Determining Protein Structures from NOESY Distance Constraints by Semidefinite Programming
Babak Alipanahi, Nathan Krislock, Ali Ghodsi, Henry Wolkowicz, Logan Donaldson, Ming Li

To cite this version:

HAL Id: hal-00684488
https://hal.archives-ouvertes.fr/hal-00684488v2
Submitted on 12 Jul 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Determining Protein Structures from NOESY Distance Constraints by Semidefinite Programming

Babak Alipanahi\textsuperscript{1*}, Nathan Krislock\textsuperscript{2}, Ali Ghodsi\textsuperscript{3}, Henry Wolkowicz\textsuperscript{4}, Logan Donaldson\textsuperscript{5}, and Ming Li\textsuperscript{1**}

\textsuperscript{1} David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada
\textsuperscript{2} INRIA Grenoble Rhône-Alpes, France
\textsuperscript{3} Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, Ontario, Canada
\textsuperscript{4} Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada
\textsuperscript{5} Department of Biology, York University, Toronto, Ontario, Canada

Abstract. Contemporary practical methods for protein NMR structure determination use molecular dynamics coupled with a simulated annealing schedule. The objective of these methods is to minimize the error of deviating from the NOE distance constraints. However, the corresponding objective function is highly nonconvex and, consequently, difficult to optimize. Euclidean distance matrix (EDM) methods based on semidefinite programming (SDP) provide a natural framework for these problems. However, the high complexity of SDP solvers and the often noisy distance constraints provide major challenges to this approach. The main contribution of this paper is a new SDP formulation for the EDM approach that overcomes these two difficulties. We model the protein as a set of intersecting two- and three-dimensional cliques. Then, we adapt and extend a technique called semidefinite facial reduction to reduce the SDP problem size to approximately one quarter of the size of the original problem. The reduced SDP problem can be solved approximately 100 times faster, and it is also more resistant to numerical problems from erroneous and inexact distance bounds.

Key words: Molecular structural biology, nuclear magnetic resonance, semidefinite programming, facial reduction

1 Introduction

Computing three-dimensional protein structures from their amino acid sequences is one of the most widely studied problems in bioinformatics. Knowing the structure of the protein is key to understanding its physical, chemical, and biological properties. The protein nuclear magnetic resonance (NMR) method is fundamentally different from the X-ray method: It is not a “microscope with atomic resolution”; rather, it provides a network of distance measurements between spatially proximate hydrogen atoms (Güntert, 1998). As a result, the NMR method relies heavily on complex computational algorithms. The existing methods for protein NMR can be categorized into four major groups: (i) methods based on Euclidean distance matrix completion (EDMC) (Braun et al., 1981; Havel and Wüthrich, 1984; Biswas et al., 2008; Leung and Toh, 2009); (ii) methods based on molecular dynamics and simulated
annealing (Nilges et al., 1988; Brünger, 1993; Schwieters et al., 2003; Güntert et al., 1997; Güntert, 2004); (iii) methods based on local/global optimization (Braun and Go, 1985; Moré and Wu, 1997; Williams et al., 2001); and (iv) methods originating from sequence-based protein structure prediction algorithms (Shen et al., 2008; Raman et al., 2010; Alipanahi et al., 2011).

In the early years of protein NMR, many EDMC-based methods directly worked on the corresponding Euclidean distance matrix (EDM). The first method to use EDMC for protein NMR was developed by Braun et al. (Braun et al., 1981). Other notable methods include EMBED (Havel et al., 1983) and DISGEO (Havel and Wüthrich, 1984). These methods faced two major drawbacks: randomly guessing the unknown distances is ineffective and after several iterations of distance correction, the error in the distances tended to become large (Güntert, 1998); and, there was no way to control the embedding dimensionality, which also tended to get large.

A major breakthrough came that combined simulated annealing with molecular dynamics (MD) simulation. Nilges et al. made some improvements in the MD-based protein NMR structure determination (Nilges et al., 1988). Instead of an empirical energy function, they proposed a simple geometrical energy function based on the NOE restraints that penalized large violations and they also combined simulated annealing (SA) with MD. These methods were able to search the massive conformation space without being trapped in one of numerous local minima. The XPLOR method (Brünger, 1993; Schwieters et al., 2003, 2006) was one of the first successful and widely-adapted methods that was built on the molecular dynamics simulation package CHARMM (Brooks et al., 1983). The number of degrees of freedom in torsion angle space is nearly 10 times smaller than in Cartesian coordinates space, while being equivalent under mild assumptions. The torsion angle dynamics algorithm implemented in the program CYANA (Güntert, 2004), and previously in the program DYANA (Güntert et al., 1997), is one of the fastest and most widely-used methods.

### 1.1 Gram Matrix Methods

The EDMC methods are based on using the Gram matrix, or the matrix of inner products. This has many advantages. For example: (i) the Gram matrix and Euclidean distance matrix (EDM) are linearly related; (ii) instead of enforcing all of the triangle inequality constraints, it is sufficient to enforce that the Gram matrix is positive semidefinite; (iii) the embedding dimension and the rank of the Gram matrix are directly related.

Semidefinite programming (SDP) is a natural choice for formulating the EDMC problem using the Gram matrix. SDP-based EDMC methods have demonstrated great success in solving the sensor network localization (SNL) problem (Doherty et al., 2001; Biswas and Ye, 2004; Biswas et al., 2006; Wang et al., 2008; Kim et al., 2009; Krislock and Wolkowicz, 2010). In the SNL problem, the location of a set of sensors is determined based on given short-range distances between spatially proximate sensors. As a result, the SNL problem is inherently similar to the protein NMR problem. The major obstacle in extending SNL methods to protein NMR is the complexity of SDP solvers. To overcome this limitation Biswas et al. proposed DAFGAL, which is built on the idea of divide-and-stitch (Biswas et al., 2008). Leung and Toh proposed the DISCO method (Leung and Toh, 2009). It is an extension of DAFGAL that can determine protein molecules with more than 10,000 atoms using a
**divide-and-conquer** technique. The improved methods for partitioning the partial distance matrix and iteratively aligning the solutions of the subproblems, boosted the performance of DISCO in comparison to DAFGAL.

### 1.2 Contributions of the Proposed SPROS Method

Most of the existing methods make some of the following assumptions: (i) (nearly) exact distances between atoms are known; (ii) distances between any type of nuclei (not just hydrogens) are known; (iii) ignoring the fact that not all hydrogen atoms can be uniquely assigned; and (iv) overlooking the ambiguity in the NOE cross-peak assignments. In order to automate the NMR protein structure determination process, we need a robust structure calculation method that tolerates more errors. In this paper, we give a new SDP formulation that does not assume (i–iv) above. Moreover, the new method, called “SPROS” (Semidefinite Programming-based Protein structure determination), models the protein molecule as a set of intersecting two- and three-dimension cliques. We adapt and extend a technique called semidefinite facial reduction which makes the SDP problem strictly feasible and reduces its size to approximately one quarter the size of the original problem. The reduced problem is more numerically stable and can be solved nearly 100 times faster.

**Outline** We have divided the presentation of the SPROS method into providing the necessary background, followed by giving a description of techniques used for problem size reduction, and finally, showing the performance of the method on experimentally derived data.

**Preliminaries** Scalars, vectors, sets, and matrices are shown in lower case, lower case bold italic, script, and upper case italic letters, respectively. We work only on real finite-dimensional Euclidean Spaces $\mathbb{E}$ and define an inner product operator $\langle \cdot, \cdot \rangle : \mathbb{E} \times \mathbb{E} \to \mathbb{R}$ for these spaces: (i) for the space of real $p$-dimensional vectors, $\mathbb{R}^p$, $\langle \mathbf{x}, \mathbf{y} \rangle := \mathbf{x}^\top \mathbf{y} = \sum_{i=1}^{p} x_i y_i$, and (ii) for the space of real $p \times q$ matrices, $\mathbb{R}^{p \times q}$, $\langle A, B \rangle := \text{trace}(A^\top B) = \sum_{i=1}^{p} \sum_{j=1}^{q} A_{ij} B_{ij}$.

