Antonio Mucherino · Carlile Lavor
Leo Liberti · Nelson Maculan *Editors*

# Distance Geometry

## Theory, Methods, and Applications

Springer

# Distance Geometry

Antonio Mucherino • Carlile Lavor
Leo Liberti • Nelson Maculan
Editors

# Distance Geometry

## Theory, Methods, and Applications

Springer

*Editors*

Antonio Mucherino
IRISA, University of Rennes 1,
Rennes, France

Carlile Lavor
Department of Applied Maths
University of Campinas
Campinas-SP, Brazil

Leo Liberti
Laboratoire d'Informatique
Ecole Polytechnique
Palaiseau, France

Nelson Maculan
Alberto Luiz Coimbra Institute
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil

ISBN 978-1-4614-5127-3 ISBN 978-1-4614-5128-0 (eBook)
DOI 10.1007/978-1-4614-5128-0
Springer New York Heidelberg Dordrecht London

Library of Congress Control Number: 2012952099

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

*To our parents*

# Preface

The idea for this book was born during the Toulouse global optimization (TOGO) workshop in 2010, one in a series of global optimization workshops (GOWs) that are held all around the world every two or three years. We mentioned the idea to Panos Pardalos, the director of the Springer optimization and its applications book series, who found it good and told us to get in touch with the publisher.

Distance geometry (DG) sets the concept of distance on the basis of Euclidean geometry. The fundamental problem of DG is an inverse problem, i.e. finding a set of points in space whose pairwise distances are equal to some given distances. Among the most important applications we find molecular conformations, localization of sensor networks, and statics. Molecules consist of atoms; some pairwise distances are known either because of chemical reasons or because of nuclear magnetic resonance (NMR) experiments. The problem is then to compute 3D molecular structures using distance information. In mobile sensor networks, pairwise distances can be measured because the power required to establish a point-to-point communication is proportional to the distance: the problem faced by the network administrator is to locate all sensors in space. In statics, the interest is to construct bar-and-joint frameworks that serve a given architectural purpose whilst being rigid, i.e. whose static equilibrium is hard to break. Other applications, notably to data visualization, robotic arms, unmanned underwater vehicles, clock synchronization, and music, also exist.

Our first motivation for editing this book is that we are not aware of the existence of an edited book on the subject of DG. Some come close, such as, e.g., [6] or [19], but neither focuses principally on DG. The papers in [6] are mostly about protein folding using partial information related to interatomic distances: this is certainly one of the main applicative drivers behind DG, but the theoretical aspects are neglected. The book [19] comes closer to the target, since it is a very nice mix of theoretical and application papers; yet its main topic is rigidity rather than DG. Rigid graphs are sometimes used as models for proteins or sensor networks, but it is often the case that not enough distances are known to make the graph rigid.

The second motivation stems from the fact that the DG community is broken into segments which reflect the motivating applications (molecular conformation, sensor networks, statics, and others) as well as the discipline (mathematics, computer science, biochemistry, engineering). There is neither journal nor a regular conference or workshop that is wholly dedicated to DG. This state of affairs is strange for a discipline that was born in the late 1920s and still produces a lot of valuable results. We therefore wanted to gently help the researchers working in this area to acquire a stronger community identity. As far as this motivation is concerned, we obtained a partial success: our call for papers was answered positively by around half of the people we had initially asked. On the one hand, this is to be expected in edited books: some early authors changed subjects or retired, whilst some of the active ones have administrative duties or simply cannot afford the time. On the other hand, this book shows the bias of its editors: most application papers are devoted to molecular conformation, which is what initially motivated us, as authors, to work in DG. Although we did send invitations to researchers in the sensor network and statics communities, few knew of us, so few of them answered.

A third motivation is that we are planning to organize a workshop on DG in June 2013 in Manaus, Brazil, which, we hope, will further help shaping the existing magma into a true DG community. Although the usual process derives edited books from conference proceedings, we decided to use this book as a basis to form a kernel of researchers who might be interested in attending the workshop.

Distance geometry was born in 1928, when Karl Menger characterized several geometrical concepts, such as congruence and convexity, by means of the primary notion of *distance*. This attempt was far from uncommon in the 50 years covering the turn of the century: several geometers asked themselves what would happen if some of the Euclidean axioms which had always formed the basis of geometry were removed. Hyperbolic, spherical, projective, and other types of geometries were studied and applied to engineering problems. When Hilbert pushed mathematicians around the world to pay attention to the precision of their proofs, the community underwent a profound change involving the axiomatization of every mathematical discipline, including geometry. This eventually stimulated logic and set theory and the remarkable works of Gödel, Tarski, Post, and Turing. Alternative axiomatizations of Euclidean geometry were sought in order to establish which geometrical concepts were truly fundamental and which were not. The axiomatization work on geometry, together with the study of Grassmann–Cayley algebras, gave rise to the use of several different types of distance coordinates which were then generalized in the framework of solving distance constraint systems, e.g. in [21].

Menger's work was eventually continued by Blumenthal: the main question in DG, at the time, was to find necessary and sufficient conditions to decide whether a given matrix was a distance matrix, i.e. a symmetric $n \times n$ matrix $D = (d_{ij})$ such that there exists an integer $K > 0$ and a set of $n$ points $\{x_1, \ldots, x_n\}$ in $\mathbb{R}^K$ with the property that

$$\forall u < v \quad \|x_u - x_v\| = d_{uv}, \tag{1}$$

where the norm $\|\cdot\|$ is usually taken to be Euclidean (in fact, all chapters in this book make this assumption aside from [7], which strays from the path and considers walk distances, instead of edge distances).

A quarter century later, Yemini defined the positioning problem in sensor networks as: *locate a set of geographically distributed objects using [...] the distances between some object pairs*. The main difference with the earlier problem is the word "some": not all distances are given. Equivalently, some of the entries from the matrix $D$ are missing. Accordingly, we define the fundamental problem in DG (in terms of graphs instead of matrices) as follows.

> Distance Geometry Problem (DGP). Given an integer $K > 0$ and a simple weighted undirected graph $G = (V,E,d)$, where $d : E \to \mathbb{Q}_+$, decide whether there exist $|V| = n$ points $\{x_1,\ldots,x_n\}$ in $\mathbb{R}^K$ with
> $$\forall\{u,v\} \in E \quad \|x_u - x_v\| = d_{uv}. \tag{2}$$

We remark that in the DGP, $K$ is given as part of the input. Sometimes $K$ needs to be decided as part of the output: for example, find the minimum $K$ such that a solution $x$ to Eq. (2) exists. This is known as the Euclidean distance matrix completion problem (EDMCP). This seemingly minor difference has a noticeable impact on the problem complexity: whereas the DGP is known to be **NP**-hard, the complexity status of the EDMCP is currently unknown. Whether the DGP itself is in **NP** or not remains an open question, explored in [5].

The number of solutions of Eq. (2) can be either uncountable or finite, but never countably infinite because of some well-known properties of semi-algebraic sets.

In the first case, finding the certificates (i.e. the feasible realizations $x = (x_1,\ldots,x_n) \in \mathbb{R}^{Kn}$ of $G$) usually requires continuous or mixed-continuous methods. Most of these reformulate (2) into a minimization of squared penalty terms:

$$\min_{x\in\mathbb{R}^{Kn}} \sum_{\{u,v\}\in E} (\|x_u - x_v\|^2 - d_{uv}^2)^2. \tag{3}$$

In this reformulation, a solution $x$ of the minimization problem is a solution of the corresponding DGP if and only if the optimal objective function value is zero. Among the chapters in this book that deal with this case, Locatelli and Schoen [14] proposes a population-based metaheuristic, Lima and Martínez [13] uses a local nonlinear programming (NLP) solution algorithm, and An and Tao [4] transforms (3) into a difference of convex (DC) functions and solves it using a DC programming method. The paper [1] proposes a heuristic method (SPE) which is directly based on the original formulation (2). SPE's simplicity is its strong point: it is specially suited to massive data sets and can be easily parallelized.

In the second case, the graph $G$ is said to be rigid. Rigidity theory is a beautiful and well-developed mathematical theory, whose main motivating application is statics. A very nice mathematical overview of some important theoretical achievements in rigidity theory is given in [2]. In sensor networks the focus is on uniquely localizable graphs (i.e. graphs with sufficiently many edges so that Eq. (2) has a unique solution up to congruences), because it is known that the sensors have unique positions in the physical space. Realizing such graphs can be done in polynomial time using semidefinite programming (SDP) duality. On the other hand, given atoms may combine in different ways to yield proteins with different chiralities: the correct model for this situation is a rigid graph whose corresponding system (2) has $h > 1$ incongruent solutions: the paper [10] discusses some bounds on $h$. Although it is certainly possible to simply ignore the rigidity of $G$ and employ continuous search methods for general graphs, most such methods will stop at the first solution found, and none will guarantee finding the whole incongruent solution set. For protein-related applications, however, finding the whole solution set is important, because new proteins with different chirality may be discovered this way. We believe the only method that can currently address this need is the branch-and-prune (BP) algorithm, which we first proposed in a technical report in 2006, published in 2008, and improved in several ways in a long and continuing sequence of papers. BP is based on an extension of trilateration, which is discussed in [18]: the intersection of $K+1$ spheres in $\mathbb{R}^K$ identifies at most one point with probability 1. This, incidentally, is why trilateration graphs can be realized in polynomial time. BP exploits the fact that the intersection of only $K$ spheres in $\mathbb{R}^K$ identifies at most two points with probability 1: the class of DGP instances that the BP can solve is called DISCRETIZABLE DGP (DDGP) and is itself **NP** hard. A development of BP involving the segmentation of the branching process (with a view to parallelization) is given in [17]. We made two empirical observations about the behaviour of BP over the years: the number of solutions (modulo translations and rotations) always turns out to be a power of two, and the CPU time on proteins looks linear in function of $n$ (rather than exponential, as BP has worst-case exponential running time): the theoretical properties that justify these observations are given in [12].

Some approaches such as ASAP [9], some generalizations of the geometric build-up algorithm [20], and the DISCO method (enhanced in [11] with restrictions originating from molecular chemistry) decompose the graph into smaller (typically rigid or uniquely localizable) components, apply specific graph embedding methodologies that exploit the structure of each component, and then stitch the partial realizations together with some form of continuous search.

The main applicative drive behind the DGP is possibly the determination of molecular structure using NMR data. The output of an NMR experiment is a value associated to a triplet consisting of two atomic labels and a distance. It can be interpreted to mean that the distance between the two atoms occurs as frequently as measured by the given value in the molecule under observation; these "raw" data are then processed into a weighted graph [8]. It turns out that distance measurements are most precise when the atoms involved are two hydrogens. In this setting, the distances $d_{ij}$ which form the main input of the DGP are often experimental measures, and as such are better described by intervals $[d^L_{uv}, d^U_{uv}]$ [3]. The corresponding problem, called INTERVAL DGP ($i$DGP), asks to find $n$

points in $\mathbb{R}^K$ satisfying $\forall\{u,v\} \in E \; \|x_u - x_v\| \in [d^L_{uv}, d^U_{uv}]$. The methods proposed in [1, 4, 11, 14, 20] all extend to the interval case. Graphs weighted by intervals usually fail to be rigid: the BP algorithm, however, can be adapted to work in the presence of a subset of interval distances; this and further improvements that consider chemical restrictions are discussed in [15].

Rennes, France — Antonio Mucherino
Campinas, Brazil — Carlile Lavor
Paris, France — Leo Liberti
Rio de Janeiro, Brazil — Nelson Maculan

## References

1. Agrafiotis, D., Bandyopadhyay, D., Young, E.: Stochastic proximity embedding (SPE): A simple, fast and scalable algorithm for solving the distance geometry problem. In: Mucherino et al. [16]
2. Alfakih, A.: Universal rigidity of bar frameworks in general position: a Euclidean distance matrix approach. In: Mucherino et al. [16]
3. Almeida, F., Moraes, A., Neto, F.G.: Overview on protein structure determination by NMR — Historical and future perspectives of the use of distance geometry. In: Mucherino et al. [16]
4. An, L.H., Tao, P.D.: DC programming approaches for distance geometry problems. In: Mucherino et al. [16]
5. Beeker, N., Gaubert, S., Glusa, C., Liberti, L.: Is the distance geometry problem in NP? In: Mucherino et al. [16]
6. Bohr, H., Brunak, S. (eds.): Protein Folds, a Distance Based Approach. CRC, Boca Raton (1996)
7. Chebotarev, P., Deza, M.: A topological interpretation of the walk distances. In: Mucherino et al. [16]
8. Crippen, G.: Distance geometry for realistic molecular conformations. In: Mucherino et al. [16]
9. Cucuringu, M.: ASAP – an eigenvector synchronization algorithm for the graph realization problem. In: Mucherino et al. [16]
10. Emiris, I., Tsigaridas, E., Varvitsiotis, A.: Mixed volume and distance geometry techniques for counting euclidean embeddings of rigid graphs. In: Mucherino et al. [16]
11. Fang, X., Toh, K.C.: Using a distributed SDP approach to solve simulated protein molecular conformation problems. In: Mucherino et al. [16]
12. Liberti, L., Lavor, C., Mucherino, A.: The discretizable molecular distance geometry problem seems easier on proteins. In: Mucherino et al. [16]
13. Lima, R., Martínez, J.M.: Solving molecular distance geometry problems using a continuous optimization approach. In: Mucherino et al. [16]
14. Locatelli, M., Schoen, F.: Global optimization for atomic cluster distance geometry problems. In: Mucherino et al. [16]
15. Malliavin, T., Mucherino, A., Nilges, M.: Distance geometry in structural biology: new perspectives. In: Mucherino et al. [16]
16. Mucherino, A., Lavor, C., Liberti, L., Maculan, N. (eds.): Distance Geometry: Theory, Methods, and Applications. Springer, Berlin (2013)
17. Nucci, P., Nogueira, L., Lavor, C.: Solving the discretizable molecular distance geometry problem by multiple realization trees. In: Mucherino et al. [16]

18. Petitjean, M.: Sphere unions and intersections and some of their applications in molecular modeling. In: Mucherino et al. [16]
19. Thorpe, M., Duxbury, P. (eds.): Rigidity Theory and Applications. Springer, New York (2002)
20. Voller, Z., Wu, Z.: Distance geometry methods for protein structure determination. In: Mucherino et al. [16]
21. Yang, L.: Solving spatial constraints with generalized distance geometry. In: Mucherino et al. [16]

# Contents

Part I Theory

Part II Methods

**Part III Applications to Protein Conformations**

# Contributors

**Dimitris K. Agrafiotis** Janssen Research & Development, Johnson & Johnson, Welsh & McKean Roads, Spring House, PA, USA

**Abdo Y. Alfakih** Department of Mathematics and Statistics, University of Windsor, Windsor, ON, Canada

**Fabio C.L. Almeida** National Center of Nuclear Magnetic Resonance, Institute of Medical Biochemistry, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

**Deepak Bandyopadhyay** GlaxoSmithKline, Collegeville, PA 19426, USA

**Nathanael Beeker** CMAP, École Polytechnique, Palaiseau, France

**Pavel Chebotarev** Institute of Control Sciences of the Russian Academy of Sciences, Moscow, Russia

**Gordon M. Crippen** University of Michigan, Ann Arbor, MI, USA

**Mihai Cucuringu** PACM, Princeton University, Fine Hall, Princeton, NJ, USA

**Michel Deza** Laboratoire de Geometrie Appliquee, LIGA, Ecole Normale Superieure, Paris, France

**Ioannis Z. Emiris** University of Athens, Athens, Greece

**Xingyuan Fang** Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ, USA

**Stéphane Gaubert** INRIA Rocquencourt, Cedex, France

**Christian Glusa** CMAP, École Polytechnique, Palaiseau, France

**Francisco Gomes-Neto** National Center of Nuclear Magnetic Resonance, Institute of Medical Biochemistry, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

**Carlile Lavor** Department of Applied Maths, University of Campinas, Campinas-SP, Brazil

**Hoai An Le Thi** Laboratory of Theoretical and Applied Computer Science (LITA), University of Lorraine, Metz, France

**Leo Liberti** LIX, École Polytechnique, Palaiseau, France

**Rodrigo S. Lima** ICE-UNIFEI, Federal University of Itajubá, Itajubá-MG, Brazil

**Marco Locatelli** Università di Parma, Parma, Italy

**Thérèse E. Malliavin** Institut Pasteur, Paris, France

**J.M. Martínez** IMECC-UNICAMP, State University of Campinas, Campinas-SP, Brazil

**Adolfo H. Moraes** National Center of Nuclear Magnetic Resonance, Institute of Medical Biochemistry, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil

**Antonio Mucherino** IRISA, University of Rennes 1, Rennes, France

**Michael Nilges** Institut Pasteur, Paris, France

**Loana Tito Nogueira** Instituto de Computação, Universidade Federal Fluminense, Rio de Janeiro, Brazil

**Pedro Nucci** Navy Arsenal of Rio de Janeiro, Brazilian Navy, Brazil

**Michel Petitjean** MTi, University Paris 7, INSERM UMR-S 973, Paris, France

**Tao Pham Dinh** Laboratory of Mathematics, National Institute for Applied Sciences, Rouen, Saint-Etienne-du-Rouvray, France

**Fabio Schoen** Università di Firenze, Firenze, Italy

**Kim-Chuan Toh** Department of Mathematics, National University of Singapore, Singapore

**Elias P. Tsigaridas** INRIA Paris-Rocquencourt, Paris, France

**Antonios Varvitsiotis** Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands

**Zachary Voller** Department of Mathematics, Iowa State University, Ames, IA, USA

**Eric Yang** Janssen Research & Development, Johnson & Johnson, Welsh & McKean Roads, Spring House, PA, USA

**Lu Yang** Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai, China

**Zhijun Wu** Department of Mathematics, Iowa State University, Ames, IA, USA

# Part I
# Theory

# Chapter 1
# Universal Rigidity of Bar Frameworks in General Position: A Euclidean Distance Matrix Approach

**Abdo Y. Alfakih**

**Abstract** A configuration $p$ in $r$-dimensional Euclidean space is a finite collection of labeled points $p^1,\dots,p^n$ in $\mathbb{R}^r$ that affinely span $\mathbb{R}^r$. Each configuration $p$ defines a Euclidean distance matrix $D_p = (d_{ij}) = (||p^i - p^j||^2)$, where $||\cdot||$ denotes the Euclidean norm. A fundamental problem in distance geometry is to find out whether or not a given proper subset of the entries of $D_p$ suffices to uniquely determine the entire matrix $D_p$. This problem is known as the universal rigidity problem of bar frameworks. In this chapter, we present a unified approach for the universal rigidity of bar frameworks, based on Euclidean distance matrices (EDMs), or equivalently, on projected Gram matrices. This approach makes the universal rigidity problem amenable to semidefinite programming methodology. Using this approach, we survey some recently obtained results and their proofs, emphasizing the case where the points $p^1,\dots,p^n$ are in general position.

## 1.1 Introduction

A *configuration* $p$ in $r$-dimensional Euclidean space is a finite collection of labeled points $p^1,\dots,p^n$ in $\mathbb{R}^r$ that affinely span $\mathbb{R}^r$. Each configuration $p$ defines the $n\times n$ matrix $D_p = (d_{ij}) = (||p^i - p^j||^2)$, where $||.||$ denotes the Euclidean norm. $D_p$ is called the *Euclidean distance matrix (EDM)* generated by configuration $p$. Obviously, $D_p$ is a real symmetric matrix whose diagonal entries are all zeros. A fundamental problem in distance geometry is to find out whether or not, a given proper subset of the entries of $D_p$, the EDM generated by configuration $p$, suffices

A.Y. Alfakih (✉)
Department of Mathematics and Statistics, University of Windsor,
Windsor, ON, Canada, N9B 3P4
e-mail: alfakih@uwindsor.ca

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_1,

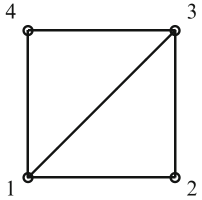


**Fig. 1.1** A bar framework $G(p)$ on 4 vertices in $\mathbb{R}^2$, where $V(G) = \{1,2,3,4\}$, $E(G) = \{(1,2),(2,3),(3,4),(4,1),(1,3)\}$ and $p^1, p^2, p^3, p^4$ are the vertices of the unit square

to uniquely determine the entire matrix $D_p$, i.e., to uniquely recover $p$, up to a rigid motion. This problem is known as the universal rigidity problem of bar frameworks.

A *bar framework*, or framework for short, denoted by $G(p)$, in $\mathbb{R}^r$ is a configuration $p$ in $\mathbb{R}^r$ together with a simple graph $G$ on the vertices $1,2,\ldots,n$. To avoid trivialities, we assume throughout this chapter that the graph $G$ is connected and not complete. It is useful to think of each node $i$ of $G$ in a framework $G(p)$ as a universal joint located at $p^i$ and of each edge $(i,j)$ of $G$ as a stiff bar of length $||p^i - p^j||$. Hence, a bar framework is often defined as a collection of stiff bars joined at their ends by universal joints. Figure 1.1 depicts a framework $G(p)$ on 4 vertices in $\mathbb{R}^2$, where $G$ is the complete graph $K_4$ minus an edge and the points $p^1,\ldots,p^4$ are the vertices of the unit square.

We say that two frameworks $G(p)$ and $G(q)$ in $\mathbb{R}^r$ are *congruent* if $D_p = D_q$. Furthermore, let $H$ denote the adjacency matrix of graph $G$, then two frameworks $G(p)$ in $\mathbb{R}^r$ and $G(q)$ in $\mathbb{R}^s$ are said to be *equivalent* if $H \circ D_p = H \circ D_q$, where $\circ$ denotes the *Hadamard product*, i.e., the element-wise product. We say that framework $G(q)$ in $\mathbb{R}^r$ is *affinely equivalent* to framework $G(p)$ in $\mathbb{R}^r$ if $G(q)$ is equivalent to $G(p)$ and configuration $q$ is obtained from configuration $p$ by an affine motion, i.e., $q^i = Ap^i + b$, for all $i = 1,\ldots,n$, for some $r \times r$ matrix $A$ and an $r$-vector $b$.

A framework $G(p)$ in $\mathbb{R}^r$ is said to be *universally rigid* if every framework $G(q)$ in any dimension that is equivalent to $G(p)$ is in fact congruent to $G(p)$, i.e., if for every framework $G(q)$ in any dimension such that $H \circ D_q = H \circ D_p$, it follows that $D_q = D_p$.

Thus, given $D_p = (d_{ij})$, the EDM generated by configuration $p$, let $K \subset \{(i,j) : i < j;\ \text{for}\ i,j = 1,2,\ldots,n\}$. Then the proper subset of entries of $D_p$ given by $\{d_{ij} : (i,j) \in K\}$ suffices to uniquely determine the entire matrix $D_p$ if and only if framework $G(p)$ is universally rigid, where $G = (V,E)$ is the graph with vertex set $V = \{1,2,\ldots,n\}$ and edge set $E = K$. For example, consider the framework $G(p)$ given in Fig. 1.1 and its corresponding EDM

$$D_p = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{bmatrix}.$$

In the process of folding $G(p)$ along the edge $(1,3)$, the distance $||p^2 - p^4||$ varies between $\sqrt{2}$ and 0. Thus, the subset of entries of $D_p$ given by $\{d_{ij} : (i,j) \in E(G)\}$ does not uniquely determine the entire matrix $D_p$ since the entry $d_{24}$ can assume any value between 2 and 0. Accordingly, $G(p)$ is not universally rigid.

The notion of dimensional rigidity is closely related to that of universal rigidity. A framework $G(p)$ in $\mathbb{R}^r$ is said to be *dimensionally rigid* if there does not exist a framework $G(q)$ that is equivalent to $G(p)$ in any Euclidean space of dimension $\geq r+1$. For example, the framework $G(p)$ given in Fig. 1.1 is obviously not dimensionally rigid since there is an infinite number of frameworks $G(q)$ in $\mathbb{R}^3$ that are equivalent to $G(p)$. This can be seen by rotating the triangle 123 of the framework $G(p)$ about the edge $(1,3)$.

In this chapter, we survey some recently obtained results concerning framework universal as well as dimensional rigidity. These results are given in Sect. 1.2 and their proofs are given in Sect. 1.4. Section 1.3 is dedicated to the mathematical preliminaries needed for our proofs. Our EDM approach of universal rigidity of bar frameworks extends to the closely related notion of "local" rigidity. However, due to space limitation, "local" rigidity [3] will not be considered here. Also, we will not consider the other closely related notion of global rigidity [10, 14].

Throughout this chapter, $|C|$ denotes the cardinality of a finite set $C$. We denote the node set and the edge set of a simple graph $G$ by $V(G)$ and $E(G)$, respectively. $\mathscr{S}_n$ denotes the space of $n \times n$ real symmetric matrices. Positive semi-definiteness (positive definiteness) of a symmetric matrix $A$ is denoted by $A \succeq \mathbf{0}$ ($A \succ \mathbf{0}$). For a matrix $A$ in $\mathscr{S}_n$, $\text{diag}(A)$ denotes the $n$-vector formed from the diagonal entries of $A$. $e$ denotes the vector of all ones in $\mathbb{R}^n$. Finally, the $n \times n$ identity matrix is denoted by $I_n$; and $\mathbf{0}$ denotes the zero matrix or the zero vector of the appropriate dimension.

## 1.2 Main Results

The following theorem characterizes universal rigidity in terms of dimensional rigidity and affine-equivalence.

**Theorem 1.1 (Alfakih [2]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$, $r \leq n-2$. Then $G(p)$ is universally rigid if and only if the following two conditions hold:*

1. *$G(p)$ is dimensionally rigid.*
2. *There does not exist a bar framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$.*

The proof of Theorem 1.1 is given in Sect. 1.4. The notion of a stress matrix $S$ of a framework $G(p)$ plays an important role in the characterization of universal rigidity of $G(p)$. Let $G(p)$ be a framework on $n$ vertices in $\mathbb{R}^r$, $r \leq n-2$. An *equilibrium stress* of $G(p)$ is a real valued function $\omega$ on $E(G)$, the set of edges of $G$, such that

$$\sum_{j:(i,j)\in E(G)} \omega_{ij}(p^i - p^j) = \mathbf{0} \quad \text{for all } i = 1,\ldots,n. \tag{1.1}$$

Let $\omega$ be an equilibrium stress of $G(p)$. Then the $n \times n$ symmetric matrix $S = (s_{ij})$ where

$$s_{ij} = \begin{cases} -\omega_{ij} & \text{if } (i,j) \in E(G), \\ 0 & \text{if } i \neq j \text{ and } (i,j) \notin E(G), \\ \sum_{k:(i,k)\in E(G)} \omega_{ik} & \text{if } i = j, \end{cases} \tag{1.2}$$

is called the *stress matrix* associated with $\omega$, or a stress matrix of $G(p)$.

Given framework $G(p)$ on $n$ vertices in $\mathbb{R}^r$, we define the following $n \times r$ matrix

$$P := \begin{bmatrix} {p^1}^T \\ {p^2}^T \\ \vdots \\ {p^n}^T \end{bmatrix}. \tag{1.3}$$

$P$ is called the *configuration matrix* of $G(p)$. Note that $P$ has full column rank since $p^1, \ldots, p^n$ affinely span $\mathbb{R}^r$. The following lemma provides an upper bound on the rank of a stress matrix $S$.

**Lemma 1.1.** *Let $G(p)$ be a bar framework on $n$ nodes in $\mathbb{R}^r$, $r \leq n-2$, and let $S$ and $P$ be a stress matrix and the configuration matrix of $G(p)$, respectively. Then $SP = \mathbf{0}$ and $Se = \mathbf{0}$, where $e$ is the vector of all 1's. Consequently, rank $S \leq n-r-1$.*

*Proof.* It follows from Eqs. (1.1) and (1.2) that the $i$th row of $SP$ is given by

$$s_{ii}(p^i)^T + \sum_{k=1,k\neq i}^{n} s_{ik}(p^k)^T = \sum_{k:(i,k)\in E(G)} \omega_{ik}(p^i - p^k)^T = \mathbf{0}.$$

Also, $e$ is obviously in the null space of $S$. Hence, the result follows. □

### 1.2.1 Dimensional and Universal Rigidity in Terms of Stress Matrices

The following theorem provides a sufficient condition for the dimensional rigidity of frameworks.

**Theorem 1.2 (Alfakih [2]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$ for some $r \leq n-2$. If $G(p)$ admits a positive semidefinite stress matrix $S$ of rank $n-r-1$, then $G(p)$ is dimensionally rigid.*

The proof of Theorem 1.2 is given in Sect. 1.4. It is worth pointing out that the converse of Theorem 1.2 is not true. Consider the following framework [2] $G(p)$ on 5 vertices in $\mathbb{R}^2$ (see Fig 1.2), where the configuration matrix $P$ is given by

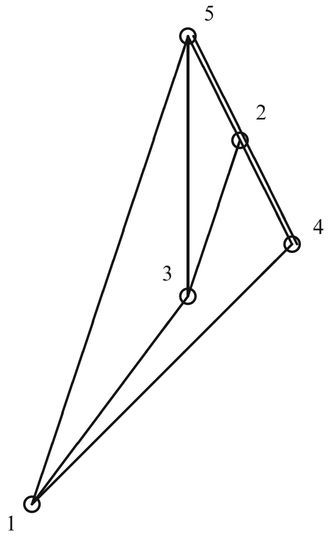


**Fig. 1.2** A dimensionally rigid framework $G(p)$ in $\mathbb{R}^2$ (in fact $G(p)$ is also universally rigid) that does not admit a positive semidefinite stress matrix of rank 2. Note that the points $p^2, p^4$, and $p^5$ are collinear, i.e., $G(p)$ is not in general position

$$P = \begin{bmatrix} -3 & -5 \\ 1 & 2 \\ 0 & -1 \\ 2 & 0 \\ 0 & 4 \end{bmatrix},$$

and where the missing edges of $G$ are $(1,2)$ and $(3,4)$. It is clear that $G(p)$ is dimensionally rigid (in fact $G(p)$ is also universally rigid) while $G(p)$ has no positive semidefinite stress matrix of rank 2.

The following result, which provides a sufficient condition for the universal rigidity of a given framework, is a direct consequence of Theorems 1.1 and 1.2.

**Theorem 1.3 (Connelly [8, 9], Alfakih [2]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$, for some $r \leq n-2$. If the following two conditions hold:*

1. *$G(p)$ admits a positive semidefinite stress matrix $S$ of rank $n-r-1$.*
2. *There does not exist a bar framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$.*

*Then $G(p)$ is universally rigid.*

A configuration $p$ (or a framework $G(p)$) is said to be *generic* if all the coordinates of $p^1, \ldots, p^n$ are algebraically independent over the integers. That is, if there does not exist a nonzero polynomial $f$ with integer coefficients such that $f(p^1, \ldots, p^n) = 0$. Thus, for a generic framework, Theorem 1.3 reduces to the following theorem.

**Theorem 1.4 (Connelly [9], Alfakih [4]).** *Let $G(p)$ be a generic bar framework on $n$ nodes in $\mathbb{R}^r$, for some $r \leq n-2$. If $G(p)$ admits a positive semidefinite stress matrix $S$ of rank $n-r-1$, then $G(p)$ is universally rigid.*

The proof of Theorem 1.4 is given in Sect. 1.4. The converse of Theorem 1.4 is also true.

**Theorem 1.5 (Gortler and Thurston [15]).** *Let $G(p)$ be a generic bar framework on $n$ nodes in $\mathbb{R}^r$, for some $r \leq n-2$. If $G(p)$ is universally rigid, then there exists a positive semidefinite stress matrix $S$ of $G(p)$ of rank $n-r-1$.*

The proof of Theorem 1.5 given in [15] goes beyond the scope of this chapter and will not be presented here.

At this point, one is tempted to ask whether a result similar to Theorem 1.4 holds if the genericity assumption of $G(p)$ is replaced by the weaker assumption of general position. A configuration $p$ (or a framework $G(p)$) in $\mathbb{R}^r$ is said to be in *general position* if no $r+1$ points in $p^1,\ldots,p^n$ are affinely dependent. For example, a set of points in the plane are in general position if no 3 of them are collinear. The following theorem answers this question in the affirmative.

**Theorem 1.6 (Alfakih and Ye [7]).** *Let $G(p)$ be a bar framework on $n$ nodes in general position in $\mathbb{R}^r$, for some $r \leq n-2$. If $G(p)$ admits a positive semidefinite stress matrix $S$ of rank $n-r-1$, then $G(p)$ is universally rigid.*

The proof of Theorem 1.6 is given in Sect. 1.4. The following result shows that the converse of Theorem 1.6 holds for frameworks $G(p)$ where graph $G$ is an $(r+1)$-lateration graph. Such frameworks were shown to be universally rigid in [22]. However, it is still an open question whether the converse of Theorem 1.6 holds for frameworks of general graphs.

A graph $G$ on $n$ vertices is called an $(r+1)$*-lateration graph* [12, 18] if there is a permutation $\pi$ of the vertices of $G$, $\pi(1),\pi(2),\ldots,\pi(n)$, such that

1. The first $(r+1)$ vertices, $\pi(1),\ldots,\pi(r+1)$, induce a clique in $G$.
2. Each remaining vertex $\pi(j)$, for $j=(r+2),(r+3),\ldots,n$, is adjacent to $(r+1)$ vertices in the set $\{\pi(1),\pi(2),\ldots,\pi(j-1)\}$.

**Theorem 1.7 (Alfakih et al [6]).** *Let $G(p)$ be a bar framework on $n$ nodes in general position in $\mathbb{R}^r$, for some $n \geq r+2$, where $G$ is an $(r+1)$-lateration graph. Then there exists a positive semidefinite stress matrix $S$ of $G(p)$ of rank $n-r-1$.*

The proof of Theorem 1.7 is given in Sect. 1.4. The preceding theorems have been stated in terms of stress matrices. The same theorems can be equivalently stated in terms of Gale matrices, as will be shown in the next section.

### 1.2.2 Dimensional and Universal Rigidity in Terms of Gale Matrices

Let $G(p)$ be a framework on $n$ vertices in $\mathbb{R}^r$, $r \leq n-2$, and let $P$ be the configuration matrix of $G(p)$. Then the following $(r+1) \times n$ matrix

$$\mathscr{P} := \begin{bmatrix} P^T \\ e^T \end{bmatrix} = \begin{bmatrix} p^1 & \dots & p^n \\ 1 & \dots & 1 \end{bmatrix} \tag{1.4}$$

has full row rank since $p^1, \dots, p^n$ affinely span $\mathbb{R}^r$. Note that $r \leq n-1$. Let

$$\bar{r} = \text{the dimension of the null space of } \mathscr{P}, \text{ i.e., } \bar{r} = n - 1 - r. \tag{1.5}$$

**Definition 1.1.** Suppose that the null space of $\mathscr{P}$ is nontrivial, i.e., $\bar{r} \geq 1$. Any $n \times \bar{r}$ matrix $Z$ whose columns form a basis of the null space of $\mathscr{P}$ is called a *Gale matrix* of configuration $p$ (or framework $G(p)$). Furthermore, the $i$th row of $Z$, considered as a vector in $\mathbb{R}^{\bar{r}}$, is called a *Gale transform* of $p^i$ [13].

