A DISTRIBUTED SDP-BASED ALGORITHM FOR LARGE NOISY ANCHOR-FREE GRAPH REALIZATION

LEUNG NGAI-HANG ZACHARY

B. SC. (HONS.), NUS

A THESIS SUBMITTED
FOR THE DEGREE OF
MASTER OF SCIENCE

DEPARTMENT OF MATHEMATICS
NATIONAL UNIVERSITY OF SINGAPORE

2008

Acknowledgements

I would like to thank the following people:

Dr. Toh Kim-Chuan, my thesis supervisor, for starting me on this project. During these two years, he has been a guide and companion in this journey of learning and problem-solving. I have learnt much from him, and will treasure our time and work together as I continue in my future resesarch.

My parents, for bringing me up and teaching me to be the man I am today. I would not be anything without them!

My brother, for his companionship and camaraderie.

My friends for their prayer and support.

Gloria, my fiancé, for being my inspiration and comforter.

The Lord, for providing me with a project that suits my skills and interests, strength for the journey, light for the way, and hope for the future!

Though I walk in the midst of trouble,
you preserve my life;
you stretch out your hand against the wrath of my enemies,
and your right hand delivers me.
The LORD will fulfill his purpose for me;
your steadfast love, O LORD, endures forever.
Do not forsake the work of your hands.

Contents

Abstract

We propose the DISCO algorithm for graph realization in $\mathbb{R}^d$, given sparse and noisy short-range inter-vertex distances as inputs. Our divide-and-conquer algorithm works as follows. When a group has a sufficiently small number of vertices, the basis step is to form a graph realization by solving a semidefinite program. The recursive step is to break a large group of vertices into two smaller groups with overlapping vertices. These two groups are solved recursively, and the sub-configurations are stitched together, using the overlapping atoms, to form a configurations for the larger group. At intermediate stages, the configurations are improved by gradient descent refinement. The algorithm is applied to the problem of determining protein molecule structure. Tests are performed on molecules taken from the Protein Data Bank database. Given 20–30% of the inter-atom distances less than 6 Å that are corrupted by a high level of noise, DISCO is able to reliably and efficiently reconstruct the conformation of large molecules. In particular, given 30% of distances with 20% multiplicative noise, a 13000-atom conformation problem is solved within an hour with an RMSD of 1.6 Å.

List of Tables

List of Figures

1 Introduction

The field of *distance geometry* is the study of sets of points based on only pairwise distances between points. One of the particular problems in distance geometry is the *graph realization problem*—to assign coordinates to vertices in a graph, with the restriction that distances between certain pairs of vertices are specified to lie in given intervals. Two practical instances of the graph realization problem are the molecular conformation problem and the sensor network localization problem.

The *molecular conformation problem* is to determine the structure of a protein molecule based on pairwise distances between atoms. Determining protein conformations is central to biology, because knowledge of the protein structure aids in the understanding of protein functions, which would lead to further applications in pharaceuticals and medicine. In this problem, the distance constraints are obtained from knowledge of the sequence of constituent amino acids; minimum separation distances (MSDs) derived from van der Waals interactions; and nuclear magnetic resonance (NMR) spectroscopy experiments. We take note of two important characteristics of molecular problems: the number of atoms may number in the tens of thousands, and the distance data may be very sparse and highly noisy.

The *sensor network localization problem* is to determine the location of wireless sensors in a network. In this problem, there are two classes of objects: anchors (whose locations are known a priori) and sensors (whose locations are unknown and to be determined). In practical situations, the anchors and sensors are able to communicate with one another, if they are not too far apart (say within radio range), and obtain an estimate of the distance between them.

While the two problems are very similar, the key difference between molecular conformation and sensor network localization is that the former is anchor-free, whereas in the latter the positions of the anchor nodes are known a priori.

Recently, semidefinite programming (SDP) relaxation techniques have been applied to the sensor network localization problem [1]. While this approach was successful for moderately-size problems with sensors in the order of a few hundreds, it was unable to solve problems with more sensors, due to limitations in SDP algorithms, software and hardware. A distributed SDP-based algorithm for sensor network localization was proposed in [3], with the objective of localizing larger networks. One critical assumption required for the algorithm to work well is that there exist anchor nodes distributed uniformly throughout the physical space. The algorithm relies on the anchor nodes to divide the sensors into clusters, and solves each cluster separately using an SDP relaxation. In general, a

divide-and-conquer algorithm must address the issue of combining the solutions of smaller subproblems into a solution for the larger subproblem. This is not an issue in the sensor network localization problem, because the solutions to the clusters automatically form a global configuration, as the anchors endow the sensors with global coordinates.

A natural question arises as to whether the distributed method proposed in [3] can be applied to molecular conformation. Unfortunately, it does not, as the assumption of uniformly distributed anchor nodes does not hold in the case of molecules.

The authors of [3] proposed a distributed SDP-based algorithm (the DAFGL algorithm) for the molecular problem [2]. The results of the DAFGL algorithm are satisfactory when given 50% of pairwise distances less than 6 Å that are corrupted by 5% multiplicative noise. The main objective of this paper is to design a robust and efiicient distributed algorithm that can handle the challenging situation [25] when 30% of short-range pairwise distances are given, and are corrupted with 10–20% multiplicative noise.

In this paper, we describe a new distributed approach, the DISCO (for DIStributed COnformation) algorithm, for the anchorless graph realization problem. By applying the algorithm to molecular conformation problems, we demonstrate its reliability and efficiency. In particular, for a 13000-atom protein molecule, we were able to estimate the positions to an RMSD of 1.6 Å given only 30% of the pairwise distances (corrupted by 20% multiplicative noise) less than 6 Å.

The remainder of the paper is organized as follows: Section 2 describes existing molecular conformation algorithms; Section 3 details the mathematical models for molecular conformation; Section 4 explains the design of DISCO; Section 5 contains the experiment setup and numerical results; Section 6 gives the conclusion.

The DISCO webpage [12] contains additional material, including the DISCO code, and a video of how DISCO solves the 1534-atom molecule 1F39.

In this paper, we adopt the following notational conventions. Lower case letters, such as n, are used to represent scalars. Lower case letters in bold font, such as $\boldsymbol{s}$, are used to represent vectors. Upper case letters, such as X, are used to represent matrices. Upper case letters in calligraphic font, such as $\mathcal{D}$, are used to represent sets. Cell arrays will be prefixed by a letter "c" and be in the math italic font, such as *cAest*. Cell arrays will be indexed by curly braces {}.

2 Related Work

In this section, we give a brief tour of select existing works. Besides presenting the algorithms, we would like to highlight that each algorithm was tested on different types of input data. For instance, some inputs were exact distances, while others were distances corrupted by low levels of noise, yet others were distances corrupted with high levels of noise; some inputs consist of all the pairwise distances less than a certain cut-off distance, while others give only a proportion of the pairwise distances less than a certain cut-off distance. It is also the case that not all the authors used the same error measure. Although the accuracy of a molecular conformation is most commonly measured by the RMSD (root mean square deviation), some of the authors did not provide the RMSD error, but only the maximum violation of lower or upper bounds for pairwise inter-atom distances. (We present more details about the RMSD measure in Section 5.) Finally, because we aim to design an algorithm which is able to scale to large molecules, we make a note of the largest molecule which each algorithm was able to solve in the tests done by the authors. We summarize this information in Table 1.

2.1 Methods Using the Inner Product Matrix

It is known from the theory of distance geometry that there is a natural correspondance between inner product matrices and distance matrices [21, 22, 23]. Thus, one approach to the molecular conformation problem is to use a distance matrix to generate an inner product matrix, which can then be factorized to recover the atom coordinates. The methods we present in §2.1 differ in how they construct the inner product matrix, but use the same procedure to compute the atom coordinates; we describe this procedure in detail below. If we denote the atom coordinates by columns $\boldsymbol{x}_i$, and let $X = [\boldsymbol{x}_1 \dots \boldsymbol{x}_n]$, then the inner product matrix Y is given by $Y = X^T X$. We can recover approximate coordinates $\tilde{X}$ from a noisy $\tilde{Y}$ by taking the best rank-3 approximation $\tilde{Y} \approx \tilde{X}^T \tilde{X}$, based on the eigenvalue decomposition of $\tilde{Y}$.

The EMBED algorithm [9] was developed by Havel, Kuntz and Crippen in 1983. Given lower and upper bounds on some of the pairwise distances as input, EMBED attempts to find a feasible conformation as follows. Initially, we only have bounds on some of the distance pairs. EMBED begins by using the triangle and tetrangle inequalities to compute distance bounds for all pairs of points. EMBED then chooses random numbers within the bounds to form an estimated distance matrix $\tilde{D}$, and checks if $\tilde{D}$ is close to a valid dimension-three Euclidean

distance matrix by considering the three largest absolute-value eigenvalues of $\tilde{Y}$, the inner product matrix corresponding to $\tilde{D}$. In the fortunate case, the three eigenvalues are positive, and are much larger than the rest. This would indicate that the estimated distance matrix $\tilde{D}$ is close to a true distance matrix, and the coordinates obtained from the inner product matrix are likely to be fairly accurate. In the unfortunate case where at least one of the three eigenvalues is negative, the estimated distance matrix $\tilde{D}$ is far from a valid distance matrix. In this case, EMBED repeats the step of choosing an estimated distance matrix until it obtains one that is close to a valid distance matrix. As a postprocessing step, the coordinates are improved by applying local optimization methods.

The DISGEO package [10], was developed by Havel and Wüthrich in 1984, so as to solve larger conformation problems. The EMBED algorithm is unable to compute a conformation of the whole protein structure, due to the high dimensionality of the problem. DISGEO works around this limitation by using two passes of EMBED. In the first pass, coordinates are computed for a subset of atoms subject to constraints inherited from the whole structure. This step forms a "skeleton" for the structure. The second pass of EMBED computes coordinates for the remaining of the atoms, building upon the skeleton computed in the first pass. As Havel and Wüthrich are biologists, their desired to design an algorithm that can compute protein structures based on realistic input data. They tested the performance of DISGEO on the BPTI protein, which has 454 atoms. The input consists of distance (3290) and chirality (450) constraints needed to fix the covalent structure, and bounds (508) for distances between hydrogen atoms less than 4 Å apart and in different amino acide residues, to simulate the distance constraints available from a NOESY experiment. Using a pseudostructure representation, they were able to solve for 666 geometric points[1] given 3798 distance and 450 chirality constraints, with three computed structures having an average RMSD of 2.08 Å from the known crystal structure. Havel's DG-II package [8], published in 1991, improves upon DISGEO by producing from the same input as DISGEO five structures having an average RMSD of 1.76 Å from the crystal structure.

The alternating projections algorithm (APA) for molecular conformation was developed in 1990 [5, 16]. As in EMBED, APA begins by using the triangle inequality to compute distance bounds for all pairs of points. We can think of the lower and upper bounds as forming a rectangular parallepiped, which the authors

[1] In NMR experiments, certain protons may not be stereospecifically assigned. For such pairs of protons, the upper bounds are modified via the creation of "pseudoatoms", as is the standard practice in NOE experiments.

refer to as a *data box*. Next, a random *dissimilarity matrix* Δ in the data box is chosen. (The dissimilarity matrix serves the same function as the estimated distance matrix in EMBED.) The dissimilarity matrix is smoothed by column metrization, so that it adheres to the triangle inequality. Next, Δ is projected onto the cone of matrices that are negative semidefinite on the orthogonal complement of $e = (1, 1, \ldots, 1)^T$, then back onto the data box. The alternating projections are repeated five times. The theoretical basis of this procedure is that as the number of projection steps goes to infinity, the resultant matrix converges to a distance matrix that satisfies the lower and upper bounds [16]. Finally, the atom coordinates are obtained from the inner product matrix, which is computed from the last dissimilarity matrix. The postprocessing step involves performing stress minimization on the resultant structure. In [16], APA was applied to the BPTI protein to compare its performance to DISGEO and DG-II. Under the exact same inputs as DISGEO and DG-II, the five best structures out of thirty produced by APA had an average RMSD of 2.39 Å compared with the crystal structure.