The Euclidean distance norm of $\mathbf{x} \in \mathbb{R}^p$ is defined as $\| \mathbf{x} \| := \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$. We use the MATLAB notation that $1:n := \{1, 2, \ldots, n\}$. For a matrix $A \in \mathbb{R}^{n \times n}$ and an index set $\mathcal{I} \subseteq 1:n$, $B = A[\mathcal{I}]$ is the $|\mathcal{I}| \times |\mathcal{I}|$ matrix formed by rows and columns of $A$ indexed by $\mathcal{I}$. Finally, we let $\mathcal{S}^p$ the space of symmetric $p \times p$ matrices.

### 2 The SPROS Method

#### 2.1 Euclidean Distance Geometry

**Euclidean Distance Matrix** A symmetric matrix $D$ is called a Euclidean Distance Matrix (EDM) if there exists a set of points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^r$ such that:

$$
D_{ij} = \| \mathbf{x}_i - \mathbf{x}_j \|^2, \quad \forall i, j.
$$

The smallest value of $r$ is called the embedding dimension of $D$, and is denoted $\text{embdim}(D)$. The space of all $n \times n$ EDMs is denoted $\mathcal{E}^n$. 
**The Gram Matrix** If we define $X := [x_1, \ldots, x_n] \in \mathbb{R}^{r \times n}$, then the matrix of inner-products, or *Gram Matrix*, is given by $G := X^TX$. It immediately follows that $G \in \mathcal{S}_n^+$, where $\mathcal{S}_n^+$ is the set of symmetric positive semidefinite $n \times n$ matrices. The Gram matrix and the Euclidean distance matrix are linearly related:

$$D = K(G) := \text{diag}(G) \cdot 1^T + 1 \cdot \text{diag}(G)^T - 2G,$$

where $1$ is the all-ones vector of the appropriate size, and $\text{diag}(G)$ is the vector formed from the diagonal of $G$. To go from the EDM to the Gram matrix, we use the Moore-Penrose generalized inverse $K^\dagger : \mathcal{S}^n \to \mathcal{S}^n$:

$$G = K^\dagger(D) := -\frac{1}{2}HDH, \quad D \in \mathcal{S}_H^n,$$

where $H = I - \frac{1}{n}11^T$ is the *centering* matrix, and $\mathcal{S}_H^n := \{A \in \mathcal{S}^n : \text{diag}(A) = 0\}$, is the set of symmetric matrices with zero diagonal.

**Schoenberg’s Theorem** Given a matrix $D$, we can determine if it is an EDM with the following well-known theorem (Schoenberg, 1935):

**Theorem 1.** A matrix $D \in \mathcal{S}_H^n$ is a Euclidean distance matrix if and only if $K^\dagger(D)$ is positive semidefinite. Moreover, $\text{embdim}(D) = \text{rank}(K^\dagger(D))$, for all $D \in \mathcal{E}^n$.

### 2.2 The SDP Formulation

Semidefinite optimization or, more commonly, semidefinite programming, is a class of convex optimization problems that has attracted much attention in the optimization community and has found numerous applications in different science and engineering fields. Notably, several diverse convex optimization problems can be formulated as SDP problems (Vandenberghe and Boyd, 1996). Current state-of-the-art SDP solvers are based on *primal-dual interior-point* methods.

**Preliminary Problem Formulation** There are three types of constraints in our formulation: (i) equality constraints, which are the union of equality constraints preserving bond lengths ($\mathcal{B}$), bond angles ($\mathcal{A}$), and planarity of the coplanar atoms ($\mathcal{P}$), giving $\mathcal{E} = \mathcal{E}_B \cup \mathcal{E}_A \cup \mathcal{E}_P$; (ii) upper bounds, which are the union of NOE-derived ($\mathcal{N}$), hydrogen bonds ($\mathcal{H}$), disulfide and salt bridges ($\mathcal{D}$), and torsion angle ($\mathcal{T}$) upper bounds, giving $\mathcal{U} = \mathcal{U}_N \cup \mathcal{U}_H \cup \mathcal{U}_D \cup \mathcal{U}_T$; (iii) lower bounds, which are the union of steric or van der Waals ($\mathcal{W}$) and torsion angle ($\mathcal{T}$) lower bounds, giving $\mathcal{L} = \mathcal{L}_W \cup \mathcal{L}_T$. We assume the target protein has $n$ atoms, $a_1, \ldots, a_n$. The preliminary problem formulation is given by:

$$\begin{align*}
\text{minimize} & \quad \gamma \langle I, K \rangle + \sum_{ij} w_{ij} \xi_{ij} + \sum_{ij} w'_{ij} \zeta_{ij} \\
\text{subject to} & \quad \langle A_{ij}, K \rangle = \epsilon_{ij}, \quad (i, j) \in \mathcal{E} \\
& \quad \langle A_{ij}, K \rangle \leq u_{ij} + \xi_{ij}, \quad (i, j) \in \mathcal{U} \\
& \quad \langle A_{ij}, K \rangle \geq l_{ij} - \zeta_{ij}, \quad (i, j) \in \mathcal{L} \\
& \quad \xi_{ij} \in \mathbb{R}_+, \quad (i, j) \in \mathcal{U}, \quad \zeta_{ij} \in \mathbb{R}_+, \quad (i, j) \in \mathcal{L} \\
& \quad K \mathbf{1} = \mathbf{0}, \quad K \in \mathcal{S}_+^n,
\end{align*}$$

where $\mathbf{1}$ is the all-ones vector of the appropriate size, and $\mathbf{0}$ is the all-zeros vector of the appropriate size.
where \( A_{ij} = (e_i - e_j)(e_i - e_j)^\top \) and \( e_i \) is the \( i \)th column of the identity matrix. The centering constraint \( K1 = 0 \), ensures that the embedding of \( K \) is centered at the origin. Since both upper bounds and lower bounds may be inaccurate and noisy, non-negative penalized slacks, \( \zeta_{ij} \)'s and \( \xi_{ij} \)'s, are included to prevent infeasibility and manage ambiguous upper bounds. The heuristic rank reduction term, \( \gamma(I, K) \), with \( \gamma < 0 \), in the objective function, produces lower-rank solutions (Weinberger and Saul, 2004).

### Bond lengths and angles
Covalent bonds are very stable, and since their fluctuations cannot be detected in NMR experiments, all bond lengths and angles must be set to ideal values computed from accurate X-ray structures; see (Engh and Huber, 1991). Bonds length and angle constraints are written in terms of the distance between an atom and its immediate neighbor and an atom and its second nearest neighbor, respectively.

### Planarity constraints
Proteins contain several coplanar atoms, from HCON in the peptide planes, and from side chain in moieties found in nine amino acids (Hooft et al., 1996). We have enforced planarity by preserving the distances between all coplanar atoms.

### Torsion angle constraints
Another source of structural information in protein NMR is the set of torsion angle restraints, defined as \( \theta_{i}^{\text{min}} \leq \theta_{i} \leq \theta_{i}^{\text{max}}, \ i \in T \). We extend the idea proposed in (Sussman, 1985) and define upper and lower bounds on the torsion angles based on the distance between the first and the fourth atom in the torsion angle. Thus, for example, we can constrain the \( \Phi_i \) angle by constraining the distance between the \( C_i \) and \( C_{i-1} \) atoms.