The Gale transform plays an important role in the theory of polytopes [17]. It follows from Lemma 1.1 and Eq. (1.2) that $S$ is a stress matrix of $G(p)$ if and only if

$$\mathscr{P}S = \mathbf{0}, \text{ and } s_{ij} = 0 \text{ for all } ij : i \neq j, (i,j) \not\in E(G). \tag{1.6}$$

Equivalently, $S$ is a stress matrix of $G(p)$ if and only if there exists an $\bar{r} \times \bar{r}$ symmetric matrix $\Psi$ such that

$$S = Z \Psi Z^T, \text{ and } s_{ij} = (z^i)^T \Psi z^j = 0 \text{ for all } ij : i \neq j, (i,j) \not\in E(G), \tag{1.7}$$

where $(z^i)^T$ is the $i$th row of $Z$. Therefore, the stress matrix $S = Z\Psi Z^T$ attains its maximum rank of $\bar{r} = n - 1 - r$ if and only if $\Psi$ is nonsingular, i.e., rank $\Psi = \bar{r}$, since rank $S$ = rank $\Psi$.

Then Theorems 1.2, 1.4, 1.5, and 1.6 can be stated in terms of Gale matrices as follows.

**Theorem 1.8 (Alfakih [2]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$ for some $r \leq n-2$, and let $Z$ be a Gale matrix of $G(p)$. If there exists a positive definite symmetric matrix $\Psi$ such that*

$$(z^i)^T \Psi z^j = 0 \textit{ for each } ij : i \neq j, (i,j) \not\in E(G),$$

*where $(z^i)^T$ is the ith row of $Z$, then $G(p)$ is dimensionally rigid.*

**Theorem 1.9 (Connelly [9], Alfakih [4], Gortler and Thurston [15]).** *Let $G(p)$ be a generic bar framework on $n$ nodes in $\mathbb{R}^r$, for some $r \leq n-2$. Let $Z$ be a Gale* *matrix of $G(p)$. Then $G(p)$ is universally rigid if and only if there exists a positive definite symmetric matrix $\Psi$ such that*

$$(z^i)^T \Psi z^j = 0 \textit{ for each } ij : i \neq j, (i,j) \notin E(G),$$

*where $(z^i)^T$ is the ith row of Z.*

**Theorem 1.10 (Alfakih and Ye [7]).** *Let $G(p)$ be a bar framework on n nodes in general position in $\mathbb{R}^r$, for some $r \leq n-2$. Let Z be a Gale matrix of $G(p)$. Then $G(p)$ is universally rigid if there exists a positive definite symmetric matrix $\Psi$ such that*

$$(z^i)^T \Psi z^j = 0 \textit{ for each } ij : i \neq j, (i,j) \notin E(G),$$

*where $(z^i)^T$ is the ith row of Z.*

## 1.3 Preliminaries

In this section we give the mathematical preliminaries needed for our proofs. In particular, we review some basic terminology and results concerning Euclidean distance matrices (EDMs) and affine motions.

### *1.3.1 Euclidean Distance Matrices*

It is well known [11, 16, 20, 21] that a symmetric $n \times n$ matrix $D$ whose diagonal entries are all zeros is EDM if and only if $D$ is negative semidefinite on the subspace

$$M := \{x \in \mathbb{R}^n : e^T x = 0\},$$

where $e$ is the vector of all 1's.

Let $V$ be the $n \times (n-1)$ matrix whose columns form an orthonormal basis of $M$, i.e., $V$ satisfies

$$V^T e = \mathbf{0}\,, \quad V^T V = I_{n-1}\,. \tag{1.8}$$

Then the orthogonal projection on $M$, denoted by $J$, is given by $J := VV^T = I_n - ee^T/n$.

Recall that $\mathscr{S}_{n-1}$ denotes the subspace of symmetric matrices of order $n-1$ and let $\mathscr{S}_H = \{A \in \mathscr{S}_n : \mathrm{diag}(A) = \mathbf{0}\}$. Consider the linear operator $\mathscr{T}_V : \mathscr{S}_H \to \mathscr{S}_{n-1}$ such that

$$\mathscr{T}_V(D) := -\frac{1}{2} V^T D V, \tag{1.9}$$

Then we have the following lemma.

**Lemma 1.2 ([5]).** *Let $D \in \mathscr{S}_H$. Then $D$ is a Euclidean distance matrix of embedding dimension $r$ if and only if $\mathscr{T}_V(D) \succeq \mathbf{0}$ and rank $\mathscr{T}_V(D) = r$.*

Let $\mathscr{K}_V : \mathscr{S}_{n-1} \to \mathscr{S}_H$ defined by

$$\mathscr{K}_V(X) := \operatorname{diag}(VXV^T)\, e^T + e\, (\operatorname{diag}(VXV^T))^T - 2VXV^T. \tag{1.10}$$

Then it is not difficult to show that the operators $\mathscr{T}_V$ and $\mathscr{K}_V$ are mutually inverse [5]. Thus, Lemma 1.2 implies that $D$ in $\mathscr{S}_H$ is an EDM of embedding dimension $r$ if and only if $D = \mathscr{K}_V(X)$ for some positive semidefinite matrix $X$ of rank $r$.

Lemma 1.2 is used in the following section to characterize the set of equivalent frameworks.

### 1.3.2 Characterizing Equivalent Bar Frameworks

Since all congruent frameworks have the same EDM (or equivalently, the same projected Gram matrix), in the rest of this chapter, we will identify congruent frameworks. Accordingly, for a given framework $G(p)$, we assume without loss of generality that the centroid of the points $p^1, \ldots, p^n$ coincides with the origin, i.e., $P^T e = \mathbf{0}$, where $P$ is the configuration matrix of $G(p)$.

Let $D = (d_{ij})$ be the EDM generated by framework $G(p)$ in $\mathbb{R}^r$ and let $P$ be the configuration matrix of $G(p)$ defined in Eq. (1.3). Let $X = \mathscr{T}_V(D)$, or equivalently, $D = \mathscr{K}_V(X)$; and let $B = PP^T$ be the Gram matrix generated by the points $p^1, \ldots, p^n$. Clearly, $B$ is positive semidefinite of rank $r$. Observe that

$$\begin{aligned} d_{ij} &= \; ||p^i - p^j||^2 \\ &= \; (p^i)^T p^i + (p^j)^T p^j - 2\,(p^i)^T p^j \\ &= \; (PP^T)_{ii} + (PP^T)_{jj} - 2\,(PP^T)_{ij}. \end{aligned}$$

Therefore,

$$D = \operatorname{diag}(B)e^T + e(\operatorname{diag}(B))^T - 2B = \mathscr{K}_V(X).$$

Hence,

$$B = VXV^T, \text{ and } X = V^T BV = V^T PP^T V. \tag{1.11}$$

Furthermore, matrix $X$ is $(n-1) \times (n-1)$ positive semidefinite of rank $r$. Accordingly, $X$ is called the *projected Gram matrix* of $G(p)$.

Now let $G(q)$ in $\mathbb{R}^s$ be a framework equivalent to $G(p)$. Let $D_q$ and $D_p$ be the EDMs generated by $G(q)$ and $G(p)$, respectively. Then $H \circ D_q = H \circ D_p$ where $H$ is the adjacency matrix of graph $G$. Thus,

$$H \circ (D_q - D_p) = H \circ \mathscr{K}_V(X_q - X_p) = \mathbf{0}, \tag{1.12}$$

where $X_q$ and $X_p$ are the projected Gram matrices of $G(q)$ and $G(p)$, respectively.

Let $E^{ij}$ be the $n \times n$ symmetric matrix with 1's in the $ij$th and $ji$th entries and zeros elsewhere. Further, let

$$M^{ij} := \mathscr{T}_V(E^{ij}) = -\frac{1}{2} V^T E^{ij} V. \tag{1.13}$$

Then one can easily show that the set $\{M^{ij} : i \neq j,\ (i,j) \not\in E(G)\}$ forms a basis for the null space of $H \circ \mathscr{K}_V$. Hence, it follows from Eq. (1.12) that

$$X_q - X_p = \sum_{ij: i\neq j, (i,j) \not\in E(G)} y_{ij} M^{ij}, \tag{1.14}$$

for some scalars $y_{ij}$. Therefore, given a framework $G(p)$ in $\mathbb{R}^r$, the set of projected Gram matrices of all frameworks $G(q)$ that are equivalent to $G(p)$ is given by

$$\{X : X = X_p + \sum_{ij: i\neq j, (i,j) \not\in E(G)} y_{ij} M^{ij} \succeq \mathbf{0}\}. \tag{1.15}$$

The following lemma establishes the connection between Gale matrices and projected Gram matrices.

**Lemma 1.3 (Alfakih [1]).** *Let $G(p)$ be a bar framework in $\mathbb{R}^r$ and let $P$ and $X$ be the configuration matrix and the projected Gram matrix of $G(p)$, respectively. Further, let $U$ and $W$ be the matrices whose columns form orthonormal bases for the null space and the column space of $X$. Then*

1. *$VU$ is a Gale matrix of $G(p)$.*
2. *$VW = PQ$ for some $r \times r$ nonsingular matrix $Q$.*

*Proof.* It follows from Eq. (1.11) that $XU = V^T P P^T V U = \mathbf{0}$. Thus $P^T V U = \mathbf{0}$. Hence, $VU$ is a Gale matrix of $G(p)$ since obviously $e^T V U = \mathbf{0}$.

Now, $(VW)^T VU = \mathbf{0}$. Thus $VW = PQ$ for some matrix $Q$ since $P^T e = \mathbf{0}$. Moreover, $Q$ is nonsingular since rank $PQ = r$. □

### 1.3.3 Affine Motions

Affine motions play an important role in the problem of universal rigidity of bar frameworks. An affine motion in $\mathbb{R}^r$ is a map $f : \mathbb{R}^r \rightarrow \mathbb{R}^r$ of the form

$$f(p^i) = Ap^i + b,$$

for all $p^i$ in $\mathbb{R}^r$, where $A$ is an $r \times r$ matrix and $b$ is an $r$-vector. A rigid motion is an affine motion where matrix $A$ is orthogonal.

Vectors $v^1, \dots, v^m$ in $\mathbb{R}^r$ are said to lie on a *quadratic at infinity* if there exists a nonzero symmetric $r \times r$ matrix $\boldsymbol{\Phi}$ such that

$$(v^i)^T \boldsymbol{\Phi} v^i = 0, \text{ for all } i = 1, \dots, m.$$

The following lemma establishes the connection between the notion of a quadratic at infinity and affine motions.

**Lemma 1.4 (Connelly [10]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$. Then the following two conditions are equivalent:*

1. *There exists a bar framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$.*
2. *The vectors $p^i - p^j$ for all $(i,j) \in E(G)$ lie on a quadratic at infinity.*

*Proof.* Suppose that there exists a framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$; and let $q^i = Ap^i + b$ for all $i = 1, \dots, n$. Then $(q^i - q^j)^T (q^i - q^j) = (p^i - p^j)^T A^T A (p^i - p^j) = (p^i - p^j)^T (p^i - p^j)$ for all $(i,j) \in E(G)$. Note that matrix $A$ is not orthogonal since $G(q)$ and $G(p)$ are not congruent. Therefore, $(p^i - p^j)^T \boldsymbol{\Phi} (p^i - p^j) = 0$ for all $(i,j) \in E(G)$, where $\boldsymbol{\Phi} = I_r - A^T A$ is a nonzero symmetric matrix.

On the other hand, suppose that there exists a nonzero symmetric matrix $\boldsymbol{\Phi}$ such that $(p^i - p^j)^T \boldsymbol{\Phi} (p^i - p^j) = 0$, for all $(i,j) \in E(G)$. Then $I_r - \delta \boldsymbol{\Phi} \succ \mathbf{0}$ for sufficiently small $\delta$. Hence, there exists a matrix $A$ such that $I_r - \delta \boldsymbol{\Phi} = A^T A$. Note that matrix $A$ is not orthogonal since $\boldsymbol{\Phi}$ is nonzero. Thus, $(p^i - p^j)^T (I_r - A^T A)(p^i - p^j) = 0$ for all $(i,j) \in E(G)$. Therefore, there exists a framework $G(q)$ in $\mathbb{R}^r$ that is equivalent to $G(q)$, where $q^i = Ap^i$ for all $i = 1, \dots, n$. Furthermore, $G(q)$ is not congruent to $G(p)$ since $A$ is not orthogonal. □

Note that Condition 2 in Lemma 1.4 is expressed in terms of the edges of $G$. An equivalent condition in terms of the missing edges of $G$ can also be obtained using Gale matrices. To this end, let $\bar{m}$ be the number of missing edges of graph $G$ and let $y = (y_{ij})$ be a vector in $\mathbb{R}^{\bar{m}}$. Let $\mathscr{E}(y)$ be the $n \times n$ symmetric matrix whose $ij$th entry is given by

$$\mathscr{E}(y)_{ij} = \begin{cases} y_{ij} & \text{if } i \neq j \text{ and } (i,j) \notin E(G), \\ 0 & \text{Otherwise.} \end{cases} \tag{1.16}$$

Then we have the following result.

**Lemma 1.5 (Alfakih [4]).** *Let $G(p)$ be a bar framework on $n$ vertices in $\mathbb{R}^r$ and let $Z$ be any Gale matrix of $G(p)$. Then the following two conditions are equivalent:*

1. *The vectors $p^i - p^j$ for all $(i,j) \in E(G)$ lie on a quadratic at infinity.*
2. *There exists a nonzero $y = (y_{ij}) \in \mathbb{R}^{\bar{m}}$ such that:*

$$V^T \mathscr{E}(y) Z = \mathbf{0}, \tag{1.17}$$

*where $V$ is defined in Eq. (1.8).*

*Proof.* Let $P$ be the configuration matrix of $G(p)$, and let $U$ and $W$ be the matrices whose columns form orthonormal bases for the null space and the column space of $X$, the projected Gram matrix of $G(p)$. Then by Lemma 1.3 we have

$$\begin{aligned}(p^i - p^j)^T \boldsymbol{\Phi}(p^i - p^j) &= (p^i)^T \boldsymbol{\Phi} p^i + (p^j)^T \boldsymbol{\Phi} p^j - 2(p^i)^T \boldsymbol{\Phi} p^j \\ &= (P\boldsymbol{\Phi}P^T)_{ii} + (P\boldsymbol{\Phi}P^T)_{jj} - 2(P\boldsymbol{\Phi}P^T)_{ij} \\ &= (VW\boldsymbol{\Phi}'W^TV^T)_{ii} + (VW\boldsymbol{\Phi}'W^TV^T)_{jj} - 2(VW\boldsymbol{\Phi}'W^TV^T)_{ij} \\ &= \mathscr{K}_V(W\boldsymbol{\Phi}'W^T)_{ij},\end{aligned}$$

where $\boldsymbol{\Phi}' = Q\boldsymbol{\Phi}Q^T$ for some nonsingular matrix $Q$, and where $\mathscr{K}_V$ is defined in Eq. (1.10).

Therefore, $p^i - p^j$ for all $(i,j) \in E(G)$ lie on a quadratic at infinity if and only if there exists a nonzero matrix $\boldsymbol{\Phi}'$ such that $H \circ \mathscr{K}_V(W\boldsymbol{\Phi}'W^T) = \mathbf{0}$. But since the set $\{M^{ij} : i \neq j, (i,j) \notin E(G)\}$ forms a basis for the null space of $H \circ \mathscr{K}_V$, it follows that vectors $p^i - p^j$ for all $(i,j) \in E(G)$ lie on a quadratic at infinity if and only if there exists a nonzero $r \times r$ matrix $\boldsymbol{\Phi}'$ and a nonzero $y = (y_{ij})$ in $\mathbb{R}^{\bar{m}}$ such that

$$W\boldsymbol{\Phi}'W^T = \sum_{ij:i\neq j,(i,j)\notin E(G)} y_{ij}M^{ij} = -\frac{1}{2} \sum_{ij:i\neq j,(i,j)\notin E(G)} y_{ij}V^TE^{ij}V = -\frac{1}{2}V^T\mathscr{E}(y)V. \tag{1.18}$$

Next we show that Eq. (1.18) is equivalent to Eq. (1.17). Suppose there exists a nonzero $y$ that satisfies Eq. (1.18). Then by multiplying Eq. (1.18) from the right by $U$ we have that $y$ also satisfies Eq. (1.17). Now suppose that there exists a nonzero $y$ that satisfies Eq. (1.17). Then

$$\begin{aligned}V^T\mathscr{E}(y)V &= [W\,U]\begin{bmatrix}W^T \\ U^T\end{bmatrix} V^T\mathscr{E}(y)V\ [W\,U]\begin{bmatrix}W^T \\ U^T\end{bmatrix}, \\ &= [W\,U]\begin{bmatrix}-2\boldsymbol{\Phi}' & 0 \\ 0 & 0\end{bmatrix}\begin{bmatrix}W^T \\ U^T\end{bmatrix}, \\ &= -2W\boldsymbol{\Phi}'W^T.\end{aligned}$$

Thus $y$ also satisfies Eq. (1.18) and the result follows. □

### *1.3.4 Miscellaneous Lemmas*

We conclude this section with the following lemmas that will be needed in our proofs. We begin with the following well-known Farkas lemma on the cone of positive semidefinite matrices.

**Lemma 1.6.** *Let $A^1,\dots,A^k$ be given $n \times n$ symmetric matrices. Then exactly one of the following two statements hold:*

1. *There exists $Y \succ \mathbf{0}$ such that trace $(A^iY) = 0$ for all $i = 1,\dots,k$.*
2. *There exists $x = (x_i) \in \mathbb{R}^k$ such that $x_1A^1 + \cdots + x_kA^k \succeq \mathbf{0}, \neq \mathbf{0}$.*

*Proof.* Assume that statement 1 does not hold, and let

$$\mathscr{L} = \{Y \in \mathscr{S}_n : \text{trace}(A^iY) = 0, \text{ for all } i = 1,\dots,k\}.$$

Then the subspace $\mathscr{L}$ is disjoint from the interior of the cone of $n \times n$ positive semidefinite matrices. By the separation theorem [19, page 96], there exists a nonzero symmetric matrix $\Theta$ such that trace$(\Theta Y) = 0$ for all $Y \in \mathscr{L}$ and trace$(\Theta C) \geq 0$ for all $C \succ \mathbf{0}$. Therefore, $\Theta \succeq \mathbf{0}$ and $\Theta = \sum_{i=1}^k x_iA^i$ for some nonzero $x = (x_i) \in \mathbb{R}^k$. Hence, statement 2 holds.

Now assume that statements 1 and 2 hold and let $\Theta = x_1A^1 + \cdots + x_kA^k$. Then on one hand, trace $(\Theta Y) > 0$; and on the other hand trace $(\Theta Y) = \sum_{i=1}^k x_i\text{trace}(A^iY) = 0$, a contradiction. Hence, the result follows. □

The following lemma shows that Gale matrices have a useful property under the general position assumption.

**Lemma 1.7.** *Let $G(p)$ be a bar framework on $n$ nodes in general position in $\mathbb{R}^r$ and let $Z$ be any Gale matrix of $G(p)$. Then any $\bar{r} \times \bar{r}$ sub-matrix of $Z$ is nonsingular.*

*Proof.* Assume $\bar{r} \leq r$. The proof of the case where $\bar{r} \geq r+1$ is similar. Let $Z'$ be any $\bar{r} \times \bar{r}$ sub-matrix of $Z$, and without loss of generality, assume that it is the sub-matrix defined by the rows $\bar{r}+1, \bar{r}+2,\dots,2\bar{r}$. Then, $Z'$ is singular if and only if there exists a nonzero $\xi \in \Re^{\bar{r}}$ such that $Z'\xi = \mathbf{0}$. Clearly, $Z\xi$ is in the null space of $\mathscr{P}$. Furthermore, $Z'\xi = \mathbf{0}$ if and only if the components $(Z\xi)_{\bar{r}+1} = (Z\xi)_{\bar{r}+2} = \ldots = (Z\xi)_{2\bar{r}} = \mathbf{0}$. Now since $Z\xi \neq \mathbf{0}$, this last statement holds if and only if the following $r+1$ points $p^1, p^2,\dots,p^{\bar{r}}, p^{2\bar{r}+1},\dots,p^n$ are affinely dependent, i.e., $G(p)$ is not in general position. □

## 1.4 Proofs

In this section we present the proofs of the theorems stated in Sect. 1.2.

## *Proof of Theorem 1.1*

Let $G(p)$ be a given framework on $n$ vertices in $\mathbb{R}^r$ for some $r \leq n-2$. Clearly, if $G(p)$ is universally rigid, then $G(p)$ is dimensionally rigid, and there does not exist a framework $G(p)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$.

To prove the other direction, let $X_p$ be the projected Gram matrix of $G(p)$. Let $Q = [W\ U]$ be the orthogonal matrix whose columns are the eigenvectors of $X_p$, where the columns of $U$ form an orthonormal basis for the null space of $X_p$.

Now suppose that $G(p)$ is not universally rigid. Then there exists a framework $G(q)$ in $\mathbb{R}^s$, which is equivalent, but not congruent, to $G(p)$, for some $s$: $1 \leq s \leq n-1$. Therefore, there exists a nonzero $\hat{y}$ in $\mathbb{R}^{\bar{m}}$ such that $X(\hat{y}) = X_p + \mathscr{M}(\hat{y}) \succeq \mathbf{0}$ where $\mathscr{M}(\hat{y}) = \sum_{(i,j)\notin E} \hat{y}_{ij} M^{ij}$. Now for a sufficiently small positive scalar $\delta$ we have[1]

$$X(t\hat{y}) = X_p + \mathscr{M}(t\hat{y}) \succeq \mathbf{0}, \text{ and } \operatorname{rank}(X(t\hat{y})) = \operatorname{rank}(X_p + \mathscr{M}(t\hat{y})) \geq r,$$

for all $t : 0 \leq t \leq \delta$. But

$$Q^T(X_p + \mathscr{M}(t\hat{y}))Q = \begin{bmatrix} \Lambda + tW^T\mathscr{M}(\hat{y})W & tW^T\mathscr{M}(\hat{y})U \\ tU^T\mathscr{M}(\hat{y})W & tU^T\mathscr{M}(\hat{y})U \end{bmatrix} \succeq \mathbf{0},$$

where $\Lambda$ is the $r \times r$ diagonal matrix consisting of the positive eigenvalues of $X_p$. Thus $U^T\mathscr{M}(\hat{y})U \succeq \mathbf{0}$ and the null space of $U^T\mathscr{M}(\hat{y})U \subseteq$ the null space of $W^T\mathscr{M}(\hat{y})U$.

Therefore, if rank $(X(t_0\hat{y})) \geq r+1$ for some $0 < t_0 \leq \delta$ we have a contradiction since $G(p)$ is dimensionally rigid. Hence, rank $(X(t\hat{y})) = r$ for all $t : 0 \leq t \leq \delta$. Thus, both matrices $U^T\mathscr{M}(\hat{y})U$ and $W^T\mathscr{M}(\hat{y})U$ must be zero. This implies that $\mathscr{M}(\hat{y})U = \mathbf{0}$, i.e., $V^T\mathscr{E}(\hat{y})Z = \mathbf{0}$ which is also a contradiction by Lemma 1.5. Therefore, $G(p)$ is universally rigid. □

## *Proof of Theorem 1.2*

Let $G(p)$ be a given framework on $n$ vertices in $\mathbb{R}^r$ for some $r \leq n-2$ and let $Z$ be a Gale matrix of $G(p)$. Let $X_p$ be the projected Gram matrix of $G(p)$, and let $Q = [W\ U]$ be the orthogonal matrix whose columns are the eigenvectors of $X_p$, where the columns of $U$ form an orthonormal basis for the null space of $X_p$.

Assume that $G(p)$ admits a positive semidefinite stress matrix $S$ of rank $n-r-1$. Therefore, there exists a positive definite symmetric matrix $\Psi$ such that $(z^i)^T\Psi z^j = 0$ for all $ij : i \neq j, (i,j) \notin E(G)$. Hence, by lemma 1.6, there does not exist $y = (y_{ij}) \in \mathbb{R}^{\bar{m}}$ such that $\sum_{ij:i\neq j,(i,j)\notin E(G)} y_{ij}(z^i(z^j)^T + z^j(z^i)^T)$ is a non zero

[1] the rank function is lower semi-continuous on the set of matrices of order $n-1$.

positive semidefinite matrix. But $z^i(z^j)^T + z^j(z^i)^T = Z^T E^{ij} Z$. Thus, there does not exist $y = (y_{ij}) \in \mathbb{R}^{\bar{m}}$ such that $Z^T \mathscr{E}(y) Z$ is a nonzero positive semidefinite matrix. Hence, there does not exist $y = (y_{ij}) \in \mathbb{R}^{\bar{m}}$ such that $U^T \mathscr{M}(y) U$ is a nonzero positive semidefinite matrix.

Now assume that $G(p)$ is not dimensionally rigid, then there exists a nonzero $y$ such that $X = X_p + \mathscr{M}(y) \succeq \mathbf{0}$ and rank $X \geq r+1$. But

$$Q^T(X_p + \mathscr{M}(y))Q = \begin{bmatrix} \Lambda + W^T \mathscr{M}(y) W & W^T \mathscr{M}(y) U \\ U^T \mathscr{M}(y) W & U^T \mathscr{M}(y) U \end{bmatrix} \succeq \mathbf{0}.$$

Since $\Lambda + W^T \mathscr{M}(y) W$ is $r \times r$, it follows that $U^T \mathscr{M}(y) U$ is a nonzero positive semidefinite, a contradiction. □

## *Proof of Theorem 1.4*

We begin with the following lemma.

**Lemma 1.8 (Connelly [10]).** *Let $G(p)$ be a generic bar framework on $n$ vertices in $\mathbb{R}^r$. Assume that each node of $G$ has degree at least $r$. Then the vectors $p^i - p^j$ for all $(i,j) \in E(G)$ do not lie on a quadratic at infinity.*

Now let $G(p)$ be a generic bar framework on $n$ vertices in $\mathbb{R}^r$. If $G(p)$ admits a positive semidefinite stress matrix of rank $n-r-1$, then each vertex of $G$ has degree at least $r+1$ [2, Theorem 3.2]. Thus Theorem 1.4 follows from Lemmas 1.4 and 1.8 and Theorem 1.3.

## *Proof of Theorem 1.6*

The main idea of the proof is to show that Condition 2 of Lemma 1.5 does not hold under the assumptions of the theorem. The choice of the particular Gale matrix to be used in Eq. (1.17) is critical in this regard. The proof presented here is that given in [7].

Let $\bar{N}(i)$ denote the set of nodes of graph $G$ that are nonadjacent to node $i$, i.e.,

$$\bar{N}(i) = \{j \in V(G) : j \neq i \text{ and } (i,j) \notin E(G)\}. \tag{1.19}$$

**Lemma 1.9.** *Let $G(p)$ be a bar framework on $n$ nodes in general position in $\mathbb{R}^r$, $r \leq n-2$. Assume that $G(p)$ has a stress matrix $S$ of rank $n-1-r$. Then there exists a Gale matrix $\hat{Z}$ of $G(p)$ such that $\hat{z}_{ij} = 0$ for all $j = 1,\ldots,\bar{r}$ and $i \in \bar{N}(j+r+1)$.*

*Proof.* Let $G(p)$ be in general position in $\mathbb{R}^r$ and assume that it has a stress matrix $S$ of rank $\bar{r} = (n-1-r)$. Let $Z$ be any Gale matrix of $G(p)$, then it follows from

Eq. (1.7) that $S = Z\Psi Z^T$ for some nonsingular symmetric $\bar{r} \times \bar{r}$ matrix $\Psi$. Let us write $Z$ as:

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix},$$

where $Z_2$ is $\bar{r} \times \bar{r}$. Then it follows from Lemma 1.7 that $Z_2$ is nonsingular. Now let

$$\hat{Z} = (\hat{z}_{ij}) = Z\Psi Z_2{}^T. \tag{1.20}$$

Then $\hat{Z}$ is a Gale matrix of $G(p)$ since both $\Psi$ and $Z_2$ are nonsingular. Furthermore,

$$S = Z\Psi Z^T = Z\Psi\,[Z_1^T \;\; Z_2^T] = [Z\Psi Z_1^T \;\; \hat{Z}].$$

In other words, $\hat{Z}$ consists of the last $\bar{r}$ columns of $S$. Thus $\hat{z}_{ij} = s_{i,j+r+1}$. It follows by the definition of $S$ that $s_{i,j+r+1} = 0$ for all $i, j$ such that $i \neq (j+r+1)$ and $(i, j+r+1) \notin E(G)$. Therefore, $\hat{z}_{ij} = 0$ for all $j = 1, \ldots, \bar{r}$ and $i \in \bar{N}(j+r+1)$. □

**Lemma 1.10.** *Let the Gale matrix in Eq. (1.17) be $\hat{Z}$ as defined in Eq. (1.20). Then the system of Eq. (1.17) is equivalent to the system of equations*

$$\mathscr{E}(y)\hat{Z} = \mathbf{0}. \tag{1.21}$$

*Proof.* System of Eq. (1.17) is equivalent to the following system of equations in the unknowns, $y_{ij}$ ($i \neq j$ and $(i,j) \notin E(G)$) and $\xi = (\xi_j) \in \mathbb{R}^{\bar{r}}$:

$$\mathscr{E}(y)\hat{Z} = e\,\xi^T. \tag{1.22}$$

Now for $j = 1, \ldots, \bar{r}$, we have that the $(j+r+1, j)$th entry of $\mathscr{E}(y)\hat{Z}$ is equal to $\xi_j$. But using Eq. (1.16) and Lemma 1.9 we have

$$(\mathscr{E}(y)\hat{Z})_{j+r+1,j} = \sum_{i=1}^{n} \mathscr{E}(y)_{j+r+1,i}\,\hat{z}_{ij} = \sum_{i:i \in \bar{N}(j+r+1)} y_{j+r+1,i}\,\hat{z}_{ij} = 0.$$

Thus, $\xi = 0$ and the result follows. □

**Lemma 1.11.** *Let $G(p)$ be a bar framework on $n$ nodes in general position in $\mathbb{R}^r$, $r \leq n-2$. Assume that $G(p)$ has a positive semidefinite stress matrix $S$ of rank $\bar{r} = n-1-r$. Then there does not exist a framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$.*

*Proof.* Under the assumption of the lemma, we have that $\deg(i) \geq r+1$ for all $i \in V(G)$, i.e., every node of $G$ is adjacent to at least $r+1$ nodes (for a proof see [2, Theorem 3.2]). Thus

$$|\bar{N}(i)| \leq n - r - 2 = \bar{r} - 1 \text{ for all } i \in V(G). \tag{1.23}$$

Furthermore, it follows from Lemmas 1.9, 1.10, and 1.5 that the vectors $p^i - p^j$ for all $(i,j) \in E(G)$ lie on a quadratic at infinity if and only if system of Eq. (1.21) has a nonzero solution $y$. But Eq. (1.21) can be written as

$$\sum_{j: \in \bar{N}(i)} y_{ij} \hat{z}^j = 0, \text{ for } i = 1, \ldots, n,$$

where $(\hat{z}^i)^T$ is the $i$th row of $\hat{Z}$. Now it follows from Eq. (1.23) that $y_{ij} = 0$ for all $(i,j) \not\in E(G)$ since by Lemma 1.7 any subset of $\{\hat{z}^1, \ldots, \hat{z}^n\}$ of cardinality $\leq \bar{r} - 1$ is linearly independent.

Thus system (1.21) does not have a nonzero solution $y$. Hence the vectors $p^i - p^j$, for all $(i,j) \in E(G)$, do not lie on a quadratic at infinity. Therefore, by Lemma 1.4, there does not exist a framework $G(q)$ in $\mathbb{R}^r$ that is affinely equivalent, but not congruent, to $G(p)$. □

Thus, Theorem 1.6 follows from Lemma 1.11 and Theorem 1.3.

## *Proof of Theorem 1.7*

The proof of Theorem 1.7 is constructive, i.e., an algorithm is presented to construct the desired stress matrix. The proof presented here is a slight modification of that given in [6].

Let $G(p)$ be a framework on $n$ vertices in general position in $\mathbb{R}^r$, $n \geq r+2$, and let $Z$ be a Gale matrix of $G(p)$. An $n \times n$ symmetric matrix $S$ that satisfies

$$\mathscr{P} S = 0, \text{ or equivalently } S = Z \Psi Z^T \text{ for some symmetric matrix } \Psi,$$

is called a *prestress matrix*, where $\mathscr{P}$ is defined in Eq. (1.4). Thus, it follows from Eqs. (1.6) and (1.7) that $S$ is a stress matrix of $G(p)$ if and only if $S$ is a prestress matrix and $s_{ij} = 0$ for all $ij : i \neq j$, $(i,j) \not\in E(G)$.

Clearly, $S^n = ZZ^T$ is a positive semidefinite prestress matrix of rank $\bar{r} = n - r - 1$. If $S^n$ satisfies $s^n_{ij} = 0$ for all $ij : i \neq j$, $(i,j) \not\in E(G)$, then we are done since $S^n$ is the desired stress matrix. Otherwise, if $S^n$ is not a stress matrix, we need to zero out the entries which should be zero but are not, i.e., the entries $s^n_{ij} \neq 0$, $i \neq j$ and $(i,j) \not\in E(G)$. We do this in reverse order by column (row); first, we zero out the entries $s^n_{in} \neq 0$, for $i < n$ and $(i,n) \not\in E(G)$, and then do the same for columns (rows) $(n-1), (n-2), \ldots, (r+3)$. This "purification" process will keep the prestress matrix positive semidefinite and maintain rank $n - r - 1$.

Let $G$ be an $(r+1)$-lateration graph with lateration order $1, 2, \ldots, n$, i.e., the vertices, $1, 2, \ldots, r+2$, induce a clique in $G$, and each remaining vertex $k$, for $k = r+3, \ldots, n$, is adjacent to $(r+1)$ vertices in the set $\{1, 2, \ldots, k-1\}$. Let

$$\bar{N}'(k) = \{i \in V(G) : i < k \text{ and } (i,k) \not\in E(G)\}. \tag{1.24}$$

Then for $k = r+3,\dots,n$,

$$|\bar{N}'(k)| = k-r-2. \tag{1.25}$$

We first show how to purify the last column (or row) of $S^n = ZZ^T$. Let $Z^n$ denote the sub-matrix of $Z$ obtained by keeping only rows with indices in $\bar{N}'(n) \cup \{n\}$. Then $Z^n$ is a square matrix of order $\bar{r} = n-r-1$. Furthermore, by Lemma 1.7, it follows that $Z^n$ is nonsingular. Let $b^n$ denote the vector in $\mathbb{R}^{\bar{r}}$ such that

$$b_i^n = \begin{cases} -s_{in}^n & \text{if } i \in \bar{N}'(n), \\ 1 & \text{if } i = n. \end{cases}$$

Now let $\xi_n \in \mathbb{R}^{\bar{r}}$ be the unique solution of the system of equations

$$Z^n \xi_n = b^n.$$

**Lemma 1.12.** *Let $S^{n-1} = S^n + Z\xi_n{\xi_n}^T Z^T = Z(I + \xi_n {\xi_n}^T)Z^T$. Then*

1. *$S^{n-1}$ is a prestress matrix of $G(p)$, i.e., $\mathscr{P}S^{n-1} = 0$.*
2. *$S^{n-1} \succeq \mathbf{0}$ and the rank of $S^{n-1}$ remains $n-r-1$.*
3. *$s_{in}^{n-1} = 0$ for all $i : i < n,\ (i,n) \notin E(G)$.*

*Proof.* The first statement is obvious. The second statement follows since $I + \xi_n \xi_n^T \succ \mathbf{0}$. The third statement is also true by construction. For all $i < n,\ (i,n) \notin E(G)$, i.e., for all $i \in \bar{N}'(n)$, we have $s_{in}^{n-1} = s_{in}^n + b_i^n b_n^n = s_{in}^n - s_{in}^n = 0$. □

We continue this purification process for columns $(n-1),\dots,k,\dots,(r+3)$. Before the $k$th purification step, we have $S^k \succeq \mathbf{0}$, $\mathscr{P}S^k = 0$, rank $S^k = n-r-1$, and

$$s_{ij}^k = 0, \text{ for all } ij : i \neq j,\ (i,j) \notin E(G), \text{ and for all } j = k+1,\dots,n.$$

Let $Z^k$ denote the sub-matrix of $Z$ obtained by keeping only rows with indices in $\bar{N}'(k) \cup \{k, k+1,\dots,n\}$. Then $Z^k$ is a square matrix of order $\bar{r} = n-r-1$. Furthermore, by Lemma 1.7, it follows that $Z^k$ is nonsingular. Let $b^k$ denote the vector in $\mathbb{R}^{\bar{r}}$ such that

$$b_i^k = \begin{cases} -s_{ik}^k & \text{if } i \in \bar{N}'(k), \\ 1 & \text{if } i = k, \\ 0 & \text{if } i = k+1,\dots,n. \end{cases}$$

Now let $\xi_k \in \mathbb{R}^{\bar{r}}$ be the unique solution of the system of equations

$$Z^k \xi_k = b^k.$$

The following lemma shows results analogous to those in Lemma 1.12, for the remaining columns.