Classical multidimensional scaling (MDS) is a collection of techniques for constructing configurations of points from pairwise distances. Trosset has applied MDS to the molecular conformation problem [21, 22, 23] since 1998. Again, the first step is to use the triangle inequality to compute distance bounds for all pairs of points. Trosset's approach is to solve the problem of finding the squared dissimilarity matrix that minimizes the distances to the cone of symmetric positive semidefinite matrices of rank less than d, while satisfying the squared lower and upper bounds. The problem is solved by applying a local optimization method, namely a limited memory approximate Hessian method. The coordinates can be extracted from an inner product matrix that is computed from the squared dissimilarity matrix. In [23], MDS is applied to five molecules with less than 700 atoms. For points with pairwise distances d_{ij} less than 7 Å, lower and upper bounds of the form $(d_{ij} - 0.01\text{Å}, d_{ij} + 0.01\text{Å})$ are given; for pairwise distances greated than 7 Å, a lower bound of 7 Å is specified. The method was able to produce estimated configurations that had a maximum bound violation of less than 0.1 Å. The author did not report the RMSD of the computed configurations, but mentioned that the configurations are "quite acceptable by the standards of computational chemistry".

More recently, in 2006, Trosset with coauthors Grooms and Lewis did work on a dissimilarity parameterized approach [6]. The authors advocate using a dissimilarity parametrization rather than a coordinate-based parametrization. Although the latter has fewer independent variables, the former seems to have converge to

"better" minimizers. Their method is named StrainMin because of its origins in the strain criterion of classical MDS. They formulate the problem as that of minimizing an objective which is the sum of the fit of the dissimilarity matrix to the data and the distance of the dissimilarity matrix to the space of rank d positive semidefinite matrices (the *strain*). By analyzing the properties of the objective function, they developed an efficient local optimization method that makes use of second-order information. The approach was tested on input data that consists of exact distances between atoms less than 6 Å apart, and a 2.5 Å lower bound as a representative van der Waal radii for atoms whose distance is unknown. They were able to satisfy the distance bounds with a maximum violation of 0.2 Å, for an ensemble of 6 PDB molecules. However, the RMSD errors were not reported.

The DAFGL algorithm of Biswas, Toh and Ye in 2008 [2] is a "parent" of this work. DAFGL differs from the previous methods in that it applies SDP relaxation methods to obtain the inner product matrix. Due to limitations in SDP algorithms, software and hardware, the largest SDP problems that can be solved are of the order of a few hundred atoms. In order to solve larger problems, DAFGL employs a distributed approach. It applies the symmetric reverse Cuthill-Mckee matrix permutation to divide the atoms into smaller groups with overlapping atoms. Each group is solved using SDP, and the overlapping groups are used to align the local solutions to form a global solution. Tests were performed on 14 molecules with number of atoms ranging from 400–5600. The input data consists of 70% of the distances d_{ij} below 6 Å, given as lying in intervals $[\underline{d}_{ij}, \overline{d}_{ij}]$ where

$$\underline{d}_{ij} = \max\big(0, (1 - 0.05|\underline{Z}_{ij}|)d_{ij}\big), \quad \overline{d}_{ij} = (1 + 0.05|\overline{Z}_{ij}|)d_{ij},$$

and $\underline{Z}_{ij}, \overline{Z}_{ij}$ are standard normal random variables with zero mean and unit variance. Given such input, DAFGL is able to produce a conformation for most molecules with an RMSD of 2–3 Å.

2.2 Buildup Methods

The ABBIE program [11] was developed by Hendrickson in 1995, to solve molecular conformation problems given exact distance data. As embedding problems in one dimension are strongly NP-complete, and in two and higher spatial dimensions are NP-hard [17], ABBIE uses a divide-and-conquer approach to make the computation more tractable. ABBIE aims to divide the problem into smaller pieces by identifying *uniquely realizable subgraphs*—subgraphs that permit a unique realization. The first step is to use graph algorithms to divide the

atoms into maximally uniquely realizable subgraphs. If at the end of this step, a subgraph is too large to be solved directly, then ABBIE continues by using small vertex separators to break a subgraph into smaller pieces, and recurse on the pieces. ABBIE proceeds to use heuristics to group vertices into *chunks*—subsets of vertices whose relative positions to one another are fixed. This step is important because combinatorial methods are faster than optimization methods. Finally, ABBIE uses an optimization routine to combine chunks and vertices together. Hendrickson tested ABBIE on the protein molecule with PDB ID 7RSA. After discarding end chains, the molecule had 1849 atoms. The input data included the exact distances between all pairs of atoms in the same amino acid (13879), and 1167 additional distances between H atoms less than 3.5 Å apart. This made for a total of 15046 edges so that the mean degree of a vertex is 16.3. Although it was not explicitly mentioned in the paper, we presume he was able to get the exact solution up to roundoff error.

Dong and Wu [4, 26], presented their geometric buildup algorithm in 2003, which also relies on having exact distances. The essential idea of this algorithm is that if four atoms form a four-clique—four atoms with distances between all pairs known—the atom positions are fixed relative to one another. The algorithm starts by finding a four-clique and fixing the coordinates of the four atoms. The other atom positions are determined atom-by-atom; when the distance of an atom to four atoms with determined coordinates is known, that atom position can be uniquely determined. The authors conducted numerical experiments on ten protein molecules, the largest of which has 4200 atoms. When given all the distances less than 8 Å, the geometric buildup algorithm is able to accurately estimate all atoms; when given all the distances less than 5 Å, the geometric buildup algorithm is able to accurately estimate nine of the ten atoms.

2.3 Global Optimization Methods

For an introduction to optimization-based methods for molecular conformation, see [13]. Here we describe briefly two such methods.

The DGSOL code [14, 15] by Moré and Wu in 1999 treats the molecular conformation problem as a large nonlinear least squares problem. As the objective function has many local minima, they apply Gaussian smoothing to the objective function to increase the likelihood of finding the global minima. They applied DGSOL to two protein fragments consisting of 100 and 200 atoms respectively. Distances were specified for atoms in the same or neighboring residues, and given as lower bounds $\underline{d}_{ij} = 0.84 d_{ij}$ and upper bounds $\overline{d}_{ij} = 1.16 d_{ij}$, where d_{ij} denotes

the true distance between atoms i and j. DGSOL was able to compute structures with a minimum and average RMSD of 0.37 Å and 1.0 Å respectively for 100 atoms and a minimum and average RMSD of 0.7 Å and 2.9 Å respectively for 200 atoms.

The GNOMAD algorithm [25] by Williams, Dugan and Altman in 2001 attempts to satisfy the input distance constraints as well as MSD constraints. Their algorithm applies to the situation when we are given sparse but exact distances. The knowledge of MSD constraints is useful in limiting the search space, but if they are not applied intelligently, then they may keep the algorithm stuck in an unsatisfactory local minimum. Since it is difficult to optimize all of the atom positions simultaneously, because of the high dimensionality of the system, GNOMAD updates the positions of the atoms one atom at a time. The reduced dimensionality allows GNOMAD to more easily satisfy the input data and MSD constraints. The authors tested GNOMAD on the protein molecule with PDB ID 1TIM, which has 1870 atoms. Given all the convalent distances and distances between atoms that share covalent bonds to the same atom, as well as 30% of short-range distances less than 6 Å, they were able to compute estimated positions with an RMSD of 1.07 Å(but see footnote[2]).

We end this section by noting that while the GNOMAD algorithm would increasing get stuck in an unsatisfactory local minimum with more stringent MSD constraints, the addition of such lower bound constraints are highly beneficial for DISCO.

3 Mathematics of Molecular Conformation

We begin this section with the SDP models for sensor network localization in §3.1. These are closely related to the SDP models for molecular conformation, which we present next in §3.2. We then introduce the gradient descent method for improving sensor positions in §3.3. Finally, we present the alignment problem in §3.4.

3.1 SDP Models for Sensor Network Localization

The setting of the sensor network localization problem is as follows. We are given a set of n_a anchor nodes with known coordinates $\boldsymbol{a}_i \in \mathbb{R}^d$, $i = 1, \ldots, n_a$, and we wish to determine the coordinates of n_s sensor nodes $\boldsymbol{s}_i \in \mathbb{R}^d$, $i = 1, \ldots, n_s$. The

[2] The number reported in Figure 11 in [25] is inconsistent with that appearing in Figure 8. It seems that the correct RMSD should be about 2–3 Å.

information that is available is measured distances or distance bounds for some of the pairwise distances $\|\boldsymbol{a}_i - \boldsymbol{s}_j\|$ for $(i,j) \in \mathcal{N}^a$ and $\|\boldsymbol{s}_i - \boldsymbol{s}_j\|$ for $(i,j) \in \mathcal{N}^s$. In the "measured distances" model, we have measured distances for certain pairs of nodes,

$$\begin{aligned} \tilde{d}^a_{ij} &\approx \|\boldsymbol{a}_i - \boldsymbol{s}_j\| \qquad (i,j) \in \mathcal{N}^a, \\ \tilde{d}^s_{ij} &\approx \|\boldsymbol{s}_i - \boldsymbol{s}_j\| \qquad (i,j) \in \mathcal{N}^s. \end{aligned} \tag{1}$$

In this model, the unknown positions $\{\boldsymbol{s}_i\}_{i=1}^{n_s}$ is the best fit to the measured distances, obtained by solving the following nonconvex minimization problem:

$$\min \left\{ \sum_{(i,j)\in\mathcal{N}^s} \left| \|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 - (\tilde{d}^s_{ij})^2 \right| + \sum_{(i,j)\in\mathcal{N}^a} \left| \|\boldsymbol{a}_i - \boldsymbol{s}_j\|^2 - (\tilde{d}^a_{ij})^2 \right| \right\}. \tag{2}$$

We denote the measured anchor-sensor and sensor-sensor distance matrices by $\tilde{D}^a$ and $\tilde{D}^s$ respectively. In the "distance bounds" model, we have lower and upper bounds on the distances between certain pairs of nodes,

$$\begin{aligned} \underline{d}^a_{ij} \le \|\boldsymbol{a}_i - \boldsymbol{s}_j\| \le \overline{d}^a_{ij} \qquad (i,j) \in \mathcal{N}^a, \\ \underline{d}^s_{ij} \le \|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le \overline{d}^s_{ij} \qquad (i,j) \in \mathcal{N}^s. \end{aligned} \tag{3}$$

In this model, the unknown positions $\{\boldsymbol{s}_i\}_{i=1}^{n_s}$ is the best fit to the measured distances, obtained by solving the following nonconvex minimization problem:

$$\begin{aligned} \min \Bigg\{ & \sum_{(i,j)\in\mathcal{N}^s} \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 - (\underline{d}^s_{ij})^2\right)_- + \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 - (\overline{d}^s_{ij})^2\right)_+ \\ & + \sum_{(i,j)\in\mathcal{N}^a} \left(\|\boldsymbol{a}_i - \boldsymbol{s}_j\|^2 - (\underline{d}^a_{ij})^2\right)_- + \left(\|\boldsymbol{a}_i - \boldsymbol{s}_j\|^2 - (\overline{d}^a_{ij})^2\right)_+ \Bigg\}, \end{aligned} \tag{4}$$

where $\alpha_+ = \max\{0, \alpha\}, \alpha_- = \max\{0, -\alpha\}$. We denote the lower and upper bound anchor-sensor and sensor-sensor distance matrices by $\underline{D}^a, \overline{D}^a$ and $\underline{D}^s, \overline{D}^s$ respectively.