### Penalizing incorrect bounds
Let \( \xi \in \mathbb{R}^{|U|} \) be a vector containing all of the slacks for upper bounds. Since \( \xi_{ij} \in \mathbb{R}_+ \), assuming that all the weights are the same, i.e., \( w_{ij} = w \), we have \( w \sum_{ij} \xi_{ij} = w \| \xi \|_1 \), where \( \| \xi \|_1 \) is the \( \ell_1 \)-norm of vector \( \xi \). The fact that minimizing the \( \ell_1 \)-norm finds sparse solutions is a widely known and used heuristic (Boyd and Vandenberghe, 2004). In our problem, \( \xi_{ij} = 0 \) implies no violation; consequently, SPROS tends to find a solution that violates a minimum number of upper bounds.

### Pseudo-atoms
Not all hydrogens can be uniquely assigned, such as the hydrogens in the methyl groups; therefore, upper bounds involving these hydrogens are ambiguous. To overcome this problem, pseudo-atoms are introduced (Güntert, 1998). Given an ambiguous constraint between one of the hydrogens and atom \( A \), by using the triangle inequality, we modify the constraint as follows:

\[
\| \text{HB}_i - A \| \leq b, \ i \in \{1, 2, 3\} \quad \Rightarrow \quad \| \text{QB} - A \| \leq b + \| \text{HB}_i - \text{QB} \|, \tag{5}
\]

where \( \| \text{HB}_i - \text{QB} \| \) is the same for \( i = 1, 2, 3 \). Pseudo-atoms are named corresponding to the hydrogens they represent; only H is changed to Q and the rightmost number is dropped. For example, in leucine, QD1 represents HD11, HD12, and HD13. We adapt the pseudo-atoms used in CYANA (Güntert, 2004).

### Side chain simplification
In CYANA, hydrogens that do not participate directly in the structural solution are discarded initially and then added at later stages (Güntert, 2004). We have adapted this approach by discarding hydrogens only if they make our problem smaller. In the side chain simplification process, we temporarily discard (i) all of the methyl hydrogens, (ii) all of the methylene hydrogens, (iii) hydroxyl hydrogens of tyrosine and serine, (iv) amino hydrogens of arginine and threonine, and (v) sulfhydryl hydrogen of cysteine. After the SDP problem is solved, the omitted hydrogen atoms are replaced and remain in all post-processing stages.
Challenges in Solving the SDP Problem Solving the optimization problem in (4) can be challenging: For small to medium sized proteins, the number of atoms, \( n \), is 1,000-3,500, and current primal-dual interior-point SDP solvers cannot solve problems with \( n > 2,000 \) efficiently. Moreover, the optimization problem in (4) does not satisfy strict feasibility, causing numerical problems; see (Wei and Wolkowicz, 2010).

It can be observed that the protein contains many small intersecting cliques. For example, peptide planes or aromatic rings, are 2D cliques, and tetrahedral carbons form 3D cliques. As we show later, whenever there is a clique in the protein, the corresponding Gram matrix, \( K \), can never be full-rank, which violates strict feasibility. By adapting and extending a technique called semidefinite facial reduction, not only do we obtain an equivalent problem that satisfies strict feasibility, but we also significantly reduce the SDP problem size.

2.3 Cliques in a Protein Molecule

A protein molecule with \( \ell \) amino acid residues has \( \ell + 1 \) planes in its backbone. Moreover, each amino acid has a different side chain with a different structure; therefore, the number of cliques in each side chain varies (see Table 4 in Appendix B for the number of cliques in each amino acid side chain). We assume that the \( i \)th residue, \( r_i \), has \( s_i \) cliques in its side chain, denoted by \( S_i^{(1)}, \ldots, S_i^{(s_i)} \). For all amino acids (except glycine and proline), the first side chain clique is formed around the tetrahedral carbon CA, \( S_i^{(1)} = \{N_i, CA_i, HA_i, CB_i, C_i\} \), which intersects with two peptide planes \( P_{i-1} \) and \( P_i \) in two atoms: \( S_i^{(1)} \cap P_{i-1} = \{N_i, CA_i\} \) and \( S_i^{(1)} \cap P_i = \{CA_i, C_i\} \). Side chain cliques for all twenty amino acids are listed in Table 4 (see Appendix B). There is a total of \( q = \ell + 1 + \sum_{i=1}^{\ell} s_i \) cliques in the distance matrix of any protein. To simplify, let \( C_i = P_{i-1}, 1 \leq i \leq \ell + 1 \), and \( C_{\ell+2} = S_1^{(1)}, C_{\ell+3} = S_1^{(2)} \), \ldots, \( C_q = S_1^{(s_q)} \). For properties of the cliques in the protein molecule, see Appendix A. See Figure 1 for an example with the Glycine amino acid as the \( i \)-th residue together with its neighboring backbone planes.

2.4 Algorithm for Finding the Face of the Structure

For \( t < n \) and \( U \in \mathbb{R}^{n \times t} \), the set of matrices \( US_i^+ U^T \) is a face of \( S_+^n \) (in fact every face of \( S_+^n \) can be described in this way); see, e.g., (Ramana et al., 1997). We let \( \text{face}(F) \) represent the smallest face containing a subset \( F \) of \( S_+^n \); then we have the important property that \( \text{face}(F) = US_i^+ U^T \) if and only if there exists \( Z \in S_i^+ \) such that \( UZU^T \in F \). Furthermore, in this case, we have that every \( Y \in F \) can be decomposed as \( Y = UZU^T \), for some \( Z \in S_i^+ \), and the reduced feasible set \( \{Z \in S_i^+ : UZU^T \in F\} \) has a strictly feasible point, giving us a problem that is more numerically stable to solve (problems that are not strictly feasible have a dual optimal set that is unbounded and therefore can be difficult to solve numerically; for more information, see (Wei and Wolkowicz, 2010)). Moreover, if \( t \ll n \), this results in a significant reduction in the matrix size.

The Face of a Single Clique Here, we solve the SINGLE CLIQUE problem, which is defined as follows: Let \( D \) be a partial EDM of a protein. Suppose the first \( n_1 \) points form a clique in the protein, such that for \( C_1 = \{1, \ldots, n_1\} \), all distances are known. That is, the matrix \( D_1 = D[C_1] \) is completely specified. Moreover, let \( r_1 = \text{embdim}(D_1) \). We now show how to compute the smallest face containing the feasible set \( \{K \in S_+^n : K(K[C_1]) = D_1\} \).
Fig. 1. The Glycine amino acid as the \( i \)-th residue with its neighboring backbone planes.

**Theorem 2 (Single Clique, (Krislock and Wolkowicz, 2010)).** Let the matrix \( U_1 \in \mathbb{R}^{n \times (n-n_1+r_1+1)} \) be defined as follows:

- let \( V_1 \in \mathbb{R}^{n_1 \times r_1} \) be a full column rank matrix such that \( \text{range}(V_1) = \text{range}(K^\dagger (D_1)) \);
- let \( \bar{U}_1 := [V_1 \, 1] \) and \( U_1 := \begin{bmatrix} \bar{U}_1 & 0 \\ 0 & I \end{bmatrix} \in \mathbb{R}^{n \times (n-n_1+r_1+1)} \).

Then \( U_1 \) has full column rank, \( 1 \in \text{range}(U) \), and

\[
\text{face}\{K \in S^n_+: K[K[C_1]] = D[C_1]\} = U_1 S^{n-n_1+r_1+1}_+ U_1^\top.
\]

**Computing the \( V_1 \) Matrix** In Theorem 2, we can find \( V_1 \) by computing the eigendecomposition of \( K^\dagger (D[C_1]) \) as follows:

\[
K^\dagger (D[C_1]) = V_1 A_1 V_1^\top, \quad V_1 \in \mathbb{R}^{n_1 \times r_1}, \quad A_1 \in S^{r_1}_+.
\]

It can be seen that \( V_1 \) has full column rank (columns are orthonormal) and also that \( \text{range}(V_1) = \text{range}(K^\dagger (D_1)) \).