**Lemma 1.13.** *Let $S^{k-1} = S^k + Z\xi_k\xi_k^T Z^T$. Then*

1. *$S^{k-1}$ is a prestress matrix of $G(p)$, i.e., $\mathscr{P}S^{k-1} = 0$.*
2. *$S^{k-1} \succeq \mathbf{0}$ and the rank of $S^{k-1}$ remains $n-r-1$.*
3. *$s_{ij}^{k-1} = 0$ for all $i : i < j$, $(i,j) \notin E(G)$ and for all $j = k,\dots,n$.*

*Proof.* The proof of the first two statements is identical to that in Lemma 1.12. The third statement is again true by construction. For each $i < k$, $(i,k) \notin E(G)$, i.e., for all $i \in \bar{N}'(k)$, we have

$$s_{ik}^{k-1} = s_{ik}^k + b_i^k b_k^k = s_{ik}^k - s_{ik}^k = 0.$$

Furthermore, for $j = k+1,\dots,n$, the $j$th column (or row) of $Z\xi_k\xi_k^T Z^T$ has all zero entries, which means that the entries in the $j$th column (or row) of $S^{k-1}$ remain unchanged from $S^k$. □

## *Now we are ready to prove Theorem 1.7*

The matrix

$$S^{r+2} = S^{r+3} + Z\xi_{r+3}\xi_{r+3}^T Z^T = Z(I + \xi_n\xi_n^T + \cdots + \xi_{r+3}\xi_{r+3}^T)Z^T,$$

obtained at the "$(r+3)$th" step of the above process, is by Lemmas 1.12 and 1.13 a positive semidefinite prestress matrix of rank $n-r-1$. Furthermore, $s_{ij}^{r+2} = 0$ for all $ij : i \neq j$, $(i,j) \notin E(G)$ and for all $j = r+3, r+4, \dots, n$. But since the vertices $1,2,\dots,r+2$ induce a clique in $G$, it follows that

$$s_{ij}^{r+2} = 0 \text{ for all } ij : i \neq j,\ (i,j) \notin E(G).$$

Hence, $S = S^{r+2}$ is a positive semidefinite stress matrix of $G(p)$ of rank $n-r-1$, i.e., $S^{r+2}$ is the desired stress matrix.

**Acknowledgements** Research supported by the Natural Sciences and Engineering Research Council of Canada.

## References

1. Alfakih, A.Y.: On rigidity and realizability of weighted graphs. Lin. Algebra. Appl. **325**, 57–70 (2001)
2. Alfakih, A.Y.: On dimensional rigidity of bar-and-joint frameworks. Discrete. Appl. Math. **155**, 1244–1253 (2007)
3. Alfakih, A.Y.: On the dual rigidity matrix. Lin. Algebra. Appl. **428**, 962–972 (2008)
4. Alfakih, A.Y.: On the universal rigidity of generic bar frameworks. Contrib. Discrete. Math. **5**, 7–17 (2010)

5. Alfakih, A.Y., Khandani, A., Wolkowicz, H.: Solving Euclidean distance matrix completion problems via semidefinite programming. Comput. Optim. Appl. **12**, 13–30 (1999)
6. Alfakih, A.Y., Taheri, N., Ye, Y.: On stress matrices of $(d+1)$-lateration frameworks in general position, to appear in Mathematical Programming (2012)
7. Alfakih, A.Y., Ye, Y: On affine motions and bar frameworks in general positions. Lin. Algebra. Appl. arXiv 1009.3318 (2010)
8. Connelly, R.: Rigidity and energy, Invent. Math. **66**, 11–33 (1982)
9. Connelly, R.: Tensegrity structures: Why are they stable? In: Thorpe, M.F., Duxbury, P.M. (eds.) Rigidity Theory and Applications, pp. 47–54. Kluwer Academic/Plenum Publishers (1999)
10. Connelly, R.: Generic global rigidity. Discrete. Comput. Geom. **33**, 549–563 (2005)
11. Critchley, F.: On certain linear mappings between inner-product and squared distance matrices. Lin. Algebra. Appl. **105**, 91–107 (1988)
12. Eren, T., Goldenberg, D.K., Whiteley, W., Yang, Y.R., Morse, A.S., Anderson, B.D.O., Belhumeur, P.N.: Rigidity, computation and randomization in network localization. In: IEEE Conference Proceedings, INFOCOM proceedings, pp. 2673–2684 (2004)
13. Gale, D.: Neighboring vertices on a convex polyhedron. In: Kuhn H.W., Tucker, A.W. (Eds.), Linear Inequalities and Related System, pp. 255–263. Princeton University Press, Princeton (1956)
14. Gortler, S.J., Healy, A.D., Thurston, D.P.: Characterizing generic global rigidity, Amer. J. Math. **132**, 897–939 (2010)
15. Gortler, S.J., Thurston, D.P.: Characterizing the universal rigidity of generic frameworks, arXiv/1001.0172v1 (2009)
16. Gower, J.C.: Properties of Euclidean and non-Euclidean distance matrices. Lin. Algebra. Appl. **67**, 81–97 (1985)
17. Grünbaum, B.: Convex Polytopes. Wiley, London (1967)
18. Lavor, C., Lee, J., Lee-St.John, A., Liberti, L., Mucherino, A., Sviridenko, M.: Discretization orders for distance geometry problems. Optim. Lett. **6**, 783–796 (2012)
19. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)
20. Schoenberg, I.J.: Remarks to Maurice Fréchet's article: Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert. Ann. Math. **36**, 724–732 (1935)
21. Young G., Householder, A.S.: Discussion of a set of points in terms of their mutual distances. Psychometrika **3**, 19–22 (1938)
22. Zhu, Z., So, A.M-C., Ye, Y.: Universal rigidity: Towards accurate and efficient localization of wireless networks. In: IEEE Conference Proceedings, INFOCOM proceedings (2010)

# Chapter 2
# Mixed Volume and Distance Geometry Techniques for Counting Euclidean Embeddings of Rigid Graphs

**Ioannis Z. Emiris, Elias P. Tsigaridas, and Antonios Varvitsiotis**

**Abstract** A graph $G$ is called generically minimally rigid in $\mathbb{R}^d$ if, for any choice of sufficiently generic edge lengths, it can be embedded in $\mathbb{R}^d$ in a finite number of distinct ways, modulo rigid transformations. Here, we deal with the problem of determining tight bounds on the number of such embeddings, as a function of the number of vertices. The study of rigid graphs is motivated by numerous applications, mostly in robotics, bioinformatics, sensor networks, and architecture. We capture embeddability by polynomial systems with suitable structure so that their mixed volume, which bounds the number of common roots, yields interesting upper bounds on the number of embeddings. We explore different polynomial formulations so as to reduce the corresponding mixed volume, namely by introducing new variables that remove certain spurious roots and by applying the theory of distance geometry. We focus on $\mathbb{R}^2$ and $\mathbb{R}^3$, where Laman graphs and 1-skeleta (or edge graphs) of convex simplicial polyhedra, respectively, admit inductive Henneberg constructions. Our implementation yields upper bounds for $n \leq 10$ in $\mathbb{R}^2$ and $\mathbb{R}^3$, which reduce the existing gaps and lead to tight bounds for $n \leq 7$ in both $\mathbb{R}^2$ and $\mathbb{R}^3$; in particular, we describe the recent settlement of the case of Laman graphs with seven vertices. Our approach also yields a new upper bound for Laman graphs with eight vertices, which is conjectured to be tight. We also establish the first lower bound in $\mathbb{R}^3$ of about $2.52^n$, where $n$ denotes the number of vertices.

I.Z. Emiris (✉)
University of Athens, Athens, Greece
e-mail: emiris@di.uoa.gr

E.P. Tsigaridas
INRIA Paris-Rocquencourt, Paris, France
e-mail: elias@polsys.lip6.fr

A. Varvitsiotis
Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands
e-mail: A.Varvitsiotis@cwi.nl

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_2,

**Keywords** Rigid graph • Laman graph • Euclidean embedding • Henneberg construction • Polynomial system • Mixed volume • Cayley–Menger matrix • Cyclohexane caterpillar.

## 2.1 Introduction

Rigid graphs (or mechanisms) constitute an old but still very active area of research due to their deep mathematical and algorithmic questions, as well as numerous applications, notably in mechanism and linkage theory [10,23,44,45,48], structural bioinformatics [19, 32, 35, 42], Sensor Network Localization [22, 33, 49], and architecture [21,25].

We start this section by introducing notation and some necessary definitions. Throughout, $G=(V,E)$ denotes a simple loop-less graph with $|V|=n$. A *framework* of $G$ consists of an assignment of vectors $p_1,\dots,p_n\in\mathbb{R}^d$ to the nodes of the graph and is denoted by $G(p)$. Two frameworks $G(p)$ and $G(q)$ are called *equivalent* if $\|p_i-p_j\|=\|q_i-q_j\|$, $\forall(i,j)\in E$, and they are called *congruent* if $\|p_i-p_j\|=\|q_i-q_j\|$, $\forall i,j\in V$. A framework $G(p)$ is called *rigid* if there exists an $\varepsilon>0$ such that if $\|p-q\|<\varepsilon$ and $G(q)$ is equivalent to $G(p)$ then they are congruent. Equivalently, a framework $G(p)\in\mathbb{R}^d$ is called rigid if (the orbit of) $p$ is an isolated point in the orbit space of the group of Euclidean motions acting on the real variety $\mathbb{V}(p)=\{x\in\mathbb{R}^{dn}:\|x_i-x_j\|^2=\|p_i-p_j\|^2,\ \forall(i,j)\in E\}$. It is a well-known fact that for real closed semi-algebraic sets the sum of the Betti numbers is finite, so in particular the number of connected components is finite [2]. This implies that for a given rigid framework $G(p)$ there exists a finite number of frameworks which are equivalent but not congruent to it.

A framework $G(p)$ is called *generic* if its coordinates are algebraically independent over $\mathbb{Q}$. A graph $G$ is called *generically rigid* if every generic framework of $G$ is rigid. Equivalently, a graph is *generically rigid* in $\mathbb{R}^d$ if, for generic edge lengths, it can be embedded in $\mathbb{R}^d$ in a finite number of ways, modulo congruence transformations. Here, a congruence transformation refers to either a translation or a rotation. A graph is *minimally rigid* if it is no longer rigid once any edge is removed. The problem of interest in this chapter is to determine the maximum number of distinct planar and spatial Euclidean embeddings of generically minimally rigid, or simply rigid, graphs, up to rigid transformations, as a function of the number of vertices.

A graph $G=([n],E)$, $[n]=\{1,\dots,n\}$, is called *Laman* if $|E|=2n-3$, and, additionally, all of its vertex-induced subgraphs with $3\le k<n$ vertices to have less than $2k-3$ edges. This is related to the Chebychev–Grübler–Kutzbach's formula on the degrees of freedom for mechanical linkages, e.g., [1]. It is a fundamental theorem that the class of Laman graphs coincides with the generically minimally rigid graphs in $\mathbb{R}^2$ [34,38].

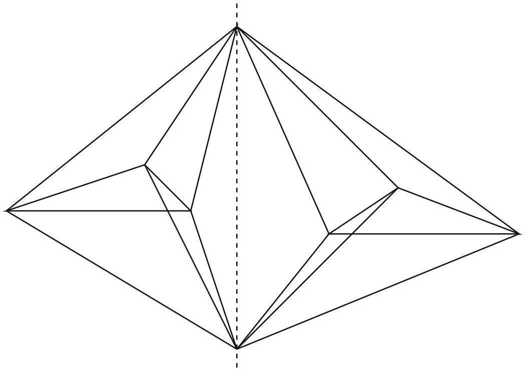


**Fig. 2.1** The double banana; a nonrigid graph which satisfies the Laman property for $d = 3$

On the other hand, an analogous combinatorial characterization of rigidity in $\mathbb{R}^3$ has proven to be elusive and remains one of the most important open problems in the area of rigidity theory. In particular, the natural generalization of Laman's property in $\mathbb{R}^3$, i.e., $|E| = 3n - 6$, and all vertex-induced subgraphs with $4 \leq k < n$ vertices have less than $3k - 6$ edges and are no longer sufficient to characterize minimal rigidity in three dimensions. The famous counterexample of the double banana (Fig. 2.1) is a nonrigid graph in $\mathbb{R}^3$ which satisfies the necessary conditions mentioned above.

Thus we restrict our attention to graphs that correspond to the one-skeleta, or edge graphs, of (convex) simplicial polyhedra; clearly, these graphs admit one convex embedding (modulo reflections as well). They are known to be generically minimally rigid in $\mathbb{R}^3$ [24].

Both Laman graphs and the one-skeleta of simplicial polyhedra admit inductive constructions that begin with a simplex of the appropriate dimension, followed by a sequence of so-called Henneberg steps [47]. There are $d$ types of such steps in $\mathbb{R}^d$, each adding one new vertex and increasing the total number of edges by $d$, for $d = 2, 3$. They will be described in the sequel. A graph is Laman, or the one-skeleton of a simplicial polytope, if it can be constructed by a sequence of the corresponding Henneberg steps.

To study upper bounds, we define a well-constrained polynomial system, i.e., polynomial systems with as many equations as unknowns, expressing the edge length constraints, whose real solutions correspond precisely to the different embeddings. When defining a straightforward system such as Eqs. (2.1) and (2.6) in $\mathbb{R}^2$ and $\mathbb{R}^3$, respectively, all nontrivial equations are quadratic. There are $2n - 4$ and $3n - 9$ equations, respectively; hence, by applying the classical Bézout bound on the number of common roots, we obtain $4^{n-2}$ and $8^{n-3}$. It is indicative of the hardness of the problem that efforts to substantially improve these bounds have failed [6, 41].

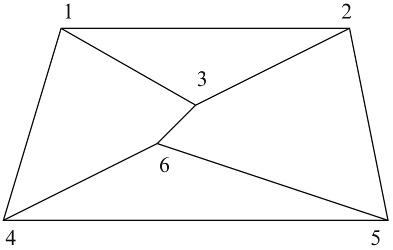


**Fig. 2.2** The Desargues, or 3-prism, graph; equivalently, a planar parallel robot

### *2.1.1 Existing Work*

The bound on the number of embeddings in dimension $d$ is

$$\prod_{k=0}^{n-d-2} \frac{\binom{n-1+k}{n-d-1-k}}{\binom{2k+1}{k}}$$

and is tight if we consider embeddings in $\mathbb{C}^d$. The bound was obtained by exploiting results from complex algebraic geometry that bound the degree of (symmetric) determinantal varieties defined by distance matrices [5, 6, 29]. For the planar and spatial case, it follows that the best known upper bounds are, respectively:

$$\binom{2n-4}{n-2} \simeq 4^{n-2}/\sqrt{\pi(n-2)} \text{ and } \frac{2^{n-3}}{n-2}\binom{2n-6}{n-3} \simeq 8^{n-3}/\left((n-2)\sqrt{\pi(n-3)}\right).$$

In [41], mixed volumes (cf. Sect. 2.2) also yield an upper bound of $4^{n-2}$, for Laman graphs.

In applications, it is crucial to know the number of embeddings for specific (small) values of $n$. The most important result in this direction was to show that the Desargues graph admits precisely 24 embeddings in the plane [26, 31]. In different communities, this graph is known as the planar parallel robot, or the three-prism graph. It is also known that the $K_{3,3}$ graph admits 16 embeddings in the plane [44], and the cyclohexane graph admits 16 embeddings in space (Sect. 2.5).

Another important question concerns lower bounds. For $\mathbb{R}^2$, there exist a lower bound of $24^{\lfloor (n-2)/4 \rfloor} \simeq 2.21^n$, obtained by a caterpillar construction, and one of $2 \cdot 12^{\lfloor (n-3)/3 \rfloor} \simeq 2.29^n/6$, obtained by a fan[1] construction [6]. Both bounds are based on the Desargues graph (Fig. 2.2), which admits 24 embeddings. This bound has been slightly improved to $\Omega(2.3^n)$ [18], by using a construction which is based on the seven-vertex graph (Fig. 2.5), which admits 56 embeddings. For the exact number of embeddings for certain rigid graphs, based on the Henneberg-1 steps we refer the reader to [36].

[1] This corrects the exponent of the original statement.

**Table 2.1** Bounds and Henneberg sequences for Laman graphs for $n \leq 10$

| $n =$ | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|
| Upper | 2 | 4 | 8 | 24 | 56 | 122 | 512 | 2048 |
| Lower | 2 | 4 | 8 | 24 | 56 | 112 | 288 | 576 |
| $H_1$ | $\triangle$ | $\triangle 1$ | $\triangle 11$ | $\triangle 111$ | $\triangle 1^4$ | $\triangle 1^5$ | $\triangle 1^6$ | |
| | | | | $\triangle\mathbf{112}$ | $\triangle\mathbf{1^3 2}$ | $\triangle\mathbf{1^4 2}$ | $\triangle 1^5 2$ | |
| $H_2$ | | | | | | | $\triangle 1^4 21$ | |
| | | | | | | | $\triangle\mathbf{1^4 22}$ | |

Exponents indicate the number of times a certain Henneberg step is repeated
Bold text indicates the simplest Henneberg sequence yielding the upper bound

### *2.1.2 New Results*

To upper bound the number of Euclidean embeddings of rigid graphs, we explore adequate polynomial systems leading to tight root bounds. This is a deep and hard question, with a wide range of applications in different fields. In the sequel, we shall shed some light to this issue. Our main tool is the mixed volume of a well-constrained polynomial system, which exploits the sparseness of the equations. This bounds the number of common roots, by Bernstein's Theorem 2.1, as described in Sect. 2.2. This bound is never larger than Bézout's and is typically much tighter.

An alternative is to actually solve a system with coefficients chosen randomly and count the real roots, though this only yields an indication on the upper bound. The advantage of mixed volume is that it treats entire classes of systems defined by their nonzero terms, without considering specific coefficient values. In addition, a tight mixed volume implies that one can solve the system efficiently, either by sparse resultants [8] or by sparse homotopies [43]. More precisely, this means, respectively, that the sparse resultant matrix, or the number of homotopy paths, shall be (close to) optimal.

In the sequel, we derive the first lower bound in $\mathbb{R}^3$:

$$16^{\lfloor (n-3)/3 \rfloor} \simeq 2.52^n, n \geq 9,$$

by designing a cyclohexane caterpillar (Fig. 2.10). Moreover, we have implemented specialized software that constructs all rigid graphs up to isomorphism, for small $n$, and computes the mixed volumes of the respective polynomial systems. We thus obtain upper and lower bounds for $n \leq 10$ in $\mathbb{R}^2$ and $\mathbb{R}^3$, which reduce the existing gaps; see Tables 2.1 and 2.2. Moreover, we establish tight bounds up to $n = 7$ in $\mathbb{R}^2$ and $\mathbb{R}^3$ by appropriately reformulating the polynomial system. We describe in detail the case of seven-vertex Laman graphs, also known as 11-bar mechanisms in robotics, and establish a tight upper bound by distance geometry. We apply Bernstein's Second theorem (Theorem 2.2) to show that the naive polynomial system (2.6) cannot yield tight mixed volumes in the spatial case. Our approach also

**Table 2.2** Bounds and Henneberg sequences for one-skeleta of simplicial polyhedra for $n \leq 10$, where $\triangle$ is the three-simplex and bold are the Henneberg sequences yielding the upper bound

| $n =$ | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|
| upper | 2 | 4 | 16 | 32 | 160 | 640 | 2560 |
| lower | 2 | 4 | 16 | 32 | 64 | 256 | 512 |
| $H_1$ | $\triangle$ | $\triangle 1$ | $\triangle 11$ | $\triangle 111$ | $\triangle 1^4$ | $\triangle 1^5$ | $\triangle 1^6$ |
| | | | $\triangle\mathbf{12}$ | $\triangle\mathbf{1^2 2}$ | $\triangle 1^3 2$ | $\triangle 1^4 2$ | $\triangle 1^5 2$ |
| | | | | | $\triangle\mathbf{1^2 2^2}$ | $\triangle 1^3 2^2$ | $\triangle 1^4 2^2$ |
| $H_2$ | | | | | $\triangle 1^2 21$ | $\triangle 1^3 21$ | $\triangle 1^4 21$ |
| | | | | | | $\triangle\mathbf{1^2 2^3}$ | $\triangle 1^3 2^3$ |
| | | | | | | | $\triangle 1^3 21^2$ |
| | | | | | | | $\triangle 1^3 212$ |
| | | | | | | | $\triangle 1^3 2^2 1$ |
| | | | | | | | $\triangle\mathbf{1^2 2^4}$ |
| $H_3$ | | | | | | | |

yields a new upper bound for Laman graphs with eight vertices, which is conjectured to be tight. Our results indicate that mixed volume can be of general interest in enumeration problems.

The rest of the chapter is structured as follows. Section 2.2 presents our algebraic tools and our implementation, Sect. 2.3 discusses the planar case ($d = 2$), Sect. 2.4 outlines the theory of distance geometry and applies it to Laman graphs with $n = 6, 7$, Sect. 2.5 deals with $\mathbb{R}^3$, and we conclude with open questions and a conjecture. Several results appeared in [20] in preliminary form, whereas the tight count for seven-vertex Laman graphs was established in [18].

## 2.2 Polynomial Systems and Mixed Volume

This section discusses multivariate polynomial systems, introduces mixed volume, and describes our software.

Classical elimination theory characterizes every polynomial by its total degree. For a well-constrained system of polynomial equations, the classical Bézout bound on the number of isolated roots equals the product of the polynomials' total degrees. One disadvantage of this bound is that it counts complex projective roots and hence increases when there are roots at projective infinity.

We introduce sparse elimination theory in order to exploit sparseness; for details, see [11]. In sparse (or toric) elimination theory, a polynomial is characterized by its support. Given a polynomial $f$ in $n$ variables, its support is the set of exponents in $\mathbb{N}^n$ corresponding to nonzero terms (or monomials). The Newton polytope of $f$ is the convex hull of its support and lies in $\mathbb{R}^n$. Consider polytopes $P_i \subset \mathbb{R}^n$ and parameters $\lambda_i \in \mathbb{R}, \lambda_i \geq 0$, for $i = 1, \dots, n$. We denote by $\lambda_i P_i$ the corresponding scalar multiple of $P_i$. Consider the Minkowski sum of the scaled polytopes $\lambda_1 P_1 + \cdots + \lambda_n P_n \in \mathbb{R}^n$;

its $n$-dimensional (Euclidean) volume is a homogeneous polynomial of degree $n$ in the $\lambda_i$. The coefficient of the monomial $\lambda_1 \cdots \lambda_n$ is the *mixed volume* of $P_1, \dots, P_n$. If $P_1 = \cdots = P_n$, then the mixed volume is $n!$ times the volume of $P_1$. We focus on the topological *torus* $\mathbb{C}^* = \mathbb{C} - \{0\}$ in order to state the so-called BKK root bound, in terms of mixed volume.

**Theorem 2.1 ([3]).** *Let $f_1 = \cdots = f_n = 0$ be a polynomial system in n variables with real coefficients. Assume that the $f_i$ have fixed Newton polytopes; hence all coefficients corresponding to polytope vertices are nonzero. Then, the number of common isolated solutions in $(\mathbb{C}^*)^n$ is bounded above by the mixed volume of these Newton polytopes. This bound is tight for a generic choice of coefficients of the $f_i$'s.*

In fact, the theorem also holds for positive-dimensional zero sets or, to be more precise, positive-dimensional toric varieties [11].

Bernstein's second theorem below describes genericity. Given $v \in (\mathbb{R}^*)^n$ and polynomial $f_i = \sum\limits_{a \in A_i} c_{i,a} x^a$, we denote by $\partial_v f_i$ the polynomial obtained by keeping only those terms minimizing the inner product between their exponent vector and $v$. The Newton polytope of $\partial_v f_i$ is the face of the Newton polytope of $f_i$ supported by $v$.

**Theorem 2.2 ([3]).** *If for all $v \in (\mathbb{R}^*)^n$, the face system $\partial_v f_1 = \ldots = \partial_v f_n = 0$ has no solutions in $(\mathbb{C}^*)^n$, then the mixed volume of the $f_i$ exactly equals the number of solutions in $(\mathbb{C}^*)^n$, and all solutions are isolated. Otherwise, the mixed volume is a strict upper bound on the number of isolated solutions.*

This theorem was used to study planar embeddings [41]; we shall also apply it to $\mathbb{R}^3$.

We have developed specialized software that constructs all Laman graphs and all 1-skeleta of simplicial polyhedra in $\mathbb{R}^3$ with $n \leq 10$, using the respective Henneberg steps. Our computational platform is SAGE[2], whereas Henneberg steps were implemented, using SAGE's interpreter, in Python. We classify all the graphs up to isomorhism using SAGE's interface for N.I.C.E., an open-source isomorphism check engine, keeping for each graph the Henneberg sequence with largest number of $H_1$ steps. For each graph we construct a polynomial system whose real solutions express all possible embeddings, using formulation (2.8). By genericity, solutions have no zero coordinates. For each system we bound the number of its complex solutions by computing its mixed volume. This can be computed by any relevant software. We used the implementation of the Lift-Prune algorithm in C [17], which can be called from within Maple [3].

For every Laman graph, to discard translations and rotations, we pick an edge and fix the coordinates of its vertices, as shown in system (2.1). In $\mathbb{R}^3$, we choose a triangle and fix the coordinates of its vertices, as shown in Eq. (2.8). More generally,

[2] http://www.sagemath.org/.

[3] http://www.di.uoa.gr/~emiris/index-eng.html.

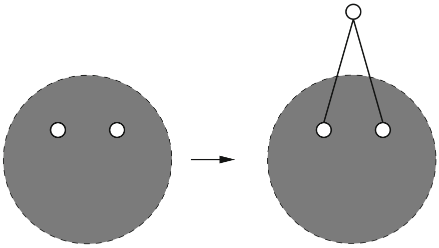


**Fig. 2.3** A Henneberg-1 step (edge addition)

in $\mathbb{R}^d$, a $(d-1)$-dimensional simplex needs to be fixed. Depending on the choice of the fixed vertices, we obtain different systems hence different mixed volumes. Since they all bound the actual number of embeddings, we use the minimum mixed volume.

We used an Intel Core2, at 2.4GHz, with 2GB of RAM. We tested more that 20,000 graphs and computed the mixed volume of more than 40,000 polynomial systems. The total time of experiments was about two days. Tables 2.1 and 2.2 summarize our results.

## 2.3 Laman Graphs

This section studies the number of embeddings of Laman graphs in $\mathbb{R}^2$. We discuss Henneberg steps, then study the number of embeddings for small $n$, and focus on $n = 6$.

Let us introduce a family of simple systems, which has been used in the past, e.g., [41], to express embeddability in $\mathbb{R}^2$. Here $x_i, y_i$ denote the coordinates of the $i$-th vertex, and the $d_{ij}$ are the given lengths.

$$\begin{cases} x_i = a_i,\ y_i = b_i, & i = 1,2, \quad a_i, b_i \in \mathbb{R}, \\ (x_i - x_j)^2 + (y_i - y_j)^2 = d_{ij}^2, & (i,j) \in E - \{(1,2)\}. \end{cases} \tag{2.1}$$

The endpoints of edge $(1,2)$ are fixed, namely we fix $(x_1, y_1)$, e.g., to $(0,0)$, so as to remove translations, and $(x_2, y_2)$, e.g., to $(1,0)$ to remove rotations and scaling, assuming without loss of generality that edge $(1,2)$ exists.

Let us consider the Henneberg steps defining Laman graphs, each adding a new vertex and a total number of two edges. A Henneberg-1 (or $H_1$) step connects the new vertex to two existing vertices (Fig. 2.3). A Hennenerg-2 (or $H_2$) step connects the new vertex to three existing vertices having at least one edge among them, and this edge is removed (Fig. 2.4). We represent each Laman graph with $n \geq 3$ by $\triangle s_4 \ldots, s_n$, where $s_i \in \{1,2\}$; this is known as its *Henneberg sequence*. The way to interpret this representation is that $G$ can be constructed inductively by starting with the simplex $\triangle$, where at step $i \geq 4$, a new vertex is inserted by performing

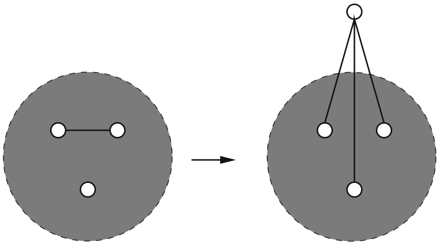


**Fig. 2.4** A Henneberg-2 step (edge split)

a Henneberg step of type $s_i \in \{1,2\}$. Given a graph represented by sequence $S$, performing a $H_1$ step yields graph $S1$, whereas performing a $H_2$ step yields graph $S2$. The Desargues graph (Fig. 2.2) is represented by $\triangle 112$.

Note that this sequence is by no means unique for a given graph; moreover, the same sequence may yield different graphs. A Laman graph is called $H_1$ if it can be constructed using only $H_1$ steps; it is called $H_2$ otherwise.

Since two circles intersect generically in two points, a $H_1$ step at most doubles the number of embeddings, and this is tight, generically. It follows that a $H_1$ graph with $n$ vertices has $2^{n-2}$ embeddings. One can easily verify that every $\triangle 2$ graph is isomorphic to a $\triangle 1$ graph and that every $\triangle 12$ graph is isomorphic to a $\triangle 11$ graph. Consequently, all Laman graphs with $n = 4,5$ are $H_1$ and have 4 and 8 embeddings, respectively.

For a Laman graph with six vertices, a tight uppper bound of 24 follows by examining the three possibilities: the graph is either $H_1$, is $K_{3,3}$, or is the Desargues graph. Now, $H_1$ graphs with six vertices have 16 embeddings. The $K_{3,3}$ graph has precisely 16 embeddings [44], a fact first conjectured in [48]. The Desargues graph has precisely 24 embeddings: the upper bound was first shown in [31] and proven more explicitly, along with the lower bound, in [26]. We return to the case $n = 6$ below, and show how the Desargues bound is obtained as a mixed volume.

Table 2.1 summarizes our results for $n \leq 10$, including the result of Theorem 2.5 for $n = 7$. Recently, the same approach has given a better upper bound of 122 for $n = 8$, which is conjectured to be tight [13]. Using our software (Sect. 2.2), we construct all Laman graphs with $n = 9,10$, and compute their respective mixed volumes, thus obtaining the shown upper bounds. The lower bound for $n = 9$ follows from the Desargues fan [6], while the others follow from the fact that a $H_1$ step exactly doubles the number of embeddings.

We now establish a general upper bound, which improves upon the existing ones when our graph contains many degree-two vertices.

**Lemma 2.1.** *Let $G$ be a Laman graph with $n > 6$ vertices among which there are $k$ degree-2 vertices. Then, the number of planar embeddings of $G$ is bounded above by* $3(2^{k+2}4^{n-k-6})$.

*Proof.* Our proof parallels that of [41] which uses mixed volumes to bound the effect of a $H_1$ step, when it is the last one in the Henneberg sequence. We start by removing all of the $k$ degree-two vertices. Notice that the removal of a degree-two vertex does not destroy other degree-two vertices (because the remaining graph should be also Laman), although it may create new ones. The remaining graph has $n-k$ vertices, and according to Table 2.1, the first six of them can only contribute 24 to the total number of embeddings. □

### 2.3.1 Algebraic Formulations for $n = 6$

Let us now focus on $n = 6$ and study good algebraic representations of the given problem so that mixed volume offers interesting bounds. For the Desargues graph, an optimal mixed volume of 24 is obtained by using the general approach of distance geometry (see next section). Interestingly, we were not able to construct a polynomial system for $K_{3,3}$ whose mixed volume were optimal. Our best result is a system defined by distance matrices, yielding a bound of 25.

Let us concentrate on planar quaternions, employed in [10] to derive a degree-six resultant in $q_4$. Quaternions are widely used to provide rational parameterizations of rigid transformations. We define a planar quaternion:

$$q := \left[\frac{1}{2}\left(d_x \cos\frac{\theta}{2} + d_y \sin\frac{\theta}{2}\right), \frac{1}{2}\left(d_y \cos\frac{\theta}{2} - d_x \sin\frac{\theta}{2}\right), \sin\frac{\theta}{2}, \cos\frac{\theta}{2}\right] \in \mathbb{R}^4,$$

to express rotation by $\theta$ and translation by $(d_x, d_y)$ in the plane, where $q_3^2 + q_4^2 = 1$.

The corresponding transformation matrix is

$$T := \begin{bmatrix} q_4^2 - {q_3}^2 & -2q_3q_4 & 2q_1q_4 - 2q_2q_3 \\ 2q_3q_4 & {q_4}^2 - {q_3}^2 & 2q_1q_3 + 2q_2q_4 \\ 0 & 0 & 1 \end{bmatrix}$$

and is used to define the reference frame of one triangle with respect to that of the other triangle, by writing equations for the three edges linking the triangles, cf. Fig. 2.2. The rest of edge lengths concern points within a reference frame and can be decoupled; their effect is to multiply the number of solutions by four.

The three equations above refer to edges $(i, i+3)$ for $i = 1,2,3$; together with $q_3^2 + q_4^2 = 1$, they define a well-constrained system. Let us take a closer look: they are all of a similar form, the simplest one being $|\langle Tv_1, v_4\rangle| = d_{14}^2$, where $\langle\cdot,\cdot\rangle$ denotes inner product and $v_1, v_4$ are the origins of the two reference frames. The observation of [10] is that these three equations are homogeneous in the $q_i$'s except from the $d_{ij}$ terms. It suffices then to multiply the latter by $q_3^2 + q_4^2$ to obtain three homogeneous equations in new variables $z_i = q_i/q_4, i = 1,2,3$. Now, the problem is reduced to a system of three nonhomogeneous equations in $z_1, z_2, z_3$; its mixed volume is six. The fourth equation becomes $z_3^2 + 1 = z_0$, which uniquely specifies $z_0 = 1/q_4$ for each of the system's solutions. Hence, the overall number of embeddings is 24, and this is optimal.

## 2.4 Distance Geometry

In this section we introduce distance geometry and Cayley–Menger matrices, then combine results from distance geometry with mixed volume, which settled the case $n = 7$ in [18]. Distance matrices were introduced by A. Cayley in 1841 with the aim to derive necessary conditions on the pairwise distances of five points in Euclidean space [9]. The theory behind distance geometry has been well developed, e.g., [4,12, 16,40]. Applications of distance geometry to structural bioinformatics, e.g., [30,42], have been quite succesful in practice, e.g., [28, 37]. We refer to [14] and further references therein for a detailed study of Euclidean distance matrices.

