embedding.

3 METHODS

3.1 A Reduced Protein Model

We work with a reduced model of protein structure in which each residue is represented by one or two hard spheres. The first hard sphere is centered on a residue's α-carbon and contains all heavy atom centers on the main chain and the β-carbon if it is present. The second hard sphere is at the barycenter of any remaining side chain atoms and contains their centers. Proline was treated specially and was represented with only a single ball which was centered on the α-carbon. The distance between bonded main chain spheres, bonded main chain and side chain spheres, and the radius of each sphere were treated as known as there is little variation for these across proteins.

For convenience, we will refer to each sphere as a single 'atom' despite the fact it represents several real atoms. Reduced models of proteins are frequently used in structure prediction methods as reviewed in [9]. Figure 1 shows an example of the reduced, hard sphere model used in our study.

3.2 Basic Concepts

We use upper-case roman letters to denote matrices (G) and and lower-case roman letters to denote vectors (b). In some cases, vectors represent *vectorized matrices* which are the result of stacking all the columns of the matrix. In these cases, upper case refers to the matrix representation (X) and lower case to the vectorized matrix (x). When referring to specific entries of a matrix we use explicit indexing: $A(i, j)$ refers to the ith column, jth row while $A(i, :)$ refers to the ith row. Entries of vectors are referred to in this way or by subscript: $b(k)$ and b_k are the kth entry of the vector. Greek letters (α) are used for scalars. The notation $||s||_1$ denotes the one-norm of vector s (sum of absolute values) while $||P||_F$ denotes the Frobenius norm of the matrix P (sum of squares of all entries). We use $\mathbf{1}$ to represent a column vector of ones of appropriate size to the situation.

A set of n atoms in d dimensions is represented as a matrix $R \in \mathbb{R}^{d \times n}$. In all cases we assume that the atoms are centered so that $R\mathbf{1} = 0$. We are most concerned with $d = 3$ as atoms in three-dimensional space are sought.

The squared distance matrix for R is referred to as D with $D(i, j)$ equal to the squared distance between the ith and jth atom. Unless otherwise noted, we will refer to a squared distance matrix simply as a distance matrix. Distance matrices are square and symmetric with a zero diagonal.

For a collection of atoms $R \in \mathbb{R}^{d \times n}$, the associated Gram matrix is defined $G = R^T R$. Gram matrices are square and symmetric. Assuming the atoms are centered, each set of atom coordinates has a unique distance matrix and corresponding Gram matrix. By definition, Gram matrices are positive semidefinite (PSD): they have d positive eigenvalues and the remainder of the eigenvalues are zero. The PSD property is denoted $G \succeq 0$.

Given a Gram matrix, corresponding atom coordinates may be recovered from it via a eigenvalue decomposition. Let $G = Q\,\mathrm{diag}(\lambda)Q^T$ where Q is the matrix of eigenvectors and λ the vector of associated eigenvalues, sorted in decreasing order. Then

$$R = \mathrm{diag}(\sqrt{\lambda})Q^T. \tag{1}$$

If G has d nonzero eigenvalues, then its atoms exist in d-dimensional space. When three-dimensional coordinates are desired and there are more than three nonzero eigenvalues, the remaining eigenvalues are set to zero.

Distance and Gram matrices are related by the entry-wise equation

$$D(i, j) = G(i, i) + G(j, j) - G(i, j) - G(j, i). \tag{2}$$

A distance matrix of size n may also be converted into a Gram matrix. Let $V = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ (the geometric centering matrix in Section 5.4.2.2 of [7]); then

$$G = -\frac{1}{2}VDV. \tag{3}$$

Note that if D is not a valid distance matrix, the resulting Gram matrix will not be positive semidefinite (i.e. it may have some negative eigenvalues).

3.3 Semidefinite Programming (SDP)

Mathematical programming deals with problems in which an objective function is to be minimized according to constraints on the optimization variables. In semidefinite programming (SDP), the optimization variable is required to be a positive semidefinite matrix and the constraints and objective are linear in this variable. The standard form of SDPs is typically written as follows:

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{s.t.} & Ax = b \\ & x \succeq 0. \end{array} \tag{4}$$

In this program, the problem data are a vectorized symmetric matrix $c \in \mathbb{R}^{n^2}$, referred to as the objective, and matrix $A \in \mathbb{R}^{m \times n^2}$ with vector

$b \in \mathbb{R}^m$ which are the constraints. The optimization variable $x \in \mathbb{R}^{n^2}$ is required to be a vectorized symmetric PSD matrix, indicated by $x \succeq 0$. The equation $Ax - b = 0$ constitutes a set of linear constraints on x.