In order to proceed to the SDP relaxation of the problem, we need to consider the matrix

$$Z = \begin{bmatrix} Y & X^T \\ X & I_d \end{bmatrix} \qquad \text{where } Y = X^T X, \quad X = [\boldsymbol{s}_1 \dots \boldsymbol{s}_n]. \tag{5}$$

By denoting the i-th unit vector in $\mathbb{R}^{n_s}$ by $\boldsymbol{e}_i$, and denoting $\boldsymbol{e}_{ij} = \boldsymbol{e}_i - \boldsymbol{e}_j$, we note

that

$$\|\boldsymbol{a}_i - \boldsymbol{s}_j\|^2 = [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z [\boldsymbol{e}_j; -\boldsymbol{a}_i],$$
$$\|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 = [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T Z [\boldsymbol{e}_{ij}; \boldsymbol{0}_d].$$

We can therefore conveniently express the constraints (1) as

$$(\tilde{d}^a_{ij})^2 \approx [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z [\boldsymbol{e}_j; -\boldsymbol{a}_i] \qquad (i,j) \in \mathcal{N}^a,$$
$$(\tilde{d}^s_{ij})^2 \approx [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T Z [\boldsymbol{e}_{ij}; \boldsymbol{0}_d] \qquad (i,j) \in \mathcal{N}^s;$$

and (3) as

$$(\underline{d}^a_{ij})^2 \le [\boldsymbol{e}_j; -\boldsymbol{a}_i] Z [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T \le (\overline{d}^a_{ij})^2 \qquad (i,j) \in \mathcal{N}^a,$$
$$(\underline{d}^s_{ij})^2 \le \ [\boldsymbol{e}_{ij}; \boldsymbol{0}_d] Z [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T \ \le (\overline{d}^s_{ij})^2 \qquad (i,j) \in \mathcal{N}^s.$$

The SDP relaxation is then rather straightforward, to relax the constraint (5) into the constraints

$$Z = \begin{bmatrix} Y & X^T \\ X & I_d \end{bmatrix} \qquad \text{where } Y \succcurlyeq X^T X, \quad X = [\boldsymbol{s}_1 \dots \boldsymbol{s}_n]. \tag{6}$$

By a Schur's complement argument, we have $Y \succcurlyeq X^T X$ if and only if $Z \succcurlyeq 0$, and thus (6) is equivalent to the following

$$Z = \begin{bmatrix} Y & X^T \\ X & I_d \end{bmatrix} \succcurlyeq 0. \tag{7}$$

We can now express the measured distances model (2) as

$$\begin{aligned}
\min \quad & \textstyle\sum_{(i,j)\in\mathcal{N}^a} t^+_{ij} + t^-_{ij} + \sum_{(i,j)\in\mathcal{N}^s} u^+_{ij} + u^-_{ij} \\
& [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z [\boldsymbol{e}_j; -\boldsymbol{a}_i] + t^+_{ij} - t^-_{ij} = (\tilde{d}^a_{ij})^2 \qquad (i,j) \in \mathcal{N}^a, \\
& [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T Z [\boldsymbol{e}_{ij}; \boldsymbol{0}_d] + u^+_{ij} - u^-_{ij} = (\tilde{d}^s_{ij})^2 \qquad (i,j) \in \mathcal{N}^s, \\
\text{s.t.} \quad & Z(n_s+1:d, n_s+1:d) = I_d, \\
& Z \succcurlyeq 0.
\end{aligned} \tag{8}$$

Similarly we can express the distance bounds model as

$$
\begin{array}{ll}
\text{Find} & Z \\
& (\underline{d}^a_{ij})^2 \le [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z [\boldsymbol{e}_j; -\boldsymbol{a}_i] \le (\overline{d}^a_{ij})^2 \qquad (i,j) \in \mathcal{N}^a, \\
& (\underline{d}^s_{ij})^2 \le \ [\boldsymbol{e}_{ij}; \mathbf{0}_d]^T Z [\boldsymbol{e}_{ij}; \mathbf{0}_d] \ \le (\overline{d}^s_{ij})^2 \qquad (i,j) \in \mathcal{N}^s, \\
\text{s.t.} & Z(n_s+1:d, n_s+1:d) = I_d, \\
& Z \succcurlyeq 0.
\end{array} \tag{9}
$$

We recover the estimated sensor positions $X = [\boldsymbol{s}_1 \dots \boldsymbol{s}_{n_s}]$ from Z as follows. If there are less than $d+1$ anchors, then X is obtained from the best rank-d approximation of the $(1,1)$-block of Z; otherwise, X is set to be equal to the $(2,1)$-block of Z.

So and Ye [18] have shown that if the distance data is uniquely localizable, then the SDP relaxation (8) or (9) is able to produce the exact sensor coordinates up to rounding errors. We are not going to define rigorously the concept "uniquely localizable". Intuitively, it means that there is only one configuration in $\mathbb{R}^d$ (perhaps up to translation, rotation, reflection) that satisfies all the distance constraints. The result of So and Ye gives us a degree of confidence that the SDP relaxation technique is a strong relaxation. We can therefore hope that applying SDP relaxation to sparse and noisy problems will be successful.

We now discuss what happens when the distance data is sparse and/or noisy, so that there is no unique realization. In such a situation, it is not possible to compute the exact coordinates. Further, the X and Y extracted from the solution Z of the SDP will not satisfy $Y = X^T X$, and Y will be of dimension greater than d. We present an intuitive explanation for this phenomenon. Suppose we have points in the plane, and certain pairs of points are constrained so that the distance between them is fixed. If the distances are perturbed slightly, then some of the points may be forced out of the plane in order to satisfy the distance constraints. Therefore, under noise, Y will tend to have a rank higher than d. Another reason for Y having a higher rank is that if there are multiple solutions, the interior-point methods used by many SDP solvers converge to max-rank solutions [7].

This situation presents us with potential problems. If Y has a higher rank than X, then the solution X extracted from Z is likely not to be an accurate solution. To ameliorate this situation, we add the following regularization term into the objective function

$$-\gamma \langle I - \boldsymbol{a}\boldsymbol{a}^T, Z \rangle, \tag{10}$$

with $\boldsymbol{a} = [\hat{\boldsymbol{e}}; \hat{\boldsymbol{a}}]$, $\hat{\boldsymbol{a}} = \sum_{i=1}^{n_a} \boldsymbol{a}_i/\sqrt{n_a+n_s}$, $\hat{\boldsymbol{e}} = \boldsymbol{e}/\sqrt{n_a+n_s}$, and γ a positive regularization parameter. This term spreads the sensors further apart and induces them to exist in a lower-dimensional space. We refer interested readers to [1] for details on the derivation of the regularization term. Thus the measured distances model (8) becomes

$$\begin{aligned}
\min \quad & \textstyle\sum_{(i,j)\in\mathcal{N}^a} t_{ij}^+ + t_{ij}^- + \sum_{(i,j)\in\mathcal{N}^s} u_{ij}^+ + u_{ij}^- - \gamma\langle I - \boldsymbol{a}\boldsymbol{a}^T, Z\rangle \\
& [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z[\boldsymbol{e}_j; -\boldsymbol{a}_i] + t_{ij}^+ - t_{ij}^- = (\tilde{d}_{ij}^a)^2 \qquad (i,j)\in\mathcal{N}^a, \\
& [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T Z[\boldsymbol{e}_{ij}; \boldsymbol{0}_d] + u_{ij}^+ - u_{ij}^- = (\tilde{d}_{ij}^s)^2 \qquad (i,j)\in\mathcal{N}^s, \\
\text{s.t.} \quad & Z(n_s+1:d, n_s+1:d) = I_d, \\
& Z \succcurlyeq 0.
\end{aligned} \tag{11}$$

and the distance bounds model (9) becomes

$$\begin{aligned}
\min \quad & -\langle I - \boldsymbol{a}\boldsymbol{a}^T, Z\rangle \\
& (\underline{d}_{ij}^a)^2 \le [\boldsymbol{e}_j; -\boldsymbol{a}_i]^T Z[\boldsymbol{e}_j; -\boldsymbol{a}_i] \le (\overline{d}_{ij}^a)^2 \qquad (i,j)\in\mathcal{N}^a, \\
& (\underline{d}_{ij}^s)^2 \le \ [\boldsymbol{e}_{ij}; \boldsymbol{0}_d]^T Z[\boldsymbol{e}_{ij}; \boldsymbol{0}_d] \ \le (\overline{d}_{ij}^s)^2 \qquad (i,j)\in\mathcal{N}^s, \\
\text{s.t.} \quad & Z(n_s+1:d, n_s+1:d) = I_d, \\
& Z \succcurlyeq 0.
\end{aligned} \tag{12}$$

3.2 SDP Models for Molecular Conformation

The setting of the molecular conformation problem is as follows. We wish to determine the coordinates of n atoms $\boldsymbol{s}_i \in \mathbb{R}^d, i = 1, \ldots, n_s$, given measured distances or distance bounds for some of the pairwise distances $\|\boldsymbol{s}_i - \boldsymbol{s}_j\|$ for $(i,j) \in \mathcal{N}$. One can observe that the molecular conformation problem can be viewed as a sensor network localization problem without anchors. Since the molecular conformation problem is a special class of sensor network localization problems, we can apply simplifications to the SDP formulations which we have derived previously. For reasons of clarity and convenience, we shall borrow the notation and terminology of the sensor network localization in this section. We shall henceforth refer to atoms as sensors.

In this problem, there are no anchors, so the $(2,2)$-block of Z no longer serves any purpose. Instead, we only need to consider the smaller matrix Y to express

the distance between sensors,

$$\|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 = \boldsymbol{e}_{ij}^T Y \boldsymbol{e}_{ij}.$$

The constraint that $Z \succcurlyeq 0$ is correspondingly replaced by the constraint $Y \succcurlyeq 0$. The regularization term (10) is replaced by

$$-\gamma \langle I - \hat{\boldsymbol{e}}\hat{\boldsymbol{e}}^T, Y \rangle$$

where γ is a positive regularization parameter and $\hat{\boldsymbol{e}} = \boldsymbol{e}/\sqrt{n_s}$. Since anchors are absent, the sensors have translational, rotational and reflective freedom. This can cause numerical difficulties when solving the SDP relaxation of the problem. The situation is improved when we remove the translational freedom, by introducing a constraint that mimics setting the center of mass to be the origin,

$$\langle Y, E \rangle = 0,$$

where E is the matrix of all ones. Finally, as before, the estimated sensor positions $X = [\boldsymbol{s}_1 \dots \boldsymbol{s}_{n_s}]$ are obtained from the best rank-d approximation of Y.

Putting all this together, we have the measured distances model

$$\begin{array}{ll} \min & \sum_{(i,j)\in\mathcal{N}^s} u_{ij}^+ + u_{ij}^- - \gamma \langle I - \boldsymbol{a}\boldsymbol{a}^T, Z \rangle \\ & \boldsymbol{e}_{ij}^T Y \boldsymbol{e}_{ij} + u_{ij}^+ - u_{ij}^- = (\tilde{d}_{ij}^s)^2 \qquad (i,j) \in \mathcal{N}^s, \\ \text{s.t.} & \langle Y, E \rangle = 0, \\ & Y \succcurlyeq 0,\ u^+, u^- \geq 0, \end{array} \tag{13}$$

and the distance bounds model

$$\begin{array}{ll} \min & -\langle I - \boldsymbol{a}\boldsymbol{a}^T, Z \rangle \\ & (\underline{d}_{ij}^s)^2 \leq \boldsymbol{e}_{ij}^T Y \boldsymbol{e}_{ij} \leq (\overline{d}_{ij}^s)^2 \qquad (i,j) \in \mathcal{N}^s, \\ \text{s.t.} & \langle Y, E \rangle = 0, \\ & Y \succcurlyeq 0. \end{array} \tag{14}$$

3.3 Coordinate Refinement via Gradient Descent

If we are given measured pairwise distances $\tilde{d}_{ij}$, then the sensor coordinates can be computed as the minimizer of

$$\min f(X) := \sum_{(i,j)\in\mathcal{N}^a} \left(\|\boldsymbol{a}_i - \boldsymbol{s}_j\| - \tilde{d}^a_{ij}\right)^2 + \sum_{(i,j)\in\mathcal{N}^s} \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\| - \tilde{d}^s_{ij}\right)^2. \tag{15}$$

Note that this formulation is different from (2). Similarly, if we are given bounds for pairwise distances $\underline{d}_{ij}$ and $\overline{d}_{ij}$, then the configuration can be computed as the solution of

$$\begin{aligned}\min f(X) := &\left[\sum_{(i,j)\in\mathcal{N}^a} \left(\|\boldsymbol{a}_i - \boldsymbol{s}_j\| - \underline{d}^a_{ij}\right)^2_- + \left(\|\boldsymbol{a}_i - \boldsymbol{s}_j\| - \overline{d}^a_{ij}\right)^2_+\right] \\ &+ \left[\sum_{(i,j)\in\mathcal{N}^s} \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\| - \underline{d}^s_{ij}\right)^2_- + \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\| - \overline{d}^s_{ij}\right)^2_+\right].\end{aligned} \tag{16}$$

Again, note that this formulation is different from (4). We can solve (15) or (16) by applying local optimization methods. For simplicity, we choose to use a gradient descent method with backtracking line search. The implemmentation of this method is rather straightforward. It is a simple exercise in calculus to compute the gradient of f with respect to the sensor coordinate $\boldsymbol{s}_i$, and so the gradient of f is easy to obtain.

The problems (15) and (16) are highly nonconvex problems with many local minimizers. If the initial iterate X_0 is not close to a good solution, then it is extremely unlikely that the X obtained from a local optimization method will be a good solution. In our case however, when we set X_0 to be the conformation produced from solving the SDP relaxation, local optimization methods are often able to produce an X with higher accuracy than the original X_0.