**2.5 The Face of a Protein Molecule**

The protein molecule is made of \( q \) cliques, \( \{C_1, \ldots, C_q\} \), such that \( D[C_i] \) is known, and we have \( r_l = \text{embdim}(D[C_i]) \), and \( n_l = |C_l| \). Let \( F \) be the feasible set of the SDP problem.
for each clique $C_l$, we define $F_l := \{K \in S^n_+ : K(K[C_l]) = D[C_l]\}$, then
\[
F \subseteq \left( \bigcap_{l=1}^{q} F_l \right) \cap S^n_C,
\]
where $S^n_C := \{K \in S^n : K1 = 0\}$ are the centered symmetric matrices. For $l = 1, \ldots, q$, let $F_l := \text{face}(F_l) = U_l S^{n-n_l+r_l+1}_+ U_l^T$, where $U_l$ is computed as in Theorem 2. We have (Krislock and Wolkowicz, 2010):
\[
\left( \bigcap_{l=1}^{q} F_l \right) \cap S^n_C \subseteq \left( \bigcap_{l=1}^{q} U_l S^{n-n_l+r_l+1}_+ U_l^T \right) \cap S^n_C,
\]
(8)
where $U \in \mathbb{R}^{n \times k}$ is a full column rank matrix that satisfies $\text{range}(U) = \bigcap_{l=1}^{q} \text{range}(U_l)$.

We now have an efficient method for computing the face of the feasible set $F$. To have better numerical accuracy, we developed a bottom-up algorithm for intersecting subspaces (see Algorithm 1 in Appendix C).

After computing $U$, we can decompose the Gram matrix as $K = UZU^T$, for $Z \in S^k_+$. However, by exploiting the centering constraint, $K1 = 0$, we can reduce the matrix size one more. If $V \in \mathbb{R}^{k \times (k-1)}$ has full column rank and satisfies $\text{range}(V) = \text{null}(1^T U)$, then we have (Krislock and Wolkowicz, 2010):
\[
F \subseteq (UV) S^{k-1}_+(UV)^T.
\]
(9)

For more details on facial reduction for Euclidean distance matrix completion problems, see (Krislock, 2010).

**Constraints for Preserving the Structure of Cliques** If we find a base set of points $B_l$ in each clique $C_l$ such that $\text{embdim}(D[B_l]) = r_l$, then by fixing the distances between points in the base set and fixing the distances between points in $C_l \setminus B_l$ and points in $B_l$, the entire clique is kept rigid. Therefore, we need to fix only the distances between base points (Alipanahi et al., 2012), resulting in a three- to four-fold reduction in the number of equality constraints. We call the reduced set of equality constraints $E_{FR}$.

### 2.6 Solving and Refining the Reduced SDP Problem

The SPROS method flowchart is depicted in Appendix D (see Fig. 2). In it, we describe the blocks for solving the SDP problem and for refining the solution. From equation (9), we can formulate the reduced SDP problem as follows:

\[
\begin{align*}
\text{minimize} & \quad \gamma \langle I, Z \rangle + \sum_{i,j} w_{ij} \xi_{ij} + \sum_{i,j} w'_{ij} \zeta_{ij} \\
\text{subject to} & \quad \langle A'_{ij}, Z \rangle = e_{ij}, \quad (i,j) \in E_{FR} \\
& \quad \langle A'_{ij}, Z \rangle \leq w_{ij} + \xi_{ij}, \quad (i,j) \in U \\
& \quad \langle A'_{ij}, Z \rangle \geq l_{ij} - \zeta_{ij}, \quad (i,j) \in L \\
& \quad \xi_{ij} \in \mathbb{R}_+, \quad (i,j) \in U, \quad \zeta_{ij} \in \mathbb{R}_+, \quad (i,j) \in L \\
& \quad Z \in S^{k-1}_+,
\end{align*}
\]
(10)

where $A'_{ij} = (UV)^T A_{ij}(UV)$. 
Weights and the regularization parameter

For each type of upper and lower bound, we define a fixed penalizing weight for violations. For example, for upper bounds (similarly for lower bounds) we have $\forall (i, j) \in U_X, w_{ij} = w_X$. We set $w_N = 1$ and $w_H = w_D = w_T = 10$ because upper bounds from hydrogen bonds and disulfide/salt bridges are assumed to be more accurate than are NOE-derived upper bounds. Moreover, the range of torsion angle violations is ten times smaller than NOE violations.

Let $m_U = |U|$ and $R$ be the radius of the protein. Then, the maximum upper bound violation is $2R$. Moreover, $\langle I, Z \rangle \leq nR$. Discarding the role of lower bound violations, with the goal of approximately balancing the two terms, a suitable $\gamma$ is:

$$\gamma nR \approx 2\varepsilon w m_U R \Rightarrow \gamma = \frac{2\varepsilon w m_U}{n},$$

where $0 \leq \varepsilon \leq 1$ is the fraction of violated upper bounds. In practice $\varepsilon \approx 0.01 - 0.30$, and $\bar{\gamma} \approx w m_U / 50n$ works well.

Post-Processing

We perform a refinement on the raw structure determined by the SDP solver. For this refinement we use a BFGS-based quasi-Newton method (Lewis and Overton, 2009) that only requires the value of the objective function and its gradient at each point. Letting $X^{(0)} = X_{SDP}$, we iteratively minimize the following objective function:

$$\phi(X) = w_E \sum_{(i,j) \in E} (\|x_i - x_j\| - e_{ij})^2 + w_U \sum_{(i,j) \in U} f(\|x_i - x_j\| - u_{ij})^2$$

$$+ w_C \sum_{(i,j) \in L} g(\|x_i - x_j\| - l_{ij})^2 + w_R \sum_{i=1}^{n} \|x_i\|^2,$$

where $f(\theta) = \max(0, \theta)$ and $g(\theta) = \min(0, -\theta)$. We set $w_E = 2$, $w_U = 1$, and $w_C = 1$. In addition, to balance the regularization term, we set

$$w_R = \alpha \frac{\phi(X^{(0)})|_{w_R=0}}{25 \sum_{i=1}^{n} \|x_i^{(0)}\|^2},$$

where $-1 \leq \alpha \leq 1$ is a parameter controlling the regularization. If $\alpha < 0$, the distances between atoms are maximized, because, after projection, some of the distances have been shortened, this term helps to compensate for that error. However, if $\alpha > 0$, the distances between atoms are minimized, resulting in better packing of atoms in the protein molecule. In practice, different values for $\alpha$ can be used to generate slightly different structures, thus creating a bundle of structures.

Fixing incorrect chiralities

Chirality constraints cannot be enforced using only distances. Consequently, some chiral centers may have the incorrect enantiomer. In this step, SPROS checks the chiral centers and resolves any problems.

Improving the stereochemical quality

Williamson and Craven have described the effectiveness of explicit solvent refinement of NMR structures and suggest that it should be a standard procedure (Williamson and Craven, 2009). For protein structures that have regions of high mobility/uncertainty due to few or no NOE observations, we have successfully employed a hybrid protocol from XPLOR-NIH that incorporates thin-layer water refinement (Linge et al., 2003) and a multidimensional torsion angle database (Kuszewski et al., 1996, 1997).
3 Results

We tested the performance of SPROS on 18 proteins: 15 protein data sets from the DOCR database in the NMR Restraints Grid (Doreleijers et al., 2003, 2005) and three protein data sets from Donaldson’s laboratory at York University. We chose proteins with different sizes and topologies, as listed in Table 1. Finally, the input to the SPROS method is exactly the same as the input to the widely-used CYANA method.