A key feature of SDPs is that they are convex and thus polynomial time algorithms exist to locate a globally optimal solution or indicate that the problem is infeasible or unbounded. [3] offers a good introduction to convex programming, a wide variety of applications of SDP are discussed in [12], and [7] discusses relation use of SDPs to distance geometry extensively.

A few modifications to Program 4 are needed to accommodate our distance geometry problems. In order to handle inequality constraints on the distances between atoms, linear slack variables are introduced in addition to the semidefinite variable.

$$\begin{array}{ll} \min\limits_{x_l, x_s} & c_l^T x_l + c_s^T x_s \\ \text{s. t.} & A_l x_l + A_s x_s = b \\ & x_l \geq 0,\ x_s \succeq 0. \end{array} \tag{5}$$

The semidefinite variable x_s represents the unknown Gram matrix of the a protein structure while the linear variables x_l are slacks that will prevent atoms from clashing. Associated with the linear variables is a linear constraint matrix A_l and with the semidefinite variable another constraint matrix A_s. The combination of A_l, A_s, and the right-hand side vector b allow distance constraints to be encoded.

Program 5 can include both exact distance constraints and minimum distance constraints. Suppose atoms i and j have radii r_i and r_j respectively and we wish to add a constraint to prevent these atoms from clashing, i.e. they must be at a minimum distance of $r_i + r_j$. Let this constraint have index k. It is encoded by setting row k of the constraint matrices so that

$$\begin{array}{rl} A_l(k,:)x_l + A_s(k,:)x_s = & X_s(i,i) + X_s(j,j) \\ & -X_s(i,j) - X_s(j,i) - x_l(k) \end{array} \tag{6}$$

and setting $b(k) = (r_i + r_j)^2$. The expression

$$X_s(i,i) + X_s(j,j) - X_s(i,j) - X_s(j,i),$$

corresponds to the squared distance between atoms i and j (see Equation 2). Along with the constraint that $x_l(k) \geq 0$, Equation 6 enforces the desired minimum distance between atoms i and j. Omitting the slack variable $x_l(k)$ creates a distance equality constraint.

The result of solving Program 5 is the optimal point (x_l^*, x_s^*). The vectorized PSD variable x_s^* may be converted into a Gram matrix simply by reshaping it into a square matrix.

There are a number of issues regarding effective implementation of Program 5 which are discussed in the appendices of this paper. These include reducing the rank of solutions (Appendix A) and efficiently handling the inequality constraints (Appendix B).

3.4 Existing Methods

Two standard distance geometry approaches to coordinate realization are discussed here which are compared to our new method (presented in Section 3.5). All of the methods adhere to the following protocol during evaluation.

1. Provide the method with a noisy distance matrix $\hat{D}$.
2. Obtain a predicted Gram matrix $\hat{G}$ from the method.
3. Convert $\hat{G}$ to predicted structure coordinates $\hat{R}$ according to Equation 1.
4. Compute the RMSD between the prediction $\hat{R}$ and true structure R.

Thus, each method uses a noisy distance matrix as input and produces a Gram matrix which is converted to structure coordinates for evaluation.

3.4.1 Multidimensional Scaling Multidimensional scaling (MDS) has long been used to convert dissimilarity information into coordinates which may be visualized in low dimensional space [2]. *Classical multidimensional scaling (CMDS)* converts a distance matrix into a Gram matrix according to Equation 3. If negative eigenvalues appear in the matrix resulting from Equation 3, they are zeroed to produce the final Gram matrix for CMDS.

CMDS can be performed very fast compared to the other methods as it involves only a few matrix multiplications and an eigenvalue decomposition. A major drawback of CMDS is that it does not handle constraints. As a result, very noisy distance matrices produce highly unrealistic structures with many atom clashes.

3.4.2 Nearest Euclidean Distance Matrix Method Given a distance matrix, we may formulate a SDP to locate a Gram matrix matching the target distances as closely as possible while fulfilling known exact distance constraints. This is referred to as the *nearest Euclidean distance matrix method* and abbreviated *NEDM*.

Typically there is a threshold beyond which no distance information is available. Here we use 15 Å as the threshold. Given noisy (squared) distance matrix $\hat{D}$, let $\mathcal{D} = \{(i,j) | \hat{D}(i,j) \leq 15\}$ be the set of noisy distances less than 15Å.