3.4 Alignment of Configurations

The molecular conformation problem is anchor-free. so that a configuration has translational, rotational, and reflective freedom. Nevertheless, we need to be able to compare two configurations, to determine how similar they are. In particular, we need to compare a computed configuration to the true configuration. In order to perform a comparision of two configurations, it is necessary to align them in a common coordinate system. We can define the "best" alignment as the affine

transformation T that minimizes

$$\min\left\{\sum_{i=1}^{n}\|T(\boldsymbol{a}_i)-\boldsymbol{b}_i\| : T(\boldsymbol{x})=Q\boldsymbol{x}+\boldsymbol{c}, Q\in\mathbb{R}^{d\times d}, Q \text{ is orthogonal}\right\}. \quad (17)$$

The constraint on the form of T restricts T to be a combination of translation, rotation and reflection. In the special case when A and B are centered at the origin, (17) reduces to an orthogonal procrustes problem

$$\min\big\{\|QA-B\|_F : Q\in\mathbb{R}^{d\times d}, Q \text{ is orthogonal}\big\}.$$

It is well known that the optimal Q can be computed from the singular value decomposition of AB^T.

4 The DISCO Algorithm

Here we present the DISCO algorithm (for DIStributed COnformation). In §4.1, we explain the essential ideas that are incorporated into the design of DISCO. We present the procedures for the recursive and basis cases in §4.2 and §4.3 respectively.

4.1 The Basic Ideas of DISCO

Prior to this work, it was known that the SDP relaxation technique and gradient descent are able to accurately localize moderately sized problems (say the number of atoms is less than 500). However, many protein molecules have more than 10000 atoms. In this work, we develop techniques to solve large-scale problems.

A natural idea is to employ a divide-and-conquer approach, which will follow the general framework: **If** the number of atoms is not too large, then solve the atom positions via SDP, and utilize gradient descent refinement to compute improved coordinates; **Otherwise** break the atoms into two subgroups, solve each subgroup recursively, and align and combine them together, again postprocessing the coordinates by applying gradient descent refinement.

How should we divide an atom group into two subgroups? We would wish to minimize the number of edges between the two subgroups. This is because when we attempt to localize the first subgroup of atoms, the edges with atoms in the second subgroup are lost. On the other hand, we wish to maximize the number of edges within a subgroup. The more edges within a subgroup, the more constraints on the atoms, and the more likely that the subgroup is localizable.

How should we join the two localized subgroups together to localize the atom group? Our strategy is for the two subgroups to have overlapping atoms. If the overlapping atoms are accurately localized in the estimated configurations, then they can be used to align the two subgroup configurations. If the overlapping atoms are *not* accurately localized, it would be disasterous to use them in aligning the two subgroup configurations. Therefore, DISCO incorporates a criterion for determining when the overlapping atoms are accurately localized.

It is important to realize that not all the atoms in a group may be localizable, for instance, some atoms may have fewer than four neighbors in that group. This must be taken into account when we are aligning two subgroup configurations together. If a significant number of the overlapping atoms are not localizable in either of the subgroups, the alignment may be highly errornous (see Figure 4). This problem could be avoided if we identify and discard the unlocalizable atoms in a group A heuristic algorithm is used by DISCO to identify atoms which are likely to be unlocalizable.

The pseudocode of the DISCO algorithm is presented in Algorithm 1. We illustrate how the DISCO algorithm solves a small molecule in Figure 1.

4.2 Recursive Case: How to Split and Combine

4.2.1 Partitioning into Subgroups

Before we discuss DISCO's partitioning procedure, we briefly describe the procedure used by DISCO's parent, the DAFGL algorithm [2]. The DAFGL algorithm partitions the set of atoms into consecutive subgroups, such that consecutive subgroups have overlapping atoms (see Figure 2). It then solves each subgroup separately, and combines the solutions together. Partitioning in DAFGL is done by repeatedly applying the symmetric reverse Cuthill-McKee (RCM) matrix permutation to submatrices of the distance matrix. The RCM permutation is specially designed to cluster the nonzero entries of a matrix (which in this case are the known distances) towards the diagonal. We observe in Figure 2 that many of the edges are not available to any subgroup, as they lie outside all the pink squares. We believe that DAFGL's partitioning procedure loses too many edges, and this is the reason why DAFGL performs poorly when the given distances are sparse, say less than 50% of pairwise distances less than 6 Å.

We hope that the above discussion has helped us to learn from our parents' mistakes; namely, in the design of DISCO's partitioning method, to make an extra effort to keep as many edges as possible.

Algorithm 1 The DISCO algorithm

procedure DISCO(L, U)
  if number of atoms $<$ basis size **then**
    $[cAest, cI] \leftarrow$ DISCOBASIS(L, U)
  else
    $[cAest, cI] \leftarrow$ DISCORECURSIVE(L, U)
  end if
  return $cAest, cI$
end procedure

procedure DISCOBASIS(L, U)
  $cI \leftarrow$ LIKELYLOCALIZABLECOMPONENTS(L, U)
  for $i = 1, \ldots,$LENGTH(cI) **do**
    $cAest\{i\} \leftarrow$ SDPLOCALIZE($cI\{i\}, L, U$)
    $cAest\{i\} \leftarrow$ REFINE($cAest\{i\}, cI\{i\}, L, U$)
  end for
  return $cAest, cI$
end procedure

procedure DISCORECURSIVE(L, U)
  $[L_1, U_1, L_2, U_2] \leftarrow$ PARTITION(L, U)
  $[cAest_1, cI_1] \leftarrow$ DISCO(L_1, U_1)
  $[cAest_2, cI_2] \leftarrow$ DISCO(L_2, U_2)
  $cAest \leftarrow [cAest_1, cAest_2]$
  $cI \leftarrow [cI_1, cI_2]$
  repeat
    $[cAest, cI] \leftarrow$ COMBINECHUNKS($cAest, cI$)
    $[cAest, cI] \leftarrow$ REFINE($cAest, cI, L, U$)
  until no change
  return $cAest, cI$
end procedure

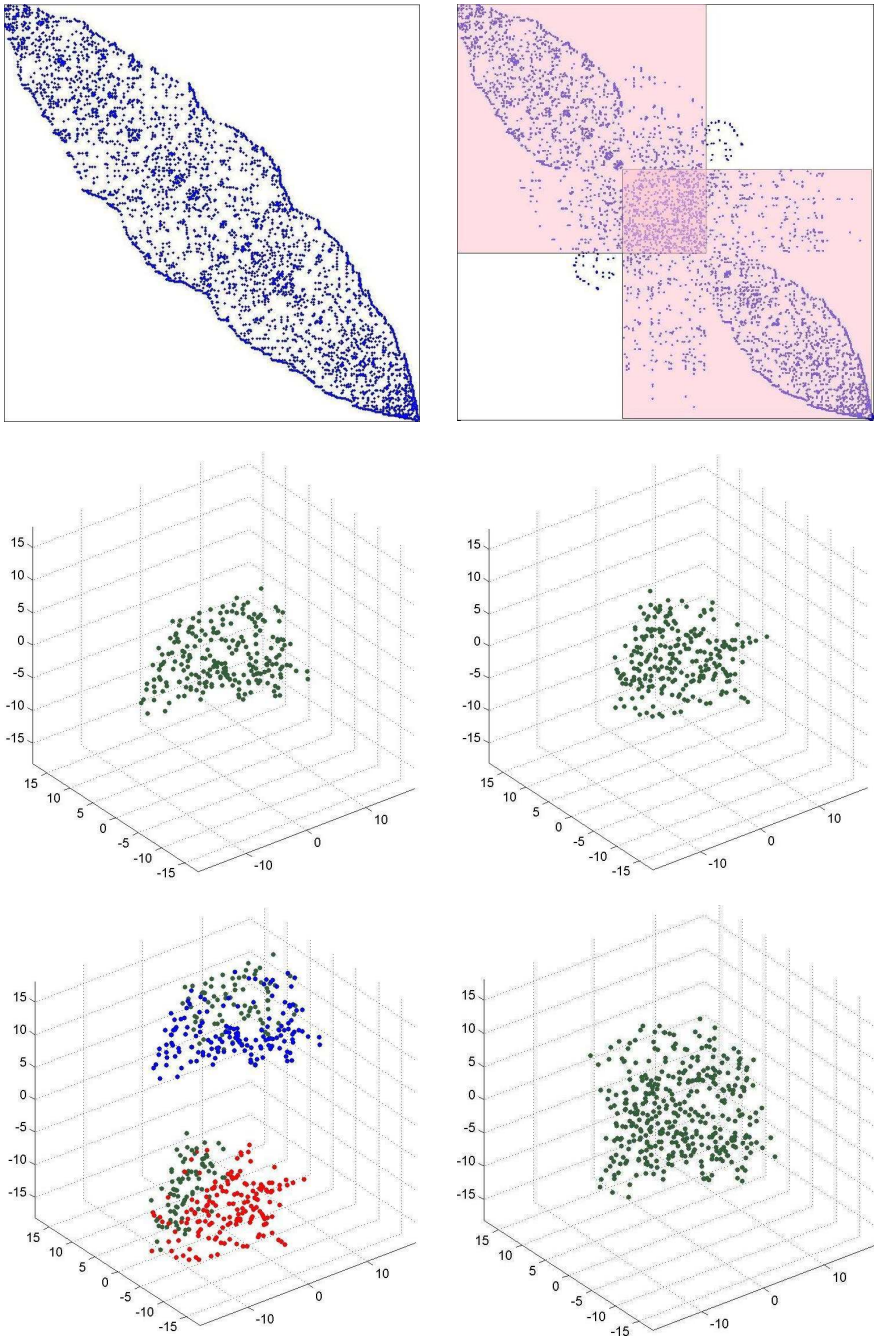

Figure 1: (top left and right) Since the number of atoms is too large ($n = 402 >$ basis size $= 300$), we divide the atoms into two subgroups. (middle left and right) We solve the subgroups independently. (bottom left) The subgroups have overlapping atoms, which are colored in green. (bottom right) The overlapping atoms allow us to align the two subgroups to form a realization of the molecule.

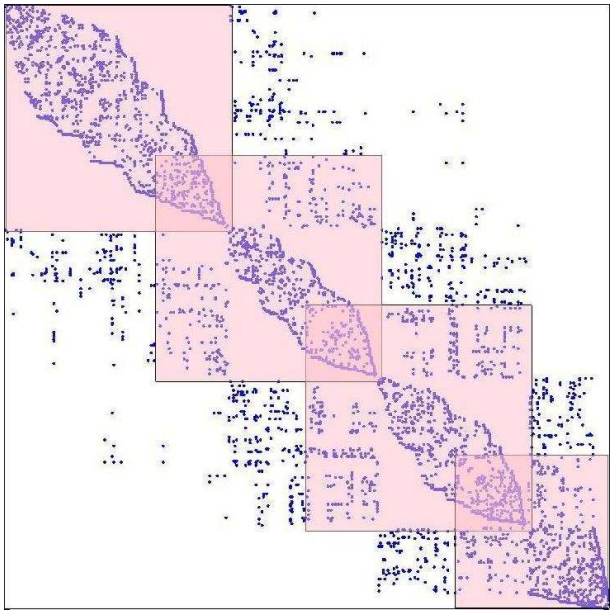

Figure 2: This permutation of the distance matrix illustrates DAFGL's partitioning strategy. The blue dots represent the known distances, and the pink squares represent the subgroups.

Suppose we wish to localize the set of atoms $\mathcal{A}$, but there are too many atoms in $\mathcal{A}$ for us to apply an SDP relaxation. We therefore divide $\mathcal{A}$ into two nonoverlapping subgroups $\mathcal{A}_1$ and $\mathcal{A}_2$. The two objectives in this division are that the number of edges between subgroups is approximately minimized, to maximize the chance that each subgroup will be localizable; and the subgroups are approximately equal in size, so that the recursive procedure will be fast.