3.1 Implementation

The SPROS method has been implemented and tested in MATLAB 7.13 (apart from the water refinement, which is done by XPLOR-NIH). For solving the SDP problem, we used the SDPT3 method (Tütüncü et al., 2003). For minimizing the post-processing objective function (12), we used the BFGS-based quasi-Newton method implementation by Lewis and Overton (Lewis and Overton, 2009). All the experiments were carried out on an Ubuntu 11.04 Linux PC with a 2.8 GHz Intel Core i7 Quad-Core processor and 8 GB of memory.

3.2 Determined Structures

From the 18 test proteins, 9 of them were calculated with backbone RMSDs less than or equal to 1.0 Å, and 16 have backbone RMSDs less than 1.5 Å. Detailed analysis of calculated structures is listed in Table 2. The superimposition of the SPROS and reference structures for three of the proteins are depicted in Fig. 1. More detailed information about the determined structures can be found in (Alipanahi, 2011).

To further assess the performance of SPROS, we compared the SPROS and reference structures for 1G6J, Ubiquitin, and 2GJY, PTB domain of Tensin, with their corresponding X-ray structures, 1UBQ and 1WVH, respectively. For 1G6J, the backbone (heavy atoms) RMSDs for SPROS and the reference structures are 0.42 Å (0.57 Å) and 0.73±0.04 Å (0.98±0.04 Å), respectively. For 2GJY, the backbone (heavy atoms) RMSDs for SPROS and the reference structures are 0.88 Å (1.15 Å) and 0.89 ± 0.08 Å (1.21 ± 0.06 Å), respectively.

3.3 Discussion

The SPROS method was tested on 18 experimentally derived protein NMR data sets of sequence lengths ranging from 76 to 307 (weights ranging from 8 to 35 KDa). Calculation
<table>
<thead>
<tr>
<th>ID</th>
<th>topo.</th>
<th>len.</th>
<th>weight</th>
<th>n</th>
<th>n'</th>
<th>cliques (2D/3D)</th>
<th>m_E</th>
<th>m'_E</th>
<th>m_U</th>
<th>bound types</th>
<th>m_U ± s_U</th>
<th>m_N</th>
<th>p_U</th>
<th>m_T</th>
<th>m_H</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G6J</td>
<td>a+b</td>
<td>76</td>
<td>8.58</td>
<td>1434</td>
<td>405</td>
<td>304 (201/103)</td>
<td>5543</td>
<td>1167</td>
<td>1354</td>
<td>21/ 29/ 17/ 33</td>
<td>31.9±15.3</td>
<td>1291</td>
<td>32</td>
<td>63</td>
<td>0</td>
</tr>
<tr>
<td>1B4R</td>
<td>B</td>
<td>80</td>
<td>7.96</td>
<td>1281</td>
<td>346</td>
<td>248 (145/103)</td>
<td>4887</td>
<td>1027</td>
<td>787</td>
<td>26/ 25/ 6/ 43</td>
<td>17.1±10.8</td>
<td>687</td>
<td>30</td>
<td>22</td>
<td>78</td>
</tr>
<tr>
<td>2E80</td>
<td>A</td>
<td>103</td>
<td>11.40</td>
<td>1523</td>
<td>419</td>
<td>317 (212/105)</td>
<td>5846</td>
<td>1214</td>
<td>3157</td>
<td>19/ 29/ 26/ 26</td>
<td>71.4±35.4</td>
<td>3070</td>
<td>24</td>
<td>87</td>
<td>0</td>
</tr>
<tr>
<td>1C7N</td>
<td>a/b</td>
<td>104</td>
<td>11.30</td>
<td>1927</td>
<td>532</td>
<td>393 (253/140)</td>
<td>7399</td>
<td>1540</td>
<td>1560</td>
<td>46/ 24/ 12/ 18</td>
<td>23.1±13.4</td>
<td>1418</td>
<td>31</td>
<td>80</td>
<td>62</td>
</tr>
<tr>
<td>2KTS</td>
<td>a/b</td>
<td>117</td>
<td>12.85</td>
<td>2075</td>
<td>593</td>
<td>448 (299/140)</td>
<td>7968</td>
<td>1719</td>
<td>2279</td>
<td>22/ 28/ 14/ 36</td>
<td>34.6±17.4</td>
<td>2276</td>
<td>25</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2K49</td>
<td>a+b</td>
<td>118</td>
<td>13.10</td>
<td>2017</td>
<td>574</td>
<td>433 (291/142)</td>
<td>7710</td>
<td>1657</td>
<td>2612</td>
<td>22/ 27/ 18/ 38</td>
<td>40.9±21.1</td>
<td>2374</td>
<td>27</td>
<td>146</td>
<td>92</td>
</tr>
<tr>
<td>2K62</td>
<td>B</td>
<td>125</td>
<td>15.10</td>
<td>2328</td>
<td>655</td>
<td>492 (327/165)</td>
<td>8943</td>
<td>1886</td>
<td>2367</td>
<td>21/ 32/ 15/ 32</td>
<td>33.9±18.6</td>
<td>2187</td>
<td>32</td>
<td>180</td>
<td>0</td>
</tr>
<tr>
<td>2L30</td>
<td>A</td>
<td>127</td>
<td>14.30</td>
<td>1867</td>
<td>512</td>
<td>393 (269/124)</td>
<td>7143</td>
<td>1492</td>
<td>1270</td>
<td>24/ 38/ 20/ 18</td>
<td>22.5±12.7</td>
<td>1055</td>
<td>25</td>
<td>156</td>
<td>59</td>
</tr>
<tr>
<td>2GJY</td>
<td>a+b</td>
<td>144</td>
<td>15.67</td>
<td>2337</td>
<td>639</td>
<td>474 (302/172)</td>
<td>8919</td>
<td>1875</td>
<td>1710</td>
<td>7/ 30/ 19/ 44</td>
<td>25.0±16.6</td>
<td>1536</td>
<td>29</td>
<td>98</td>
<td>76</td>
</tr>
<tr>
<td>2KTE</td>
<td>a/b</td>
<td>152</td>
<td>17.21</td>
<td>2576</td>
<td>717</td>
<td>542 (360/182)</td>
<td>9861</td>
<td>2089</td>
<td>1899</td>
<td>17/ 31/ 22/ 30</td>
<td>24.3±20.8</td>
<td>1669</td>
<td>30</td>
<td>124</td>
<td>106</td>
</tr>
<tr>
<td>1XPW</td>
<td>B</td>
<td>153</td>
<td>17.44</td>
<td>2578</td>
<td>723</td>
<td>541 (355/186)</td>
<td>9837</td>
<td>2081</td>
<td>1206</td>
<td>0/ 31/ 11/ 58</td>
<td>17.0±10.8</td>
<td>934</td>
<td>37</td>
<td>210</td>
<td>62</td>
</tr>
<tr>
<td>2K7H</td>
<td>a/b</td>
<td>157</td>
<td>16.66</td>
<td>2710</td>
<td>756</td>
<td>563 (363/200)</td>
<td>10452</td>
<td>2196</td>
<td>2768</td>
<td>29/ 33/ 13/ 25</td>
<td>30.3±11.3</td>
<td>2481</td>
<td>19</td>
<td>239</td>
<td>48</td>
</tr>
<tr>
<td>2KP</td>
<td>A</td>
<td>165</td>
<td>17.28</td>
<td>2533</td>
<td>722</td>
<td>535 (344/191)</td>
<td>9703</td>
<td>2094</td>
<td>2504</td>
<td>31/ 26/ 23/ 20</td>
<td>59.2±25.0</td>
<td>4972</td>
<td>22</td>
<td>232</td>
<td>0</td>
</tr>
<tr>
<td>2YT0</td>
<td>a+b</td>
<td>176</td>
<td>19.17</td>
<td>2940</td>
<td>828</td>
<td>627 (419/208)</td>
<td>11210</td>
<td>2404</td>
<td>3357</td>
<td>23/ 14/ 35</td>
<td>34.9±22.3</td>
<td>3237</td>
<td>30</td>
<td>120</td>
<td>0</td>
</tr>
<tr>
<td>2L7B</td>
<td>A</td>
<td>307</td>
<td>35.30</td>
<td>5603</td>
<td>1567</td>
<td>1205 (836/369)</td>
<td>21421</td>
<td>4521</td>
<td>4355</td>
<td>10/ 30/ 16</td>
<td>27.6±14.4</td>
<td>3459</td>
<td>23</td>
<td>408</td>
<td>488</td>
</tr>
<tr>
<td>1Z1V</td>
<td>A</td>
<td>80</td>
<td>9.31</td>
<td>1259</td>
<td>362</td>
<td>272 (181/91)</td>
<td>4836</td>
<td>1046</td>
<td>1261</td>
<td>46/ 24/ 18/ 13</td>
<td>28.6±16.3</td>
<td>1189</td>
<td>15</td>
<td>0</td>
<td>72</td>
</tr>
<tr>
<td>HACS1</td>
<td>B</td>
<td>87</td>
<td>9.63</td>
<td>1150</td>
<td>315</td>
<td>237 (156/81)</td>
<td>4401</td>
<td>923</td>
<td>828</td>
<td>46/ 21/ 5/ 27</td>
<td>20.2±14.2</td>
<td>828</td>
<td>20</td>
<td>0</td>
<td>36</td>
</tr>
<tr>
<td>2LJG</td>
<td>a+b</td>
<td>153</td>
<td>17.03</td>
<td>2343</td>
<td>662</td>
<td>495 (327/168)</td>
<td>9009</td>
<td>1909</td>
<td>1347</td>
<td>40/ 29/ 8/ 22</td>
<td>16.4±11.9</td>
<td>1065</td>
<td>28</td>
<td>204</td>
<td>78</td>
</tr>
</tbody>
</table>