### 2.4.1 Preliminaries

We begin by introducing some basic notation and necessary definitions. Let $\mathbb{S}_n$ denote the set of symmetric $n \times n$ matrices and $\mathbb{S}^n_+$ the set of $n \times n$ positive semidefinite matrices. Throughout, $e$ will denote the all ones vector of the appropriate size. Consider a matrix $D \in \mathbb{S}_n$ such that $D_{ij} = d_{ij}^2$ for every $i \neq j$ and $D_{ii} = 0$ for $i \in [n]$. Such a matrix is usually called a *pre-distance matrix* (in other fields it is known as a dissimilarity matrix). A *Euclidean distance matrix* (EDM) is a pre-distance matrix $D$ for which there exist $p_1, \dots, p_n \in \mathbb{R}^k$ such that

$$D_{ij} = ||p_i - p_j||^2, \; 1 \leq i, j \leq n, \tag{2.2}$$

where $|| \cdot ||$ denotes Euclidean distance. Let $\mathrm{EDM}_n$ denote the cone of $n \times n$ Euclidean distance matrices. Vectors $p_1, \dots, p_n$ *realize* the pre-distance matrix $D$ if they satisfy relation (2.2). Given $D \in \mathrm{EDM}_n$, let $\mathrm{ed}(D)$ denote the smallest dimension $k$ in which there exist vectors realizing $D$. This is called the *embedding dimension* of $D$.

The previous definitions lead to some natural questions. Can we identify necessary and sufficient conditions that will ensure that a given pre-distance matrix is a EDM? Moreover, given $D \in \mathrm{EDM}_n$, can we compute its embedding dimension? Theorem 2.3, provides positive answers to both questions.

**Theorem 2.3 ([27, 40]).** *Let $D \in \mathbb{S}_n$ be a pre-distance matrix and vector $s \in \mathbb{R}^n$ such that $s^T e = 1$. Then, $D \in \mathrm{EDM}_n$ if and only if*

$$F_s(D) := -\frac{1}{2}\left(I - es^T\right) D \left(I - se^T\right) \textit{ is positive semidefinite.}$$

*Moreover, if $D \in EDM_n$, then* $\mathrm{ed}(D) = \mathrm{rank} F_s(D)$.

Theorem 2.3 implies that testing embeddabililty of a pre-distance matrix $D$ can be done in polynomial time, since it amounts to checking whether some matrix is positive semidefinite. Additionally, the embedding dimension can also be efficiently computed, since it amounts to a rank computation. We now obtain a useful reformulation of Theorem 2.3 by *Cayley–Menger determinants*. The Cayley–Menger matrix associated with a given pre-distance matrix: $D \in \mathbb{S}_n$ is the $(n+1) \times (n+1)$ matrix

$$\mathrm{CM}(D) = \begin{pmatrix} 0 & e \\ e^T & D \end{pmatrix}.$$

It is an easy exercise to check that the Schur complement of $\mathrm{CM}(D)$ with respect to the $2 \times 2$ submatrix $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ indexed by the first row/column and the $i$-th row/column is equal to $-2F_{e_i}(D)$. Consequently,

$$\begin{aligned} \det \mathrm{CM}(D) &= (-1)^n 2^{n-1} \det F_{e_i}(D), \\ \mathrm{rank}\mathrm{CM}(D) &= \mathrm{rank} F_{e_i}(D) + 2. \end{aligned} \tag{2.3}$$

Now, we express the condition of $F_{e_i}(D)$ being positive semidefinite in terms of Cayley–Menger determinants. It is well-known that $F_{e_i}(D) \in \mathbb{S}_n^+$ if and only if every principal minor of $F_{e_i}(D)$ has nonnegative determinant. But the principal minors of $F_{e_i}(D)$ have the form $F_{e_i}(D[X])$, for some $X \subseteq [n]$, where $D[X]$ denotes the principal minor of $D$ indexed by $X$. This observation, combined with Eq. (2.3), forms the crux of the following reformulation of Theorem 2.3.

**Theorem 2.4 ([39]).** *Let $D \in \mathbb{S}_n$ be a pre-distance matrix. Then $D \in \mathrm{EDM}_n$ if and only if*

$$\forall\, X \subseteq [n],\ \ (-1)^{|X|} \mathrm{det} CM(D[X]) \geq 0. \tag{2.4}$$

Condition (2.4) yields inequalities. For $|X| = 2$, it expresses the fact that all entries of $D$ must be nonnegative. For $|X| = 3$, it captures the triangular inequality. Indeed, if we apply it to $X = \{1,2,3\}$, then condition (2.4) becomes:

$$(d_{12}+d_{13}+d_{23})(d_{12}+d_{13}-d_{23})(d_{12}+d_{23}-d_{13})(d_{13}+d_{23}-d_{12}) \leq 0,$$

with equality satisfied precisely when the corresponding points are collinear and strict inequality satisfied when the points define a triangle. Equivalently, for $X = \{i,j,k\}$, condition (2.4) can be written as $d_{ik} + d_{jk} \geq d_{ij}$ for all triplets $i,j,k \in \{1,\ldots,n\}$. For $k = 4$ the condition captures the tetrangular inequality.

We are now ready to summarize our approach. Consider a Laman graph $G = ([n],E)$, together with an assignment of generic weights $d_{ij} \in \mathbb{R}_+$ to its edges. Let $D$ be the corresponding partial (symmetric) matrix, i.e., $D_{ij} = d_{ij}^2, \forall (i,j) \in E$ and $\mathrm{diag}(D) = 0$. A matrix $D' \in \mathbb{S}_n$ is called an *EDM completion* of $D$, if $D' \in \mathrm{EDM}_n$ and $D'_{ij} = D_{ij}, \forall (i,j) \in E$. Clearly, the number of embeddings of the distances $d = (d_{ij})$ in the plane, modulo rigid transformations, is equal to the number of EDM completions of $D$ that satisfy $\mathrm{rank}\mathrm{CM}(D) \leq 4$.

The condition $\mathrm{rank}\,\mathrm{CM}(D) \leq 4$ imposes the vanishing of all $5 \times 5$ minors of $\mathrm{CM}(D)$. Thus, we obtain a polynomial system of $\binom{n+1}{5}$ equations in the $\binom{n}{2} - 2n +$

3 unknowns $x_{ij}$, corresponding to the lengths of the edges that are not present in $G$. Clearly, the number of its real solutions is an upper bound on the number of embeddings of $G$ in $\mathbb{R}^2$. We employ mixed volumes in order to bound the number of complex solutions.

For this, we identify all square subsystems which correspond to Laman subgraphs of $G$ and compute their mixed volume. Focusing on Laman subgraphs implies that we make use of Cayley–Menger minors only; in other words our system is defined by submatrices whose first row and column are filled with ones. This is indispensable for the system to have a finite number of solutions. We thus construct several well-constrained polynomial systems, each yielding a mixed volume; the smallest of which provides the desired upper bound.

### 2.4.2 *The Case $n = 6$*

Consider the Desargues graph and its associated Cayley–Menger matrix seen below. Here $c_{ij} = d_{ij}^2$ correspond to the fixed distances, and $x_{15}, x_{16}, x_{24}, x_{26}, x_{34}, x_{35}$ to the unspecified ones. The numbering of the vertices corresponds to that in Fig. 2.2. Moreover, we assume that the matrix is indexed by $0, 1, \ldots, 6$, and we use the shorthand notation $\mathrm{CM}(X)$ for the principal minor defined by the indices in $X$:

$$\begin{array}{c|ccccccc|}
 & & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\
\hline
 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
v_1 & 1 & 0 & c_{12} & c_{13} & c_{14} & x_{15} & x_{16} \\
v_2 & 1 & c_{12} & 0 & c_{23} & x_{24} & c_{25} & x_{26} \\
v_3 & 1 & c_{13} & c_{23} & 0 & x_{34} & x_{35} & c_{36} \\
v_4 & 1 & c_{14} & x_{24} & x_{34} & 0 & c_{45} & c_{46} \\
v_5 & 1 & x_{15} & c_{25} & x_{35} & c_{45} & 0 & c_{56} \\
v_6 & 1 & x_{16} & x_{26} & c_{36} & c_{46} & c_{56} & 0
\end{array} .$$

By Theorem 2.4 or conditions (2.3), all $5 \times 5$ principal minors of this matrix that contain the first row/column of ones vanish, hence yielding polynomial equations on the variables $x_{15}, x_{16}, x_{24}, x_{26}, x_{34}, x_{35}$. There are $\binom{6}{4} = 15$ such minors, nine of which yield bivariate and the rest trivariate equations. We restrict attention to bivariate equations, which turn out to be sufficient in this case. There are six quadratics and three cubics. No $3 \times 3$ subsystem exists, but it is possible to find several $4 \times 4$ subsystems, including some with only one cubic. Take, for example, the minors

$$\begin{aligned}
\mathrm{CM}(0,2,4,5,6)(x_{24}, x_{26}) &= \mathrm{CM}(0,1,4,5,6)(x_{15}, x_{16}) \\
&= \mathrm{CM}(0,1,2,4,5)(x_{15}, x_{24}) = \mathrm{CM}(0,1,2,3,6)(x_{16}, x_{26}) = 0,
\end{aligned}$$

where we have indicated the variables per equation. They define a system of three quadratic and one cubic equation in $x_{15}, x_{16}, x_{24}, x_{26}$, and its mixed volume is 24. The other two unknowns are uniquely defined from each solution because one can easily construct linear equations expressing each of the $x_{34}, x_{35}$, in terms of the $x_{15}, x_{16}, x_{24}, x_{26}$. For instance $x_{16}\mathrm{CM}(0,1,2,3,4) - \mathrm{CM}(0,1,3,4,6)$ is linear in $x_{34}$ and quadratic in $x_{16}, x_{24}$.

Let us examine $K_{33}$, with the same approach. We obtain the following Cayley–Menger matrix, where $c_{ij} = d_{ij}^2$ correspond to the fixed distances for $i = 1,3,5$ and $j = 2,4,6$, and $x_{13}, x_{15}, x_{24}, x_{26}, x_{35}, x_{46}$ are the unspecified distances.

$$\begin{array}{c|ccccccc|}
 & & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\
 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
v_1 & 1 & 0 & c_{12} & x_{13} & c_{14} & x_{15} & c_{16} \\
v_2 & 1 & c_{12} & 0 & c_{23} & x_{24} & c_{25} & x_{26} \\
v_3 & 1 & x_{13} & c_{23} & 0 & c_{34} & x_{35} & c_{36} \\
v_4 & 1 & c_{14} & x_{24} & c_{34} & 0 & c_{45} & x_{46} \\
v_5 & 1 & x_{15} & c_{25} & x_{35} & c_{45} & 0 & c_{56} \\
v_6 & 1 & c_{16} & x_{26} & c_{36} & x_{46} & c_{56} & 0
\end{array}\,.$$

We consider the $5 \times 5$ minors that yield cubic bivariate polynomials in $x_{13}$, $x_{15}$, $x_{24}$, $x_{26}$, $x_{35}$, $x_{46}$. No $3 \times 3$ subsystem exists, but it is possible to find several $4 \times 4$ subsystems, such as the following:

$$\begin{aligned}\mathrm{CM}(0,3,4,5,6)(x_{35}, x_{46}) &= \mathrm{CM}(0,2,3,5,6)(x_{26}, x_{35}) \\ &= \mathrm{CM}(0,1,4,5,6)(x_{15}, x_{46}) = \mathrm{CM}(0,1,2,5,6)(x_{15}, x_{26}) = 0.\end{aligned}$$

This system has a mixed volume of 25. Furthermore, the above equations are sparse enough to reveal precise information about their common solutions via univariate (i.e., Sylvester) resultants. Specifically, we can consider the first two and the last two polynomials, respectively, as univariate in $x_{35}$ and $x_{15}$, and obtain resultants $R_1$ and $R_2$ in $x_{26}, x_{46}$. Seen as univariate polynomials in $x_{26}$, they yield a resultant in $x_{46}$ that factorizes into the product of $x_{46}^2$, a factor of degree 16, and another of degree 8; analyzing these factors may yield an upper bound of 16.

### 2.4.3 The Case $n = 7$

Let us consider Laman graphs with seven vertices. We will prove the main result of this section, namely a tight count of embeddings, which was established in [18].

If a $H_1$ step is applied to any graph with $n = 6$, the resulting graph with $n = 7$ admits exactly 48 embeddings. To maximize the number of embeddings, we shall

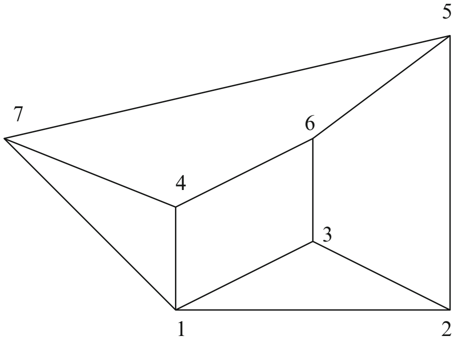


**Fig. 2.5** The graph that gives the tight bound for $n = 7$

apply a $H_2$ step to any graph with $n = 6$. By checking graph isomorphisms and taking into account the various symmetries, it was shown [20] that there are only three relevant graphs to be considered. These are obtained by a $H_2$ step applied to the Desargues graph as follows, where numbering refers to Fig. 2.2: We remove edge $(4,5)$ and add edges $(4,7), (5,7)$ and one of the following three edges: $(1,7)$, $(3,7)$, or $(6,7)$. The first case corresponds to the topology that shall be studied extensively in the sequel, since it leads to the maximum number of 56 embeddings. We will call this graph $G_7$, and it is shown in Fig. 2.5. The other two graphs admit at most 44 and 48 embeddings, respectively; hence they are not studied any further. These upper bounds are obtained as mixed volumes of polynomial systems, as explained above.

**Theorem 2.5 ([18]).** *The maximum number of planar Euclidean embeddings for a Laman graph with seven vertices is 56.*

*Proof.* It is known that for the graph of Fig. 2.5, there exist edge lengths for which it has 56 embeddings in $\mathbb{R}^2$ [18]. This settles the lower bound; the rest of the proof focuses on the upper bound.

The Cayley–Menger matrix for graph $G_7$ is below, where the $c_{ij} = d_{ij}^2$ correspond to the fixed distances, and $x_{ij}$ are the unspecified distances.

$$\begin{array}{c|cccccccc|}
 & & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \\
 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
v_1 & 1 & 0 & c_{12} & c_{13} & c_{14} & x_{15} & x_{16} & c_{17} \\
v_2 & 1 & c_{12} & 0 & c_{23} & x_{24} & c_{25} & x_{26} & x_{27} \\
v_3 & 1 & c_{13} & c_{23} & 0 & x_{34} & x_{35} & c_{36} & x_{37} \\
v_4 & 1 & c_{14} & x_{24} & x_{34} & 0 & x_{45} & c_{46} & c_{47} \\
v_5 & 1 & x_{15} & c_{25} & x_{35} & x_{45} & 0 & c_{56} & c_{57} \\
v_6 & 1 & x_{16} & x_{26} & c_{36} & c_{46} & c_{56} & 0 & x_{67} \\
v_7 & 1 & c_{17} & x_{27} & x_{37} & c_{47} & c_{57} & x_{67} & 0
\end{array} .$$

By Theorem 2.4, any $5 \times 5$ minor of this matrix must vanish, which yields polynomial equations on the variables $x_{ij}$. There are $\binom{8}{5}$ such minors, but only $\binom{7}{4} = 35$ Cayley–Menger minors, each in two to four variables. Among these polynomials, no $4 \times 4$ subsystem exists that corresponds to a rigid mechanism, as was verified by checking Laman's condition in our Maple implementation.

However, it is possible to find certain $5 \times 5$ subsystems whose subgraph is Laman and, moreover, uniquely define the configuration of the overall graph. One of these systems has four bivariate equations and one trivariate equation and is defined by taking the following minors:

$$\begin{cases} \mathrm{CM}(0,4,5,6,7)(c_{46},c_{47},c_{56},c_{57},x_{45},x_{67}) = 0, \\ \mathrm{CM}(0,1,4,6,7)(c_{14},c_{17},c_{46},c_{47},x_{16},x_{67}) = 0, \\ \mathrm{CM}(0,1,4,5,7)(c_{14},c_{17},c_{47},c_{57},x_{15},x_{45}) = 0, \\ \mathrm{CM}(0,1,2,3,5)(c_{12},c_{13},c_{25},c_{23},x_{15},x_{35}) = 0, \\ \mathrm{CM}(0,1,3,5,6)(c_{13},c_{36},c_{56},x_{15},x_{16},x_{35}) = 0. \end{cases} \tag{2.5}$$

These define a system of three quadratic and two cubic equations in $x_{15},x_{16},x_{35}$, $sx_{45},x_{67}$; the cubics are the first and last polynomials. The corresponding subgraph is Laman, and, moreover, once the unknown lengths are fixed, they uniquely define the configuration of the overall graph. The reason is that, once we solve for the unknowns in the system, the resulting graph has more constraints than given by Laman's condition.

The mixed volume turns out to be equal to 56. This bounds the number of complex common solutions of the system, hence the number of real embeddings. Notice that this bound does not take into account solutions with zero coordinates, in other words some zero length. However, a graph has embeddings with some zero length only when the input bar lengths form a singular set, in the sense that they would satisfy a non-generic algebraic dependency. For example, by letting some input distance be exactly 0, some graph may theoretically have infinitely many configurations. However, generically, it is impossible to have such an embedding. Thus for $n = 7$ we have a tight bound of 56. □

## 2.5 1-skeleta of Simplicial Polyhedra

This section extends the previous results to one-skeleta, or edge-skeleta, of (convex) simplicial polyhedra, which are known to be rigid in $\mathbb{R}^3$.

Fewer results are known on the number of embeddings of such graphs, despite their relevance in applications. One related work in robotics studies all classes of parallel robots [23] but focuses on the generic number of complex configurations.

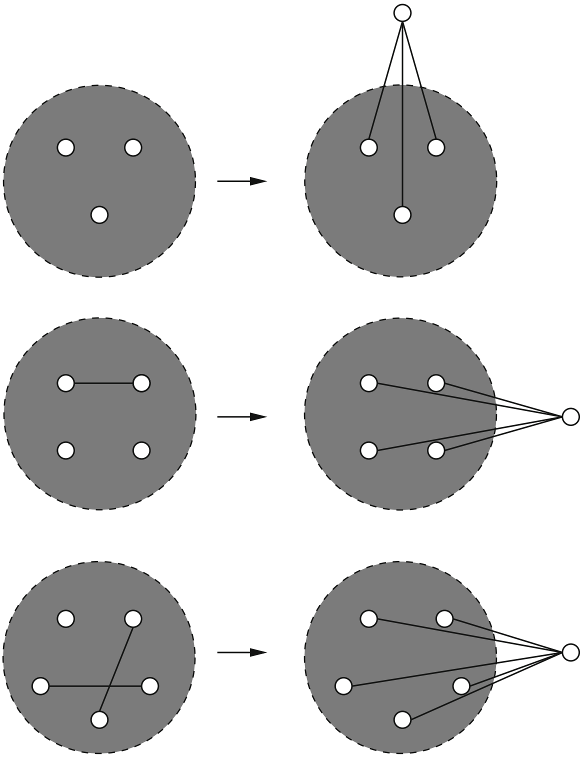


**Fig. 2.6** The spatial Henneberg steps

Concerning real configurations, the most celebrated result states that the general Stewart (or Gough) platform has 40 real positions [15].

To express embeddability in $\mathbb{R}^3$, we extend system (2.1), where $x_i, y_i, z_i$ denote the coordinates of the $i$-th vertex, and the $d_{ij}$ are the given lengths:

$$\begin{cases} x_i = a_i,\ y_i = b_i,\ z_i = c_i, & i = 1,2,3, \quad a_i, b_i, c_i \in \mathbb{R}, \\ (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 = d_{ij}^2, & (i,j) \in E - \{(1,2),(1,3),(2,3)\}. \end{cases} \tag{2.6}$$

We fix $(x_1, y_1, z_1) = (0,0,0)$ to remove translations, $(x_2, y_2, z_2) = (1,0,0)$ to remove two rotational degrees of freedom and scaling, and set $x_3 = 0, z_3 > 0$ to remove the third rotational degree of freedom. If we choose three points forming a triangle, then we can compute the unique position of $(x_3, y_3, z_3)$ as in system (2.6).

Consider any $k+2$ vertices forming a cycle with $\geq k-1$ diagonals, $k \geq 1$. The (extended) Henneberg-$k$ step (or $H_k$), $k = 1,2,3$, corresponds to adding a vertex, connecting it to the $k+2$ vertices and removing $k-1$ diagonals among them, as illustrated in Fig. 2.6. Since three spheres intersect generically in two points, a $H_1$ step at most doubles the number of spatial embeddings, and this is tight, generically.

**Proposition 2.1 ([7]).** *A graph is the 1-skeleton of a simplicial polyhedron in* $\mathbb{R}^3$ *if and only if it has a construction that begins with a tetrahedron, followed by any sequence of* $H_1, H_2, H_3$ *steps.*

System (2.6) has $3n$ unknowns. Our first observation is that this system does not capture the structure of the problem. Specifically, by choosing direction

$$v = (\underbrace{0,0,0}_{v_1},\underbrace{0,0,0}_{v_2},\underbrace{0,0,0}_{v_3},\underbrace{-1,-1-1}_{v_4},\dots,\underbrace{-1,-1,-1}_{v_n}) \in \mathbb{R}^{3n},$$

the corresponding face system becomes

$$\begin{cases} x_i = a_i, y_i = b_i, z_i = c_i, & i = 1,2,3, \quad a_i, b_i, c_i \in \mathbb{R}, \\ x_i^2 + y_i^2 + z_i^2 = 0, & (i,j) \in E:\ i \notin \{1,2,3\},\ j \in \{1,2,3\}, \\ (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 = 0, & (i,j) \in E,\ i,j \notin \{1,2,3\}. \end{cases} \tag{2.7}$$

This system has $(a_1, b_1, c_1, \dots, a_3, b_3, c_3, 1, 1, \gamma\sqrt{2}, \dots, 1, 1, \gamma\sqrt{2}) \in (\mathbb{C}^*)^{3n}$ as a solution, where $\gamma = \pm\sqrt{-1}$. Consequently, according to Theorem 2.2, the mixed volume of system (2.6) is not a tight bound on the number of solutions in $(\mathbb{C}^*)^{3n}$. To remove spurious solutions (at toric infinity), we introduce variables $w_i = x_i^2 + y_i^2 + z_i^2$, for $i = 1, \dots, n$. This yields the following equivalent system, but with lower mixed volume:

$$\begin{cases} x_i = a_i, y_i = b_i, z_i = c_i, & i = 1,2,3, \\ w_i = x_i^2 + y_i^2 + z_i^2, & i = 4, \dots, n, \\ w_i + w_j - 2x_ix_j - 2y_iy_j - 2z_iz_j = d_{ij}^2, & (i,j) \in E - \{(1,2),(1,3),(2,3)\}. \end{cases} \tag{2.8}$$

This is the formulation we have used in our computations. The reduced mixed volume is indicated by a face system similar to the one previously defined. To be more specific, we consider $v \in \mathbb{R}^{4n-3}$ such that $v_i = 0$ for $i \leq 9$ and $i > 3n$, where the latter coordinates correspond to $w_4, \dots, w_n$ and $v_i = -1$ for $i = 10, \dots, 3n$. The corresponding face system becomes

$$\begin{cases} x_i = a_i, y_i = b_i, z_i = c_i, & i = 1,2,3, \quad a_i, b_i, c_i \in \mathbb{R}, \\ x_i^2 + y_i^2 + z_i^2 = 0, & i \in \{4, \dots, n\}, \\ x_ix_j + y_iy_j + z_iz_j = 0, & (i,j) \in E,\ i,j \notin \{1,2,3\}, \\ x_ix_j + y_iy_j + z_iz_j = 0, & (i,j) \in E:\ i \notin \{1,2,3\},\ j \in \{1,2,3\}, \end{cases}$$

where the first two classes of equations are similar to the corresponding ones in system (2.7). In particular, this face system does not admit roots like those above, with $(x_i, y_i, z_i) = (1, 1, \gamma\sqrt{2})$ for $i = 4, \dots, n$, because of the fourth set of equations. Of course, there are probably other face systems with nontrivial roots.

We continue by establishing tight bounds for some small cases. For $n = 4$, the only simplicial polytope is the three-simplex, which clearly admits only two embeddings. For $n = 5$, there is a unique graph that corresponds to a one-skeleton of a simplicial polyhedron [7], cf. Fig. 2.7. This graph is obtained from the three-simplex through a $H_1$ step, so for $n = 5$, there is a tight bound of 4.

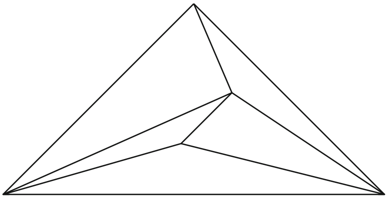


**Fig. 2.7** The only one-skeleton of a simplicial polytope on five vertices

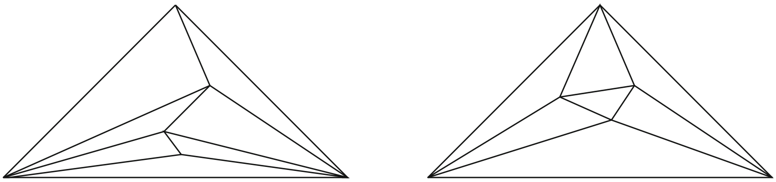


**Fig. 2.8** All one-skeleta of convex simplicial polyhedra on six vertices

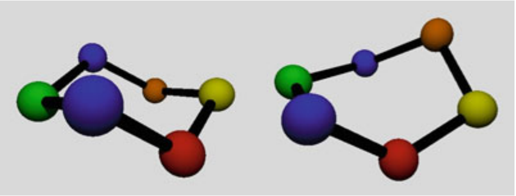


**Fig. 2.9** A chair and a boat configuration of the cyclohexane molecule

**Lemma 2.2.** *The one-skeleton of a simplicial polyhedron on six vertices has at most 16 embeddings, and this bound is tight.*

*Proof.* There are two non-isomorphic graphs $G_1, G_2$ for $n = 6$ [7], cf. Fig. 2.8. For $G_1$, the mixed volume is eight. Since all facets of $G_2$ are symmetric, we fix one and compute the mixed volume, which equals 16, so the overall upper bound is 16. We shall obtain a matching lower bound.

Notice that $G_2$ is the graph of the cyclohexane, which admits 16 distinct Euclidean embeddings for generic edge lengths [19]. To see the equivalence, recall that the cyclohexane is essentially a six-cycle (cf. Fig. 2.9), with known lengths between vertices at distance 1 (adjacent) and 2.

The former are bond lengths whereas the latter are specified by the angle between consecutive bonds. In [19], the upper bound was obtained as the mixed volume of two different systems; one was composed of equations obtained by distance matrices.

Alternatively, $G_2$ corresponds to a Stewart platform parallel robot with 16 configurations, where two triangles define the platform and base and six lengths link the triangles in a jigsaw shape. A folklore argument gives the configurations

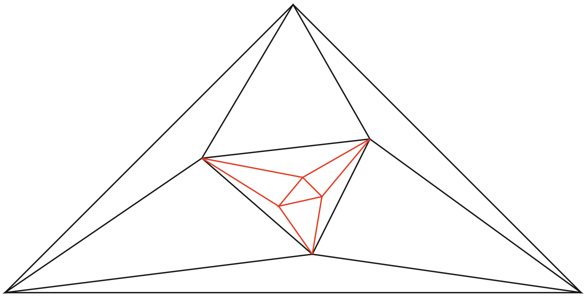


**Fig. 2.10** A cyclohexane caterpillar with two copies

by placing two points per axis [4]. On each axis, one point belongs to the platform, the other to the base. There are two choices for labeling points on each axis, giving eight configurations; by reflection about the origin, we obtain another eight. □

We now pass to general $n$ and establish the first lower bound in $\mathbb{R}^3$.

**Theorem 2.6.** *There exist edge lengths for which the cyclohexane caterpillar construction has* $16^{\lfloor (n-3)/3 \rfloor} \simeq 2.52^n$ *embeddings, for* $n \geq 9$.

*Proof.* We glue together copies of cyclohexanes sharing a common triangle. The resulting graph is the one-skeleton of a simplicial polytope. Each new copy adds three vertices, and since there exist edge lengths for which the cyclohexane graph has 16 embeddings, the claim follows. □

A caterpillar made of two cyclohexane copies is illustrated in Fig. 2.10.

Table 2.2 summarizes our computational results for $n \leq 10$, where $\triangle$ is the three-simplex and bold text indicates the Henneberg sequence of the graph that yields the upper bound. The upper bounds for $n = 7, \ldots, 10$ are computed by our software employing mixed volumes. The lower bound for $n = 9$ follows from Theorem 2.6. All other lower bounds are obtained by applying a $H_1$ step to a graph with one fewer vertex.

Our computation also shows that, in constructing all non-isomorphic one-skeleta of simplicial polyhedra, $H_3$ need not be applied before $n = 13$. Lastly, we state a result similar to Lemma 2.1.

**Lemma 2.3.** *Let* $G$ *be the 1-skeleton of a simplicial polyhedron on* $n > 7$ *vertices among which there are* $k$ *degree-three vertices. Then the number of embeddings of* $G$ *is bounded above by* $2^{k+5}8^{n-k-7}$.

*Proof.* We start by removing all of the $k$ degree-three vertices. Notice that the removal of a degree-three vertex does not destroy other degree-three vertices (because the remaining graph should be also the edge graph of a simplicial

[4] Personal communication with Daniel Lazard.

polyhedron), although it may create new ones. The remaining graph has $n-k$ vertices, and according to Table 2.2, the first seven of them can only contribute 32 to the total number of embeddings. □

## 2.6 Further Work

Undoubtedly, the most important and oldest problem in rigidity theory is the full combinatorial characterization of rigid graphs in $\mathbb{R}^3$. In the planar case, existing bounds are not tight. This is due to the fact that root counts include rotated copies of certain embeddings. Our approach based on distance matrices may offer an algorithmic process for obtaining good algebraic representations, in particular low mixed volumes, including in the spatial case. For high $n$ the issue is that the number of equations produced is quite large, with algebraic dependancies among them.

Since we deal with Henneberg constructions, it is important to determine the effect of each step on the number of embeddings: a $H_1$ step always doubles their number; we conjecture that $H_2$ multiplies it by $\leq 4$ and spatial $H_3$ by $\leq 8$, but these may not always be tight. Our conjecture has been verified for small $n$.

The structures studied here are called point-and-bar structures; they generalize to body-and-bar, where edges can be connected to different points of a rigid body, and to body-and-hinges structures, where the allowed degrees of freedom model hinges, useful in structural bioinformatics. It is known that a body-and-bar structure in $\mathbb{R}^d$ is rigid if and only if the associated graph is the edge-disjoint union of $\binom{d+1}{2}$ spanning trees [47]. A similar result holds for body-and-hinges structures, where every hinge corresponds to $\binom{d+1}{2}-1$ edges [46].

**Acknowledgements** I.Z. Emiris is partially supported by FP7 contract PITN-GA-2008-214584 SAGA: Shapes, Algebra, and Geometry. Part of this work was done while he was on sabbatical at team Salsa of INRIA Rocquencourt. E. Tsigaridas is partially supported by an individual postdoctoral grant from the Danish Agency for Science, Technology and Innovation, and also acknowledges support from the Danish National Research Foundation and the National Science Foundation of China (under grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation, within which part of this work was performed and from the EXACTA grant of the National Science Foundation of China (NSFC 60911130369) and the French National Research Agency (ANR-09-BLAN-0371-01). E. Tsigaridas performed part of this work while he was with the Aarhus University, Denmark. A. Varvitsiotis started work on this project as a graduate student at the University of Athens.