We solve the nearest EDM problem using the following SDP:

$$\begin{array}{ll} \min\limits_{s, x_s, x_l} & ||s||_1 \\ \text{s. t.} & A_l x_l + A_s x_s = b \\ & X_s(i,i) + X_s(j,j) \\ & -X_s(i,j) - X_s(j,i) + s_{ij} \quad = \hat{D}(i,j), \forall (i,j) \in \mathcal{D} \\ & s \in \mathbb{R}^{|\mathcal{D}|}, x_l \geq 0, x_s \succeq 0. \end{array} \tag{7}$$

Here we use the same constraint matrices A_l and A_s as are used in Program 5 to enforce the minimum distances and known backbone equality distances. The norm of slack vector s is minimized which guides the resulting structure towards the distances in $\hat{D}$ (values of $s(k)$ close to zero indicate the target distances have been matched well). The process of solving Program 7 is very similar to solving Program 5. The same SDP solver is used for both while employing dimensionality reduction and efficient inequality handling (see Appendix A and Appendix B).

We chose to ignore distances larger than 15Å in part because using complete distance information in NEDM enlarges the resulting optimization problem significantly.

3.5 Design of Minimization Objective (Domo)

The NEDM method of Section 3.4.2 has the disadvantage of being a large optimization problem. In NEDM, we begin with distance equality constraints between bonded atoms and inequality constraints between unbonded atoms. Then, for each available estimated distance $\hat{D}(i,j)$, additional constraints are required to target that distance. It would be beneficial to avoid these additional constraints.

The method we introduce here does just that. Rather than create constraints and minimize slacks on the available distances as is done in NEDM, it derives an objective which directly targets the available distances. The objective is then minimized to obtain the predicted Gram matrix. This process is referred to as the *design of minimization objective* method, abbreviated *Domo*. At a high level the following steps are taken.

1. Accept a noisy distance matrix $\hat{D}$ as input.
2. Use CMDS used to obtain an initial Gram matrix $\hat{G}$ (Section 3.4.1).
3. Use *convex design* on $\hat{G}$ to obtain an objective c_l .
4. Minimize c_l in *convex prediction* to obtain a new predicted Gram matrix $\hat{G}'$ which is returned as the answer .

Convex design (Step 3) requires a problem model for which it will design an objective. This model is provided by convex prediction (Step 4). Thus we first describe convex prediction and then convex design.

3.5.1 Convex Prediction Convex prediction is simply the minimization of a given objective subject to distance constraints. The exact form for

convex prediction is

$$\begin{array}{ll} \min\limits_{x_l, x_s} & c_l^T x_l \\ \text{s. t.} & A_l x_l + A_s x_s = b \\ & x_l \geq 0,\ x_s \succeq 0. \end{array} \tag{8}$$

The only difference from Program 5 is that Program 8 uses only the linear objective c_l and omits the SDP objective c_s. This choice is explained next.

One interpretation of Program 8 is that it represents a very crude model of the potential energy of a protein configuration. Recall that x_l represents the excess distances between nonbonded atoms in the model above their minimum distance. Associated with each element of x_l is a term in c_l which can be interpreted as the level of attraction or repulsion for a pair of nonbonded atoms. This means energy in the system is calculated based only on the weighted sum of squared distances between nonbonded atoms. Though crude, this model of protein structures will be seen to have several desirable properties, one of the most novel being efficient designability which is discussed in the sequel.

3.5.2 Convex Design The natural question to pose is how one might obtain objectives for convex prediction. Given the Gram matrix for a protein, we would like a way to produce an objective which, when minimized in Program 8, closely reproduces the structure of interest. Finding a good objective for a Gram matrix can be described as another mathematical problem based on the duality theory of semidefinite programming. We call the process of finding the objective convex design.

We present convex design in terms of the standard SDP in Program 4 for brevity. Our implementation actually designs objectives for Program 8 which is a straight-forward extension.

Convex programming problems like SDP come in pairs: the primal and dual problems. Program 4 is referred to as the *primal* problem and its associated *dual* problem is the following.

$$\begin{array}{ll} \max\limits_{y, z} & b^T y \\ \text{s. t.} & A^T y + z - c = 0 \\ & y \in \mathbb{R}^m, z \succeq 0. \end{array} \tag{9}$$

The dual SDP uses the same data (A, b, c) as the primal but is a maximization problem over free vector $y \in \mathbb{R}^m$ and PSD variable z. Provided both primal and dual problems are feasible and bounded solving either will yield the same optimal value. The primal/dual optimal points (x^*, y^*, z^*) have an important property know as complementary slackness.