Table 1. Information about the proteins used in testing SPROS. The second, third, and fourth columns, list the topologies, sequence lengths, and molecular weight of the proteins, the fifth and sixth columns, n and n', list the original and reduced SDP matrix sizes, respectively. The seventh column lists the number of cliques in the protein. The eights and ninth columns, m_E and m'_E, list the number of equality constraints in the original and reduced problems, respectively. The 10th column, m_U, lists the total number of upper bounds for each protein. The 11th column, bound types, lists intra-residue, |i − j| = 0, sequential, |i − j| = 1, medium range, 1 < |i − j| ≤ 4, and long range, |i − j| > 4, respectively, in percentile. The 12th column, m_U ± s_U, lists the average number of upper bounds per residue, together with the standard deviation. The 13th column, m_N, lists the number of NOE-inferred upper bounds. The 14th column, p_U, lists the fraction of pseudo-atoms in the upper bounds in percentile. The last two columns, m_T and m_H, list the number of upper bounds inferred from torsion angle restraints, and hydrogen bonds, disulfide and salt bridges, respectively.
<table>
<thead>
<tr>
<th>ID</th>
<th>t</th>
<th>s</th>
<th>t</th>
<th>backbone heavy atoms</th>
<th>CBd.</th>
<th>0.1˚A</th>
<th>0.1˚A violations</th>
<th>RMDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G6J</td>
<td>44.5</td>
<td>175.5</td>
<td>241.0</td>
<td>0.68 ± 0.05</td>
<td>0.90 ± 0.05</td>
<td>0</td>
<td>4.96 (0.08 ± 0.07)</td>
<td>0.85 (0)</td>
</tr>
<tr>
<td>1B4R</td>
<td>21.4</td>
<td>138.0</td>
<td>179.0</td>
<td>0.85 ± 0.06</td>
<td>1.06 ± 0.06</td>
<td>0</td>
<td>20.92 (13.87 ± 0.62)</td>
<td>6.14 (2.28 ± 0.21)</td>
</tr>
<tr>
<td>2E8O</td>
<td>129.8</td>
<td>181.3</td>
<td>340.9</td>
<td>0.58 ± 0.02</td>
<td>0.68 ± 0.01</td>
<td>0</td>
<td>31.33 (31.93 ± 0.14)</td>
<td>9.98 (10.75 ± 0.13)</td>
</tr>
<tr>
<td>1CN7</td>
<td>75.0</td>
<td>230.1</td>
<td>339.7</td>
<td>1.53 ± 0.11</td>
<td>1.80 ± 0.10</td>
<td>0</td>
<td>10.27 (7.63 ± 0.80)</td>
<td>3.18 (2.11 ± 0.52)</td>
</tr>
<tr>
<td>2KTS</td>
<td>116.7</td>
<td>231.0</td>
<td>398.5</td>
<td>0.92 ± 0.06</td>
<td>1.13 ± 0.06</td>
<td>0</td>
<td>25.36 (27.44 ± 0.58)</td>
<td>6.49 (10.36 ± 0.68)</td>
</tr>
<tr>
<td>2K49</td>
<td>140.7</td>
<td>240.7</td>
<td>422.7</td>
<td>0.99 ± 0.14</td>
<td>1.24 ± 0.16</td>
<td>0</td>
<td>13.75 (15.79 ± 0.67)</td>
<td>2.80 (4.94 ± 0.46)</td>
</tr>
<tr>
<td>2K62</td>
<td>156.1</td>
<td>259.0</td>
<td>464.2</td>
<td>1.40 ± 0.08</td>
<td>1.72 ± 0.08</td>
<td>1</td>
<td>33.74 (42.92 ± 0.95)</td>
<td>10.79 (21.20 ± 1.20)</td>
</tr>
<tr>
<td>2GJY</td>
<td>113.7</td>
<td>285.9</td>
<td>455.7</td>
<td>0.99 ± 0.07</td>
<td>1.29 ± 0.09</td>
<td>0</td>
<td>11.67 (8.36 ± 0.59)</td>
<td>0.36 (0.49 ± 0.12)</td>
</tr>
<tr>
<td>2KTE</td>
<td>139.9</td>
<td>297.7</td>
<td>503.2</td>
<td>1.39 ± 0.17</td>
<td>1.85 ± 0.16</td>
<td>1</td>
<td>35.55 (31.97 ± 0.46)</td>
<td>11.94 (11.96 ± 0.40)</td>
</tr>
<tr>
<td>1XPW</td>
<td>124.8</td>
<td>297.1</td>
<td>489.7</td>
<td>1.30 ± 0.10</td>
<td>1.68 ± 0.10</td>
<td>0</td>
<td>9.74 (0.17 ± 0.09)</td>
<td>1.20 (0.01 ± 0.02)</td>
</tr>
<tr>
<td>2K7H</td>
<td>211.7</td>
<td>312.0</td>
<td>591.0</td>
<td>1.24 ± 0.07</td>
<td>1.49 ± 0.07</td>
<td>0</td>
<td>17.60 (16.45 ± 0.30)</td>
<td>4.39 (4.92 ± 0.35)</td>
</tr>
<tr>
<td>2KVP</td>
<td>462.0</td>
<td>282.4</td>
<td>814.8</td>
<td>0.94 ± 0.08</td>
<td>1.05 ± 0.09</td>
<td>0</td>
<td>15.15 (17.43 ± 0.29)</td>
<td>4.01 (5.62 ± 0.21)</td>
</tr>
<tr>
<td>2YT0</td>
<td>292.1</td>
<td>421.5</td>
<td>800.1</td>
<td>0.79 ± 0.05</td>
<td>1.04 ± 0.06</td>
<td>1</td>
<td>29.04 (28.9 ± 0.36)</td>
<td>6.64 (6.60 ± 0.30)</td>
</tr>
<tr>
<td>2L7B</td>
<td>1101.1</td>
<td>593.0</td>
<td>1992.1</td>
<td>2.15 ± 0.11</td>
<td>2.55 ± 0.11</td>
<td>3</td>
<td>19.15 (21.72 ± 0.36)</td>
<td>4.23 (4.73 ± 0.23)</td>
</tr>
<tr>
<td>1Z1V</td>
<td>30.6</td>
<td>158.8</td>
<td>209.2</td>
<td>1.44 ± 0.17</td>
<td>1.74 ± 0.15</td>
<td>0</td>
<td>3.89 (2.00 ± 0.25)</td>
<td>0.62 (0)</td>
</tr>
<tr>
<td>HACS1</td>
<td>17.4</td>
<td>145.0</td>
<td>176.1</td>
<td>1.00 ± 0.07</td>
<td>1.39 ± 0.10</td>
<td>0</td>
<td>20.29 (15.68 ± 0.43)</td>
<td>4.95 (3.73 ± 0.33)</td>
</tr>
<tr>
<td>2L1V</td>
<td>212.0</td>
<td>190.0</td>
<td>403.6</td>
<td>1.63 ± 0.11</td>
<td>1.92 ± 0.11</td>
<td>1</td>
<td>5.56 (5.35 ± 0.34)</td>
<td>0.60 (0.10 ± 0.07)</td>
</tr>
<tr>
<td>1LJE</td>
<td>178.3</td>
<td>202.7</td>
<td>271.4</td>
<td>1.07 ± 0.07</td>
<td>1.36 ± 0.08</td>
<td>0</td>
<td>21.65 (21.72 ± 0.03)</td>
<td>4.28 (4.61 ± 0.24)</td>
</tr>
</tbody>
</table>