## References

1. Angeles, J.: Rational Kinematics. Springer, New York (1989)
2. Basu, S., Pollack, R., Roy, M-F.: Algorithms in real algebraic geometry. In: Algorithms and Computation in Mathematics, vol. 10 2nd edn. Springer, New york (2006)

3. Bernstein, D.N.: The number of roots of a system of equations. Funct. Anal. Appl. **9**(2), 183–185 (1975)
4. Blumenthal, L.M.: Theory and Applications of Distance Geometry, vol. 15, 2nd edn. Chelsea Publishing Company, Bronx, NY (1970)
5. Borcea, C.: Point configurations and Cayley-Menger varieties, arXiv:math/0207110 (2002)
6. Borcea, C., Streinu, I.: The number of embeddings of minimally rigid graphs. Discrete Comput. Geom. **31**(2), 287–303 (2004)
7. Bowen, R., Fisk, S.: Generation of triangulations of the sphere. Math. Comput. **21**(98), 250–252 (1967)
8. Canny, J.F., Emiris, I.Z.: A subdivision-based algorithm for the sparse resultant. J. ACM **47**(3), 417–451 (2000)
9. Cayley, A.: On a theorem in the geometry of position. Camb. Math. J.**2**, 267–271 (1841)
10. Collins, C.L.: Forward kinematics of planar parallel manipulators in the Clifford algebra of $P^2$. Mech. Mach. Theor. **37**(8), 799–813 (2002)
11. Cox, D., Little, J., O'Shea, D.: Using Algebraic Geometry. Number 185 in GTM. 2nd edn. Springer, New York (2005)
12. Dattorro, J.: Convex Optimization and Euclidean Distance Geometry. Meboo, USA (2011)
13. Despotakis, S.C., Emiris, I.Z., Psarros, I.: An upper bound on Euclidean embeddings of rigid graphs with 8 vertices, Manuscript (2012)
14. Deza, M.M., Laurent, M.: Geometry of Cuts and Metrics. Springer, Berlin (1997)
15. Dietmeier, P.: The Stewart-Gough platform of general geometry can have 40 real postures, In: Lenarcic, J., Husty, M. (eds.) Advances in Robot Kinematics: Analysis and Control, pp. 7–16. Springer, New York (1998)
16. Dress, A.W.M., Havel, T.F.: Distance geometry and geometric algebra. Found. Phys. **23**(10), 1357–1374 (1991)
17. Emiris, I.Z., Canny, J.F.: Efficient incremental algorithms for the sparse resultant and the mixed volume. J. Symbolic. Comput. **20**(2), 117–149 (1995)
18. Emiris, I.Z., Moroz, G.: The assembly modes of rigid 11-bar linkages. In: Proceedings of IFToMM World Congress in Mechanism and Machine Science, Guanajuato, Mexico (2011)
19. Emiris, I.Z., Mourrain, B.: Computer algebra methods for studying and computing molecular conformations. Algorithmica, Special Issue on Algorithms for Computational Biology **25**, 372–402 (1999)
20. Emiris, I.Z., Tsigaridas, E., Varvitsiotis, A.: Algebraic methods for counting Euclidean embeddings of rigid graphs. Lecture Notes in Computer Science "Graph drawing", **5849**, 195–200 (2009)
21. Emmerich, D.G.: Structures Tendues et Autotendantes, In Monographies de géométrie constructive, d. cole d'Architecture Paris-La-Villette, 1988
22. Eren, T., Goldenberg, D.K., Whiteley, W., Yang, Y.R., Morse, A.S., Anderson, B.D.O., Belhumeur, P.N.: Rigidity, computation and randomization in network localization. In: Proceedings of IEEE INFOCOM'04, Hong Kong, 2673–2684 (2004)
23. Faugère, J.C., Lazard, D.: The combinatorial classes of parallel manipulators combinatorial classes of parallel manipulators. Mech. Mach. Theor. **30**(6), 765–776 (1995)
24. Gluck, H.: Almost all simply connected closed surfaces are rigid. Lect. Notes. Math. **438**, 225–240 (1975)
25. Gomez-Jauregui, V.: Tensegrity Structures and their Application to Architecture, MSc Thesis, School of Architecture, Queen's University, Belfast (2004)
26. Gosselin, C.M., Sefrioui, J., Richard, M.J.: Solutions polynomiales au problème de la cinématique directe des manipulateurs parallèles plans à trois degrés de liberté. Mech. Mach. Theor. **27**(2), 107–119 (1992)
27. Gower, J.C.: Euclidean distance geometry. J. Math. Sci. **1**, 1–14 (1982)
28. Guentert, P., Mumenthaler, C., Wüthrich, K.: Torsion angle dynamics for NMR structure calculation with the new program Dyana. J. Mol. Biol. **273**, 283–298 (1997)
29. Harris, J., Tu, L.W.: On symmetric and skew-symmetric determinantal varieties. Topology **23**(1), 71–84 (1984)

30. Havel, T.F.: Distance geometry: Theory, algorithms, and chemical applications. In:von Ragué, P., Schreiner, P.R., Allinger, N.L., Clark, T., Gasteiger, J., Kollman, P.A., Schaefer III, H.F. (eds.) Encyclopedia of Computational Chemistry, pp. 723–742. Wiley, New York (1998)
31. Hunt, K.N.: Structural kinematics of in parallel actuated robot arms. Transactions of the American Society of Mechanical Engineers, Journal of Mechanisms, Transmissions, Automation in Design, 705–712 (1983)
32. Jacobs, D.J., Rader, A.J., Kuhn, L.A., Thorpe, M.F.: Protein flexibility predictions using graph theory. Protein. Struct. Funct. Genet. **44**(2), 150–165 (2001)
33. Krislock, N., Wolkowicz, H.: Explicit sensor network localization using semidefinite representations and facial reductions. SIAM J. Optim. **20**(5), 2679–2708 (2010)
34. Laman, G.: On graphs and rigidity of plane skeletal structures. J. Eng. Math. **4**, 331–340 (1970)
35. Lavor, C., Mucherino, A., Liberti, L., Maculan, N.: On the computation of protein backbones by using artificial backbones of hydrogens. J. Global Optim.**50**(2), 329–344 (2011)
36. Liberti, L., Lavor, C., Masson, B., Mucherino, A.: Polynomial cases of the discretizable molecular distance geometry problem, arXiv:1103.1264 (2011)
37. Malliavin, T., Dardel, F.: Structure des protéines par RMN, In:Sciences Fondamentales, volume AF, pp. 6608 (1–18). Techniques de l'Ingénieur, Paris (2002)
38. Maxwell, J.C.: On the calculation of the equilibrium and stiffness of frames, Phil. Mag. **27**(182), 294–299 (1864)
39. Menger, , Géométrie Générale, Mem. Sci. Math., no. 124, Académie des Sciences de Paris (1954).
40. Schönberg, I.J.: Remarks to M. Frechet's article "Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert". Ann. Math. **36**, 724–732 (1935)
41. Steffens, R., Theobald, T.: Mixed volume techniques for embeddings of Laman graphs. Comput. Geom.: Theor. Appl. **43**(2), 84–93 (2010)
42. Thorpe, M.F., Duxbury, P.M. (eds.): Rigidity Theory and Applications. Fund. Materials Res. Ser., Kluwer, New York (1999)
43. Verschelde, J.: Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. ACM Trans. Math. Software **25**(2), 251–276 (1999)
44. Walter, D., Husty, M.: On a 9-bar linkage, its possible configurations and conditions for paradoxical mobility. In: Proceedings of IFToMM World Congress in Mechanism and Machine Science, Besançon, France (2007)
45. Walter, D., Husty, M.L.: A spatial 9-bar linkage, possible configurations and conditions for paradoxical mobility. In: Proceedings of NaCoMM, Bangalore, India, pp. 195–208 (2007)
46. Whiteley, W.: Rigidity and scene analysis, In: Goodman, J.E., O'Rourke, J. (eds.): Handbook of Discrete and Computational Geometry, 2nd edn. chapter 60, pp. 893–916. CRC Press, Boca Raton, Florida (2004)
47. Whiteley, W., Tay, T.S.: Generating isostatic frameworks. Struct. topology **11**, 21–69 (1985)
48. Wunderlich, W.: Gefärlice annahmen der trilateration und bewegliche afchwerke I. Z. Angew. Math. Mech.**57**, 297–304 (1977)
49. Zhu, Z., So, A.M.C., Ye, Y.: Universal rigidity and edge sparsification for sensor network localization. SIAM J. Optim. **20**(6), 3059–3081 (2010)

# Chapter 3
# The Discretizable Molecular Distance Geometry Problem seems Easier on Proteins

**Leo Liberti, Carlile Lavor, and Antonio Mucherino**

**Abstract** Distance geometry methods are used to turn a set of interatomic distances given by Nuclear Magnetic Resonance (NMR) experiments into a consistent molecular conformation. In a set of papers (see the survey [8]) we proposed a Branch-and-Prune (BP) algorithm for computing the set $X$ of all incongruent embeddings of a given protein backbone. Although BP has a worst-case exponential running time in general, we always noticed a linear-like behaviour in computational experiments. In this chapter we provide a theoretical explanation to our observations. We show that the BP is fixed-parameter tractable on protein-like graphs and empirically show that the parameter is constant on a set of proteins from the Protein Data Bank.

**Keywords** Branch-and-Prune • Symmetry • Distance geometry • Fixed-parameter tractable • Protein conformation

L. Liberti
LIX, École Polytechnique, 91128 Palaiseau, France
e-mail: liberti@lix.polytechnique.fr

C. Lavor
Department of Applied Maths, University of Campinas, Campinas-SP, Brazil
e-mail: clavor@ime.unicamp.br

A. Mucherino (✉)
IRISA, University of Rennes 1, Rennes, France
e-mail: antonio.mucherino@irisa.fr

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_3,

## 3.1 Introduction

We consider the following decision problem [9]:

> Discretizable Molecular Distance Geometry Problem (DMDGP). Given a simple weighted undirected graph $G = (V, E, d)$ where $d : E \to \mathbb{R}_+$, $V$ is ordered so that $V = [n] = \{1, \ldots, n\}$, and the following assumptions hold:
>
> 1. For all $v > 3$ and $u \in V$ with $1 \le v - u \le 3$, $\{u, v\} \in E$ (discretization).
> 2. Strict triangular inequalities $d_{v-2,v} < d_{v-2,v-1} + d_{v-1,v}$ hold for all $v > 2$ (non-collinearity)
>
> and given an embedding $x' : \{1,2,3\} \to \mathbb{R}^3$, is there an embedding $x : V \to \mathbb{R}^3$ extending $x'$, such that
>
> $$\forall \{u,v\} \in E \quad \|x_u - x_v\| = d_{uv} \text{ ?} \tag{3.1}$$

An embedding $x$ on $V$ extends an embedding $x'$ on $U \subseteq V$ if $x'$, as a function, is the restriction of $x$ to $U$; an embedding is feasible if it satisfies (3.1). We also consider the following problem variants:

- DMDGP$_K$, i.e. the family of decision problems (parametrized by the positive integer $K$) obtained by replacing each symbol '3' in the DMDGP definition by the symbol '$K$'.
- The $^{\mathsf{K}}$DMDGP, where $K$ is given as part of the input (rather than being a fixed constant as in the DMDGP$_K$).

In both variants, strict triangular inequalities are replaced by strict simplex inequalities, see Eq. (11) in [7]. We remark that DMDGP = DMDGP$_3$. Other related problems also exist in the literature, such as the discretizable distance geometry problem (DDGP) [18], where the discretization axiom is relaxed to require that each vertex $v > K$ has at least $K$ adjacent predecessors. The original results in this chapter, however, only refer to the DMDGP and its variants.

Statements such as "$\forall p \in P\, F(p)$ holds with probability 1", for some uncountable set $P$ and valid sentence $F$, actually mean that there is a Lebesgue-measurable $Q \subseteq P$ with Lebesgue measure 1 w.r.t. $P$ such that $\forall p \in Q\, F(p)$ holds. This notion is less restrictive than genericity based on algebraic independence [2]. We also point out that a statement might hold with probability 1 with respect to a set which has itself Lebesgue measure 0 in a larger set. For example, we will show that the set of $^{\mathsf{K}}$DMDGP instances having an incongruent solution set $X$ with $|X| = 2^\ell$ for some $\ell \in \mathbb{N}$ has measure 1 into the set of all YES instances, which itself is a set of measure 0 in the set of all $^{\mathsf{K}}$DMDGP instances.

The discretization axiom guarantees that the locus of the points that embed $v$ in $\mathbb{R}^3$ is the intersection of the three spheres centred at $v-3, v-2, v-1$ with radii $d_{v-3,v}, d_{v-2,v}, d_{v-1,v}$. If this intersection is non-empty, then it contains two points with probability 1. The complementary zero-measure set contains instances that do not satisfy the NON-COLLINEARITY axiom and which might yield loci for $v$ with zero or uncountably many points. We remark that if the intersection of the three spheres is empty, then the instance is a NO one. We solve ${}^{\mathsf{K}}$DMDGP instances using a recursive algorithm called Branch-and-Prune (BP) [13]: at level $v$, the search is branched according to the (at most two) possible positions for $v$. The BP generates a (partial) binary search tree of height $n$, each full branch of which represents a feasible embedding for the given graph. The BP has exponential worst-case complexity.

The ${}^{\mathsf{K}}$DMDGP and its variants are related to the MOLECULAR DISTANCE GEOMETRY PROBLEM (MDGP), i.e. find an embedding in $\mathbb{R}^3$ of a given simple weighted undirected graph. We denote the $K$-dimensional generalization of the MDGP (with $K$ part of the input) by DISTANCE GEOMETRY PROBLEM (DGP) and the variant with $K$ fixed by DGP$_K$. The MDGP is a good model for determining the structure of molecules given a set of interatomic distances [11, 14], which are usually given by nuclear magnetic resonance (NMR) experiments [21], a technique which allows the detection of interatomic distances below 5.5Å. The DGP has applications to wireless sensor networks [5], statics, robotics and graph drawing among others. In general, the MDGP and DGP implicitly require a search in a continuous Euclidean space [14]. ${}^{\mathsf{K}}$DMDGP instances describe rigid graphs [6], in particular Henneberg type I graphs [12].

The DMDGP is a model for protein backbones. For any atom $v \in V$, the distances $d_{v-1,v}$ and $d_{v-2,v-1}$ are known because they refer to covalent bonds. Furthermore, the angle between $v-2$, $v-1$ and $v$ is known because it is adjacent to two covalent bonds, which implies that $d_{v-2,v}$ is also known by triangular geometry. In general, the distance $d_{v-3,v}$ is smaller than 5Å and can therefore be assumed to be known by NMR experiments; in practice, there are ways to find atomic orders which ensure that $d_{v-3,v}$ is known [7]. There is currently no known protein with $d_{v-3,v-1}$ being *exactly equal* to $d_{v-3,v-2} + d_{v-2,v-1}$ [13].

Over the years, we noticed that the CPU time behaviour of the BP on protein instances looked more linear than exponential. In this chapter we give a theoretical motivation for this observation. More precisely, there are cases where BP is actually fixed-parameter tractable (FPT), and we empirically verify on 45 proteins from the Protein Data Bank (PDB) [1] that they belong to these cases, and always with the parameter value set to the constant 4. The strategy is as follows: we first show that DMDGP$_K$ is NP-hard (Sect. 3.3), then we show that the number of leaf nodes in the BP search tree is a power of 2 with probability 1 (Sect. 3.4.2), and finally we use this information to construct a directed acyclic graph (DAG) representing the number of leaf nodes in function of the graph edges (Sect. 3.5). This DAG allows us to show that the BP is FPT on a class of graphs which provides a good model for proteins (Sect. 3.5.1).

## 3.2 The BP Algorithm

For all $v \in V$ we let $N(v) = \{u \in V \mid \{u,v\} \in E\}$ be the set of vertices *adjacent* to $v$. An embedding of a subgraph of $G$ is called a *partial embedding* of $G$. Let $X$ be the set of embeddings (modulo translations and rotations) solving a given $^{\mathsf{K}}$DMDGP instance.

Since vertex $v$ can be placed in at most two possible positions (the intersection of $K$ spheres in $\mathbb{R}^K$), the BP algorithm tests each in turn and calls itself recursively for every feasible position. BP exploits other edges (than those granted by the DISCRETIZATION axiom) in order to prune branches: a position might be feasible with respect to the distances to the $K$ immediate predecessors $v-1,\ldots,v-K$ but not necessarily with distances to other adjacent predecessors.

For a partial embedding $\bar{x}$ of $G$ and $\{u,v\} \in E$ let $S^{\bar{x}}_{uv}$ be the sphere centered at $x_u$ with radius $d_{uv}$. The BP algorithm is BP($K+1$, $x'$, $\emptyset$) (see Alg. 1), where $x'$ is the initial embedding of the first $K$ vertices mentioned in the $^{\mathsf{K}}$DMDGP definition. By the $^{\mathsf{K}}$DMDGP axioms, $|T| \leq 2$. At termination, $X$ contains all embeddings (modulo rotations and translations) extending $x'$ [9, 13]. Embeddings $x \in X$ can be represented by sequences $\chi(x) \in \{-1,1\}^n$ representing left/right choices when traversing a branch from root to leaf of the search tree. More precisely, (i) $\chi(x)_i = 1$ for all $i \leq K$; (ii) for all $i > K$, $\chi(x)_i = -1$ if $ax_i < a_0$ and $\chi(x)_i = 1$ if $ax_i \geq a_0$, where $ax = a_0$ is the equation of the hyperplane through $x_{i-K},\ldots,x_{i-1}$. For an embedding $x \in X$, $\chi(x)$ is the *chirality* [3] of $x$ (the formal definition of chirality actually states $\chi(x)_0 = 0$ if $ax_i = a_0$, but since this case holds with probability 0, we do not consider it here).

The BP (Alg. 1) can be run to termination to find all possible embeddings of $G$, or stopped after the first leaf node at level $n$ is reached, in order to find just one embedding of $G$. In the last few years we have conceived and described several BP variants targeting different problems [8], including, very recently, problems with interval-type uncertainties on some of the distance values [10]. The BP algorithm is currently the only method which is able to find all incongruent embeddings for

**Algorithm 1** BP($v$, $\bar{x}$, $X$)

```
Require: A vertex $v \in V \setminus [K]$, a partial embedding $\bar{x} = (x_1,\ldots,x_{v-1})$, a set $X$.
$T = \bigcap_{u \in N(v),\, u<v} S^{\bar{x}}_{uv}$;
for $p \in T$ do
    $x \leftarrow (\bar{x}, p)$
    if $v = n$ then
        $X \leftarrow X \cup \{x\}$
    else
        BP($v+1, x, X$)
    end if
end for
```

a given protein backbone. Compared to continuous search algorithms (e.g. [17]), the performance of the BP algorithm is impressive from the point of view of both efficiency and reliability.

## 3.3 Complexity

Any class of YES instances where each vertex $v$ only has distances to the $K$ immediate predecessors provides a full BP binary search tree (after level $K$), and therefore shows that the BP is an exponential-time algorithm in the worst case. One remarkable feature of the computational experiments conducted on our BP implementation [19] on protein instances is that the exponential-time behaviour of the BP algorithm was never noticed empirically.

Restricting $d$ to only take integer values, the DGP$_1$ is NP-complete by reduction from SUBSET-SUM, the DGP$_K$ is (strongly) NP-hard by reduction from 3-SAT, and the DGP is (strongly) NP-hard by induction on $K$ [20]. Only the DGP$_1$ is known to be NP-complete, because if $d$ takes integer values then the YES-certificate $x$ (the embedding) can be chosen to have integer values too.

The DMDGP is NP-hard by reduction from SUBSET-SUM (Theorem 3 in [9]). We generalize this result to the ${}^{\mathsf{K}}$DMDGP.

**Theorem 3.1.** *The DMDGP$_K$ is NP-hard for all $K \geq 2$.*

*Proof.* Let $a = (a_1, \ldots, a_N)$ be an instance of SUBSET-SUM consisting of positive integers, and define an instance of DMDGP$_K$ where $V = \{0, \ldots, KN\}$, $E$ includes $\{i, i+j\}$ for all $j \in \{1, \ldots, K\}$ and $i \in \{0, \ldots, KN - j\}$, and

$$\forall i \in \{0, \ldots, KN-1\} \quad d_{i,i+1} = a_{\lfloor i/K \rfloor} \tag{3.2}$$

$$\forall j \in \{2, \ldots, K\}, i \in \{0, \ldots, KN-j\} \quad d_{i,i+j} = \sqrt{\sum_{\ell=1}^{j} d^2_{i+\ell-1,i+\ell}} \tag{3.3}$$

$$d_{0,KN} = 0. \tag{3.4}$$

Let $s \in \{-1, 1\}^N$ be a solution of the SUBSET-SUM instance $a$. We let $x_0 = 0$ and for all $i = K(\ell - 1) + j > 0$ we let $x_i = x_{i-1} + s_\ell a_\ell e_j$, where $e_j$ is the vector with a one in component $j$ and zero elsewhere. Because $\sum_{\ell \leq N} s_\ell a_\ell = 0$, if $s$ solves the SUBSET-SUM instance $a$, then, by inspection, $x$ solves the corresponding DMDGP instance Eqs. (3.2)–(3.4). Conversely, let $x$ be an embedding that solves Eqs. (3.2)–(3.4), where we assume without loss of generality that $x_0 = 0$. Then Eq. (3.3) ensures that the line through $x_i, x_{i-1}$ is orthogonal to the line through $x_{i-1}, x_{i-2}$ for all $i > 1$, and again we assume without loss of generality that, for all $j \in \{1, \ldots, K\}$, the lines through $x_{j-1}, x_j$ are parallel to the $i$th coordinate axis. Now consider the chirality $\chi$ of $x$: because all distance segments are orthogonal, for each $j \leq K$ the $j$th coordinate is given by $x_{KN,j} = \sum_{i \bmod K = j} \chi_i a_{\lfloor i/K \rfloor}$. Since $d_{0,KN} = 0$, for all $j \le K$, we have $0 = x_{KN,j} = \sum_{\ell \le N} \chi_{K(\ell-1)+j} a_\ell$, which implies that, for all $j \le K$, $s^j = (\chi_{K(\ell-1)+j} \mid 1 \le \ell \le N)$ is a solution for the SUBSET-SUM instance $a$. □

**Corollary 3.1.** *The* $^{\mathsf{K}}$*DMDGP is NP-hard.*

*Proof.* Every specific instance of the $^{\mathsf{K}}$DMDGP specifies a fixed value for $K$ and hence belongs to the DMDGP$_K$. Hence the result follows by inclusion. □

## 3.4 Partial Reflection Symmetries

The results in this section are also found in [16], but the presentation below, which is based on group theory, is new and (we hope) clearer. We partition $E$ into the sets $E_D = \{\{u,v\} \mid |v-u| \le K\}$ and $E_P = E \smallsetminus E_D$. We call $E_D$ the *discretization edges* and $E_P$ the *pruning edges*. Discretization edges guarantee that a DGP instance is in the $^{\mathsf{K}}$DMDGP. Pruning edges are used to reduce the BP search space by pruning its tree. In practice, pruning edges might make the set $T$ in Alg. 1 have cardinality 0 or 1 instead of 2. We assume $G$ is a YES instance of the $^{\mathsf{K}}$DMDGP.

### *3.4.1 The Discretization Group*

Let $G_D = (V, E_D, d)$ and $X_D$ be the set of embeddings of $G_D$; since $G_D$ has no pruning edges, the BP search tree for $G_D$ is a full binary tree and $|X_D| = 2^{n-K}$. The discretization edges arrange the embeddings so that, at level $\ell$, there are $2^{\ell-K}$ possible positions for the vertex $v$ with rank $\ell$. We assume that $|T| = 2$ (see Alg. 1) at each level $v$ of the BP tree, an event which, in absence of pruning edges, happens with probability 1 — thus many results in this section are stated with probability 1. Let therefore $T = \{x_v^0, x_v^1\}$ be the two possible embeddings of $v$ at a certain recursive call of Alg. 1 at level $v$ of the BP tree; then because $T$ is an intersection of $K$ spheres, $x_v^1$ is the reflection of $x_v^0$ through the hyperplane defined by $x_{v-K}, \ldots, x_{v-1}$. Denote this reflection operator by $R_x^v$.

**Theorem 3.2 (Cororollary 4.6 and Theroem 4.9 in [16]).** *With probability 1, for all* $v > K$ *and* $u < v - K$ *there is a set* $H^{uv}$*, with* $|H^{uv}| = 2^{v-u-K}$*, of real positive values such that for each* $x \in X$ *we have* $\|x_v - x_u\| \in H^{uv}$*. Furthermore,* $\forall x \in X$ $\|x_v - x_u\| = \|R_x^{u+K}(x_v) - x_u\|$ *and* $\forall x' \in X$*, if* $x'_v \notin \{x_v, R_x^{u+K}(x_v)\}$ *then* $\|x_v - x_u\| \neq \|x'_v - x_u\|$*.*

We sketch the proof in Fig. 3.1 for $K = 2$; the solid circles at levels $3,4,5$ mark equidistant levels from 1. The dashed circles represent the spheres $S_{uv}^x$ (see Alg. 1). Intuitively, two branches from level 1 to level 4 or 5 will have equal segment lengths but different angles between consecutive segments, which will cause the end nodes to be at different distances from the node at level 1.

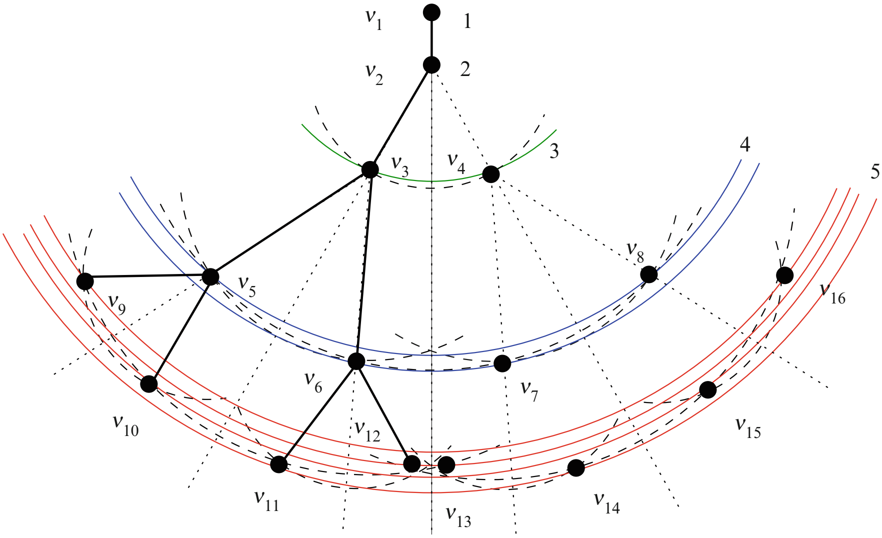


**Fig. 3.1** A pruning edge $\{1,4\}$ prunes either $v_6, v_7$ or $v_5, v_8$

For any nonzero vector $y \in \mathbb{R}^K$ let $\rho^y$ be the reflection operator through the hyperplane passing through the origin and normal to $y$. If $y$ is normal to the hyperplane defined by $x_{v-K}, \ldots, x_{v-1}$, then $\rho^y = R_x^v$.

**Lemma 3.1.** *Let $x \neq y \in \mathbb{R}^K$ and $z \in \mathbb{R}^K$ such that $z$ is not in the hyperplanes through the origin and normal to $x, y$. Then $\rho^x \rho^y z = \rho^{\rho^x y} \rho^x z$.*

*Proof.* Fig. 3.2 gives a proof sketch for $K = 2$. By considering the reflection $\rho^{\rho^x y}$ of the map $\rho^y$ through $\rho^x$, we get $\|z - \rho^y z\| = \|\rho^x z - \rho^{\rho^x y} \rho^x z\|$. By reflection through $\rho^x$ we get $\|O - z\| = \|O - \rho^x z\|$ and $\|O - \rho^y z\| = \|O - \rho^x \rho^y z\|$. By reflection through $\rho^y$ we get $\|O - z\| = \|O - \rho^y z\|$. By reflection through $\rho^{\rho^x y}$ we get $\|O - \rho^x z\| = \|O - \rho^{\rho^x y} \rho^x z\|$. The triangles $\triangle(z, O, \rho^y z)$ and $\triangle(\rho^x z, O, \rho^{\rho^x y} \rho^x z)$ are then equal because the side lengths are pairwise equal. Also, reflection of $\triangle(z, O, \rho^y z)$ through $\rho^x$ yields $\triangle(z, O, \rho^y z) = \triangle(\rho^x z, O, \rho^x \rho^y z)$, whence $\rho^{\rho^x y} \rho^x z = \rho^x \rho^y z$. □

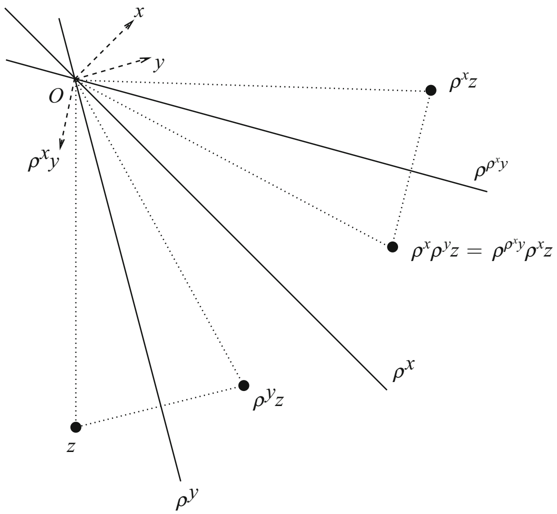
For $v > K$ and $x \in X$ we now define partial reflection operators:

$$g_v(x) = (x_1, \ldots, x_{v-1}, R_x^v(x_v), \ldots, R_x^v(x_n)). \tag{3.5}$$

The $g_v$'s map an embedding $x$ to its partial reflection with first branch at $v$. It is easy to show that the $g_v$'s are injective with probability 1 and idempotent. Further, the $g_v$'s commute.

**Lemma 3.2.** *For $x \in X$ and $u, v \in V$ such that $u, v > K$, $g_u g_v(x) = g_v g_u(x)$.*

**Fig. 3.2** Reflecting through $\rho^y$ first and $\rho^x$ later is equivalent to reflecting through $\rho^x$ first and (the reflection of $\rho^y$ through $\rho^x$) later



*Proof.* Assume without loss of generality $u < v$. Then:

$$\begin{aligned}
g_u g_v(x) &= g_u(x_1, \ldots, x_{v-1}, R_x^v(x_v), \ldots, R_x^v(x_n)) \\
&= (x_1 \ldots, x_{u-1}, R_{g_v(x)}^u(x_u), \ldots, R_{g_v(x)}^u R_x^v(x_v), \ldots, R_{g_v(x)}^u R_x^v(x_n)) \\
&= (x_1 \ldots, x_{u-1}, R_x^u(x_u), \ldots, R_{g_u(x)}^v R_x^u(x_v), \ldots, R_{g_u(x)}^v R_x^u(x_n)) \\
&= g_v(x_1, \ldots, x_{u-1}, R_x^u(x_u), \ldots, R_x^u(x_n)) \\
&= g_v g_u(x),
\end{aligned}$$

where $R_{g_v(x)}^u R_x^v(x_w) = R_{g_u(x)}^v R_x^u(x_w)$ for each $w \geq v$ by Lemma 3.1. □

We define the *discretization group* to be the group $\mathscr{G}_D = \langle g_v \mid v > K \rangle$ generated by the $g_v$'s.

**Corollary 3.2.** *With probability 1, $\mathscr{G}_D$ is an Abelian group isomorphic to $C_2^{n-K}$ (the Cartesian product consisting of $n-K$ copies of the cyclic group of order 2).*

For all $v > K$ let $\gamma_v = (1, \ldots, 1, -1_v, \ldots, -1)$ be the vector consisting of one's in the first $v-1$ components and $-1$ in the last components. Then the $g_v$ actions are naturally mapped onto the chirality functions.

**Lemma 3.3.** *For all $x \in X$, $\chi(g_v(x)) = \chi(x) \circ \gamma_v$, where $\circ$ is the Hadamard (i.e. component-wise) product.*

This follows by definition of $g_v$ and of chirality of an embedding.

Because, by Alg. 1, each $x \in X$ has a different chirality, for all $x, x' \in X$ there is $g \in \mathscr{G}_D$ such that $x' = g(x)$, i.e. the action of $\mathscr{G}_D$ on $X$ is transitive. By Theorem 3.2, the distances associated to the discretization edges are invariant with respect to the discretization group.

### 3.4.2 The Pruning Group

Consider a pruning edge $\{u,v\} \in E_P$. By Theroem 3.2, with probability 1 we have $d_{uv} \in H^{uv}$, otherwise $G$ cannot be a YES instance (against the hypothesis). Also, again by Theroem 3.2, $d_{uv} = \|x_u - x_v\| \neq \|g_w(x)_u - g_w(x)_v\|$ for all $w \in \{u+K+1,\ldots,v\}$ (e.g. the distance $\|v_1 - v_9\|$ in Fig. 3.1 is different from all its reflections $\|v_1 - v_h\|$, with $h \in \{10,11,12\}$, w.r.t. $g_4, g_5$). We therefore define the *pruning group*

$$\mathscr{G}_P = \langle g_w \mid w > K \wedge \forall \{u,v\} \in E_P \ (w \notin \{u+K+1,\ldots,v\}) \rangle.$$

By definition, $\mathscr{G}_P \leq \mathscr{G}_D$ and the distances associated with the pruning edges are invariant with respect to $\mathscr{G}_P$.

**Theorem 3.3.** *The action of $\mathscr{G}_P$ on X is transitive with probability 1.*

*Proof.* This theorem follows from Theorem 5.4 in [15], but here is another, hopefully simpler, proof. Let $x, x' \in X$, we aim to show that $\exists g \in \mathscr{G}_P$ such that $x' = g(x)$ with probability 1. Since the action of $\mathscr{G}_D$ on $X$ is transitive, $\exists g \in \mathscr{G}_D$ with $x' = g(x)$. Now suppose $g \notin \mathscr{G}_P$, then there is a pruning edge $\{u,v\} \in E_P$ and an $\ell \in \mathbb{N}$ s.t. $g = \prod_{h=1}^{\ell} g_{v_h})$ for some vertex set $\{v_1,\ldots,v_\ell > K\}$ including at least one vertex $w \in \{u+K+1,\ldots,v\}$. By Theorem 3.2, as remarked above, this implies that $d_{uv} = \|x_u - x_v\| \neq \|g_w(x)_u - g_w(x)_v\|$ with probability 1. If the set $Q = \{v_1,\ldots,v_\ell\} \cap \{u+K+1,\ldots,v\}$ has cardinality 1, then $g_w$ is the only component of $g$ not fixing $d_{uv}$, and hence $x' = g(x) \notin X$, against the hypothesis. Otherwise, the probability of another $z \in Q \smallsetminus \{w\}$ yielding $\|x_u - x_v\| = \|g_z g_w(x)_u - g_z g_w(x)_v\|$, notwithstanding the fact that $\|g_w(x)_u - g_w(x)_v\| \neq \|x_u - x_v\| \neq \|g_z(x)_u - g_z(x)_v\|$, is zero; and by induction this also covers any cardinality of $Q$. Therefore $g \in \mathscr{G}_P$ and the result follows. □

**Theorem 3.4.** *With probability 1, $\exists \ell \in \mathbb{N}\ |X| = 2^\ell$.*

*Proof.* Since $\mathscr{G}_D \cong C_2^{n-K}$, $|\mathscr{G}_D| = 2^{n-K}$. Since $\mathscr{G}_P \leq \mathscr{G}_D$, $|\mathscr{G}_P|$ divides the order of $|\mathscr{G}_D|$, which implies that there is an integer $\ell$ with $|\mathscr{G}_P| = 2^\ell$. By Theorem 3.3, the action of $\mathscr{G}_P$ on $X$ only has one orbit, i.e. $\mathscr{G}_P x = X$ for any $x \in X$. By idempotency, for $g, g' \in \mathscr{G}_P$, if $gx = g'x$ then $g = g'$. This implies $|\mathscr{G}_P x| = |\mathscr{G}_P|$. Thus, for any $x \in X$, $|X| = |\mathscr{G}_P x| = |\mathscr{G}_P| = 2^\ell$. □

## 3.5 Bounded Treewidth

The results of the previous section allow us to express the number of nodes at each level of the BP search tree in function of the level rank and the pruning edges. Fig. 3.3 shows a DAG $\mathscr{D}_{uv}$ that represents the number of valid BP search tree nodes in function of pruning edges between two vertices $u,v \in V$ such that $v > K$ and $u < v - K$. The first line shows different values for the rank of $v$ w.r.t. $u$; an arc labelled with an integer $i$ implies the existence of a pruning edge $\{u+i,v\}$ (arcs with $\vee$-expressions replace parallel arcs with different labels). An arc is unlabelled if there is no pruning edge $\{w,v\}$ for any $w \in \{u,\ldots,v-K-1\}$. The vertices of the DAG are arranged vertically by BP search tree level and are labelled with the number of BP nodes at a given level, which is always a power of two by Theroem 3.4. A path in this DAG represents the set of pruning edges between $u$ and $v$, and its incident vertices show the number of valid nodes at the corresponding levels. For example, following unlabelled arcs corresponds to no pruning edge between $u$ and $v$ and leads to a full binary BP search tree with $2^{v-K}$ nodes at level $v$.

### 3.5.1 Fixed-Parameter Tractable Behaviour

For a given $G_D$, each possible pruning edge set $E_P$ corresponds to a path spanning all columns in $\mathscr{D}_{1n}$. Instances with diagonal (Proposition 3.1) or below-diagonal (Proposition 3.2) $E_P$ paths yield BP trees whose width is bounded by $O(2^{v_0})$ where $v_0$ is small w.r.t. $n$.