However, it is not apparent, after localizing $\mathcal{A}_1$ and $\mathcal{A}_2$, how to combine them to form a configuration for $\mathcal{A}$. Our method is to make use of overlapping atoms between the subgroups. If the overlapping atoms are localized in both groups, then the two configurations can be aligned via a combination of translation, rotation, and refection. Of course, $\mathcal{A}_1$ and $\mathcal{A}_2$ were constructed to have no overlapping atoms. Thus we need to enlarge them to subgroups $\mathcal{B}_1 \supset \mathcal{A}_1, \mathcal{B}_2 \supset \mathcal{A}_2$ which have overlapping atoms. We construct $\mathcal{B}_i, i = 1, 2$, by adding some atoms $\tilde{\mathcal{A}}_{i\oplus 1} \subset \mathcal{A}_{i\oplus 1}$ to $\mathcal{A}_i$. (We define $\oplus$ by $1 \oplus 1 = 2, 2 \oplus 1 = 1$.) The set of atoms $\tilde{\mathcal{A}}_1, \tilde{\mathcal{A}}_2$ are auxilliary atoms added to $\mathcal{A}_1$ and $\mathcal{A}_2$ to create overlap. While $\mathcal{A}_1$ and $\mathcal{A}_2$ were constructed so as to *minimize* the number of edges $(i, j) \in \mathcal{N}$ with $i \in \mathcal{A}_1, j \in \mathcal{A}_2$; $\tilde{\mathcal{A}}_1$ and $\tilde{\mathcal{A}}_2$ are constructed so as to *maximize* the number of edges $(i, j) \in \mathcal{N}$ with $i \in \mathcal{A}_1, j \in \tilde{\mathcal{A}}_2$, and $(i, j) \in \mathcal{N}$ with $i \in \tilde{\mathcal{A}}_1, j \in \mathcal{A}_2$. The reason for this is that we want the set of atoms $\mathcal{B}_i = \mathcal{A}_i \cup \tilde{\mathcal{A}}_{i\oplus 1}, i = 1, 2$ to be localizable, so we want as many edges within $\mathcal{B}_1$ and $\mathcal{B}_2$ as possible.

We can succintly describe the partitioning as splitting A into two localizable groups $\mathcal{A}_1$ and $\mathcal{A}_2$, then growing $\mathcal{A}_1$ into $\mathcal{B}_1$ and $\mathcal{A}_2$ into $\mathcal{B}_2$ so that $\mathcal{B}_1$ and $\mathcal{B}_2$ are both likely to be localizable. The splitting step should minimize inter-group edges, to maximize the likelihood that $\mathcal{A}_1$ and $\mathcal{A}_2$ are localizable; while the growing step should maximize inter-group edges, to maximize the likelihood that $\tilde{\mathcal{A}}_2$ and $\tilde{\mathcal{A}}_1$ are localizable in $\mathcal{B}_1$ and $\mathcal{B}_2$.

To make our description more concrete, we give the pseudocode of the partition algorithm as Algorithm 2. The operation of the algorithm is also illustrated in Figure 3. We elaborate on the details of the pseudocode below. The PARTITION method consists of three stages: SPLIT, REFINE and OVERLAP.

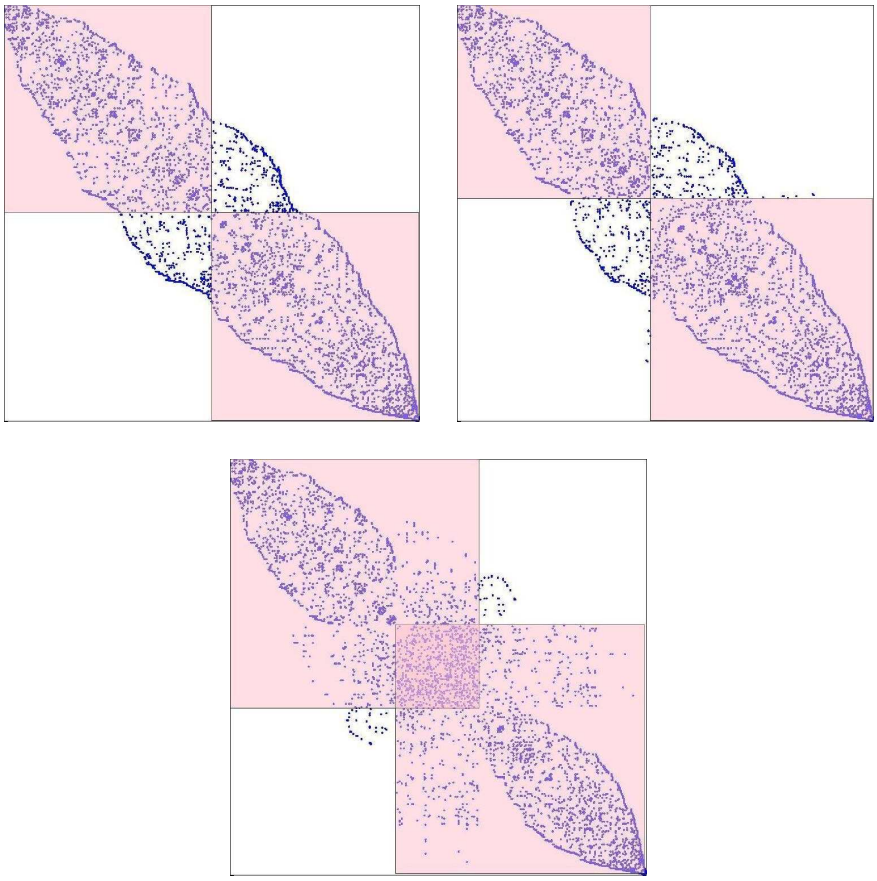

Figure 3: (top left) An RCM permutation gives us a balanced cut, with few cross-edges (number of cross-edges = 317). (top right) Refining the split reduces the number of cross-edges (number of cross-edges = 266). (bottom) Expanding the nonoverlapping subgroups $\mathcal{A}_1, \mathcal{A}_2$ into the overlapping subgroups $\mathcal{B}_1, \mathcal{B}_2$.

In the SPLIT method, we use compute the RCM permutation $\boldsymbol{p}$ of the rows and columns of the distance matrix D, that approximately minimizes the bandwidth of the matrix $D(\boldsymbol{p}, \boldsymbol{p})$. This is conveniently implemmented as the `symrcm`

command in MATLAB. The RCM permutation has the effect of clustering the edges towards the main diagonal, so that if we split the matrix with a vertical cut and horizontal cut through the center of the matrix, then the majority of the edges are in the (1,1) and (2,2) blocks, and only a few of the edges are in the (1,2) and (2,1) blocks. Thus if we select $\mathcal{A}_1 = \boldsymbol{p}(1 : \lfloor n/2 \rfloor)$ and $\mathcal{A}_2 = \boldsymbol{p}(\lfloor n/2 \rfloor + 1 : n)$, this approximately minimizes the number of edges from $\mathcal{A}_1$ to $\mathcal{A}_2$ (see Figure 3, top left).

In the REFINE method, we can reduce the number of inter-group edges as follows: If an atom $a \in \mathcal{A}_1$ is "closer" to $\mathcal{A}_2$ than $\mathcal{A}_1$, then switch it over to $\mathcal{A}_2$. An atom is "closer" to group $\mathcal{A}_1$ rather than group $\mathcal{A}_2$ if one of the two conditions hold:

1. it has more neighbors in group $\mathcal{A}_1$;
2. it has the same number of neighbors in groups $\mathcal{A}_1$ and $\mathcal{A}_2$, and its closest neighbor is in $\mathcal{A}_1$.

In the OVERLAP method, we compute $\mathcal{B}_1$ and $\mathcal{B}_2$. We begin by setting $\mathcal{B}_i$ to $\mathcal{A}_i$, then add to $\mathcal{B}_i$ the atom a not in $\mathcal{B}_i$ that is closest to $\mathcal{B}_i$, repeating until $\mathcal{B}_i$ is of the desired number of atoms.

4.2.2 Alignment of Atom Groups

Here we describe how to combine the computed configurations for $\mathcal{B}_1$ and $\mathcal{B}_2$ to form a configuration for $\mathcal{A}$. We shall adopt the following notation to facillitate our discussion. Let B_1, B_2 be the cooordinates for the atoms in $\mathcal{B}_1, \mathcal{B}_2$ respectively, and let C_1, C_2 be the coordinates for the overlapping atoms in $\mathcal{B}_1, \mathcal{B}_2$ respectively. If a is an atom in $\mathcal{A}$, then let $\boldsymbol{a}$ denote its coordinates in the configuration for $\mathcal{A}$. If $a \in \mathcal{B}_1$ (resp. $a \in \mathcal{B}_2$), then let $\boldsymbol{b}_1$ (resp. $\boldsymbol{b}_2$) denote its coordinates in the configuration B_1 (resp. B_2).

The first method we used to produce a configuration for $\mathcal{A}$ was to consider the composition of translation, rotation and reflection T that best aligns C_2 to C_1. If a is in $\mathcal{B}_1$ but not in $\mathcal{B}_2$, then we set $\boldsymbol{a} = \boldsymbol{b}_1$; if a is not in $\mathcal{B}_1$ but is in $\mathcal{B}_2$, then we set $\boldsymbol{a} = T(\boldsymbol{b}_2)$; if a is in both $\mathcal{B}_1$ and $\mathcal{B}_2$, then we set $\boldsymbol{a} = (\boldsymbol{b}_1 + T(\boldsymbol{b}_2))/2$, the average of $\boldsymbol{b}_1$ and $T(\boldsymbol{b}_2)$. While this method is simple, it suffers from the disadvantage that a few outliers can have a high degree of influence on the alignment. If a significant number of the overlapping atoms are poorly localized, the alignment may be destroyed.

The method used by DISCO is slightly more sophisticated, so as to be more robust. Our strategy is to use an iterative alignment process. If we again let T

Algorithm 2 The partition algorithm

procedure PARTITION(D)
  $[\mathcal{A}_1, \mathcal{A}_2] \leftarrow$ SPLIT(D)
  $[\mathcal{A}_1, \mathcal{A}_2] \leftarrow$ REFINE($D, \mathcal{A}_1, \mathcal{A}_2$)
  $[\mathcal{B}_1, \mathcal{B}_2] \leftarrow$ OVERLAP($D, \mathcal{A}_1, \mathcal{A}_2$)
  return $\mathcal{B}_1, \mathcal{B}_2$
end procedure
procedure SPLIT(D)
  $P \leftarrow$ SYMRCM(D)
  $\mathcal{A}_1 \leftarrow \boldsymbol{p}(1 : \lfloor \frac{n}{2} \rfloor)$, $\mathcal{A}_2 \leftarrow \boldsymbol{p}(\lfloor \frac{n}{2} \rfloor + 1 : n)$
  return $\mathcal{A}_1, \mathcal{A}_2$
end procedure
procedure REFINE($D, \mathcal{A}_1, \mathcal{A}_2$)
  for $i = 1, 2$ **do**
    while exists $a \in \mathcal{A}_i$ closer to $\mathcal{A}_{i\oplus 1}$ **do**
      $\mathcal{A}_i \leftarrow \mathcal{A}_i \setminus \{a\}$
      $\mathcal{A}_{i\oplus 1} \leftarrow \mathcal{A}_{i\oplus 1} \cup \{a\}$
    end while
  end for
end procedure
procedure OVERLAP($D, \mathcal{A}_1, \mathcal{A}_2$)
  for $i = 1, 2$ **do**
    repeat
      $a \leftarrow$ the closest point to $\mathcal{A}_i$ that is not in $\mathcal{A}_i$
      $\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \{a\}$
    until $\mathcal{A}_i$ is of desired size
  end for
end procedure

be the composition of translation, rotation and reflection that best aligns C_2 to C_1, we find the overlapping point a such that the distance between its position in the two configurations $\|\boldsymbol{b}_1 - T(\boldsymbol{b}_2)\|$ is greatest. If it is greater than two times the mean distance over all overlapping points, then it is likely that this point is not accurately localized in either of the two configurations, so that we remove this outlier point, and repeat the process; if the distance is less than two times the mean distance over all overlapping points, then we conclude that this point is not an outlier, and we this T may give us a good alignment. By discarding points whose coordinates do not agree, it is hoped that our alignment only uses points that are well-localized in both groups. The T obtained from discarding outlier points goes through a second test. If the alignment of the remaining overlapping points has high error, that is if the RMSD is greater than a certain threshold, this indicates that it is not possible to accurately align $\mathcal{B}_1$ and $\mathcal{B}_2$ together; if the RMSD is not too great, then we will proceed to align $\mathcal{B}_1$ and $\mathcal{B}_2$.

4.3 Basis Case: Localizing An Atom Group

4.3.1 When DISCO Fails

A prototype of DISCO was able to accurately localize certain molecules, but would produce high-error structures for other molecules. Usually, the root of the problem was that the configuration for one particular subgroup had high error. Unfortunately, aligning a good configuration and a bad configuration often produces a bad configuration, so that the error propogates up to the complete protein configuration (see Figure 4).