**Table 2:** Information about determined structures of the test proteins. The second, third, and fourth columns list SDP time, water refinement time, and total time, respectively. For the backbone and heavy atoms columns, the mean and standard deviation between the determined structure and the reference structures is reported (backbone RMSDs less than 1.5˚A are shown in bold). The seventh column, CBd., lists the number of residues with "CB deviations" larger than 0.25˚A, as defined by MolProbity. The eighth and ninth columns list the percentage of upper bound violations larger than 0.1˚A and 1.0˚A, respectively. The last three columns list the percentage of residues with favorable and allowed backbone torsion angles and outliers, respectively.
times were in the order of a few minutes per structure. Accurate results were obtained for all of the data sets, although with some variability in precision. The best attribute of the SPROS method is its tolerance for, and efficiency at, managing many incorrect distance constraints (that are typically defined as upper bounds).

The reduction methodology developed for SPROS is an ideal choice for protein-ligand docking. If the side chains participating at the interaction surface are only declared to be flexible, it has the effect of reducing the SDP matrix size to less than 100. Calculations under these specific parameters can be achieved in a few seconds thereby making SPROS a worthwhile choice for automated, high-throughput screening.

Our final goal is a fully automated system for NMR protein structure determination, from peak picking (Alipanahi et al., 2009) to resonance assignment (Alipanahi et al., 2011), to protein structure determination. An automated system, without the laborious human intervention will have to tolerate more errors than usual. This was the initial motivation of designing SPROS. The key is to tolerate more errors. Thus, we are working towards incorporating an adaptive violation weight mechanism to identify the most significant outliers in the set of distance restraints automatically.

Acknowledgments

This work was supported in part by NSERC Grant OGP0046506, NSERC grant RGPIN3274-87-09, NSERC grant 238934, an NSERC Discovery Accelerator Award, Canada Research Chair program, an NSERC Collaborative Grant, David R. Cheriton Graduate Scholarship, OCRiT, Premier’s Discovery Award, and Killam Prize.


Appendix A: Properties of Cliques

Let $C_i = P_{i-1}$, $1 \leq i \leq \ell + 1$, and $C_{\ell+2} = S^{(1)}_1$, $C_{\ell+2} = S^{(2)}_1$, \ldots, $C_q = S^{(s_{\ell})}_\ell$. Let $r_i = \text{embdim}(D[\mathcal{C}_i])$. The following properties hold for cliques in a protein molecule:

1. $P_i \cap P_{i'} = \emptyset$, given $|i - i'| > 1$.
2. $P_i \cap S^{(j)}_i = \emptyset$, given $i' \neq i, i + 1$.
3. $S^{(j)}_i \cap S^{(j')}_{i'} = \emptyset$, given $i' \neq i$.
4. $|C_i| \geq r_i + 1$.
5. $3 \leq |C_i| \leq 16$.
6. $\forall i, i', |C_i \cap C_{i'}| \leq 2$.
7. $\exists i$ such that $\forall i' \neq i$, $C_i \cap C_{i'} = \emptyset$.
8. If $\mathcal{I}_i = \{i' : C_i \cap C_{i'} \neq \emptyset\}$, then $\forall i, |\mathcal{I}_i| \leq 4$.
9. $\bigcup_{i=1}^{q} C_i = 1:n$. 
Appendix B: Additional Tables

Table 3. Table summarizing properties of different amino acids: $p$ denotes abundance of amino acids in percentile, $t$ denotes the number of torsion angles (excluding $\omega$), $a$ denotes the total number of atoms and pseudo-atoms, $s$ denotes the total number of atoms and pseudo-atoms in the side chains, $q$ denotes the number of cliques in each side chain (the number in the parenthesis is the number of 3D cliques), and $k$ denotes the increase in the SDP matrix size. The values in the Reduced column denote the same values in the side chain simplified case. The weighted average ($w.a.$) of quantity $x$ is computed as $\sum_{i \in A} p_i x_i$, where $A$ is the set of twenty amino acids.