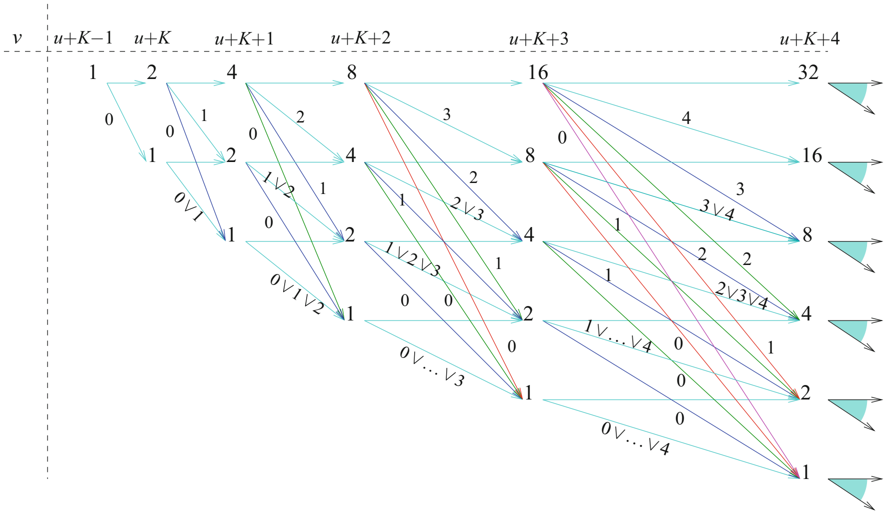


**Fig. 3.3** Number of valid BP nodes (vertex label) at level $u+K+\ell$ (column) in function of the pruning edges (path spanning all columns)

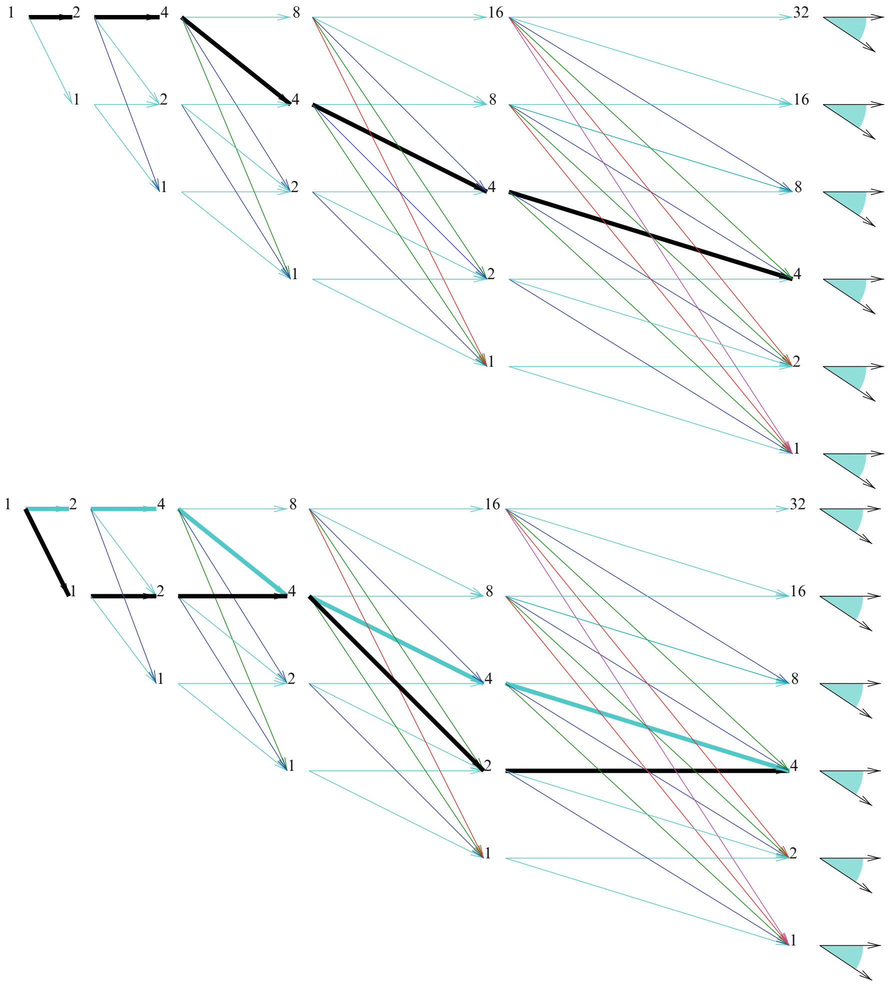
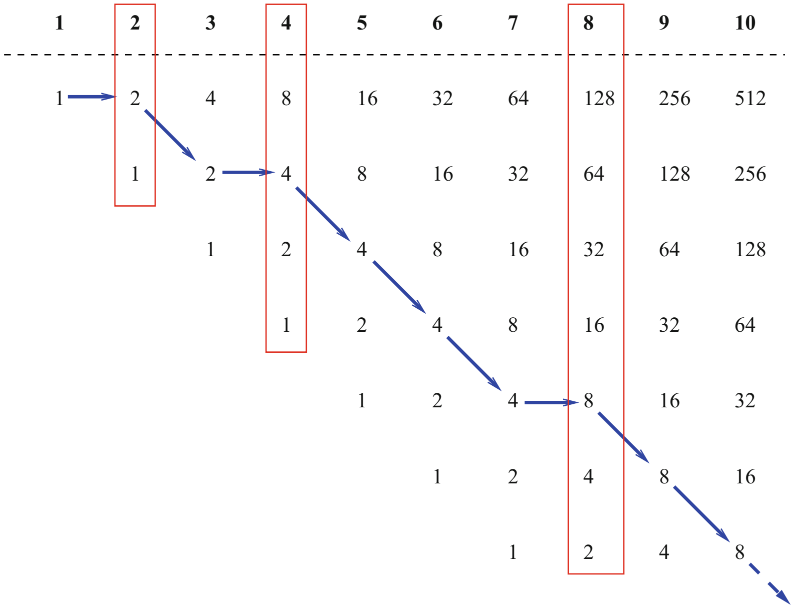


**Fig. 3.4** A path $\mathsf{p}_0$ yielding treewidth 4 (*above*) and another path below $\mathsf{p}_0$ (*below*)

**Proposition 3.1.** *If* $\exists v_0 > K$ *s.t.* $\forall v > v_0 \exists u < v - K$ *with* $\{u, v\} \in E_P$ *then the BP search treewidth is bounded by* $2^{v_0 - K}$.

This corresponds to a path $\mathsf{p}_0 = (1, 2, \ldots, 2^{v_0 - K}, \ldots, 2^{v_0 - K})$ that follows unlabelled arcs up to level $v_0$ and then arcs labelled $v_0 - K - 1$, $v_0 - K - 1 \vee v_0 - K$ and so on, leading to nodes that are all labelled with $2^{v_0 - K}$ (Fig. 3.4, left).

**Proposition 3.2.** *If* $\exists v_0 > K$ *such that every subsequence* $s$ *of consecutive vertices* $> v_0$ *with no incident pruning edge is preceded by a vertex* $v_s$ *such that* $\exists u_s < v_s$ $(v_s - u_s \geq |s| \wedge \{u_s, v_s\} \in E_P)$*, then the BP search treewidth is bounded by* $2^{v_0 - K}$.



| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
|  | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 |
|  |  | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
|  |  |  | 1 | 2 | 4 | 8 | 16 | 32 | 64 |
|  |  |  |  | 1 | 2 | 4 | 8 | 16 | 32 |
|  |  |  |  |  | 1 | 2 | 4 | 8 | 16 |
|  |  |  |  |  |  | 1 | 2 | 4 | 8 |

**Fig. 3.5** A path yielding treewidth $O(n)$

This situation corresponds to a below-diagonal path, Fig. 3.4 (right). In general, for those instances for which the BP search treewidth has a $O(2^{v_0}\log n)$ bound, the BP has a worst-case running time $O(2^{v_0}L2^{\log n}) = O(Ln)$, where $L$ is the complexity of computing $T$. Since $L$ is typically constant in $n$ [4], for such cases the BP runs in time $O(2^{v_0}n)$. Let $V' = \{v \in V \mid \exists \ell \in \mathbb{N}\ (v = 2^{\ell})\}$.

**Proposition 3.3.** *If $\exists v_0 > K$ s.t. for all $v \in V \smallsetminus V'$ with $v > v_0$ there is $u < v - K$ with $\{u,v\} \in E_P$ then the BP search treewidth at level n is bounded by $2^{v_0}n$.*

This corresponds to a path along the diagonal $2^{v_0}$ apart from logarithmically many vertices in $V$ (those in $V'$), at which levels the BP doubles the number of search nodes (Fig. 3.5). For a pruning edge set $E_P$ as in Proposition 3.3, or yielding a path below it, the BP runs in $O(2^{v_0}n^2)$.

### *3.5.2 Empirical Verification*

We consider a set of 45 protein instances from the PDB. Since PDB data include the Euclidean embedding of the protein, we compute all distances, then filter all those that are longer than 5.5Å, which is a reasonable estimate for NMR measures [21]. We then form the weighted graph and embed it with our implementation of the BP

**Table 3.1** Computation of minimum $v_0$ in PDB instances

| | | $v_0$ | | | $v_0$ | |
|---|---|---|---|---|---|---|
| *PDB ID* | $\|V\|$ | Proposition 3.1 | *PDB ID* | $\|V\|$ | Proposition 3.1 | Proposition 3.2 |
| 1brv | 57 | 4 | 1a70 | 291 | 269 | 4 |
| 1a11 | 75 | 4 | 1ag4 | 309 | 4 | – |
| 1acw | 87 | 4 | 2hsy | 312 | 4 | – |
| 1ppt | 108 | 4 | 1acz | 324 | 4 | – |
| 1bbl | 111 | 4 | 1poa | 354 | 4 | – |
| 1erp | 114 | 4 | 1fs3 | 372 | 4 | – |
| 1aqr | 120 | 4 | 1itm | 390 | 4 | – |
| 1k1v | 123 | 4 | 1mbn | 459 | 369 | 4 |
| 1h1j | 132 | 4 | 1ngl | 537 | 4 | – |
| 1ed7 | 135 | 4 | 1b4c | 552 | 280 | 4 |
| 1dv0 | 135 | 4 | 1la3 | 564 | 4 | – |
| 1crn | 138 | 4 | 1a23 | 567 | 4 | – |
| 1jkz | 138 | 4 | 1oy2 | 573 | 4 | – |
| 1ahl | 147 | 4 | 2ron | 726 | 4 | – |
| 1ptq | 150 | 4 | 1d8v | 789 | 4 | – |
| 1brz | 159 | 4 | 1rgs | 792 | 203 | 4 |
| 1ccq | 180 | 4 | 1q8k | 900 | 4 | – |
| 1hoe | 222 | 4 | 1ezo | 1110 | 4 | – |
| 1bqx | 231 | 4 | 1m40 | 1224 | 4 | – |
| 1pht | 249 | 4 | 1bpm | 1443 | 1319 | 4 |
| 1a2s | 267 | 4 | 1n4w | 1610 | 4 | – |
| 1jk2 | 270 | 4 | 1mqq | 2032 | 4 | – |
| | | | 3b34 | 2790 | 4 | – |

algorithm [19]. It turns out that 40 proteins satisfy Proposition. 3.1 and five proteins satisfy Proposition. 3.2, all with $v_0 = 4$ (see Table 3.1). This is consistent with the computational insight [9] that BP empirically displays a polynomial (specifically, linear) complexity on real proteins.

## 3.6 Conclusion

In this chapter we provide a theoretical basis to the empirical observation that the BP never seems to attain its exponential worst-case time bound on DMDGP instances from proteins. Other original contributions include a generalization of an NP-hardness proof to the $^{\mathsf{K}}$DMDGP and a new presentation, based on group theory and involving new proofs, of the fact that the cardinality of the solution set of YES instances of the $^{\mathsf{K}}$DMDGP is a power of two with probability 1.

**Acknowledgements** The authors wish to thank the Brazilian research agencies FAPESP and CNPq and the French research agency CNRS and École Polytechnique for financial support.

## References

1. Berman, H. Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I., Bourne, P.: The protein data bank. Nucleic Acid Res. **28**, 235–242 (2000)
2. Connelly, R.: Generic global rigidity. Discrete Comput. Geom. **33**, 549–563 (2005)
3. Crippen, G., Havel, T.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
4. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global Optim. **26**, 321–333 (2003)
5. Eren, T., Goldenberg, D., Whiteley, W., Yang, Y., Morse, A., Anderson, B., Belhumeur, P.: Rigidity, computation, and randomization in network localization. In: IEEE Infocom Proceedings, 2673–2684 (2004)
6. Graver, J.E., Servatius, B., Servatius, H.: Combinatorial Rigidity. Graduate Studies in Math., AMS (1993)
7. Lavor, C. Lee, J., John, A.L.S., Liberti, L., Mucherino, A., Sviridenko, M.: Discretization orders for distance geometry problems. Optim. Lett. **6**, 783–796 (2012)
8. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
9. Lavor, C., Liberti, L. Maculan, N. Mucherino, A.: The discretizable molecular distance geometry problem. Comput. Optim. Appl. **52**, 115–146 (2012)
10. Lavor, C., Liberti, L., Mucherino, A.: On the solution of molecular distance geometry problems with interval data. In: IEEE Conference Proceedings, International Workshop on Computational Proteomics (IWCP10), International Conference on Bioinformatics and Biomedicine (BIBM10), Hong Kong, 77–82 (2010)
11. Lavor, C., Mucherino, A., Liberti, L., Maculan, N.: On the computation of protein backbones by using artificial backbones of hydrogens. J. Global Optim. **50**, 329–344 (2011)
12. Liberti, L., Lavor, C.: On a relationship between graph realizability and distance matrix completion. In: Kostoglou, V., Arabatzis, G., Karamitopoulos, L. (eds.) Proceedings of BALCOR, vol. I, pp. 2–9. Hellenic OR Society, Thessaloniki (2011)
13. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
14. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
15. Liberti, L., Masson, B., Lavor, C., Lee, J., Mucherino, A.: On the number of solutions of the discretizable molecular distance geometry problem, Tech. Rep. 1010.1834v1[cs.DM], arXiv (2010)
16. Liberti, L., Masson, B., Lee, J., Lavor, C., Mucherino, A.: On the number of solutions of the discretizable molecular distance geometry problem, Lecture Notes in Computer Science. In: Wang, W., Zhu, X., Du, D-Z. (eds.) Proceedings of the 5th Annual International Conference on Combinatorial Optimization and Applications (COCOA11), Zhangjiajie, China, vol.6831, pp. 322–342 (2011)
17. Moré, J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**(3), 814–846 (1997)
18. Mucherino, A., Lavor, C., Liberti, L.: The discretizable distance geometry problem, to appear in Optimization Letters (DOI:10.1007/s11590-011-0358-3).
19. Mucherino, A., Liberti, L., Lavor, C.: `MD-jeep`: an implementation of a branch and prune algorithm for distance geometry problems, Lectures Notes in Computer Science. In: Fukuda, K., et al. (eds.) Proceedings of the Third International Congress on Mathematical Software (ICMS10), Kobe, Japan, vol. 6327, pp. 186–197 (2010)
20. Saxe, J.: Embeddability of weighted graphs in *k*-space is strongly NP-hard. In: Proceedings of $17^{th}$ Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
21. Schlick, T.: Molecular Modelling and Simulation: An Interdisciplinary Guide. Springer, New York (2002)

# Chapter 4
# Spheres Unions and Intersections and Some of their Applications in Molecular Modeling

**Michel Petitjean**

**Abstract** The geometrical and computational aspects of spheres unions and intersections are described. A practical analytical calculation of their surfaces and volumes is given in the general case: any number of intersecting spheres of any radii. Applications to trilateration and van der Waals surfaces and volumes calculation are considered. The results are compared to those of other algorithms, such as Monte Carlo methods, regular grid methods, or incomplete analytical algorithms. For molecular modeling, these latter algorithms are shown to give strongly overestimated values when the radii values are in the ranges recommended in the literature, while regular grid methods are shown to give a poor accuracy. Other concepts related to surfaces and volumes of unions of spheres are evoked, such as Connolly's surfaces, accessible surface areas, and solvent-excluded volumes.

## 4.1 Introduction

We denote by $E^d$ the $d$-dimensional Euclidean space. The relation between spheres intersections and distance geometry can be exemplified by the trilateration problem: given, in $E^3$, three fixed points $c_1$, $c_2$, $c_3$ with known coordinates, locate an unknown point $x$ from its respective distances $d(x,c_1)$, $d(x,c_2)$, $d(x,c_3)$ to these fixed points. This problem can be reformulated as a spheres intersection problem: given three spheres of respective centers $c_1, c_2, c_3$ and respective radii $R_1 = d(x,c_1)$, $R_2 = d(x,c_2)$, $R_3 = d(x,c_3)$, locate the points at the intersection of their boundaries. Such a reformulation allows us to realize immediately that, when the centers are not aligned and when the intersection of the three spheres is not void, there are in

M. Petitjean (✉)
MTi, University Paris 7, INSERM UMR-S 973, Paris, France
e-mail: petitjean.chiral@gmail.com

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_4,

general two solution points which are mirror images through the plane containing $c_1, c_2, c_3$. Then the experimentalist can decide which of these two solution points is relevant, e.g., via an appropriate determinant calculus.

Many applications occur in molecular modeling because atoms can be modeled as hard spheres and molecules can be modeled as unions of spheres. The values of the atomic radii depend on how they are defined and measured. Usually, they are needed to compute the van der Waals surface and the van der Waals volume of a molecule, i.e., the surface and the volume defined by the union of the atomic spheres of the molecule. Other molecular concepts which are based on atomic spheres include the accessible surface areas, the Connolly surfaces, and the solvent excluded volumes [4, 20, 24, 25, 36, 47, 54]. They all depend on an additional probe sphere assumed to modelize a solvent molecule. Discussions on the physical meaning of the molecular surfaces and volumes and on the adequate choice of radii values have been done in [5, 29–31], but this is not in the scope of this chapter. Practical radii values can be found in [5, 15, 23, 43, 46, 50, 55]. It happened also that whole molecules were implicitely or explicitely modeled by spheres. For example, in the alpha-shape model of pockets and channels in proteins [13], the ligand is represented by a probe sphere. The alpha-shape model is strongly connected with Delaunay triangulations and Voronoi diagrams [11, 12]. More recently these channels were computed as union of spheres centered on the vertices of a grid [41,42]. In any case, structural chemists know that the shapes of the molecules are generally far from being spherical: better models are minimal height and minimal radius enclosing cylinders [39], but spheres are much easier to handle so they are still much used for molecular modeling.

## 4.2 The Analytical Calculation

We consider $n$ spheres in $E^d$ of given centers and radii, and we look for the calculation of the surface and the volume of their union or of their intersection. Let $V_i$ be the volume of the sphere $i$, $i \in \{1..n\}$, with fixed center $c_i$ and fixed radius $R_i$. We denote by $V_{i_1 i_2}$ the volume of the intersection of the spheres $i_1$ and $i_2$, $V_{i_1 i_2 i_3}$ the volume of the intersections of the spheres $i_1$ and $i_2$ and $i_3$, etc. Similarly, $S_i$ is the surface (i.e., its area) of the sphere $i$, $S_{i_1 i_2}$ is the one of the intersection of the spheres $i_1$ and $i_2$, etc. $V$ is the volume of the union of the $n$ spheres and $S$ is its surface. Although we will exhibit the full analytical calculation of $V$ and $S$ only for $d = 3$, it is enlighting to do some parts of this calculation in $E^d$. We will specify the dimension when needed.

### *4.2.1 Spheres and Lens*

We set $n = 1$: we consider one sphere of fixed radius $R$ in $E^d$. We denote respectively by $V_{\lrcorner d}(r)$ and $S_{\lrcorner d}(r)$ the volume and the surface of this sphere as functions of the

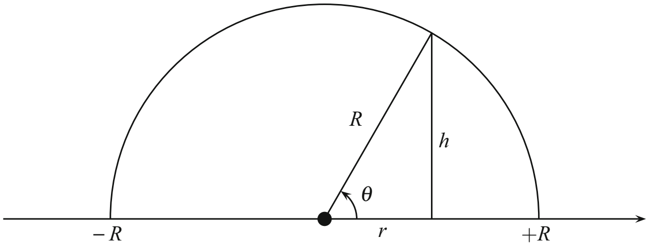


**Fig. 4.1** Calculation of the sphere in $E^d$ via summation of the volumes of spheres in $E^{d-1}$

radius $r$. For example, $V_{\_1}(R)$ is the length of a segment of half-length $R$, $S_{\_2}(R)$ is the perimeter of the circle of radius $R$, $V_{\_2}(R)$ is its surface, etc.

**Theorem 4.1.** *The volume and the surface of the sphere of radius R in $E^d$ are respectively given in Eqs. (4.1) and (4.2):*

$$V_{\_d}(R) = \frac{\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)} R^d, \quad d \geq 1 \tag{4.1}$$

$$S_{\_d}(R) = d\frac{\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)} R^{d-1}, \quad d \geq 2 \tag{4.2}$$

*Proof.* We know that $V_{\_1}(R)$ and $S_{\_2}(R)$ stand. We get $V_{\_2}(R)$ by integration: $V_{\_2}(R) = \int\limits_0^R S_{\_2}(r)dr$, and conversely $S_{\_2}(R)$ is retrieved by derivating $V_{\_2}(R)$. For similar reasons, it suffices to prove Eq. (4.1) to be true and Eq. (4.2) is proved to stand by derivation of $V_{\_d}(R)$. We proceed by recurrence and we calculate $V_{\_d}(R)$ by integration of $V_{\_(d-1)}(h)$ with $h = \sqrt{R^2 - r^2}$, as indicated in Fig. 4.1: $V_{\_d}(R) = 2\int\limits_0^R V_{\_(d-1)}((R^2-r^2)^{\frac{1}{2}})dr.$

Setting $r = R\sqrt{t}$, the integral is expressed with the $\beta$ function:

$$V_{\_d}(R) = R^d \frac{\pi^{\frac{d-1}{2}}}{\Gamma(\frac{d+1}{2})} \int_0^1 t^{-\frac{1}{2}}(1-t)^{\frac{d-1}{2}}dt = R^d \frac{\pi^{\frac{d-1}{2}}}{\Gamma(\frac{d+1}{2})} \beta\left(\frac{1}{2}, \frac{d+1}{2}\right).$$

Since $\beta(\frac{1}{2}, \frac{d+1}{2}) = \dfrac{\Gamma(\frac{1}{2})\Gamma(\frac{d+1}{2})}{\Gamma(\frac{d}{2}+1)}$ and $\Gamma(\frac{1}{2}) = \pi^{\frac{1}{2}}$, we get the desired result.

Remark: expanding the expression of the $\Gamma$ function shows that $V_{\_d}(R)$ and $S_{\_d}(R)$ are proportional to $\pi^{\frac{d}{2}}$ (when $d$ is even) or to $\pi^{\frac{d-1}{2}}$ (when $d$ is odd).

It is useful to calculate the volume $V_{\_d}(R,\theta)$ and the surface $S_{\_d}(R,\theta)$ of the spherical cap defined by the angle $\theta$ in Fig. 4.1. They can be calculated by recurrence as above via integration using respectively the expressions of $V_{\_(d-1)}(R,\theta)$ and $S_{\_(d-1)}(R,\theta)$, and with the help of incomplete $\beta$ functions.

For clarity, we recall in Eqs. (4.3) and (4.4) the results for $d = 3$, obtained, respectively, from summation of the elementary cylinders volumes $(\pi h^2)(dr)$ and of the elementary truncated cones surfaces $(2\pi h)(Rd\theta)$:

$$V_{\_3}(R,\theta) = \frac{\pi R^3}{3}(1-\cos\theta)^2(2+\cos\theta), \tag{4.3}$$

$$S_{\_3}(R,\theta) = 2\pi R^2(1-\cos\theta). \tag{4.4}$$

Remark: in Eqs. (4.3) and (4.4), $\theta$ takes values in $[0;\pi]$.

### 4.2.2 Lens and Radical Hyperplanes

We set $n = 2$. The intersection of two spheres in $E^d$ of respective centers $c_1$ and $c_2$ and radius $R_1$ and $R_2$ is either empty, or reduces to one point, or is a lens, or is the smallest sphere in the case it is included in the largest sphere. The case of interest is the one of one lens.

The lens exists when:

$$|R_1 - R_2| \leq \|c_2 - c_1\| \leq R_1 + R_2. \tag{4.5}$$

This lens is bounded by two spherical caps separated by a $(d-1)$-hyperplane orthogonal to the direction $c_2 - c_1$ and intersecting the axis $c_2 - c_1$ at the point $t_{12}$. This $(d-1)$-hyperplane is called a *radical hyperplane*, or simply a *radical plane* when $d = 3$.

**Theorem 4.2.** *The location of the intersection point $t_{12}$ is given in Eqs. (4.6)–(4.8):*

$$t_{12} = \left(\frac{c_1+c_2}{2}\right) + \frac{R_1^2 - R_2^2}{\|c_2-c_1\|^2}\left(\frac{c_2-c_1}{2}\right) \tag{4.6}$$

$$\|t_{12} - c_1\| = \frac{1}{2}\left(\|c_2-c_1\| + \frac{R_1^2-R_2^2}{\|c_2-c_1\|}\right) \tag{4.7}$$

$$\|t_{12} - c_2\| = \frac{1}{2}\left(\|c_2-c_1\| + \frac{R_2^2-R_1^2}{\|c_2-c_1\|}\right) \tag{4.8}$$

*Proof.* We denote with quotes the transposed vectors, e.g., $t'_{12}$ is the transposed of $t_{12}$, and $t'_{12}t_{12} = \|t_{12}\|^2$. The intersection of the lens with its radical hyperplane defines a $(d-1)$-sphere (i.e., a disk when $d = 3$), of radius $L_{12}$ to be calculated further. We define $y_{12}$ as being any point on the boundary of this $(d-1)$-sphere in the radical hyperplane. Considering the right triangles $c_1, t_{12}, y_{12}$ and $c_2, t_{12}, y_{12}$, we have $L_{12}^2 = R_1^2 - (t_{12}-c_1)'(t_{12}-c_1) = R_2^2 - (t_{12}-c_2)'(t_{12}-c_2)$. We express $t_{12}$ with its barycentric coordinates relative to $c_1$ and $c_2$: $t_{12} = \alpha_1 c_1 + \alpha_2 c_2$, $\alpha_1 + \alpha_2 = 1$. Solving for the unknown quantity $\alpha_1$ and after elimination of the term $t'_{12} t_{12}$, we get $\alpha_1 = \frac{1}{2} + \frac{R_2^2 - R_1^2}{2\|c_2 - c_1\|^2}$ and then $\alpha_2 = \frac{1}{2} + \frac{R_1^2 - R_2^2}{2\|c_2 - c_1\|^2}$.

Moreover, $\theta_1$ and $\theta_2$ being the angles respectively associated to each spherical cap (see Fig. 4.1), the barycentric coefficients of $t_{12}$ are the respective cosine of these angles:

$$\alpha_1 = \cos\theta_1 = \frac{1}{2} + \frac{R_2^2 - R_1^2}{2\|c_2 - c_1\|^2} \tag{4.9}$$

$$\alpha_2 = \cos\theta_2 = \frac{1}{2} + \frac{R_1^2 - R_2^2}{2\|c_2 - c_1\|^2} \tag{4.10}$$

Then the radius $L_{12}$ of the $(d-1)$-sphere bounding the lens is:

$$4L_{12}^2 = 2(R_1^2 + R_2^2) - \left(\frac{R_1^2 - R_2^2}{\|c_2 - c_1\|}\right)^2 - \|c_2 - c_1\|^2 \tag{4.11}$$

The surface of the triangle defined by $c_1, c_2, y_{12}$ is $L_{12}\|c_2 - c_1\|/2$. Then we express this surface from $R_1$, $R_2$, and $\|c_2 - c_1\|$ with the Heron formula [53]: the expression of $L_{12}$ above comes after expansion and squaring.

There is a major difference about the validity of Eqs. (4.6)–(4.8) and (4.11): the latter is valid for nonconcentric spheres if and only if the inequalities (4.5) stand, although the former stand if and only if $c_1 \neq c_2$. Thus, the radical plane exists for any radius values, even null ones, discarding whether or not the intersection of the two spheres is empty and discarding if one sphere is included in the other one. In Eqs. (4.9)–(4.10), $\alpha_1$ and $\alpha_2$ are defined for any pair of nonconcentric spheres, although the existence of $\theta_1$ and $\theta_2$ need that inequalities (4.5) stand. When $R_1 = R_2$, the radical hyperplane is just the $(d-1)$-hyperplane mediator of the segment $c_2 - c_1$.

Then, the volume $V$ and the surface $S$ of the union of the two spheres are respectively expressed from the volume and the surface of their intersection:

$$V = V_1 + V_2 - V_{12}, \tag{4.12}$$

$$S = S_1 + S_2 - S_{12}. \tag{4.13}$$

$V_{12}$ and $S_{12}$ can be respectively calculated from the volume and the surface of the two spherical caps bounding the lens. Knowing $t_{12}$ from Eqs. (4.6), the calculation is done as indicated at the end of Sect. 4.2.1 with the angles $\theta_1$ and $\theta_2$ respectively associated to each spherical cap (see Fig. 4.1). These angles are known from Eqs. (4.7) and (4.8): $|\cos\theta_1| = \|t_{12} - c_1\|/R_1$ and $|\cos\theta_2| = \|t_{12} - c_2\|/R_2$. In the case $d = 3$, we simply use Eqs. (4.3) and (4.4). The values $\cos\theta_1$ and $\cos\theta_2$ are taken from Eqs. (4.9) and (4.10). Remark: they can be negative.

### 4.2.3 More About Radical Hyperplanes

We consider $n$ spheres in $E^d$, not two of them being concentric, so that we have defined $n(n-1)/2$ radical hyperplanes.

The intersection of the radical hyperplanes orthogonal to $c_1 - c_2$ and $c_2 - c_3$ is a $(d-2)$-flat, for which any point $z$ satisfies to the two orthogonality conditions $(z-t_{12})'(c_1-c_2) = 0$ and $(z-t_{23})'(c_2-c_3) = 0$, where $t_{12}$ and $t_{23}$ are known from Eq. (4.6). By expanding these two equations and by adding them, and by expressing $t_{13}$ from Eq. (4.6), we get $(z-t_{13})'(c_1-c_3) = 0$. That proves Lemma 4.1.

**Lemma 4.1.** *Assuming that $c_1, c_2, c_3$ are not aligned, the common intersection of the three radical hyperplanes defined by $c_1, c_2, c_3$ exists and is a unique $(d-2)$-flat.*

When the three radii are equal, we retrieve for $d = 2$ that the three perpendicular bisectors of the sides of a triangle intersect at a common single point.

More generally, we look for the existence of the points $z$ at the intersection of the $n(n-1)/2$ radical planes defined by $c_i - c_j$, $1 \leq i < j \leq n$. Applying repeatedly Lemma 4.1 to all triplets of centers, it appears that the set of the points $z$ is the intersection of the $n-1$ radical planes defined by $c_1 - c_i$, $i = 2, .., n$. If existing, this intersection is a unique $(d+1-n)$-flat. This flat is orthogonal to the $(n-1)$-flat containing $c_1, c_2, \ldots, c_n$; thus it can be located from the projection $z_0$ of any of its points $z$ on the $(n-1)$-flat containing $c_1, c_2, \ldots, c_n$. Let $\gamma_j$, $j = 1, .., n$, be the barycentric coordinates of $z_0$ related to $c_1, \ldots, c_n$. We get the following linear system of order $n$:

$$\begin{pmatrix} 1 & \ldots & 1 \\ c_1'(c_1-c_2) & \ldots & c_n'(c_1-c_2) \\ \vdots & \vdots & \vdots \\ c_1'(c_1-c_n) & \ldots & c_n'(c_1-c_n) \end{pmatrix} \cdot \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_n \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 \\ c_1'c_1 - c_2'c_2 + R_1^2 - R_2^2 \\ \vdots \\ c_1'c_1 - c_n'c_n + R_1^2 - R_n^2 \end{pmatrix}. \quad (4.14)$$

The system (4.14) receives a unique solution when the determinant $\Delta$ of the matrix above is not zero. The value of $\Delta$ is obtained by substracting columns $2, .., n$ from column 1 and by developing from the first column:

$$\Delta = (-1)^n \cdot det \begin{pmatrix} (c_2-c_1)'(c_2-c_1) & \ldots & (c_n-c_1)'(c_2-c_1) \\ \vdots & \vdots & \vdots \\ (c_2-c_1)'(c_n-c_1) & \ldots & (c_n-c_1)'(c_n-c_1) \end{pmatrix}.$$

Denoting by $\Delta_{1,2,\ldots,n}$ the determinant of the simplex $c_2 - c_1, \ldots, c_n - c_1$, we get:

$$\Delta = (-1)^n \Delta_{1,2,\ldots,n}^2. \quad (4.15)$$

It means that $z_0$ and the intersection we are looking for exists if and only if the simplex $c_1, \ldots, c_n$ is not degenerated in the subspace of dimension $n-1$ defined by

$c_1,\dots,c_n$ (in which case there are neither two concentric spheres nor three aligned centers). The conclusion is still valid if we consider the $n-1$ radical planes defined by $c_k - c_i$, $i = 1,..,n$, $i \neq k$, $2 \leq k \leq n$, and we get Theorem 4.3.

**Theorem 4.3.** *Let n spheres in $E^d$, $2 \leq n \leq d+1$, their n centers $c_1,\dots,c_n$ being the vertices of a nondegenerated simplex in the $(n-1)$-flat defined by these n centers. The intersection of the resulting $n(n-1)/2$ radical planes is a $(d+1-n)$-flat orthogonal to the $(n-1)$-flat containing the n centers, and its orthogonal projection $z_0$ on this $(n-1)$-flat satisfies to Eqs. (4.14) and (4.15).*

The result given above was reported in [37]. When $n = d+1$, the $d(d+1)/2$ radical planes all intersect at a common single point $z_0$. When all radii are equal, we retrieve for $d = 3$ that the six planes mediators of the edges of a tetrahedron intersect at a common single point.

The case of $n > d+1$ spheres has interesting connections with Voronoi diagrams [10, 11], but Voronoi diagrams are out of scope of this chapter, because they do not lead to practical surfaces and volumes computations.

### 4.2.4 Intersections of Order 3

We set $n = 3$. When two spheres are tangent, either one is included in the other one or their intersection reduces to one point. For surfaces and volumes calculations, this latter case can be viewed as if the intersection is empty. Thus, in both cases, we can neglect the existence of tangent spheres. The enumeration of the possible topological configurations is done in the plane containing the three centers: we have to enumerate the configurations encountered from the intersections of the three great circles of the spheres.

When the intersection of two spheres is a lens, their great circles intersect at two contact points. Otherwise there is no contact point. For three spheres, there are either 0, or 2, or 4, or 6 contact points (we neglect the cases of contact points with multiplicities greater than 1). We enumerated in Figs. 4.2–4.5 the 14 possible configurations for three spheres.