$$x^{*T} z^* = 0. \tag{10}$$

Suppose we have a protein structure represented as a Gram matrix X^* and its equivalent vectorized form x^*. Finding a c for Program 4 with x^* as its minimizer is done by solving the dual problem with a few modifications. We allow c to vary and enforce complementary slackness using the known x^*. In addition, a trace constraint is used which guarantees that some of the eigenvalues of z are nonzero which avoids degenerate solutions. The resulting program is shown below.

$$\begin{array}{ll} \text{find} & c, y, z \\ \text{s. t.} & A^T y + z - c = 0 \\ & x^{*T} z = 0 \\ & \text{Trace}(z) = n \\ & c \in \mathbb{R}^{n^2}, y \in \mathbb{R}^m, z \succeq 0. \end{array} \tag{11}$$

The results of solving Program 11 are the dual feasible variables y, z and an objective vector c. When used in Program 4, c will produce the optimal point x^* used to design it. Some implementation details of Program 11 are discussed in the Appendix C.

3.5.3 Differences between Domo and CMDS One might wonder why Domo would produce a different structure than CMDS as Domo is targeting the CMDS answer. The reasons for this are two-fold. First, CMDS does not allow for constraints to be met so the structures it produces may be highly unrealistic. Domo can use constraints to avoid atom clashes in its predictions. Second, CMDS does not attempt to minimize the dimensionality of its answer: the given distance matrix is directly converted to a Gram matrix and high-dimensional answers may ensue. Additionally, negative eigenvalues in the Gram matrix are simply truncated which is an indication of a poor solution. In contrast, Domo employs rank minimization techniques to find low-dimensional solutions and restricts its search space to PSD matrices. At high noise levels, the resulting structure coordinates will be much closer to three dimensions than in CMDS.

3.6 Pure Convex Prediction Methods

According to the interpretation of convex prediction as a rough model for potential energy, we see that Domo estimates the energy function to be minimized using convex design. These estimated objectives produce fairly good structure predictions as reported later. Thus, if an objective can be estimated by some means, minimizing it in Program 8 will produce a structure prediction. Of particular interest is the potential to estimate objectives based on features of a protein's sequence which, coupled with convex prediction, would constitute a full system for protein structure prediction

An immediate question to be addressed is how accurate such predicted objectives would need to be in order for convex prediction to produce good estimated structures. As a first attempt to answer this question, we examine the quality of structure estimates produced by pure convex prediction when noisy objectives are provided as input.

Several variants of pure convex prediction were tested which minimize the provided objective in Program 8. The simplest variant is to use the input objective as is, referred to as *Obj*. We found that the quality of predicted structures is sensitive to the presence of large negative values in the objective. Under the potential energy model interpretation, this corresponds to repulsive forces overwhelming attractive forces which causes the structure to unfold into an extended conformation. To prevent this, the smallest 1% and 2% of objective values are zeroed in variants *Objz1* and *Objz2*, respectively. As Objz1 and Objz2 had much better results than Obj, we omit Obj from further discussion.

The inputs to Objz1 and Objz2 are objectives rather than distance matrices so a direct comparison to CMDS/NEDM/Domo is tenuous. However, the good properties of pure convex prediction indicate that it is worth pursuing as a means of ab initio protein structure prediction.

4 MATERIALS

4.1 Optimization Software and Hardware

The backbone of our methods is a SDP solver for which we chose CSDP [1]. Matlab™ was used to implement convex iteration and the active set approach [8]. Convex design problems were solved using the `lsqlin()` function from the Matlab optimization toolbox.

Computations were run on a cluster of 15 machines with a total of 80 compute nodes. Each machine ran Ubuntu Linux 8.04 and had either an AMD Opteron processor and 4GB of RAM (10 machines, 4 nodes each) or an Intel Xeon processor with 16GB of RAM (5 machines, 8 nodes each).

4.2 Data Sets

We chose 124 proteins from the ASTRAL compendium for our test set [5]. The proteins range from 50 to 100 residues long and were selected for their general compactness. There is 40% sequence identity or less between proteins. Each protein was comprised of 87 to 189 spheres with an average of 132 spheres (in the reduced model, Section 3.1).

4.3 Noise Model

Our main interest is to study how the different coordinate realization methods handle noise in the input. We use a very simple noise model to perturb the distance matrices which are used as input to the methods.

Noisy distance matrices were generated using the true distance matrix D, perturbation matrix P, and noise level γ according to the following:

$$\hat{D} = (1-\gamma)v + \gamma\frac{||D||_F}{||P||_F}P. \tag{12}$$

The perturbations P symmetric matrices generated from a Gaussian distribution with mean 0 and variance 1. If a negative distance resulted from adding a perturbation to a distance, its absolute value was taken. When the noise level is $\gamma = 0.5$, the perturbed distance matrix $\hat{D}$ is half truth and half noise.