What are the reasons for some atom groups to be badly localized? The first reason is rather obvious—some of the atoms may only have three or fewer neighbors and so are not uniquely localizable in general. The second reason is more subtle. When we plotted the estimated positions against the true positions, we noticed that the badly localized groups often consisted of two subgroups; each subgroup was well localized relative to itself, but there were not many edges between subgroups, so that the subgroups were not well-positioned relative to each other.

4.3.2 Identifying a Likely-localizable Core

If one subgroup is poorly localized, the complete protein configuration could be destroyed. Thus we must make an extra effort to ensure that we are able to accurately localize each subgroup.

Here we make a slight digression to a related result. In the case when exact distances are given, Hendrickson [11] established sufficient conditions for unique localizability. These conditions are not of great import to us, and so we give only a flavor of the conditions: (1) vertex 4-connectivity, (2) redundant rigidity—the graph is rigid after the removal of any edge, (3) stress testing—the null space of the so-called "stress matrix" has dimension 4.

Unfortunately, Hendrickson's results, while interesting, are not applicable to our situation. Imagine a conformation that is kept rigid by a set of edges, which are constrained to be of specified distances. If the specified distances are relaxed to distance bounds, it is possible that the conformation will have freedom to flex or bend into a shape that is drastically different from the original. The lesson from this exercise of our imagination is that to get a good localization with noisy distances requires stricter conditions than to get a good localization with exact distances.

To ensure that we can get a good localization of a group, we may have to discard some of the atoms, or split the group into several subgroups (see §4.3.1). Atoms with fewer than 4 neighbors should be removed, because we have no hope of localizing them accurately. We should also check if it is possible to split the atoms into two subgroups, both larger than the `MinSplitSize`[3], which have fewer than `MinCrossEdges` edges between them. If this were the case, it may not be possible to localize both subgroups together accurately, but if we split the subgroups, it may be possible to localize them accurately. The exact choice of these parameters are a matter of personal taste, but we have found that the value of 20 for `MinSplitSize` and 50 for `MinCrossEdges` seems to work well in practice. With regard to our choice for `MinCrossEdges`, in the case of exact distances, in general 6 edges are needed to align two rigid subgroups. However, in our case the distance data may be very noisy, so we may need many more edges to align the two groups well. This is why the rather conservative value of 50 is chosen.

How should we split the atoms into two subgroups, both subgroups with at least `MinSplitSize` atoms, so that there are as few edges between them as possible? This problem is familiar to us, because it is similar to the partitioning problem that has been discussed in §4.2.1. Of course, in the partitioning problem, we would like the two subgroups to have approximately the same size; while here we would like both subgroups to have at least `MinSplitSize` atoms. Nevertheless, the similarity of the two problems suggests that we could learn from

[3]We are looking for two rigid subgroups, which have few edges between them. The rigid subgroups should not be a very small group of atoms.

the partitioning approach. DISCO find the approximate minimum split by first applying the RCM permutation to permute the rows and columns of the distance matrix and cluster the nonzero entries towards the diagonal. It then tries values of p from `MinSplitSize` to $n - $ `MinSplitSize` $+ 1$, to see which is the cut such that the number of edges between atoms $1 : p$ and $(p + 1) : n$ is minimized (see Figure 5).

Again, to make our ideas more concrete, we present the pseudocode of DISCO's localizable components algorithm in Algorithm 3.

Algorithm 3 Computing the likely-localizable core

```
procedure LocalizableComponents(A)
    Remove atoms with fewer than 4 neighbors from A
    [nCrossEdges, A1, A2] ← MinSplit(A)
    if nCrossEdges < MinCrossEdges then
        cI1 ← LocalizableComponents(A1)
        cI2 ← LocalizableComponents(A2)
        return [cI1, cI2]
    else
        return A
    end if
end procedure
procedure MinSplit(A)
    p ← SymRcm(D)
    for i = MinSplitSize, ..., n − MinSplitSize − 1 do
        nCrossEdges{i} ← nCrossEdges(D, p(1 : i), p(i + 1 : n))
    end for
    i ← MinIndex(nCrossEdges)
    nCrossEdges ← nCrossEdges{i}
    A1 ← A(1 : i)
    A2 ← A(i + 1 : n)
end procedure
```

5 Numerical Experiments

In §5.1, we explain computational issues in the DISCO algorithm. In §5.2, we present the experimental setup. In §5.3, we discuss the numerical results.

5.1 Computational Issues

5.1.1 SDP Localization

In Section 3, we presented the "measured distances" and "distance bounds" SDP models for the graph realization problem. We now have to decide which model is more suitable for DISCO. In particular, we will compare the two models in terms of the running time and the accuracy of the computed configuration.

We decided to use the "measured distances" model for DISCO, because the running time is superior, while the accuracy is comparable to that of the "distance bounds" model. With regards to the running time, DISCO uses the software SDPT3 to solve the SDPs arising from the graph realization problems. The running time of SDPT3 is of the order of $O(mn^3) + O(m^2n^2) + \Theta(m^3) + \Theta(n^3)$, where m is the number of constraints and n is the dimension of the SDP matrix. In our case, m corresponds to the number of distances/bounds, and n corresponds to the number of atoms. The "distance bounds" model has (roughly) twice as many constraints as the "measured distances" model, and in practice, it may be 3–8 times slower on the same input.

In the "measured distances" model, the regularization parameter γ has to be chosen judiciously. The regularization paramter affects the configuration in the following way: the larger the regularization parameter, the more separated the computed configuration. In the extreme case when the regularization parameter is very large, the regularization term will dominate the distance error term to the extent that the objective value goes to minus infinity because the atoms move as far apart as possible rather than fitting the distance constraints.

We have found that the value $\gamma = \bar{\gamma} := m/25n$ seems to work well in practice. We present our intuition for choosing such a value in the following. It is expected that if the value of the distance terms

$$\sum_{(i,j)\in\mathcal{N}^s} u_{ij}^+ + u_{ij}^-$$

and the value of the separation term

$$\gamma\langle I - E/n, Y\rangle$$

in (13) are balanced, the computed configuration will neither be too compact nor too separated. Note that

$$\langle I - E/n, Y\rangle \approx \langle I, Y\rangle,$$

since one of the constraints in (13) is that $\langle E, Y \rangle = 0$. If we let r denote the half-diameter of the chunk, and we make the very crude estimates

$$u_{ij}^{+} + u_{ij}^{-} \approx r^2/25, \qquad Y_{ii} \approx r^2,$$

then this gives rise to the choice of $\gamma = m/25n$. The factor m/n comes from that there are m edges and n diagonal terms. In our numerical experiments, we have found that values $\gamma \in (\frac{1}{4}\bar{\gamma}, 4\bar{\gamma})$ seem to work well in practice, so the SDP model works well for a reasonably wide range of γ.

It would be useful to be able to quantify how "separated" an estimated configuration is, compared to the true configuration. We thus define the *separation* of a computed configuration as

$$\sigma(X) = \frac{1}{|\mathcal{N}^s|} \sum_{(i,j)\in\mathcal{N}^s} \frac{\|\boldsymbol{s}_i - \boldsymbol{s}_j\|}{\|\boldsymbol{s}_i^{\text{true}} - \boldsymbol{s}_j^{\text{true}}\|}.$$

Note that computing the separation requires us to know the true configuration. However, we would not normally have the true configuration available. It is more appropriate then to use the *approximate separation* of a computed configuration

$$\tau(X) = \frac{1}{|\mathcal{N}^s|} \sum_{(i,j)\in\mathcal{N}^s} \frac{\|\boldsymbol{s}_i - \boldsymbol{s}_j\|}{(\tilde{d}_{ij}^s)^2}. \tag{18}$$

The approximate separation of the computed configuration may indicate that the SDP should be resolved with a different regularization parameter. If the approximate separation indicates that the computed solution is "too compact" ($\tau(X) < 0.8$), then resolving the SDP with a larger γ (doubling γ) may produce a "more separated" configuration that is closer to the true configuration. Similarly, if the computed solution is "too separated" ($\tau(X) > 1.1$), then resolving the SDP with a smaller γ (halving γ) may produce a more accurate configuration.

The inclusion of minimum separation distance (MSD) constraints can also help us to compute a better configuration from the SDP model. Due to physical reasons, there is an MSD between any two atoms i and j, which we shall denote by α_{ij}. After solving the SDP (13), we check to see if the minimum separation condition

$$\|\boldsymbol{s}_i - \boldsymbol{s}_j\|^2 \approx Y_{ii} + Y_{jj} - 2Y_{ij} \geq \alpha_{ij}^2$$

is satisfied for all pairs (i, j). If this is not true, then we let $\mathcal{E}$ be the set of pairs (i, j) which violate the condition. We then resolve the SDP, with the additional

constraints

$$Y_{ii} + Y_{jj} - 2Y_{ij} \geq \alpha_{ij}^2, \quad \forall (i,j) \in \mathcal{E}.$$

We observed that imposing the minimum separation constraint improves the quality of the SDP solution. While it was reported in [25, p. 526] that the minimum separation constraints pose a significant global optimization challenge for molecular structure determination, we believe that the minimum separation constraints may in fact be advantageous for finding a lower rank SDP solution from (13).

In this paper, we set the minimum separation distance α_{ij} to 1 Åuniformly, regardless of the types of the i-th and j-th atoms. In a more realistic setting, it is desirable to set α_{ij} as the sum of the van der Waals radii of atoms i, j if they are not covalently bonded.

5.1.2 Gradient Descent

We have found that a regularized gradient descent refinement performs better than the nonregularized counterpart. Recall that the atom coordinates obtained via SDP localization are obtained by projecting Y onto the space of rank-d matrices. This tends to give rise to a configuration that is "too compact", because the discarded dimensions may make significant contributions to some of the pairwise distances. Introducing a separation term in the objective function may induce the atoms to spread out appropriately.

Here we describe the regularized gradient descent. Let us denote the initial iterate by $X^0 = [\boldsymbol{s}_1^0, \ldots, \boldsymbol{s}_n^0]$, which we will assume is centered at the origin. The regularized objective function is

$$\min f(X) := \sum_{(i,j)\in\mathcal{N}^s} \left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\| - \tilde{d}_{ij}^s\right)^2 - \mu \sum_{i=1}^n \|\boldsymbol{s}_i\|^2, \tag{19}$$

where $\mu > 0$ is a regularization parameter. Typically, a choice of

$$\mu = \frac{\sum_{(i,j)\in\mathcal{N}^s} \left(\|\boldsymbol{s}_i^0 - \boldsymbol{s}_j^0\| - \tilde{d}_{ij}^s\right)^2}{10 \sum_{i=1}^n \|\boldsymbol{s}_i^0\|^2}$$

works well in practice.

We remark that choosing a suitable maximum number of iterations and tolerance level to terminate the gradient descent can significantly reduce the computational time of the gradient descent component of DISCO.

5.2 Experimental Setup

The DISCO source code was written in MATLAB, and is freely available from the DISCO website [12]. DISCO uses the SDPT3 software package of Toh, Todd and Tütüncü [20, 24, 19] to solve SDPs arising from graph realization.

Our experimental platform was a dual-processor machine (2.40GHz Intel Core2 Duo processor) with 4GB RAM, running MATLAB version 7.3, which only runs on one core.

We tested DISCO using input data obtained from a set of 12 molecules taken from the Protein Data Bank (PDB). The conformation of these molecules is already known, so that our computed conformation can be compared with the true conformation.

The sparsity of the inter-atom distances was modeled by choosing at random a proportion of the short-range inter-atom edges, subject to the condition that the distance graph is connected[4]. It is important to note that the degree of some atoms may be less than 4, so that they are not localizable, but we do not discard these atoms. We have chosen to define short-range inter-atom distances as those less than 6 Å. The "magic number" of 6 Å was selected because NMR techniques are able to give us information about the distance between some pairs of atoms if they are less than approximately 6 Å apart. We have adopted this particular input data model because it is simple and fairly realistic [25, 2].