<table>
<thead>
<tr>
<th>A.A.</th>
<th>$p$</th>
<th>$t$</th>
<th>$a$</th>
<th>$s$</th>
<th>$q$</th>
<th>$k$</th>
<th>$p$</th>
<th>$t$</th>
<th>$a$</th>
<th>$s$</th>
<th>$q$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ala</td>
<td>7.3</td>
<td>3</td>
<td>11</td>
<td>5</td>
<td>2 (2)</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>1 (1)</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arg</td>
<td>5.2</td>
<td>6</td>
<td>29</td>
<td>23</td>
<td>5 (4)</td>
<td>10</td>
<td>20</td>
<td>14</td>
<td>5 (1)</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asn</td>
<td>4.6</td>
<td>4</td>
<td>16</td>
<td>10</td>
<td>3 (2)</td>
<td>6</td>
<td>13</td>
<td>7</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asp</td>
<td>5.1</td>
<td>4</td>
<td>13</td>
<td>7</td>
<td>3 (2)</td>
<td>6</td>
<td>10</td>
<td>4</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cys</td>
<td>1.8</td>
<td>4</td>
<td>12</td>
<td>6</td>
<td>3 (2)</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>2 (1)</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glu</td>
<td>4.0</td>
<td>5</td>
<td>20</td>
<td>14</td>
<td>4 (3)</td>
<td>8</td>
<td>14</td>
<td>8</td>
<td>4 (1)</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gln</td>
<td>6.2</td>
<td>5</td>
<td>17</td>
<td>11</td>
<td>4 (3)</td>
<td>8</td>
<td>11</td>
<td>5</td>
<td>4 (1)</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gly</td>
<td>6.9</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>1 (1)</td>
<td>3</td>
<td>8</td>
<td>2</td>
<td>1 (1)</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>His</td>
<td>2.3</td>
<td>4</td>
<td>18</td>
<td>12</td>
<td>3 (2)</td>
<td>6</td>
<td>15</td>
<td>9</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ile</td>
<td>5.8</td>
<td>6</td>
<td>22</td>
<td>16</td>
<td>5 (5)</td>
<td>11</td>
<td>13</td>
<td>7</td>
<td>3 (2)</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leu</td>
<td>9.3</td>
<td>6</td>
<td>23</td>
<td>17</td>
<td>5 (5)</td>
<td>11</td>
<td>14</td>
<td>8</td>
<td>3 (2)</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lys</td>
<td>5.8</td>
<td>7</td>
<td>27</td>
<td>21</td>
<td>6 (6)</td>
<td>13</td>
<td>12</td>
<td>6</td>
<td>5 (1)</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Met</td>
<td>2.3</td>
<td>6</td>
<td>20</td>
<td>14</td>
<td>5 (4)</td>
<td>10</td>
<td>11</td>
<td>5</td>
<td>4 (1)</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phe</td>
<td>4.1</td>
<td>4</td>
<td>24</td>
<td>18</td>
<td>3 (2)</td>
<td>6</td>
<td>21</td>
<td>15</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pro</td>
<td>5.0</td>
<td>1</td>
<td>17</td>
<td>12</td>
<td>1 (1)</td>
<td>3</td>
<td>17</td>
<td>12</td>
<td>1 (1)</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ser</td>
<td>7.4</td>
<td>4</td>
<td>12</td>
<td>6</td>
<td>3 (2)</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>2 (1)</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thr</td>
<td>5.8</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>4 (3)</td>
<td>8</td>
<td>11</td>
<td>5</td>
<td>2 (2)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trp</td>
<td>1.3</td>
<td>4</td>
<td>25</td>
<td>19</td>
<td>3 (2)</td>
<td>6</td>
<td>22</td>
<td>16</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tyr</td>
<td>3.3</td>
<td>5</td>
<td>25</td>
<td>19</td>
<td>4 (2)</td>
<td>7</td>
<td>21</td>
<td>15</td>
<td>3 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Val</td>
<td>6.5</td>
<td>5</td>
<td>19</td>
<td>13</td>
<td>4 (4)</td>
<td>9</td>
<td>13</td>
<td>7</td>
<td>2 (1)</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$w.a.$ - 4.5 18.2 12.2 3.6 (3.0) 7.6 12.8 6.8 2.7 (1.3) 5.0
Table 4. Cliques in the simplified side chains of amino acids. If $\mathcal{S}^{(i)}$, $2 \leq i < s'$ (s' is the number of cliques in the simplified side chain) is not listed, it is the same as Lys. 2D cliques are marked by an '∗'.

<table>
<thead>
<tr>
<th>A.A.</th>
<th>Side Chain Cliques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ala</td>
<td>$\mathcal{S}^{(1)} = {N, CA, HA, CB, QB, C}$</td>
</tr>
</tbody>
</table>
| Arg  | $\mathcal{S}^{(4)} = \{CG, CD, NE\}$*  
|      | $\mathcal{S}^{(5)} = \{CD, CE, HE, CZ, NH1, HH11, HH12\}$* |
| Asn  | $\mathcal{S}^{(3)} = \{CB, CG, OD1, ND2, HD21, HD22, QD2\}$* |
| Asp  | $\mathcal{S}^{(3)} = \{CB, CG, OD1, OD2\}$* |
| Cys  | $\mathcal{S}^{(2)} = \{CA, CB, SG\}$* |
| Glu  | $\mathcal{S}^{(4)} = \{CG, CD, OE1, OE2\}$* |
| Gln  | $\mathcal{S}^{(4)} = \{CG, CD, OE1, NE2, HE21, HE22, QE2\}$* |
| Gly  | $\mathcal{S}^{(1)} = \{N, CA, HA2, HA3, QA, C\}$ |
| His  | $\mathcal{S}^{(3)} = \{CB, CG, ND1, HD1, CD2, HD2, CE1, HE1, NE2\}$* |
| Ile  | $\mathcal{S}^{(2)} = \{CA, CB, HB, CG1, CG2, QG2\}$  
|      | $\mathcal{S}^{(3)} = \{CB, CG1, CD1, QD1\}$* |
| Leu  | $\mathcal{S}^{(3)} = \{CB, CG, HG, CD1, CD2, QD1, QD2, QOD\}$ |
| Lys  | $\mathcal{S}^{(1)} = \{N, CA, HA, CB, C\}$  
|      | $\mathcal{S}^{(2)} = \{CA, CB, CG\}$*  
|      | $\mathcal{S}^{(3)} = \{CB, CG, CD\}$*  
|      | $\mathcal{S}^{(4)} = \{CG, CD, CE\}$*  
|      | $\mathcal{S}^{(5)} = \{CD, CE, NZ, QZ\}$* |
| Met  | $\mathcal{S}^{(3)} = \{CB, CG, SD\}$*  
|      | $\mathcal{S}^{(4)} = \{CG, SD, CE QE\}$* |
| Phe  | $\mathcal{S}^{(3)} = \{CB, CG, CD1, HD1, CE1, HE1, CZ, HZ, CE2, HE2, CD2, HD2, QD, QE, QR\}$* |
| Pro  | $\mathcal{S}^{(1)} = \{N, CD, CA, HA, CB, HB2, HB3, QB, CG, HG2, HG3, QG, HD2, HD3, QD, C\}$ |
| Ser  | $\mathcal{S}^{(2)} = \{CA, CB, OG\}$* |
| Thr  | $\mathcal{S}^{(2)} = \{CA, CB, HB, OQ1, CG2, QG2\}$ |
| Trp  | $\mathcal{S}^{(3)} = \{CB, CG, CD1, HD1, CD2, CE2, CE3, HE3, NE1, HE1, CZ2, HZ2, CZ3, HZ3, CH2, HH2\}$* |
| Tyr  | $\mathcal{S}^{(3)} = \{CB, CG, CD1, HD1, CE1, HE1, CE2, HE2, CD2, HD2, CZ, OH, QD, QE, QR\}$* |
| Val  | $\mathcal{S}^{(2)} = \{CA, CB, HB, CG1, CG2, QG1, QG2, QQG\}$ |
Appendix C: Efficient Subspace Intersection Algorithm

Algorithm 1: Hierarchical bottom-up intersection

\[\text{input: Set of cliques } \{C_l\} \text{ and their matrices } \{U_l\}, l = 1,\ldots,q\]
\[\text{output: Matrix } U \text{ such that } \text{range}(U) = \bigcap_{l=1}^{q} \text{range}(U_l)\]

// Initialization
for \(i \leftarrow 1\) to \(q\) do
  \[Q_l^{(1)} = U_l\] // \(Q_l^{(i)}: U\) of the subtree rooted at the node \(j\), level \(i\)
  \[A_l^{(1)} = C_l\] // \(A_l^{(i)}: \text{points in the subtree rooted at the node } j, \text{level } i\)
end
\(v \leftarrow \lceil \log(q) \rceil + 1\) // number of levels in the tree
\(p \leftarrow q\) // number of cliques in the current level
\(p' \leftarrow p\) // number of cliques in the lower level
for \(i \leftarrow 2\) to \(v\) do
  \(p \leftarrow \lceil p'/2 \rceil\)
  for \(j \leftarrow 1\) to \(p\) do
    \(\ell_1 \leftarrow 2(j - 1) + 1\)
    \[A_j^{(i)} \leftarrow A_j^{(i-1)}\]
    \[Q_j^{(i)} \leftarrow Q_j^{(i-1)}\]
    if \(\ell_1 < p'\) then
      \(\ell_2 \leftarrow \ell_1 + 1\)
      \[A_j^{(i)} \leftarrow A_j^{(i)} \cup A_j^{(i-1)}\]
      \[Q_j^{(i)} \leftarrow \text{Intersect}(Q_j^{(i)}, Q_j^{(i-1)})\]
    end
  end
  \(p' \leftarrow p\)
end
\(U \leftarrow Q_1^{(v)}\) // For the root \(A_1^{(v)} = 1:n\)
Appendix D: SPROS Flowchart

Fig. 3. SPROS method flowchart.