Noise levels of $\gamma \in \{0.10, 0.25, 0.50, 0.60, 0.75\}$ were tested for five different perturbations per protein. This resulted in a total of $124 \times 5 \times 5 = 3100$ predicted structures per method.

When testing the pure convex prediction methods (Section 3.6), the same noise model was used but with exact objectives rather than distance matrices. For each protein, convex design (Section 3.5.2) was used to find a c_l which perfectly predicts the protein's structure in Program 8. This objective was then perturbed to $\hat{c_l}$ according to Equation 12 and used as input to the pure convex prediction methods.

4.4 Evaluation Metrics

A standard measure of quality for a model protein structure is to compute its root mean squared deviation (RMSD) to the true protein. After optimally superimposing the coordinates of the two structures, RMSD measures how far the model deviates from the truth. All of the predictions our methods produce may be reflected coordinates as we are dealing solely with distance-based predictions. For that reason, we measure the RMSD of both a given prediction and its reflection to the true structure and report the better RMSD. Under mild conditions, such as the presence of an α-helix in the predicted structure, it is possible to determine which orientation of the predicted protein is appropriate but we leave automatic structure determination to future work.

We collect the RMSDs of each prediction and use Welch's t-test on the measurements to determine the overall quality of each method. This test compares the average performance of one method versus another. In each test, we report the p-value which is typically considered significant if $p \leq 0.05$. Welch's t-test is preferred as it does not assume the two methods being evaluated have equal variance in their predictions. Different subgroups of the data are analyzed to determine where each method fares best.

5 RESULTS

5.1 Performance of Distance Geometry Methods

Our main results for predicted structure quality are shown in Table 1 and Figure 2. Domo outperforms the standard methods, CMDS and NEDM, at all noise levels. The mean RMSD of Domo is lower than the other methods and the t-tests clearly indicate it is significantly better in each case.

All three methods degrade in performance as the noise level increases, but Domo seems to do so more gracefully up to $\gamma = 0.60$. At $\gamma = 0.75$ the performance of CMDS, NEDM, and Domo is closer in terms of mean RMSD, but Domo still statistically outperforms the others. For brevity, we have omitted data concerning atom clashes in the predicted structures, but it is worth mentioning that clashes become quite severe in CMDS predictions at higher noise levels. Despite having a similar mean RMSD to the other methods, its structure predictions appear poor when the constraint violations are considered. NEDM and Domo do not suffer from this problem nearly as badly.

Table 1. RMSD performance of distance geometry methods with p-value comparisons

Noise	$\gamma = 0.10$			$\gamma = 0.25$		
Method	CMDS	NEDM	Domo	CMDS	NEDM	Domo
μ_{rmsd}	0.72	1.02	0.39	2.11	2.45	1.09
σ_{rmsd}	0.04	0.20	0.06	0.14	0.55	0.14
CMDS	-	1.00	0.00	-	1.00	0.00
NEDM	0.00	-	0.00	0.00	-	0.00
Domo	1.00	1.00	-	1.00	1.00	-

Noise	$\gamma = 0.50$			$\gamma = 0.60$		
Method	CMDS	NEDM	Domo	CMDS	NEDM	Domo
μ_{rmsd}	5.59	6.51	3.27	7.88	8.78	5.56
σ_{rmsd}	0.42	1.29	0.46	0.58	1.28	0.82
CMDS	-	1.00	0.00	-	1.00	0.00
NEDM	0.00	-	0.00	0.00	-	0.00
Domo	1.00	1.00	-	1.00	1.00	-

Noise	$\gamma = 0.75$		
Method	CMDS	NEDM	Domo
μ_{rmsd}	11.09	10.42	9.88
σ_{rmsd}	0.89	1.15	1.05
CMDS	-	0.00	0.00
NEDM	1.00	-	0.00
Domo	1.00	1.00	-

The mean RMSD and standard deviation at each noise level for each method is given in the first two rows. The remaining rows show the p-values of t-Tests measuring whether the mean RMSD performance of the column method is better than the row method. A small value indicates the column method outperforms the row method. Each comparison involves 124 proteins with five different perturbations for a total of 620 observations.