In realistic molecular conformation problems the exact inter-atom distances are not given, but only lower and upper bounds on the inter-atom distances are known. Thus after selecting a certain proportion of short-range inter-atom distances, we add noise to the distances to give us lower and upper bounds. In this paper, we have experimented with "normal" and "uniform" noise. The noise level is specified by a parameter ν, which indicates the expected value of the noise. When we say we have a noise level of 20%, what that means is that $\nu = 0.2$. In the normal noise model, the bounds are specified by

$$\underline{d}_{ij} = \max\big(1, (1 - |\underline{Z}_{ij}|)d_{ij}\big), \quad \overline{d}_{ij} = (1 + |\overline{Z}_{ij}|)d_{ij},$$

where $\underline{Z}_{ij}, \overline{Z}_{ij}$ are independent normal random variables with zero mean and standard deviation $\nu\sqrt{\pi/2}$. In the uniform noise model, the bounds are specified by

$$\underline{d}_{ij} = \max\big(1, (1 - |\underline{Z}_{ij}|)d_{ij}\big), \quad \overline{d}_{ij} = (1 + |\overline{Z}_{ij}|)d_{ij},$$

where $\underline{Z}_{ij}, \overline{Z}_{ij}$ are independent uniform random variables in the interval $[0, 2\nu]$.

[4]The interested reader may refer to the code for the details of how the selection is done.

We have defined the normal and uniform noise models in such a way that for both noise models, the expected value of $|\underline{Z}_{ij}|, |\overline{Z}_{ij}|$ is ν.

In addition to the lower and upper bounds, which are available for only some pairwise distances, we have minimum separation distances (MSDs) between all pairs of atoms. Due to physical reasons, two atoms i and j must be separated by an MSD α_{ij}, which depends on particular details such as the type of the pair of atoms (e.g. C-N, N-O), whether they are covalently bonded, etc. The MSD gives a lower bound for the distance between the two atoms. As mentioned in the previous subsection, for simplicity, we set the minimum separation distance to be uniformly 1 Å, regardless of the types of atoms.

The error of the computed configuration is measured by the root mean square deviation (RMSD). If the computed configuration X is optimally aligned to the true configuration X^*, using the procedure of §3.4, then the RMSD is defined by the following formula

$$\text{RMSD} = \frac{1}{\sqrt{n}} \left(\sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{x}_i^*\|^2 \right).$$

The RMSD basically measures the "average" deviation of the computed atom positions to the true atom positions.

5.3 Results and Discussion

To help the reader to appreciate the difficulty of the molecular conformation problem, under the setup we have just described, we solved two of the smaller molecules using sparse but exact distances. This information is presented in Table 2. Even if we solve a molecular problem in a centralized fashion, due to the sparsity of the distance data, the problem is not localizable, we can only get an approximate solution.

The performance of DISCO is listed in Tables 3 and 4, for the case when teh initial random number seed is set to zero, i.e.

```
randn('state',0); rand('state',0);
```

The RMSD plots across the molecules, with 10 runs given different initial random number seeds, is shown in Figure 6.

When given 30% of the short-range distances, corrupted by 20% noise, DISCO is able to compute an accurate structure. we have a final structure (core structure) with an RMSD of 0.9–1.6 Å (0.6–1.6 Å). The core structure is the union of the

likely-localizable components. Typically, the core structure is solved to a slightly higher accuracy, though there are a few exceptions to this.

We believe these are the best numbers which we could hope for, and we present an intuitive explanation of why this is so. For simplicity, let us assume that the mean distance of any given edge is 3 Å. This is reasonable because the maximum given distance is about 6 Å. Given 20% noise, we give a bound of about 2.4–3.6 Å for that edge. Thus each atom can move about 1.2 Å. The RMSD should therefore be approximately 1.2 Å.

When given 20% of the short-range distances, the conformation problems become more difficult, due to the sparsity of available distances. For each problem, the mean degree of each atom is 7.4–8.6, so the data is highly sparse. We set a lower level of 10% noise for these experiments. Even under such challenging input, DISCO is still able to produce a fairly accurate structure ($\approx$ 2 Å) for all the molecules except 1RGS and 1I7W ($\approx$ 3.5 Å).

In Figure 6, we plot the RMSDs for different random inputs of the same molecule. The graphs indicate that DISCO is able to produce an accurate conformation ($<$ 3 Å) for most of the molecules over different random inputs. DISCO does not perform so well on the two molecules 1RGS and 1I7W, which have less regular geometries, Although we have designed DISCO with safeguards, DISCO will nevertheless occasionally make mistakes in aligning sub-configurations.

6 Conclusion and Future Work

We have proposed a novel divide-and-conquer, SDP-based algorithm for the molecular conformation problem. Our numerical experiments demonstrate that DISCO is able to solve very sparse and highly noisy problems accurately, in a short amount of time. The largest molecule with more than 13000 atoms was solved in about one hour to an RMSD of 1.6 Å, given only 30% of pairwise distances less than 6 Å and corrupted by 20% multiplicative noise.

We hope that with the new tools and ideas developed in this paper, we will be able to tackle molecular conformation problems with realistic input data, as was done in [10].

Algorithm(s)	Largest molecule (No. of atoms)	Inputs	Output
EMBED (83), DISGEO (84), DG-II (91), APA (99)	454	All distance and chirality constraints needed to fix the covalent structure are given exactly. Some or all of the distances between hydrogen atoms less than 4 Å apart and in different amino acid residues given as bounds.	RMSD 2.08 Å
DGSOL (99)	200	All distances between atoms in successive residues given as lying in $[0.84d_{ij}, 1.16d_{ij}]$.	RMSD 0.7 Å
GNOMAD (01)	1870	All distances between atoms that are covalently bonded given exactly; all distances between atoms that share covalent bonds with the same atom given exactly; additional distances given exactly, so that 30% of the distances less than 6 Å are given; physically inviolable minimum separation distance constraints given as lower bounds.	RMSD 1.07 Å(*)
MDS (02)	700	All distances less than 7 Å were given as lying in $[d_{ij} - 0.01\text{Å}, d_{ij} + 0.01\text{Å}]$.	violations < 0.01 Å
StrainMin (06)	5147	All distances less than 6 Å are given exactly, a representative lower bound of 2.5 Å is given for other pairs of atoms.	violations < 0.1 Å
ABBIE (95)	1849	All of distances between atoms in the same amino acid given exactly. All distances corresponding to pairs of hydrogen atoms less than 3.5 Å apart from each other, given exactly.	Exact
Geometric build-up (07)	4200	All distances between atoms less than 5 Å apart from each other given exactly.	Exact
DAFGL (07)	5681	70% of the distances less than 6 Å were given as lying in $[\underline{d}_{ij}, \overline{d}_{ij}]$, where $\underline{d}_{ij} = \max(0, (1 - 0.05\|\underline{Z}_{ij}\|)d_{ij}), \overline{d}_{ij} = (1 + 0.05\|\overline{Z}_{ij}\|)d_{ij}$, and $\underline{Z}_{ij}, \overline{Z}_{ij}$ are standard normal random variables with zero mean and unit variance.	RMSD 3.16 Å

Table 1: A summary of molecular conformation algorithms. (*) The RMSD reported by GNOMAD may be incorrect, and the true value should be about 2–3 Å. The number reported in Figure 11 of [25] does not agree with that which appears in Figure 8.

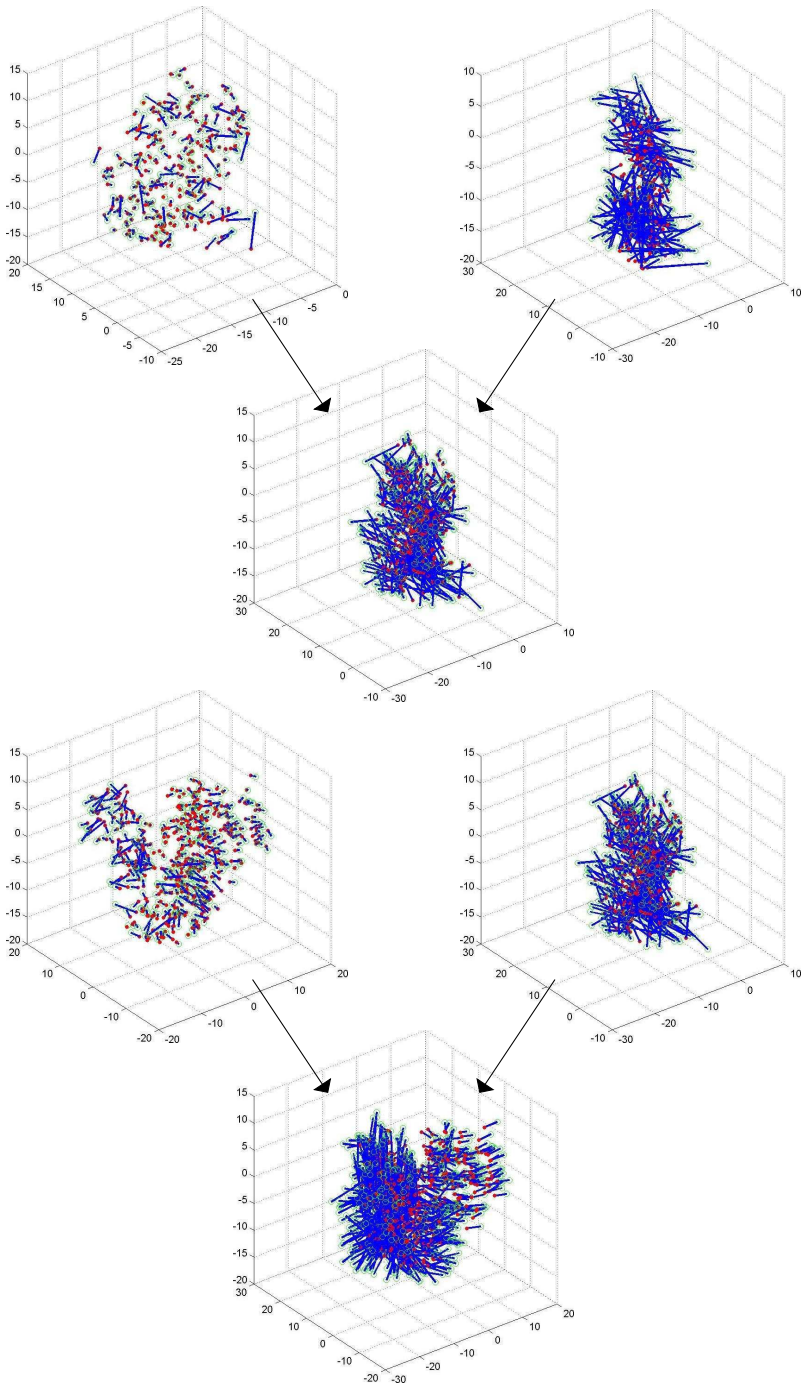

Figure 4: In each atom plot, a circle represents a true atom position, the red dot represents an estimated atom position, and the blue line joins the corresponding true and estimated atom positions. In this figure, we show how two subgroup configurations (the arrow tails) are aligned to produce a configuration for a larger group (where the arrow heads point). In this example, because one subgroup configuration is poorly localized, the resulting configurations formed from this poorly localized configuration are also unsatisfactory.

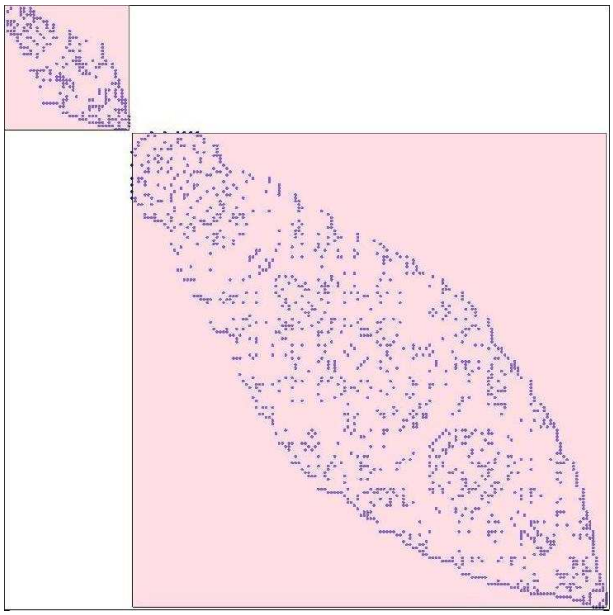

Figure 5: Finding the cut that minimizes the number of edges between subgroups.

Input data: exact distances $\leq$ 6 Å					
Molecule	n	30% distances		20% distances	
		RMSD (Å)	ℓ	RMSD (Å)	ℓ
1GM2	166	0.10	0	0.83	10
1PTQ	402	0.39	9	1.16	38

Table 2: The molecular problem with sparse but exact distance data cannot always be solved exactly. We have denoted by n the number of atoms in the molecule and by ℓ the number of atoms with degree less than 4.