Not all configurations are relevant in chemistry and for trilateration (see Sect. 4.1), but all must be considered by the programmer willing to build a software computing $V$ and $S$ in the general case. Exception made for the configuration in Fig. 4.5d, $V_{123}$ and $S_{123}$ are trivial to calculate, and so are $V$ and $S$ from Eqs. (4.16) and (4.17), obtained by iterating Eqs. (4.12) and (4.13):

$$V = V_1 + V_2 + V_3 - V_{12} - V_{13} - V_{23} + V_{123}, \tag{4.16}$$

$$S = S_1 + S_2 + S_3 - S_{12} - S_{13} - S_{23} + S_{123}. \tag{4.17}$$

Now we set $d = 3$ and we consider the case of Fig. 4.5d. The convex domain at the intersection of the three spheres is symmetric through the plane containing $c_1, c_2, c_3$, and the three arcs on the boundary of this domain intersect at two points $z_+$

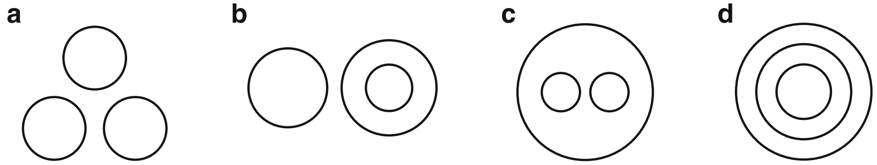


**Fig. 4.2** The four configurations in the case there is no contact point

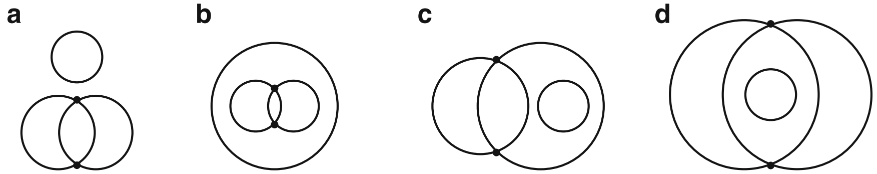
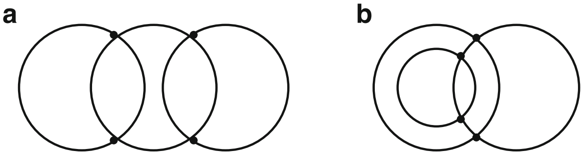


**Fig. 4.3** The four configurations in the case there are two contact points

**Fig. 4.4** The two configurations in the case there are four contact points



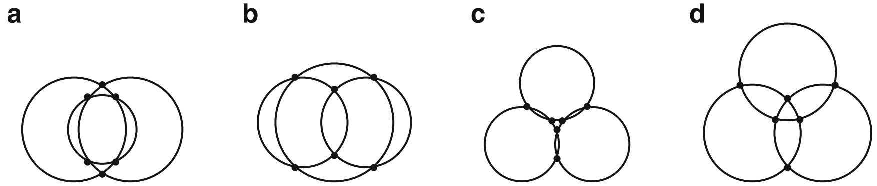


**Fig. 4.5** The four configurations in the case there are six contact points

and $z_-$ which both lie at the intersection of the three radical planes. The barycentric coordinates $\gamma_i$, $i \in \{1,2,3\}$, of $z_0 = (z_+ + z_-)/2$ are calculated in Eq. (4.18) from Eqs. (4.14) and (4.15), where $T_{123}$ is the surface of the triangle $c_1, c_2, c_3$ and $i, j, k$ are circular permutations of $1,2,3$. We get also the length of the segment $z^+ - z^-$ in Eq. (4.19), and then we deduce Theorem 4.4:

$$\begin{aligned}16T_{123}^2\gamma_i = & -2R_i^2\|c_j - c_k\|^2 \\ & + R_j^2(\|c_j - c_k\|^2 + \|c_k - c_i\|^2 - \|c_i - c_j\|^2) \\ & + R_k^2(\|c_j - c_k\|^2 - \|c_k - c_i\|^2 + \|c_i - c_j\|^2) \\ & + \|c_j - c_k\|^2(-\|c_j - c_k\|^2 + \|c_k - c_i\|^2 + \|c_i - c_j\|^2), \quad (4.18)\end{aligned}$$

$$\|\frac{z_+ - z_-}{2}\|^2 = \frac{R_1^2+R_2^2+R_3^2}{3} - \|z_0 - \frac{c_1+c_2+c_3}{3}\|^2 \\ -\frac{\|c_2-c_1\|^2+\|c_3-c_2\|^2+\|c_1-c_3\|^2}{9}. \qquad (4.19)$$

**Theorem 4.4.** *The convex domain defined by the configuration of Fig. 4.5d exists if and only if the quantity at the right member of Eq. (4.19) is positive.*

In the case of Fig. 4.5d, the point $z_0 = (z_+ + z_-)/2$ is interior to the intersection of the three spheres because this intersection is convex. Then, $z_0$, $z_+$, $z_-$ and the three contact points on the boundary of the sphere intersection in the plane $c_1, c_2, c_3$ define a partition of the intersection into six parts, each one being the intersection between one sphere and one trihedron. Each of the six trihedra originates in $z_0$ and is bounded by two radical planes and by the plane $c_1, c_2, c_3$. We denote by $x_{ij}^{(k)}$ each of the contact points at the intersection of the great circles of spheres $i$ and $j$ located in the interior of the sphere $k$, where $i, j, k$ are circular permutations of $1,2,3$. Thus the trihedra are defined by the half lines sets:

$$\left\{z_0 \to x_{31}^{(2)}, z_0 \to x_{12}^{(3)}, z_0 \to z_+\right\}$$

$$\left\{z_0 \to x_{12}^{(3)}, z_0 \to x_{23}^{(1)}, z_0 \to z_+\right\}$$

$$\left\{z_0 \to x_{23}^{(1)}, z_0 \to x_{31}^{(2)}, z_0 \to z_+\right\}$$

and their mirror symmetric images through the plane $c_1, c_2, c_3$:

$$\left\{z_0 \to x_{31}^{(2)}, z_0 \to x_{12}^{(3)}, z_0 \to z_-\right\}$$

$$\left\{z_0 \to x_{12}^{(3)}, z_0 \to x_{23}^{(1)}, z_0 \to z_-\right\}$$

$$\left\{z_0 \to x_{23}^{(1)}, z_0 \to x_{31}^{(2)}, z_0 \to z_-\right\}.$$

The six above trihedra intersect respectively with the sphere 1, 2, 3, 1, 2, 3.

It follows that the calculation of $V_{123}$ and $S_{123}$ can be done through the calculation of the portion of the area of a sphere defined by its intersection with a trihedron having its origin in the interior of the sphere. This latter calculation will be shown to be required in the case of four intersecting spheres and is done analytically in Sect. 4.2.6 using the Gauss–Bonnet theorem.

Several analytical treatments of up to three intersecting spheres appeared in the literature. Some do not involve the Gauss–Bonnet theorem [17, 18, 49], but most involve it [2, 7, 8, 24, 36, 47]. These latter are mainly based on [7] for the surfaces and on [8] for the volumes. As shown in Sect. 4.2.7, neglecting the intersections of order four and higher is numerically not acceptable for van der Waals surfaces and volumes calculation.

### 4.2.5 Intersections of Order 4

We set $n = 4$ and we iterate again Eqs. (4.12) and (4.13):

$$V = V_1 + V_2 + V_3 + V_4 - V_{12} - V_{13} - V_{14} - V_{23} - V_{24} - V_{34}$$
$$+ V_{123} + V_{124} + V_{134} + V_{234} - V_{1234}, \tag{4.20}$$
$$S = S_1 + S_2 + S_3 + S_4 - S_{12} - S_{13} - S_{14} - S_{23} - S_{24} - S_{34}$$
$$+ S_{123} + S_{124} + S_{134} + S_{234} - S_{1234}. \tag{4.21}$$

We set $d = 3$. For clarity, we do not enumerate all topological configurations generated by four intersecting spheres. If any of the four triplets of spheres is not in the configuration of Fig. 4.5d, we know how to calculate $V_{1234}$ and $S_{1234}$.

Assuming that the four sphere triplets are in the configuration of Fig. 4.5d, we can classify the topological configurations via locating the four pairs of two contact points $z_+^{(ijk)}$ and $z_-^{(ijk)}$ at the intersections of the boundaries of the three spheres $i, j, k$, $1 \le i < j < k \le 4$. Each of the two contact points defined by a triplet of spheres can be inside or outside the fourth sphere (we still neglect tangencies and multiplicities). We consider the possible locations of the six remaining contact points. When all of them are inside a sphere, one sphere is included in the union of the three other ones. When all the six remaining contact points are outside a sphere, either the intersection of three spheres is included in the remaining one or the 4-order intersection is empty. This case of empty intersection was called an empty simplicial topology (EST) in [37]. If it happens for some triplet $i, j, k$ that $z_+^{(ijk)}$ is inside the fourth sphere and $z_-^{(ijk)}$ is outside, or conversely, then that happens for all the other triplets: there are three contact points inside a sphere and three outside (this is the general case of intersection of four spheres). The two remaining cases correspond in fact to the same configuration: the intersection of two spheres is included in the union of the two other ones.

To summarize, when the four sphere triplets are in the configuration of Fig. 4.5d, the possible configurations for a non-empty 4-order intersection are:

1. One sphere is included in the union of the three other ones.
2. The intersection of two spheres is included in the union of the two other ones.
3. The intersection of three spheres is included in the remaining one.
4. None of the above ones: general case.

Calculating the 4-order intersection reduces to calculate the 3-order intersections in the cases of configurations 1,2,3. The total number of pairs of interior contact points for these latter configurations are respectively 3,2,1, and the total number of pairs of exterior contact points are respectively 1,2,3. Only the EST configuration has 4 exterior contact point pairs.

In the general case, the domain of the 4-order intersection is bounded by four spherical triangles separated by six arcs of circle and intersecting at four contact points: $z^{(123)}$, $z^{(124)}$, $z^{(134)}$, $z^{(234)}$. This convex domain is topologically similar to a tetrahedron. From Theorem 4.3, there is a unique point $z_0$ at the intersection of the 6 radical planes. Its barycentric coordinates are computed from Eq. (4.14) and (4.15). The point $z_0$ is interior to the convex domain of the 4-order intersection. This latter is partitioned into four parts, each one being the intersection of one sphere and one trihedron originating in $z_0$.

These trihedra, which intersect respectively the spheres 1, 2, 3, 4, are:

$$\{z_0 \rightarrow x^{(123)}, z_0 \rightarrow x^{(124)}, z_0 \rightarrow z^{(134)}\},$$
$$\{z_0 \rightarrow x^{(123)}, z_0 \rightarrow x^{(124)}, z_0 \rightarrow z^{(234)}\},$$
$$\{z_0 \rightarrow x^{(123)}, z_0 \rightarrow x^{(134)}, z_0 \rightarrow z^{(234)}\},$$
$$\{z_0 \rightarrow x^{(124)}, z_0 \rightarrow x^{(134)}, z_0 \rightarrow z^{(234)}\}.$$

It follows that the calculation of $V_{1234}$ and $S_{1234}$ can be done through the calculation of the portion of the area of a sphere defined by its intersection with a trihedron having its origin in the interior of the sphere. This latter calculation is done analytically in Sect. 4.2.6 using the Gauss–Bonnet theorem. An analytical treatment of the four-sphere intersection without invoking the Gauss–Bonnet theorem was reported [27].

### 4.2.6 Intersection of a Sphere with a Dihedron or with Trihedron

We consider a sphere of center $c$ and radius $R$ intersecting a salient trihedron of origin $z$ inside the sphere. The trihedron intersects the boundary of the sphere at $x_1$, $x_2$, $x_3$, these three points being ordered in the direct sense when referred to $z$ as origin. We look for the calculation of the volume $V_{st}$ of the intersection and for the surface $S_{st}$ of the spherical triangle on the boundary of the sphere. This spherical triangle is bounded by the oriented arcs of circle $(x_1,x_2)$, $(x_2,x_3)$, $(x_3,x_1)$. These arcs have respective radii $h_{12}$, $h_{23}$, $h_{31}$ and respective angles $b_{12}$, $b_{23}$, $b_{31}$, each of these angles being in $[0;\pi]$. The trihedron intersects each of the planes respectively tangent to the sphere at $x_1$, $x_2$, $x_3$, thus defining the respective angles $a_1$, $a_2$, $a_3$, each of these angles being in $[0;\pi]$. We define $c_{12}$, $c_{23}$, $c_{31}$ as the projections of $c$ on the planes containing the respective triplets $(x_1,z,x_2)$, $(x_2,z,x_3)$, $(x_3,z,x_1)$. We also define the angles associated to the three lens: $\theta_{ij} = \widehat{x_i,c,c_{ij}} = \widehat{x_j,c,c_{ij}}$, $(i,j) = (1,2)$ or $(i,j) = (2,3)$ or $(i,j) = (3,1)$, such that $\cos\theta_{ij}$ is positive when the vector $c_{ij} - c$ of origin $c$ has the same sense than the direct normal to the plane of the oriented arc $(x_i,x_j)$, and $\cos\theta_{ij}$ is negative when $c_{ij} - c$ has a sense opposite to this normal. We have also $\sin\theta_{ij} = h_{ij}/R$. The sphere has a constant Gaussian curvature and each arc of circle has a constant geodesic curvature. Thus, the Gauss–Bonnet theorem [26, 52] leads to a simple expression of $S_{st}$ in Eq. (4.22), in which the first term in parenthesis is the spherical excess:

$$\frac{S_{st}}{R^2} = (a_1 + a_2 + a_3 - \pi) - (b_{12}\cos\theta_{12} + b_{23}\cos\theta_{23} + b_{31}\cos\theta_{31}). \qquad (4.22)$$

$V_{zx}$ being the signed volume of the oriented tetrahedron $(x_1-z, x_2-z, x_3-z)$ and $V_{cx}$ being the signed volume of the oriented tetrahedron $(x_1-c, x_2-c, x_3-c)$, the volume $V_{st}$ is given below [37]:

$$\begin{aligned} V_{st} = \frac{R^3}{6} \big[&\cos\theta_{12}\sin^2\theta_{12}(\sin b_{12}-b_{12}) \\ &+\cos\theta_{23}\sin^2\theta_{23}(\sin b_{23}-b_{23}) \\ &+\ \cos\theta_{31}\sin^2\theta_{31}(\sin b_{31}-b_{31})\big] + V_{zx} - V_{cx} + \frac{S_{st}R}{3}. \end{aligned} \tag{4.23}$$

When $z$ lies at $c$, we retrieve the expression given by Girard's theorem [48]: $\frac{S_{st}}{R^2}$ is the spherical excess and $V_{st} = \dfrac{S_{st}R}{3}$.

Remark: rather than looking for the validity of Eq. (4.23) from integrating $S_{st}$ as stated in [37], we give a hint of proof as follows. We define the point $x_0$ at the intersection of the half line of origin $c$ passing through $z$ with the boundary of the sphere, and we add the three signed contributions to $V_{st}$ due to the trihedra $\{z \to x_1, z \to x_2, z \to x_0\}$ and $\{z \to x_2, z \to x_3, z \to x_0\}$ and $\{z \to x_3, z \to x_1, z \to x_0\}$, a contribution being negative when the bounding arcs are not oriented in the conventional sense. For each trihedra, say, $\{z \to x_1, z \to x_2, z \to x_0\}$, the volume of the intersection with the sphere is $V_{sc012} - V_{cz12} - V_{c12}$, in which $V_{sc012}$ is the volume of the domain bounded by $c - x_0$, $c - x_1$, $c - x_2$ and by the spherical triangle $(x_0, x_1, x_2)$ intercepted by the trihedron $\{z \to x_1, z \to x_2, z \to x_0\}$ and of area computable by Eq (4.22). Despite that this domain is not bounded by a trihedron, $V_{sc012}$ is computable by trivial integration in respect to the radius of the sphere. $V_{cz12}$ is the volume of the tetrahedron $(c, z, x_1, x_2)$. $V_{c12}$ is the volume of the portion of the cone of origin $c$ with height $R\cos\theta_{12}$ and of basis the circular segment of area $(b_{12} - \sin b_{12})(R\sin\theta_{12})^2/2$, i.e., $V_{c12} = \cos\theta_{12}\sin^2\theta_{12}(b_{12} - \sin b_{12})R^3/6$. The rest of the calculation is easy.

The case of the six trihedra listed at the end of Sect. 4.2.4 is of interest. It is such that one of the arcs, say, $(x_2, x_3)$, lies on a great circle of the sphere: $z$, $x_2$, $x_3$, $c$, $c_{12}$, $c_{31}$, are coplanar, and $c_{23}$ coincides with $c$, then $a_2 = \pi/2$, $a_3 = \pi/2$, and $\theta_{23} = \pi/2$.

Grouping each trihedra with its symmetric image through the plane of the great circle let us to the expressions of the volume $V_{sd}$ of the intersection of a sphere and a dihedron and for the surface $S_{sd}$ of the associated portion of spherical area. Keeping the notations used in Eqs. (4.22) and (4.23):

$$\frac{S_{sd}}{R^2} = 2a_1 - 2(b_{12}\cos\theta_{12} + b_{31}\cos\theta_{31}), \tag{4.24}$$

$$\begin{aligned} V_{sd} = \frac{R^3}{3} \big[&\cos\theta_{12}\sin^2\theta_{12}(\sin b_{12}-b_{12}) \\ &+\ \cos\theta_{31}\sin^2\theta_{31}(\sin b_{31}-b_{31})\ \big] + 2(V_{zx} - V_{cx}) + \frac{S_{sd}R}{3}\Bigg]. \end{aligned} \tag{4.25}$$

Care: in Eqs. (4.24) and (4.25), $b_{12}$ and $b_{31}$ are the angles of the half arcs (not the full arcs) bounding the intersection of the sphere with the dihedron. It is also recalled that the arcs must be correctly oriented and that $\cos\theta_{12}$ and $\cos\theta_{31}$ can be negative.

When $z$ is outside the sphere, the intersection with a dihedron reduces to simpler cases such as a sphere minus two lens, and the intersection with a trihedron can be reduced to intersections with appropriate dihedra or trihedra for which Eqs. (4.22)–(4.25) can be used. More generally we can extend volumes and surfaces computations to the intersections of a sphere with a tetrahedron, a convex polyhedron, and with any union of convex polyhedra. These computations are based on Eqs. (4.22)–(4.25) but are not needed in the framework of this chapter.

### 4.2.7 Intersections of Order 5 and Higher

By iterating Eqs. (4.12) and (4.13) until order $n$, the volume and the surface of the union of the $n$ spheres can be expressed from the inclusion–exclusion principle as follows:

$$V = \sum_{1\le i\le n} V_i - \sum_{1\le i_1<i_2\le n} V_{i_1 i_2} + \sum_{1\le i_1<i_2<i_3\le n} V_{i_1 i_2 i_3} + \cdots + (-1)^{n-1} V_{i_1 i_2 \cdots i_n}, \quad (4.26)$$

$$S = \sum_{1\le i\le n} S_i - \sum_{1\le i_1<i_2\le n} S_{i_1 i_2} + \sum_{1\le i_1<i_2<i_3\le n} S_{i_1 i_2 i_3} + \cdots + (-1)^{n-1} S_{i_1 i_2 \cdots i_n}. \quad (4.27)$$

Setting $d = 3$ and $n \ge 5$, the existence of intersections of order $n$ is deduced from Helly's theorem [14]. It means that the intersection of five spheres is not empty if and only if each of the five subsets of four spheres gives rise to a non-empty intersection. If $n \ge 6$, the intersection of six spheres is not empty if and only if each of the six subsets of five spheres gives rise to a nonempty intersection.

For practical computations, we need Theorem 4.5:

**Theorem 4.5.** *When $n \ge 5$ spheres in $E^3$ have a common non-empty intersection, there are m (with $1 \le m \le 3$) of these spheres such that their union J contains the intersection I of the $n - m$ remaining spheres.*

*Proof.* Assume initially that $n = 5$ spheres have a common non-empty intersection. From Helly's theorem, each of the five subsets of four spheres gives rise to a nonempty intersection. If one of these subsets of four spheres is such that the union of $m = 1, 2$, or three spheres contains the intersection of the $4 - m$ remaining spheres, the union of these $m$ spheres contains the intersection of the $5 - m$ remaining spheres and the theorem stands. As stated in Sect. 4.2.5, a set of four spheres giving rise to a nonempty intersection which is not in one of these latter configurations is in the general configuration.

Now we need to prove the theorem when the five subsets of four spheres are in the case of a general configuration. For this latter, the four contact point pairs between three spheres are such that one of these contact points lies inside the fourth sphere

**Table 4.1** Distribution of the volume and of the errors on the volume

| | Mean | Std. dev. | Min | Max | Median |
|---|---|---|---|---|---|
| Volume: exact value $V$ in Å$^3$ | 336.181 | 165.917 | 115.518 | 1162.450 | 386.702 |
| Error $\hat{V}^{(3)} - V$, from eq. (4.16) | 35.022 | 15.329 | 6.389 | 88.783 | 33.581 |
| Error $\hat{V}^{(4)} - V$, from eq. (4.20) | −6.173 | 2.745 | −12.768 | −0.498 | −5.180 |
| $100 \cdot (\hat{V}^{(3)} - V)/V$ | 10.655 | 2.973 | 3.154 | 19.107 | 8.889 |
| $100 \cdot (\hat{V}^{(4)} - V)/V$ | −1.987 | 0.838 | −5.283 | −0.161 | −1.427 |

and the other contact point lies outside this fourth sphere. Thus, enumerating the spatial arrangements of the five spheres leads to only two possible configurations:

1. One sphere contains the intersection of the four other ones, and simultaneously the union of these four spheres contains the first one. The common 5-order intersection is reduced to a 4-order intersection of the four spheres in the general configuration, which is topologically related to a tetrahedron.
2. The union of two spheres contains the intersection of the three other ones, and simultaneously the union of these three spheres contains the intersection of the two first ones. The common 5-order intersection is a spherical polyhedron topologically related to a triangular prism, i.e., bounded by two triangles and three tetragons, nine arcs, and six vertices.

Obviously the theorem stands for both configurations and thus it stands always for $n = 5$. Then it stands for $n > 5$ because the intersection of the $n - m$ spheres is itself included in the intersection of the $5 - m$ spheres. □

Theorem 4.5 is known as the *three spheres theorem* [37]. A more general expression of this theorem is given in the appendix.

Then, applying the inclusion-exclusion principle to both members of the equality $I \cup J = J$ provides a relation between the $n$-order intersection and the $(n-1)$-order intersections. Starting from 4-order intersections surfaces and volumes, we can compute 5-order and then higher-order intersection surfaces and volumes.

Some authors attempted to fully describe spheres intersections [9, 16, 20, 37] or to produce softwares dealing with more than $n = 3$ intersecting spheres [21, 40]. Since many authors did an analytical treatment with the assumption that no 4-order intersections exist (see Sect. 4.2.4), we analyzed a set of 70 molecules offering a wide structural diversity [28] with the ASV freeware [37, 40]. The number of atoms ranged from 16 to 186 (median: 45). The atomic radii (in Å) were taken from [15]: H = 1.17, C = 1.75, N = 1.55, O = 1.40, F = 1.30, I = 2.10, and from [23]: P = 1.75, S = 2.55.

The results for volumes are in Table 4.1 and those for surfaces are in Table 4.2.

All molecules gave raise to 5-order intersections, 69 to 6-order intersections,, 13 to 7-order intersections, and 2 to 8-order intersections. Owing to the results in Tables 4.1 and 4.2, truncating the calculation after the 3-order intersections is inacceptable, both for volumes and surfaces computations: the values are strongly overestimated. Correcting the overestimated volumes via the use of smaller radii is

**Table 4.2** Distribution of the surface and of the errors on the surface

| | Mean | Std. dev. | Min | Max | Median |
|---|---|---|---|---|---|
| Surface: exact value $S$ in Å$^2$ | 388.162 | 185.526 | 147.386 | 1324.635 | 461.750 |
| Error $\hat{S}^{(3)} - S$, from eq. (4.17) | 337.120 | 170.906 | 71.409 | 869.994 | 362.754 |
| Error $\hat{S}^{(4)} - S$, from eq. (4.21) | $-59.374$ | 27.288 | $-130.446$ | $-6.716$ | $-53.672$ |
| $100 \cdot (\hat{S}^{(3)} - S)/S$ | 86.511 | 25.691 | 35.609 | 139.625 | 79.243 |
| $100 \cdot (\hat{S}^{(4)} - S)/S$ | $-16.176$ | 7.064 | $-37.001$ | $-3.407$ | $-12.130$ |

of course inappropriate, and lowering the radii even does not guarantee a decrease of the surfaces values. Truncating the calculation after the 4-order intersections gives underestimated values which are still not acceptable for surfaces, although it gives a rough approximation of the volumes which could be tolerated in some contexts.

Another set was analyzed [37], containing 63 molecules: even though the hydrogens were discarded, similar conclusions were derived. It is amazing to see the success encountered during the last three decades by the analytical algorithms neglecting the existence of intersections between more than three spheres (see references cited at the end of Sect. 4.2.4). Unless working with sufficiently small atomic radii, the resulting software tools should return strongly erroneous results, including those which compute derivatives [24, 36, 47]

## 4.3 Numerical Methods

The full analytical calculation of spheres unions volumes and surfaces can be used to compute numerical approximations of derivatives, which in turn are useful in some optimization problems [38]. However, for many QSAR (quantitative structure activity relationships) applications involving the molecular surface or volume as an input variable for regression, an approximate value based on a numerical calculus may be acceptable. Several numerical methods were developed [3, 4, 25, 32, 34, 35, 54]. The brute approach is based on a regular grid, i.e., a mesh of $N$ nodes defining cubes to be counted in order to estimate the volume of the surface. Despite that more or less sophisticated possible variants of this approach are possible, these are $O(N^3)$ processing time algorithms. Thus, Monte-Carlo methods should be preferred (see further).

## 4.4 Monte Carlo Methods

These methods are highly attractive due to their great simplicity. In general, they are used in awkward situations where analytical methods are unavailable and where numerical methods are inefficient, e.g., due to a multiple integral over a domain with an untractable boundary calculation.

As shown below, Monte Carlo methods are $O(N^2)$ processing time algorithms, although regular grid methods are generally $O(N^d)$ ones. Thus, for $d > 2$, Monte-Carlo methods should be preferred. Furthermore, they are fully adequate for QSAR applications and easy to program.

### 4.4.1 Monte-Carlo Calculations of Volumes

The Monte Carlo measure of the volume $V$ of a finite domain $\mathscr{D}$ of $E^d$ is performed as follows:

1. Enclose the domain in a $d$-dimensional parallelepipedic window of volume $W$.
2. Build a function returning the status of one point: inside $\mathscr{D}$ or not.
3. Generate a sample of $N$ independent random points in $E^d$, each one following the uniform law in the window.
4. Count the number $N_{\mathscr{D}}$ of these random points which fell inside $\mathscr{D}$.
5. The Monte Carlo estimate of $V$ is $\hat{V} = W \cdot N_{\mathscr{D}}/N$.

We consider the $N$ random variables $X_i = \mathbf{1}_{\{U_i \in \mathscr{D}\}}$, $i = 1,\ldots,N$, where $U_i$ is the $i^{th}$ random vector following the uniform law in the window and $\mathbf{1}_{\{U_i \in \mathscr{D}\}}$ is the indicator function of the event $U_i \in \mathscr{D}$. In other words, $X_i = 1$ when $U_i$ takes a value in $\mathscr{D}$ and $X_i = 0$ elsewhere. Assuming that $U_1,\ldots,U_N$ are independent, then $X_1,\ldots,X_N$ are independent and identically distributed (i.i.d.). The probability for $U_i$ to take a value in $\mathscr{D}$ is $p = V/W$. Setting $q = 1 - p$, $Prob(X_i = 1) = p$ and $Prob(X_i = 0) = q$, i.e., each of the $X_i$ follows the Bernouilli law of parameter $p$. The sum of these $N$ i.i.d. random variables $X_i$ follows a binomial law $B(N, p)$, of expectation $Np$ and variance $Npq$. Their mean is the random variable $\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$, with expectation $p = \frac{V}{W}$ and variance $\frac{pq}{N} = \frac{1}{N}\left(\frac{V}{W}\right)\left(\frac{W-V}{W}\right)$. It follows that the observed mean $\frac{\hat{V}}{W} = \frac{N_{\mathscr{D}}}{N}$ is a consistent and unbiased estimator of $\bar{X}$. Thus, $\hat{V}$ is a consistent and unbiased estimator of $V$, of expectation $V$ and variance $V(W-V)/N$. Furthermore, we know from de Moivre-Laplace theorem [19, 45] that the law of $\frac{N\bar{X}-Np}{\sqrt{Npq}}$ converges to the normal law $\mathscr{N}(0,1)$ of expectation 0 and variance 1.

We notice that the best possible window is the one minimizing $V(W-V)$, so that it should be the smallest possible one containing $\mathscr{D}$. The theory still works with a non-parallelepipedic window, but this latter needs an increase of the computational cost to generate the $U_i$ following the uniform law in the window.

The precision is evaluated through a confidence interval. Several binomial proportion confidence intervals were proposed [1, 6, 44]. Among them, we selected the so-called Wald interval, which is symmetric and based under the normal approximation $[\hat{p} \pm \xi_h box(1-\alpha/2)\sqrt{\frac{\hat{p}(1-\hat{p})}{N}}]$, where $\hat{p} = \hat{V}/W$ and $\xi_{(1-\alpha/2)}$ is the $(1-\alpha/2)$ percentile of the normal law $\mathscr{N}(0,1)$ corresponding to the error $\alpha$ (e.g., for a 95% confidence level, $\alpha = 0.05$, $1-\alpha/2 = 0.975$ and $\xi_{(1-\alpha/2)} \approx 1.96$). Thus,

**Table 4.3** Monte Carlo measures of the van der Waals volume of the cyclosporine

| $N$ | $N_{\mathscr{D}}$ | $W$ | $\hat{V}$ | $\hat{\sigma}_V$ | $100 \cdot \hat{\sigma}_V / V$ |
|---|---|---|---|---|---|
| 100 | 23 | 5230.333 | 1202.977 | 220.109 | 18.297 |
| 10000 | 2220 | 5230.333 | 1161.134 | 21.737 | 1.872 |
| 1000000 | 222642 | 5230.333 | 1164.492 | 2.176 | 0.187 |

Analytical value: $V = 1162.450152$

having estimated the center of the confidence interval for $\hat{V}$, its length is estimated from the observed standard deviation $\hat{\sigma}_V = W\hat{\sigma}_p$, with $\hat{\sigma}_p = \sqrt{\frac{\hat{p}(1-\hat{p})}{N}}$. The Wald confidence interval is appropriate as long as neither $Np$ nor $Nq$ is too small, which is the case in our context.

Examples of Monte Carlo computations of a molecular volume are given in Table 4.3. The molecule is the cyclosporine. It is the largest of the dataset used at the end of Sect. 4.2.7 and it contains 186 atomic spheres.

Clearly, multiplying by 100 the number of observations led to an increase of the precision by a factor 10. This is in agreement with the proportionality of $\hat{\sigma}_V$ to $1/\sqrt{N}$.

### 4.4.2 Monte Carlo Calculations of Surfaces

The Monte Carlo measure of the surface $S$ of an union of $n$ spheres in $E^d$ is similar to the one for volumes, except that we consider the ratio of the surface of the union of the spheres to the union of their surfaces, i.e., the sum $T = S_1 + \ldots + S_n$ of the individual surfaces of the $n$ spheres plays the role of $W$ for the volumes, and the domain $\mathscr{D}$ is now defined by the surface of the union of the $n$ spheres.

The Monte Carlo measure of $S$ is performed as follows:

1. Compute the total surface $T$.
2. Build a function returning the status of one point: inside a sphere or not.
3. Generate a sample of $N$ independent random points, each one following the uniform law over the union of the $n$ surfaces. It is done as follows:
   - Select one of the $n$ spheres such that each sphere $i$ has a probability $S_i/T$ to be selected, i.e., get a random number $v$ following the uniform law over $[0;T]$ and retain the sphere index $i$ as the smallest one such that $v < S_1 + \ldots + S_i$.
   - Generate a random point of $E^d$ following the uniform distribution on the surface of the sphere $i$. It will fall in $\mathscr{D}$ if it is not interior to any of the $n-1$ other spheres.
4. Count the number $N_{\mathscr{D}}$ of these random points which fell in $\mathscr{D}$.
5. The Monte Carlo estimate of $S$ is $\hat{S} = T \cdot N_{\mathscr{D}}/N$.

The analysis of the algorithm is identical to the one for volumes, with $\hat{\sigma}_S = T\hat{\sigma}_p$, $\hat{\sigma}_p = \sqrt{\frac{\hat{p}(1-\hat{p})}{N}}$ and $\hat{p} = \hat{S}/T$.

**Table 4.4** Monte Carlo measures of the van der Waals surface area of the cyclosporine

| $N$ | $N_{\mathscr{D}}$ | $T$ | $\hat{S}$ | $\hat{\sigma}_S$ | $100 \cdot \hat{\sigma}_S / S$ |
|---|---|---|---|---|---|
| 100 | 20 | 4751.111 | 950.222 | 190.044 | 20.000 |
| 10000 | 2715 | 4751.111 | 1289.927 | 21.130 | 1.638 |
| 1000000 | 277944 | 4751.111 | 1320.543 | 2.128 | 0.161 |

Analytical value: $S = 1324.635145$

There are several methods to generate a random point following the uniform law on the boundary of a sphere. For clarity, we assume that the center of the sphere lies at the origin. The rejection method is the simplest one: generate a point following the uniform law in the smallest cube containing the sphere, and accept the point if it falls inside the sphere; if it falls outside, generate an other one, until it falls inside the sphere. Once done, normalize the corresponding vector to set its length equal to the radius of the sphere. The rejection method is rather inefficient for high $d$ values because the ratio of the volume of the sphere to its smallest enclosing cube tends to zero when $d$ increases to infinity (see Sect. 4.2.1). Nevertheless, it can be retained for $d = 3$. There are other methods, such as normalizing a vector following the isotropic multinormal law, i.e., such that its $d$ components follow $d$ uncorrelated normal laws of null expectation and of identical standard deviations. Observations from the normal law can themselves be generated from various methods, such as Box-Muller or Marsaglia [22].

Examples of Monte Carlo computations of a molecular surface are given in Table 4.4. The molecule is the cyclosporine. It is the largest of the dataset used at the end of Sect. 4.2.7 and it contains 186 atomic spheres.

Again, multiplying by 100 the number of observations led to an increase of the precision by a factor 10, which is in agreement with the proportionality of $\hat{\sigma}_S$ to $1/\sqrt{N}$.

## 4.5 Discussion and Conclusion

Selecting the analytical calculation vs. the Monte Carlo calculation of the volume or surface of a union of spheres depends on two criteria:

1. A high precision is required, e.g., for computing derivatives via finite differences.
2. A small computing time is required, due to the need of numerous repeated calls.

It is clear from Eqs. (4.26)–(4.27) that, when all spheres intersect, the computing time of the analytical calculation grows exponentially with the number of spheres. Practically, we measured computing times with the Linux- 64-Intel version of the freeware ASV [40], which can perform both analytical and Monte Carlo calculations. For the cyclosporine data mentioned in Tables 4.3 and 4.4 and containg 186 atoms, the analytical calculation took 0.23s, although the Monte Carlo calculation took 36s. However, for the 5188 "ATOM" set of the human prostate antigen (PDB code 2ZCH; contains hydrogens) the analytical calculation took 1053s, although the Monte Carlo calculation took only 131s. Thus, for small molecules or small sets of spheres, the analytical calculation is recommended. For large sets of spheres with a huge of intersections, the Monte-Carlo calculation is useful as long as a moderate accuracy suffices. Since it is the case for most molecular modeling applications, the f77 sources of the Monte Carlo calculations in $E^d$, plus a f77 implementation of a pseudo-random generator [22] of period $2^{57}$ based on the congruential method used in [33], are provided with ASV.

Geometry programs are often subject to potential numerical instabilities. It is the case of the calculation of the intersection of the radical planes via Theorem 4.3 which assumes that the centers of the spheres are the vertices of a nondegenerate simplex (see Sect. 4.2.3). Alas, chemistry data offer a huge of triplets of aligned atoms and of quadruplets of coplanar atoms. For example benzene derivatives offer 12 coplanar atoms and three sets of four aligned atoms due to the benzene ring, which is by far the ring the most frequently encountered by chemists [51]. In fact, any $sp^2$ carbon (e.g., a carbon connected with one double bond and two single bonds) gives raise to four coplanar atoms, a very common situation. Fortunately, in most cases, calculating the intersection of the radical planes is not required. Furthermore, due to the presence of a limited number of significant digits in molecular files, most cases of alignment or coplanarity are avoided. In any case, to prevent instabilities, it is possible to let the user perturbate the atomic coordinates with a given magnitude [40].

## Appendix: Segments and Disks

We give here a stronger version of Theorem 4.5.

**Theorem 4.6.** *When $n \geq d+2$ spheres in $E^d$, $d \leq 3$, have a common non-empty intersection, there are m (with $1 \leq m \leq \lfloor \frac{d+1}{2} \rfloor + 1$) of these spheres such that their union contains the intersection of the $n-m$ remaining spheres and simultaneously the union of these $n-m$ spheres, contains the intersection of the first m ones.*

*Proof.* We assume initially that $n = d+2$ spheres have a common non-empty intersection.

We set $d = 1$. A sphere in $E^1$ is a segment. The $n = 3$ segments have a common 3-order intersection. The theorem stands if a segment is included in an other one. If not, it is easy to check that there is only one possible configuration: one segment contains the intersection of the two other ones, and simultaneously the union of these two segments contains the first one.