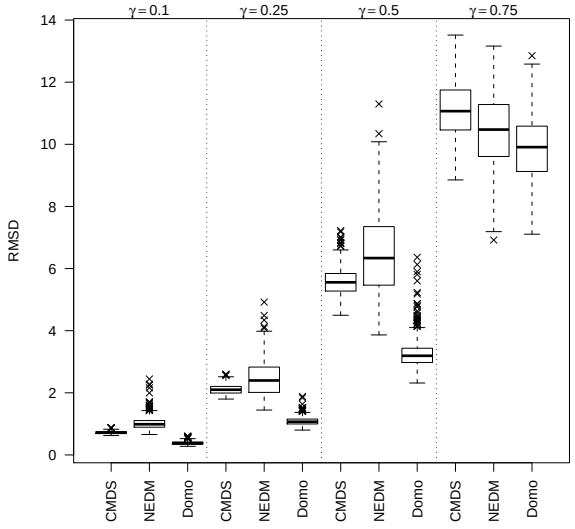

Fig. 2. Box and whisker plot of RMSDs for the distance geometry methods at four noise levels. The center line is at the data median, the box starts at 25th quartile and ends at the 75th quartile. Outliers are shown above and below the boxplots. The whiskers show $1.5 \times IQR$ beyond the top and bottom quartiles (see any introductory statistics text). Also see Table 1 for significance testing.

Table 2. Mean computation time of methods

	CMDS	NEDM	Domo	Objz1	Objz2
$\mu_{0.10}$	0.0	111.8	5.7	4.7	12.5
$\sigma_{0.10}$	0.0	133.8	6.7	4.4	17.5
$\mu_{0.25}$	0.0	153.2	6.1	5.5	12.6
$\sigma_{0.25}$	0.0	202.6	5.8	5.9	13.6
$\mu_{0.50}$	0.0	705.4	19.0	6.1	15.5
$\sigma_{0.50}$	0.0	903.3	20.8	8.0	19.3
$\mu_{0.60}$	0.0	624.5	104.2	5.6	18.0
$\sigma_{0.60}$	0.0	833.4	111.6	7.4	25.3
$\mu_{0.75}$	0.0	511.2	747.0	4.2	16.8
$\sigma_{0.75}$	0.0	602.7	1015.4	4.7	25.3

μ_γ is the mean computation time in minutes at noise level γ while σ_γ is the standard deviation. Each statistic is computed from 620 observations at each noise level.

Table 3. RMSD performance of pure convex prediction methods

	Objz1	Objz2
$\mu_{0.10}$	2.11	2.53
$\sigma_{0.10}$	0.44	0.38
$\mu_{0.25}$	2.21	2.50
$\sigma_{0.25}$	0.46	0.41
$\mu_{0.50}$	2.60	3.06
$\sigma_{0.50}$	0.38	0.43
$\mu_{0.60}$	3.28	3.43
$\sigma_{0.60}$	0.67	0.38
$\mu_{0.75}$	5.28	4.84
$\sigma_{0.75}$	1.79	0.53

μ_γ is the mean RMSD at noise level γ while σ_γ is the standard deviation. Each statistic is computed from 620 observations at each noise level.

Table 2 shows the mean and standard deviation of compute times in minutes for each method. This table includes both the distance geometry approaches (CMDS, NEDM, and Domo) as well as the pure convex prediction methods discussed in Section 5.2.

As mentioned in Section 3.4.1, CMDS takes a neglible amount of time compared to the other methods as it involves no optimization, only matrix factorizations. This speed is tempered by an inability of CMDS to handle constraints resulting in potentially unrealistic structures at high noise levels. The other two methods do not suffer from this behavior as readily.

At noise levels up to $\gamma = 0.50$, Domo compares very favorably on computational cost to NEDM. At these levels, run times are an order of magnitude lower for Domo the NEDM. At $\gamma = 0.60$, the mean time to complete a Domo computation is still 1/6th of NEDM but this trend reverses at $\gamma = 0.75$ where the mean run time for Domo is higher than that of NEDM. The median run times of the two methods are almost the same at this level, 312.9 minutes for Domo and 308.6 minutes for NEDM, but Domo has more outlier cases with high runtime which raises its mean runtime significantly.

5.2 Performance of Pure Convex Prediction Methods

Pure convex prediction methods Objz1 and Objz2 were used to estimate protein structures on the same dataset as the distance geometry methods. The structure prediction quality results are presented in Table 3 and Figure 3.

Direct comparison of pure convex prediction methods to distance geometry methods should be done with the knowledge that the inputs are different and equal levels of perturbation may not actually introduce the same amount of noise in the two cases. With that caveat aside, it seems the pure convex prediction methods are highly robust to perturbations. The structure predictions that they produce even at the highest noise level of $\gamma = 0.75$ are very good with Objz1 averaging a 5.28Å RMSD and Objz2 a 4.84Å RMSD. The level of constraint violation for these methods is also lower on average than any of the distance geometry methods (data not shown). The compute times for these Objz1 and Objz2 are quite competitive as seen in Table 2.