Input data: 30% distances ≤ 6 Å, corrupted by 20% noise								
Molecule	n	ℓ	RMSD (Å)				Time (h:m:s)	
			Normal		Uniform		Normal	Uniform
1GM2	166	0	0.92	(0.94)	0.74	(0.76)	00:00:08	00:00:13
1PTQ	402	9	1.08	(0.96)	1.00	(0.85)	00:00:23	00:00:18
1PHT	814	15	1.45	(0.69)	1.15	(0.56)	00:01:22	00:01:00
1AX8	1003	16	1.35	(1.17)	1.00	(0.80)	00:01:31	00:01:07
1TIM	1870	45	1.23	(1.03)	0.94	(0.80)	00:04:18	00:03:28
1RGS	2015	37	1.52	(1.36)	1.51	(1.41)	00:05:25	00:03:53
1KDH	2923	48	1.38	(1.16)	1.21	(0.89)	00:07:57	00:05:30
1BPM	3672	36	1.10	(1.03)	0.79	(0.73)	00:11:24	00:08:08
1TOA	4292	62	1.15	(1.07)	0.89	(0.78)	00:13:25	00:09:06
1MQQ	5681	46	0.92	(0.86)	0.82	(0.74)	00:23:56	00:17:24
1I7W	8629	134	2.45	(2.34)	1.51	(1.40)	00:40:39	00:31:26
1YGP	13488	87	1.92	(1.93)	1.50	(1.52)	01:20:35	01:00:55

Table 3: We have denoted by ℓ the number of atoms with degree less than 4. The mean degree of an atom is 10.8–12.6. The approximate core of the structure consists typically of 94–97% of the total number of atoms. For large molecules, the SDP localization consumes about 70% of the running time, while the gradient descent consumes about 20% of the running time.

Input data: 20% distances ≤ 6 Å, corrupted by 10% noise								
Molecule	n	ℓ	RMSD (Å)				Time (h:m:s)	
			Normal		Uniform		Normal	Uniform
1GM2	166	7	1.44	(1.25)	0.92	(0.77)	00:00:04	00:00:04
1PTQ	402	46	1.49	(1.17)	1.48	(1.14)	00:00:14	00:00:10
1PHT	814	53	1.53	(1.13)	1.40	(0.97)	00:01:02	00:00:54
1AX8	1003	78	1.69	(1.40)	1.52	(1.17)	00:01:00	00:00:55
1TIM	1870	143	1.77	(1.41)	1.84	(1.50)	00:02:43	00:02:33
1RGS	2015	189	9.82	(9.51)	1.83	(1.39)	00:03:32	00:03:14
1KDH	2923	210	1.74	(1.31)	1.63	(1.13)	00:04:28	00:04:45
1BPM	3672	187	1.46	(1.14)	1.31	(0.91)	00:06:52	00:06:33
1TOA	4292	251	1.67	(1.26)	1.58	(1.16)	00:08:39	00:08:49
1MQQ	5681	275	1.17	(0.95)	1.17	(0.92)	00:14:31	00:14:21
1I7W	8629	516	4.04	(3.69)	3.87	(3.52)	00:26:52	00:26:04
1YGP	13488	570	1.83	(1.63)	1.70	(1.46)	01:01:21	00:57:31

Table 4: We have denoted by ℓ the number of atoms with degree less than 4. The mean degree of an atom is 7.4–8.6. The approximate core of the structure consists typically of 88–92% of the total number of atoms. For large molecules, the SDP localization consumes about 60% of the running time, while the gradient descent consumes about 30% of the running time.

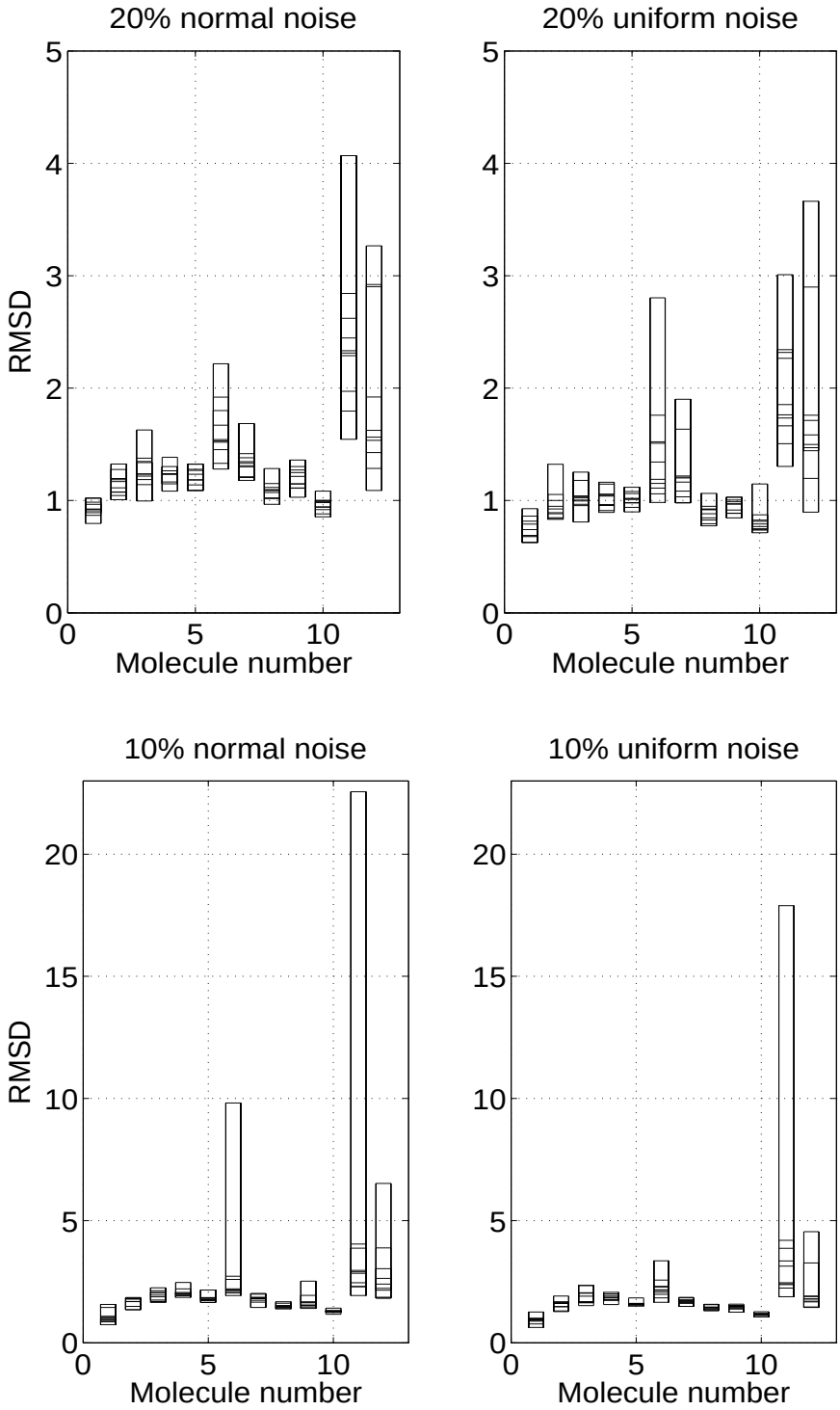

Figure 6: For each molecule, ten random inputs were generated with different initial random number seeds. We plot the the RMSDs of the ten structures produced by DISCO against the molecule number. (top left) 30% short-range distances, 20% normal noise (top right) 30% short-range distances, 20% uniform noise (bottom left) 20% short-range distances, 10% normal noise (bottom right) 20% short-range distances, 10% uniform noise

References

[1] P. Biswas, T.-C. Liang, K.-C. Toh, T.-C. Wang, and Y. Ye, *Semidefinite programming approaches for sensor network localization with noisy distance measurements*, IEEE Trans. on Auto. Sci. and Eng., 3 (2006), pp. 360–371.

[2] P. Biswas, K.-C. Toh, and Y. Ye, *A distributed SDP approach for large scale noisy anchor-free graph realization with applications to molecular conformation*, 2008.

[3] P. Biswas and Y. Ye, *A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization*, tech. rep., Dept. of Management Sci. and Eng., Stanford University, Oct 2003.

[4] Q. Dong and Z. Wu, *A geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data*, Journal of Global Optimization, 26 (2003), pp. 321–333.

[5] W. Glunt, T. Hayden, S. Hong, and J. Wells, *An alternating projection algorithm for computing the nearest euclidean distance matrix*, SIAM J. of Mat. Anal. and Appl., 11 (1990), pp. 589–600.

[6] I. G. Grooms, R. M. Lewis, and M. W. Trosset, *Molecular embedding via a second-order dissimilarity parameterized approach*, December 2006. Submitted to SIAM Journal on Scientific Computing. Revised April 2007.

[7] O. Güler and Y. Ye, *Convergence behavior of interior point algorithms*, Math. Prog., 60 (1993), pp. 215–228.

[8] T. F. Havel, *A evaluation of computational strategies for use in the determination of protein structure from distance constraints obtained by nuclear magnetic resonance*, Progress is Biophysics and Molecular Bio., 56 (1991), pp. 43–78.

[9] T. F. Havel, I. D. Kuntz, and G. M. Crippen, *The combinatorial distance geometry approach to the calculation of molecular conformation*, J. of Theor. Biol., 104 (1983), pp. 359–381.

[10] T. F. Havel and K. Wüthrich, *A distance geometry program for determining the structures of small proteins and other macromolecules from*

nuclear magnetic resonance measurements of 1h-1h proximities in solution, Bull. of Math. Bio., 46 (1984), pp. 673–698.

[11] B. Hendrickson, *The molecule problem: exploiting structure in global optimization*, SIAM J. of Optimization, 5 (1995), pp. 835–857.

[12] N.-H. Z. Leung and K.-C. Toh, *The DISCO web page*. `http://deutsixfive.googlepages.com/disco.html`.

[13] M. Locatelli and F. Schoen, *Structure prediction and global optimization*, Optima (Mathematical Programming Society Newsletter), 76 (2008), pp. 1–8.

[14] J. J. Moré and Z. Wu, *Global continuation for distance geometry problems*, SIAM J. of Optimization, 7 (1997), pp. 814–836.

[15] ——, *Distance geometry optimization for protein structures*, J. on Global Optimization, 15 (1999), pp. 219–234.

[16] R. Reams, G. Chatham, W. Glunt, D. McDonald, and T. Hayden, *Determining protein structure using the distance geometry program apa*, Computers & Chemistry, 23 (1999), pp. 153–163.

[17] J. B. Saxe, *Embeddability of weighted graphs in k-space is strongly np-hard*, in Proceedings of the 17th Allerton Conference on Communication, Control, and Computing, 1979, pp. 480–489.

[18] A. M.-C. So and Y. Ye, *Theory of semidefinite programming for sensor network localization*, in SODA: Proceedings of the sixteenth annual ACM-SIAM symposium on discrete algorithms, 2005, pp. 405–414.

[19] K.-C. Toh, M. J. Todd, and R. H. Tutuncu, *The SDPT3 web page*. `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

[20] K.-C. Toh, M. J. Todd, and R. H. Tutuncu, *SDPT3—a MATLAB software package for semidefinite programming*, Optimization Methods and Software, 11 (1999), pp. 545–581.

[21] M. W. Trosset, *Applications of multidimensional scaling to molecular conformation*, Computing Science and Statistics, 29 (1998), pp. 148–152.

[22] ——, *Distance matrix completion by numerical optimization*, Computational Optimization and Applications, 17 (2000), pp. 11–22.

[23] ——, *Extensions of classical multidimensional scaling via variable reduction*, Computational Statistics, 17 (2002), pp. 147–163.

[24] R. H. Tutuncu, K.-C. Toh, and M. J. Todd, *Solving semidefinite-quadratic-linear programs using SDPT3*, Mathematical Programming Ser. B, 95 (2003), pp. 189–217.

[25] G. A. Williams, J. M. Dugan, and R. B. Altman, *Constrained global optimization for estimating molecular structure from atomic distances*, Journal of Computational Biology, 8 (2001), pp. 523–547.

[26] D. Wu and Z. Wu, *An updated geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data*, Journal of Global Optimization, 37 (2007), pp. 661–673.