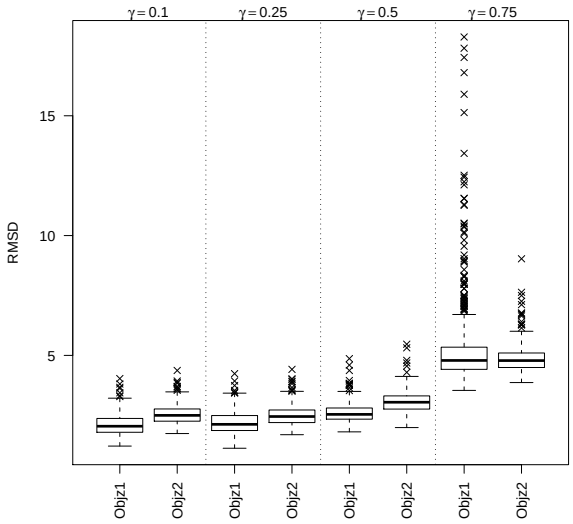

Fig. 3. Box and whisker plot of RMSDs for the two convex prediction variants at four noise levels. See Figure 2 for an explanation of the figure.

Based on these findings, it seems that estimates of the objectives used by Objz1 and Objz2 need not be terribly accurate in order for them to produce a good structure prediction. These findings strongly recommend further study of pure convex prediction methods.

6 CONCLUSION

Domo provides a new means to perform coordinate realization given distance information. We have demonstrated its utility in producing estimated protein structure coordinates based on noisy distance matrices. The method produces relatively good predictions at a lower computational cost than traditional approaches in most cases. There is nothing that precludes using Domo to realize coordinates for types of data other than proteins. We intend to investigate its potential as a general tool to embed data in low dimensions based on dissimilarity information.

The pure convex prediction methods discussed here offer an entirely new direction for protein structure prediction. They quickly produce quality structure estimates even when the objective provided is heavily perturbed. Developing this capability into a full ab initio system will require two additional advancements. First, we do not understand the sensitivity of convex prediction to large negative values in the objective at present and will need to do so for the method to be successful. Second, new machine learning techniques are required to predict an objective from a protein's sequence.

ACKNOWLEDGEMENTS

Funding: This work was supported by the National Institute of Health [T32GM008347, RLM008713A], the National Science Foundation [IIS-0431135], and the University of Minnesota Digital Technology Center.

Supplement: Supplementary materials for this work are available online at `http://bioinfo.cs.umn.edu/supplements/eccb2010`.

REFERENCES

[1]Brian Borchers. Csdp, a c library for semidefinite programming. *Optimization Methods and Software*, 11:613–623, 1999.

[2]Ingwer Borg and Patrick Groenen. *Modern Multidimensional Scaling: Theory and Applications, 2nd Edition*. Springer, 2005.

[3]Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press Cambridge, 2006.

[4]Michael W. Carter, Michael W. Carter, Holly H. Jin, Holly H. Jin, Michael A. Saunders, Michael A. Saunders, Yinyu Ye, and Yinyu Ye. Spaseloc: An adaptable subproblem algorithm for scalable wireless network localization. *SIAM J. on Optimization*, 2005.

[5]John-Marc Chandonia, Gary Hon, Nigel S Walker, Loredana Lo Conte, Patrice Koehl, Michael Levitt, and Steven E Brenner. The astral compendium in 2004. *Nucleic Acids Res*, 32(Database issue):D189–D192, Jan 2004.

[6]G. M. Crippen and T. F. Havel. *Distance Geometry and Molecular Conformation.* Wiley, 1988.

[7]Jon Dattorro. *Convex Optimization and Euclidean Distance Geometry*. Meboo Publishing, 2009.

[8]The MathWorks Inc. Matlab version 7.2. Natick, Massachusetts, 2006.

[9]Andrzej Kolinski and Jeffrey Skolnick. Reduced models of proteins and their applications. *Polymer*, 45(2):511 – 524, 2004. Conformational Protein Conformations.

[10]Enrico O. Purisima and Harold A. Scheraga. An approach to the multiple-minima problem in protein folding by relaxing dimensionality : Tests on enkephalin. *Journal of Molecular Biology*, 196:697–709, Aug 1987.

[11]Blake Shaw and Tony Jebara. Structure preserving embedding. In Léon Bottou and Michael Littman, editors, *Proceedings of the 26th International Conference on Machine Learning*, pages 937–944, Montreal, June 2009. Omnipress.

[12]Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Rev.*, 38(1):49–95, 1996.

[13]K Wurhrich. Protein structure determination in solution by NMR spectroscopy. *Journal of Biological Chemistry*, 265(36):22059–22062, 1990.

A RANK REDUCTION VIA CONVEX ITERATION

Typically the solution x^* returned by SDP solvers has a high rank, i.e. many nonzero eigenvalues. This means that the atoms represented by x^* exist in a high-dimensional space. Unfortunately both constraining rank in an SDP destroys its convexity resulting in a NP-hard problem. The optimization community is greatly interested in heuristics that reduce the rank of SDP problems. A number of these heuristics are explored in [7] and we have found that the convex iteration technique there to be particularly useful for our distance geometry problems.

In convex iteration, one first solves the usual SDP and then modifies the original objective by a search direction designed to favor a low-rank solution. Repeating this process converges to some solution that is typically lower rank than the original. Though there are no guarantees that a desired rank will be achieved, we have found convex iteration works well to produce nearly rank three solutions for our problems.

In [7], the search direction is found by solving another SDP. In this work, we use the spectral decomposition to determine the search direction. This is done to avoid the computational cost of solving another SDP and, in preliminary tests, did not seem to affect the overall accuracy of the coordinates which were ultimately produced.

Let c be the original SDP objective and $x^{(1)}$ be the solution found the first step during convex iteration with $x^{(1)} = Q\,\mathrm{diag}(\lambda)Q^T$. Assume that the eigenvalues are arranged in descending order. We are seeking three-dimensional solutions so we determine the first search direction by allowing the leading eigenvalues to grow and attempting to minimize the remaining eigenvalues. Thus the search direction w is determined by

$$\lambda' = (0, 0, 0, 1 + \lambda_4, \cdots, 1 + \lambda_n), \tag{13}$$

$$w = Q\,\mathrm{diag}(\lambda')Q^T. \tag{14}$$

For step $i > 1$ of convex iteration with solution $x^{(i)} = Q\,\mathrm{diag}(\lambda)Q^T$ we use a slightly different search direction.

$$\lambda'' = (0, 0, 0, 1, \cdots, 1), \tag{15}$$

$$w = Q\,\mathrm{diag}(\lambda'')Q^T. \tag{16}$$

Empirical performance in initial testing was the only guide for these decisions. Further study is certainly merited.

In the second and subsequent rounds of convex iteration is a weighted sum of the original objective and the search direction, e.g. $c^{(i)} = c/||c||_F + \alpha w/||w||_F$. Our experience suggests heavily favoring the search direction ($\alpha = 100$) leads to good solutions and faster convergence of convex iteration.

B EFFICIENT HANDLING OF INEQUALITY CONSTRAINTS

While solving a SDP representing a protein structure, many of the inequality constraints of the form of Equation 6 are satisfied strictly. That is, the distance between two coordinates is larger than the minimum required distance. This means the constraint was not necessary initially. Since the computational cost of solving a SDP is tied to the number of constraints, it can be advantageous in some cases to try to eliminate unnecessary constraints.

We use a simple active set method to increase efficiency. Initially, only equality distance constraints are used to solve the SDP. Violated inequality constraints are identified in the initial solution and added to the problem which is then resolved. Inequality constraints that are still violated are added in without removing previous additions until none of the original constraints are violated. The fraction of violated constraints added at each iteration was adjusted for each method based on a trial runs. The fractions by 0.30 for NEDM and Domo and 0.05 for Objz1 and Objz2.

Such an active set method does not always yield better computational efficiency as the sum of active set iteration times may exceed that of solving the whole original problem with all inequality constraints included at once. However, in many cases, we see a computational gain which could likely be augmented by further development of schemes that dynamically add and remove inequality constraints.

C NOTES ON CONVEX DESIGN

Without the trace constraint, the degenerate answer $c, y, z = 0$ would be valid for Program 11 which is not of much use. It is important to note that the trace constraint is not the only way to enforce nondegenerate solutions to the design problem. For example, an alternative is $c_1 = 1$ which will result in a different c being designed which will still lead to x^* as the answer to Program 4.

Using the eigenvalue decomposition of x^* it is possible to convert Program 11 from a semidefinite program to a linear program. Our implementation does this to save on computational costs. Complementary slackness (Equation 10) ensures that the z found during design will have the same eigenvectors Q as x^*. Thus, wherever z appears, it may be replaced by $Q\,\mathrm{diag}(\omega)Q^T$ and we merely have to enforce non-negativity of the eigenvalues of z by $\omega \geq 0$.