We set $d = 2$. A sphere in $E^2$ is a disk. The $n = 4$ disks have a common 4-order intersection. The possible configurations for three intersecting disks are enumerated in Figs. 4.2 and 4.5 (Sect. 4.2.4). If any of the four triplets of disks is not in the general case of intersection of Figs. 4.5d, the theorem stands. We assume that the four triplets of disks are in this general case (Fig. 4.5d). We consider the four

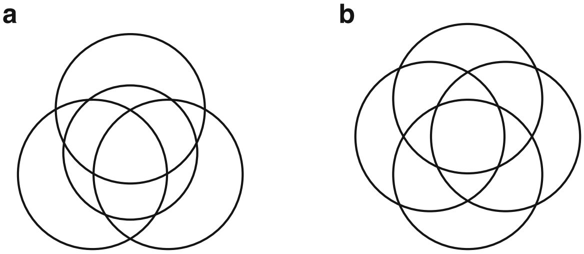


**Fig. 4.6** The two configurations in the case of 4 intersecting disks

bounding circles. There are two contact points at the intersection of each of the six pairs of circles. Enumerating the arrangements of these four circles can be done with the help of their contact points and with the two 2-Dlens at the respective intersections of the disks 1,2 and 3,4. It leads to only two possible configurations:

1. One disk contains the intersection of the three other ones, and simultaneously the union of these three disks contains the first one. The common 4-order intersection is a curvilinear triangle (Fig. 4.6a).
2. The union of two disks contains the intersection of the two other ones, and simultaneously the union of the two latter ones contains the intersection of the two former ones. The common 4-order intersection is a curvilinear tetragon (Fig. 4.6b).

The theorem stands in both cases and thus it stands always for $n = 4$. For $d = 3$, the theorem was proved in Sect. 4.2.7.

For $n = d + 2$ and $d \leq 3$ the theorem stands and we found the $1 \leq m \leq \lfloor \frac{d+1}{2} \rfloor + 1$ required spheres. For $n > d + 2$ and $d \leq 3$, we consider $n - d - 2$ additional spheres. The theorem still stands because (a) the union of the $m$ spheres contains the intersection of the $d + 2 - m$ spheres which in turn contains the intersection of the $(n - d - 2) + (d + 2 - m) = n - m$ spheres, and (b) the union of these $n - m$ spheres contains the union of the $d + 2 - m$ spheres which in turn contains the intersection of the $m$ ones. □

It is conjectured that Theorem 4.6 stands for $d > 3$. Although useful in the case $d = 3$ for spheres unions surfaces and volumes computations, this theorem can also be used in the case $d = 2$ for computing surfaces and exposed arcs lengths of disks unions.

## References

1. Agresti, A., Coull, A.B.: Approximate is better than "exact" for interval estimation of binomial proportions. The American Statistician **52**(2), 119–126 (1998)
2. Alard, P., Wodak, S.J.: Detection of cavities in a set of interpenetrating spheres. J. Comput. Chem. **12**(8), 918–922 (1991)
3. Blyznyuk, A.A., Gready, J.E.: Numerical calculation of molecular surface area. I. Assessment of errors. J. Comput. Chem. **17**(8), 962–969 (1996)

4. Bohacek, R.S., McMartin, C.: Definition and display of steric, hydrophobic, and hydrogen-bonding properties of ligand binding sites in proteins using Lee and Richards accessible surface: Validation of a high-resolution graphical tool for drug design. J. Med. Chem. **35**(10), 1671–1684 (1992)
5. Bondi, A.: Van der Waals volumes and radii. J. Phys. Chem. **68**(3), 441–451 (1964)
6. Brown, L.D., Cai, T.T., DasGupta, A.: Interval estimation for a binomial proportion. Stat. Sci. **16**(2), 101–133 (2001)
7. Connolly, M.L.: Analytical molecular surface calculation. J. Appl. Crystallogr. **16**(5), 548–558 ( 1983)
8. Connolly, M.L.: Computation of molecular volume. J. Am. Chem. Soc. **107**(5), 1118–1124 (1985)
9. Dodd, L.R., Theodorou, D.N.: Analytical treatment of the volume and surface area of molecules formed by an arbitrary collection of unequal spheres intersected by planes. Mol. Phys. **72**(6), 1313–1345 (1991)
10. Edelsbrunner, H.: Algorithms in Combinatorial Geometry, EATCS Monographs on Theoretical Computer Science, vol. 10, Springer, Berlin (1987)
11. Edelsbrunner, H.: The union of balls and its dual shape. Discrete Comput. Geom. **13**(1), 415–440 (1995)
12. Edelsbrunner, H., Facello, M., Liang, J.: On the definition and the construction of pockets in macromolecules. Discrete Appl. Math. **88**(1–3), 83–102 (1998)
13. Edelsbrunner, H., Mücke, E.P.: Three-dimensional alpha shapes, ACM Trans. Graph. **13**(1), 43–72 (1994)
14. Eggleston, H.G.: Convexity. In: Smithies, F., Todd, J.A. (eds.) Cambridge Tracts in Mathematics and Mathematical Physics **47**, chapter 2. Cambridge University Press, Cambridge (1966)
15. Gavezzotti, A.: The calculation of molecular volumes and the use of volume analysis in the investigation of structured media and of solid-state organic reactivity. J. Am. Chem. Soc. **105**(16), 5220–5225 (1983)
16. Gibson, K.D., Scheraga, H.A.: Exact calculation of the volume and surface area of fused hard-spheres molecules with unequal atomic radii. Mol. Phys. **62**(5), 1247–1265 (1987)
17. Gibson, K.D., Scheraga, H.A.: Volume of the intersection of three spheres of unequal size. A simplified analytical formula. J. Phys. Chem. **91**(15), 4121–4122 (1987) Additions and corrections: **91**(24), 6326
18. Gibson, K.D., Scheraga, H.A.: Surface area of the intersection of three spheres with unequal radii. A simplified analytical formula. Mol. Phys. **64**(4), 641–644 (1988)
19. Girardin, V., Limnios, L.: Probabilités en vue des applications: cours et exercices corrigés, Volume 1, chap 4.3. Vuibert, Paris (2008)
20. Gogonea, V., Ōsawa, E.: An improved algorithm for the analytical computation of solvent-excluded volume. The treatment of singularities in solvent-accessible surface area and volume functions. J. Comput. Chem. **16**(7), 817–842 (1995)
21. Guerrero-Ruiz, G., Ocadiz-Ramirez, A., Garduño-Juárez, R.: ESFERA: a program for exact calculation of the volume and surface area of fused hard-sphere molecules with unequal atomic radii. J. Comput. Chem. **15**(4), 351–352 (1991)
22. Hammersley, J.M., Handscomb, D.C.: Monte Carlo Methods. In: Monographs on Statistics and Applied Probability. Chapman and Hall, London (1983)
23. Hopfinger, A.J.: Conformational properties of macromolecules. In: Molecular Biology Series, vol. 17, chap. 2, sect. II. Academic, New York (1973)
24. Kundrot, C.E., Ponder, J.W., Richards, F.M.: Algorithms for calculating excluded volume and its derivatives as a function of molecular conformation and their use in energy minimization. J. Comput. Chem. **12**(3), 402–409 (1991)
25. Lee, B., Richards, F.M.: The interpretation of protein structures: Estimation of static accessibility. J. Mol. Biol. **55**(3), 379–400 (1971)
26. Lelong-Ferrand, J.: Geometrie Differentielle, Chaps. 10.5–10.6. Masson, Paris (1983)
27. Lustig, R.: Geometry of four hard Fused spheres in an arbitrary spatial configuration. J. Comput. Chem. **59**(2), 195–207 (1986)

28. Meslamani, J.E., André, F., Petitjean, M.: Assessing the geometric diversity of cytochrome P450 ligand conformers by hierarchical clustering with a stop criterion. J. Chem. Inform. Model. **49**(2), 330–337 (2009)
29. Meyer, A.Y.: The size of molecules. Chem. Soc. Rev. **15**(4), 449–474 (1986)
30. Meyer, A.Y.: Molecular mechanics and molecular shape. V. On the computation of the bare surface area of molecules. J. Comput. Chem. **9**(1), 18–24 (1988)
31. Meyer, A.Y.: More on the size of molecules. J. Struct. Chem. **1**(2–3), 265–279 (1990)
32. Müller, J.J.: Calculation of scattering curves for macromolecules in solution and comparison with results of methods using effective atomic scattering factors. J. Appl. Crystallogr. **16**(1), 74–82 (1983)
33. NAG Fortran Library Routine Document. //www.nag.co.uk/numeric/fl/manual/xhtml/G05/g05caf.xml.
34. Pavanï, R., Ranghino, G.: A method to compute the volume of a molecule. J. Comput. Chem. **6**(3), 133–135 (1982)
35. Pavlov, M.Y., Fedorov, B.A.: Improved technique for calculating X-ray scattering intensity of biopolymers in solution: Evaluation of the form, volume, and surface of a particle. Biopolymers **22**(6), 1507–1522 (1983)
36. Perrot, G., Cheng , Gibson, K.D., Vila, J., Palmer, K.A., Nayeem, A., Maigret, B., Scheraga, H.A.: MSEED: a program for the rapid analytical determination of accessible surface areas and their derivatives. J. Comput. Chem. **13**(1), 1–11 (1992)
37. Petitjean, M.: On the analytical calculation of van der Waals surfaces and volumes: Some numerical aspects. J. Comput. Chem. **15**(5), 507–523 (1994)
38. Petitjean, M.: Geometric molecular similarity from volume-based distance minimization: Application to saxitoxin and tetrodotoxin. J. Comput. Chem. **16**(1), 80–90 (1995)
39. Petitjean, M.: About the algebraic solutions of smallest enclosing cylinders problems, arXiv:1008.5259 (2010)
40. Petitjean, M.: ASV freeware, http://petitjeanmichel.free.fr/itoweb.petitjean.freeware.html.
41. Petřek, M., Košinová, P., Koča, J., Otyepka, M.: MOLE: a Voronoi diagram-based explorer of molecular channels, pores, and tunnels. Structure **15**(11), 1357–1363 (2007)
42. Petřek, M., Otyepka, M., Banáš, P., Košinová, P. Koča, J., Damborský, D.: CAVER: a new tool to explore routes from protein clefts, pockets and cavities. BMC Bioinformatics **7**(316), 1–9 (2006)
43. Ramachandran, G.N., Sasisekharan, V.: Conformations of polypeptides and proteins. In: Advances in Protein Chemistry, vol. 23, pp. 283–437 (appendix by the Editors p. 438). Academic, New-York (1968)
44. Reiczigel, J.: Confidence intervals for the binomial parameter: some new considerations. Stat. Med. **22**(4), 611–621 (2003)
45. Renyi, A.: Probability Theory. North Holland Publishing Company, Amsterdam ( 1970)
46. Richards, F.M.: Calculation of molecular volumes and areas for structures of known geometry. In: Diffraction Methods for Biological Macromolecules, part B. In: Wyckoff, H.W., Hirs, C.H.W., Timasheff, S.N. (eds.) Methods in Enzymology vol. 115, pp. 440–464. Academic, London (1985)
47. Richmond, T.J.: Solvent accessible surface area and excluded volume in proteins. Analytical equations for overlapping spheres and implications for the hydrophobic effect. J. Mol. Biol. **178**(1), 63–89 (1984)
48. Rosenfeld, B.A.: A history of non-euclidean geometry. Evolution of the concept of a geometric space. Springer, New York (1988)
49. Rowlinson, J.S.: The triplet distribution function in a fluid of hard spheres. J. Mol. Phys. **6**(5), 517–524 (1963)
50. Scott, R.A., Scheraga, H.A.: Conformational analysis of macromolecules. III. Helical structures of polyglycine and poly-L-alanine. J. Chem. Phys. **45**(6), 2091–2101 (1966)
51. Stobaugh, R.E.: Chemical abstracts service chemical registry system. 11. Substance-related statistics: Update and additions. J. Chem. Inform. Comput. Sci. **28**(4), 180–187 (1988)
52. Stoker, J.J.: Differential Geometry. Wiley, New York (1969)

53. Vakil, R.: Chapter: Geometry Revisited. A Mathematical Mosaic. Patterns and Problem Solving, pp. 160. Brendan Kelly Publishing. Burlington, Ontario (1996)
54. Wodak, S.J., Janin, J.: Analytical approximation to the accessible surface area of proteins. Proceedings of the National Academy of Sciences of the USA **77**(4), pp. 1736–1740 (1980)
55. Zefirov, Y.V., Zorkii, P.M.: Van der Waals radii and their application in chemistry Russian Chemical Review **58**(5), 421–440 (1989) Uspekhi Khimii **58**(5), 713–746, 1989 (in Russian)

# Chapter 5
# Is the Distance Geometry Problem in NP?

**Nathanael Beeker, Stéphane Gaubert, Christian Glusa, and Leo Liberti**

**Abstract** Given a weighted undirected graph $G=(V,E,d)$ with $d:E\to\mathbb{Q}_+$ and a positive integer $K$, the distance geometry problem (DGP) asks to find an embedding $x:V\to\mathbb{R}^K$ of $G$ such that for each edge $\{i,j\}$ we have $\|x_i-x_j\|=d_{ij}$. Saxe proved in 1979 that the DGP is **NP**-complete with $K=1$ and doubted the applicability of the Turing machine model to the case with $K>1$, because the certificates for YES instances might involve real numbers. This chapter is an account of an unfortunately failed attempt to prove that the DGP is in **NP** for $K=2$. We hope that our failure will motivate further work on the question.

## 5.1 Introduction

Consider the following decision problem.
DISTANCE GEOMETRY PROBLEM (DGP). Given a weighted undirected graph $G=(V,E,d)$, where $d:E\to\mathbb{F}$, and a positive integer $K$, establish whether there exists an embedding $x:V\to\mathbb{R}^K$ such that

$$\forall\{i,j\}\in E\quad \|x_i-x_j\|=d_{ij}, \tag{5.1}$$

N. Beeker • C. Glusa (✉)
CMAP, École Polytechnique, 91128 Palaiseau, France
e-mail: nathanael.beeker@polytechnique.edu; christian.glusa@polytechnique.edu

S. Gaubert
INRIA Rocquencourt, Cedex, France
e-mail: stephane.gaubert@inria.fr

L. Liberti
LIX, École Polytechnique, 91128 Palaiseau, France
e-mail: liberti@lix.polytechnique.fr

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_5,

where $F$ is a set of nonnegative numbers, which, for the purposes of this chapter, we assume to be either integers $\mathbb{N}$ or rationals $\mathbb{Q}_+$. We denote explicit dependence of the DGP on $K$ by DGP$_K$.

The DGP is **NP**-hard, but even when $\mathbb{F} = \mathbb{N}$ it is not known, whenever $K > 1$, whether it is in **NP** or not. Trying to prove that the DGP is in **NP** would involve finding a polynomial size representation for the solutions of a polynomial system of equations of degree two. Disproving the statement would probably be much more difficult. This chapter relates a possible proof technique for showing that DGP $\in$ **NP** and the corresponding failure, in the hope of enticing new efforts on this topic.

### *5.1.1 Applications*

In the Molecular Distance Geometry Problem (MDGP), $G$ is a molecule graph where the $E$ is the set of known interatomic distances and $K = 3$. Since the function of molecules depends strongly on their spatial configuration, finding an embedding of $V$ in $\mathbb{R}^3$ is of practical interest [11, 13]. A distinguishing property is that because of the experimental techniques involved, most distances are bounded above by 6Å, so that the resulting graph is 3D generalization of a Unit Disk Graph (UDG) [1].

Wireless Sensor Network Localization (WSNL) aims to embed a wireless sensor network in $\mathbb{R}^2$ (so $K = 2$). Pairs of sensors can estimate their distance by measuring the power used for a two-way communication. Since sensor networks always include a wired backbone (allowing the link between the sensor network and the external world) and the position of the wired backbone components is usually known, the distinguishing mathematical property of the WSNL is that a *partial embedding* $x' : U \to \mathbb{Q}^2$ is known in advance, where $U \subseteq V$ is the set of wired backbone components, called *anchors* in the WSNL literature [4, 21]. Again, because wireless communication can only occur below a certain distance threshold, the resulting graph is a UDG.

Lines of forces acting on static physical structure (such as a building) define a graph. If the forces sum to zero, then the structure stands. Starting from such basic definitions, a theory of bar-and-joint structures has been developed ever since the XVIII century [3, 10, 15, 18, 24]. This involves embeddings of the graph where joints are vertices and bars (with their lengths) are weighted edges; the zero sum force requirement holds if a given embedding is an isolated point in embedding space. More recently, the interest was shifted towards graphs whose topology itself guarantees that all (or almost all) embeddings are isolated points. Such graphs are termed *rigid* [6, 20].

Graph Drawing (GD) is a discipline studying algorithms for drawing graphs. The embedding might be defined for any $K \geq 1$, but of course only projections in 2D and 3D are actually represented visually. See www.graphdrawing.org for more information.

### 5.1.2 Complexity

Saxe [19] proved in 1979 that $\mathrm{DGP}_1$ with $\mathbb{F} = \mathbb{N}$ is **NP**-complete by means of a reduction from SUBSET-SUM [5]. It is in **NP** because a given embedding $x$ can be verified to satisfy (5.1) in polynomial time. Furthermore, an instance $\{a_1, \ldots, a_n\} \in \mathbb{Z}^n$ of SUBSET-SUM can be suitably reduced to the instance $G = (V, E, d)$ with $V = \{v_0, \ldots, v_{n-1}\}$, $E = \{\{v_i, v_{i+1 \bmod n}\} \mid i < n\}$, $d_{i,i+1 \bmod n} = a_i$ for all $i < n$.

For what concerns $K > 1$, in [19], Sect. 5, Saxe writes,

- **NP**-Completeness is defined for language recognition problems on Turing Machines, which inherently can deal only with integers and not with arbitrary reals.
- Given a "random" embedding of an unweighted graph into a Euclidean space, any two of the edge weights induced by the embedding will be incommensurable with probability 1. Moreover, if the graph is overconstrained and the dimension of the space is at least two, then rounding the induced edge-weights to multiples of some small distance will almost always produce a weighted graph that is not embeddable in space.

The DGP contains the $\mathrm{DGP}_1$, which is **NP**-complete, but as remarked by Saxe, the DGP itself might not be in **NP**. Thus, it is commonly stated in the literature that the DGP (and in particular the MDGP and the WSNL) is **NP**-hard (see, e.g., [4, 9]). By definition [5], a problem is **NP**-hard when every problem in **NP** can be reduced to it, independently of whether the problem itself is in **NP** or not.

In order to show that a decision problem is in **NP**, we have to perform the following steps:

1. Encoding certificates of YES instances
2. Showing that such certificates can be verified in time which is polynomial in the size of the instance

In the case of the DGP, the certificates are solutions of the system (5.1). Squaring every equation of the system yields

$$\forall \{i,j\} \in E \quad \|x_i - x_j\|^2 = d_{ij}^2. \tag{5.2}$$

System (5.2) has the same set of solutions as Eq. (5.1), since $d$ always takes nonnegative values. Notice, however, that Eq. (5.2) is a polynomial system: as such, its solutions $x = ((x_{11}, \ldots, x_{1K},), \ldots, (x_{n1}, \ldots, x_{nK}))$ always have algebraic components.

## 5.2 Representations of Algebraic Numbers

It is well known that some algebraic numbers over $\mathbb{Q}$ can be written as mathematical expressions involving integers and elementary operators such as sum, subtraction, product, fraction, and $k$-root. Let us call $\mathscr{O}$ the set of operator symbols $+, \times, \div, \sqrt[k]{}$. The statement DGP $\in$ **NP** is equivalent to stating that all components of an embedding solving the instance can always be written as meaningful strings of symbols in $\mathbb{Z}$ and $\mathscr{O}$, the size of which is bound by a polynomial in the instance size. Not all algebraic numbers, however, can be written this way: specifically, this is the case if and only if the Galois group of the minimal polynomial of the algebraic number in question is soluble [22]. What about those algebraic numbers that do not satisfy this requirement?

If $\alpha$ is a root of a polynomial $p(x)$ over $\mathbb{Q}$ whose Galois group is not soluble, then it cannot be expressed using symbols in $\mathbb{Z} \cup \mathscr{O}$ alone. What one can do, however, is to adjoin other algebraic numbers in $B = \{\beta_1, \dots, \beta_h\}$ to $\mathbb{Q}$, obtaining other fields $F = \mathbb{Q}[\beta_1, \dots, \beta_h]$, until the minimal polynomial of $\alpha$ over $F$ has a soluble Galois group. This process terminates: it suffices to adjoin all the roots of $p(x)$. Symbolic algebra packages such as Maple [14] attempt to find smallest $h$ such that the Galois group of $p(x)$ over $F$ is soluble. Then $\alpha$ can be expressed by meaningful strings of symbols in $\mathbb{Z} \cup B \cup \mathscr{O}$.

*Example 5.1.* Asking Maple to solve

$$x^5 + y + 1 = 0$$
$$y^2 + y - x = 0$$

yields the solution $x = \alpha^2 + \alpha$, $y = \alpha$, where $\alpha$ is a root of the polynomial $(x + 1)(x^8 + 3x^7 + 3x^6 + x^5 + 1)$. The Galois group of $x^8 + 3x^7 + 3x^6 + x^5 + 1$ is $S_8$, the full symmetric group over 8 elements, and $S_8$ is not soluble.

### 5.2.1 Polynomial System Representation

Each algebraic number $\alpha \in \mathbb{A}$ can be associated with a polynomial $p_\alpha \in \mathbb{Q}[x]$ such that $p(\alpha) = 0$ and a rational $\bar{\alpha} \in \mathbb{Q}$ which is closest to $\alpha$ than to the other roots of $p_\alpha$.

*Example 5.2.* For $\alpha = \sqrt[3]{\frac{1}{2} + \sqrt{[4]3}}$ we might choose its minimal polynomial over $\mathbb{Q}$, $p_\alpha(x) = x^{12} - 2x^9 + \frac{3}{2}x^6 - \frac{1}{2}x^3 - \frac{47}{16}$, and set $\bar{\alpha} = 2$, which is closest to $\alpha$ than to the other real root of $p_\alpha$.

As mentioned above, embeddings can be seen as sequences of algebraic numbers. Any sequence $S$ of $\ell$ algebraic numbers can be associated with a multivariate polynomial system $\mathbf{p}_S \in \mathbb{Z}[x_1, \dots, x_\ell]$ such that $\mathbf{p}_S(S) = 0$, together with a rational vector $q \in \mathbb{Q}^\ell$ such that $\|S - q\|_2$ is smallest.

### 5.2.2 Formal Grammar Representation

The "meaningful strings" mentioned above, used to represent algebraic numbers in a field $F = Q[B]$ where $B = \{\beta_1, \dots, \beta_h\}$, are generated by the formal grammar:

$$\mathbb{A} \longrightarrow (\mathbb{A} + \mathbb{A}) \vee (\mathbb{A} \times \mathbb{A}) \vee (\mathbb{A} \div \mathbb{A}) \vee (\sqrt[\mathbb{Z}]{\mathbb{A}}) \vee (\mathbb{Z}) \vee (B)$$

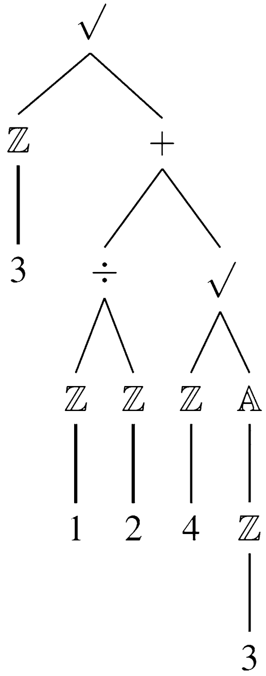


**Fig. 5.1** The derivation tree for $\alpha$ according to the algebraic grammar

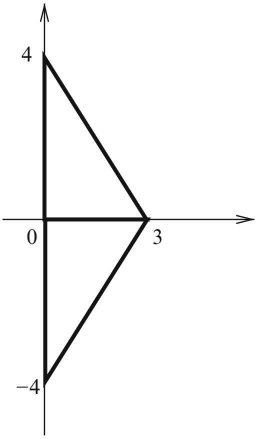
where, with a slight abuse of notation, we use $\mathbb{A}, \mathbb{Z}$ to denote the *type* of algebraic and integer numbers. Given a string consisting of symbols in $\mathbb{Z} \cup B \cup \mathscr{O}$, the string is meaningful if it matches the pattern given by the grammar. The algorithm that matches strings to grammars [12] is recursive in nature and yields a *grammar derivation trees* [16]. Each algebraic number in $\mathbb{A}$ can be represented with respect to $B$ by its corresponding derivation tree.

*Example 5.3.* The algebraic number $\alpha = \sqrt[3]{\frac{1}{2} + \sqrt[4]{3}}$ yields the grammar derivation tree shown in Fig. 5.1.

## 5.3 The Gröbner Bases Strategy

We restrict our attention to $K = 2$ and propose to pursue a line of argument showing that $\text{DGP}_2 \in \mathbf{NP}$. It is well known that any multivariate polynomial system of equations such as Eq. (5.2) can be reduced to a "triangular form" by employing Gröbner bases and the Buchberger algorithm [2] (a clear and short introduction to these concepts can be found in [8]). We represent an embedding $x : V \to \mathbb{R}^2$ solving a $\text{DGP}_2$ instance as the sequence $(x_{11}, x_{12}, x_{21}, x_{22}, \ldots, x_{n1}, x_{n2})$.

**Fig. 5.2** The two configurations given by the Gröbner system in Example 5.4



*Example 5.4.* Consider the right-angled triangle with smallest possible integer side lengths $(3,4,5)$ in $\mathbb{R}^2$ delimited by $x_1 = (0,0)$, $x_2 = (3,0)$, $x_3 = (0,4)$. System (5.2) is:

$$\begin{aligned}
(x_{11}-x_{21})^2+(x_{12}-x_{22})^2 &= 9\\
(x_{11}-x_{31})^2+(x_{12}-x_{32})^2 &= 16\\
(x_{21}-x_{31})^2+(x_{22}-x_{32})^2 &= 25.
\end{aligned}$$

The above system describes *all* $(3,4,5)$-sided triangles in $\mathbb{R}^2$. We can fix $x_{11} = x_{12} = 0$ and $x_{21} = 3$ to eliminate rotations and translations. This reduces the system to

$$\begin{aligned}
3^2+x_{22}^2 &= 9\\
x_{31}^2+x_{32}^2 &= 16\\
(3-x_{31})^2+(x_{22}-x_{32})^2 &= 25.
\end{aligned}$$

A Gröbner basis of the above system (provided by Maple 9.5 [14] with the pure lexicographic term ordering) is given by

$$\begin{aligned}
x_{31}^2 &= 0\\
x_{32}^2 &= 16\\
16x_{22}+3x_{31}x_{32} &= 0.
\end{aligned}$$

It is clear that the Gröbner system has two real solutions given by $x_{22} = x_{31} = 0$ and $x_{32} = \pm 4$, which correspond to two congruent conformations reflected along the 1st coordinate, as shown in Fig. 5.2.

Let the system (5.2) have solution set $X$, and let $x \in X$. According to Sect. 5.2.1 we can represent $x$ by Eq. (5.2) and a rational vector $q \in \mathbb{Q}^{2n}$ which is closest to $x$

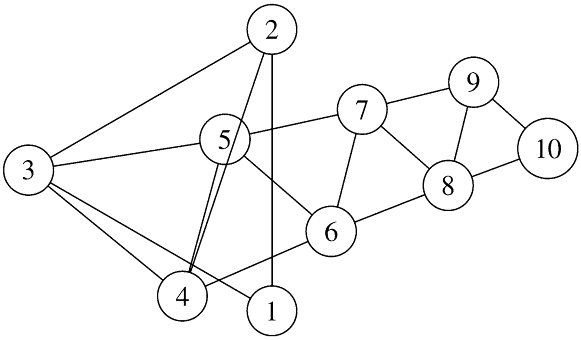


**Fig. 5.3** A triangle chain of size 10 embedded in $\mathbb{R}^2$

than any other $x' \in X$. Because of Gröbner basis theory, it follows that the very same embedding can be represented by any Gröbner basis system derived from Eq. (5.2) and $q$. The advantage in reducing the original system (5.2) to triangular form is that, by a form of back substitution, we can easily derive the set $B$ referred to in Sect. 5.2, together with the string that describes the components of $x$.

Showing that the size of a Gröbner basis is bounded by a polynomial in the instance size would be a (substantial) first step toward proving that $\text{DGP}_2 \in \mathbf{NP}$. Unfortunately, this is false in general: the size of a Gröbner basis grows doubly exponentially. The polynomial system (5.2), however, has a very special structure, which—one might hope—could provide an exception. The rest of this section will introduce an infinite class of DGP instances which provide empirical evidence to the contrary. This is, of course, not a conclusive statement.

### 5.3.1 The Empirical Evidence Against

In this section we construct an infinite class of graphs embedded in $\mathbb{R}^2$ which have a Gröbner basis whose size, obtained computationally for a few cases, indicates an exponential growth in the instance size. The graph class consists of a chain of triangles sharing a side: $V = \{1,\dots,n\}$ (with $n \geq 3$), and $E = \{\{v-2,v\},\{v-1,v\} \mid v > 2\}$. The weight function $d : E \to \mathbb{Q}_+$ is such that $d_{uv} = \frac{1}{u}$ for all $\{u,v\}$ such that $u < v$. Examples with $n = 10$ and $n = 20$ are given, respectively, in Figs. 5.3 and 5.4.

These triangle chains embedded in $\mathbb{R}^2$ provide rigid frameworks [7] and are examples of of Henneberg type I graphs [23] and of DISCRETIZABLE DGP (DDGP) instances [17]. Using Maple [14], we were able to show that the dependency of the Gröbner basis size in terms of the instance size looks exponential over a set of triangle chains with $n$ vertices with $n \in \{3,\dots,11\}$. More precisely, the number of equations in the Gröbner basis and the size of each equation both seem to grow exponentially (or worse), whereas the degree seems to grow linearly, as shown in Fig. 5.5.

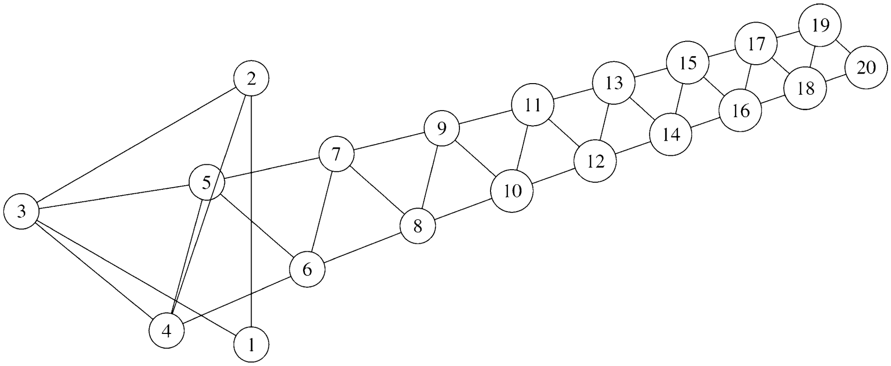


**Fig. 5.4** A triangle chain of size 20 embedded in $\mathbb{R}^2$

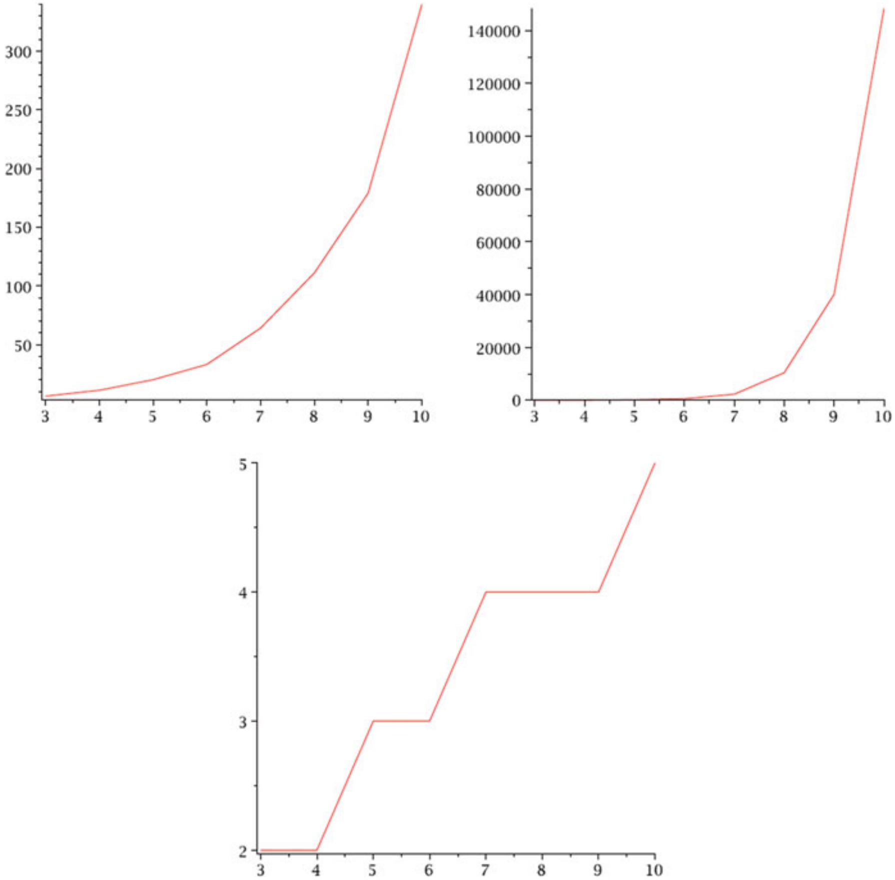


**Fig. 5.5** The growth pattern of the number of equations (*left*) in the Gröbner basis of triangle chains, the size of each equation (*center*), and the degree of the system (*right*)

## References

1. Clark, B., Colburn, C., Johnson, D.: Unit disk graph. Discrete Math. **86**, 165–177 (1990)
2. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties and Algorithms, 2nd edn. Springer, Berlin (1997)
3. Cremona, L.: Le figure reciproche nella statica grafica. In: Bernardoni, G., Milano (1872)
4. Eren, T., Goldenberg, D., Whiteley, W., Yang, Y., Morse, A., Anderson, B., Belhumeur, P.: Rigidity, computation, and randomization in network localization. In: IEEE Infocom Proceedings, 2673–2684 (2004)
5. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of **NP**-Completeness. Freeman and Company, New York (1979)
6. Graver, J.: Rigidity matroids. SIAM J. Discrete Math. **4**, 355–368 (1991)
7. Graver, J., Servatius, B., Servatius, H.: Combinatorial rigidity. Am. Math. Soc. (1993), http://books.google.com.pe/books/about/Combinatorial_Rigidity.html?id=0XwvY1GVNN4C
8. Hägglöf, K., Lindberg, P., Svensson, L.: Computing global minima to polynomial optimization problems using gröbner bases. J. Global Optim. **7**(2), 115–125 (1995)
9. Hendrickson, B.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
10. Henneberg, L.: Die Graphische Statik der starren Systeme. Teubner, Leipzig (1911)
11. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
12. Levine, R., Mason, T., Brown, D.: Lex and Yacc, 2nd edn. O'Reilly, Cambridge (1995)
13. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res.**18**, 33–51 (2010)
14. Maplesoft, Inc.: Maple 9 Getting Started Guide. Maplesoft, Waterloo (2003) http://www.maplesoft.com/products/maple/manuals/GettingStartedGuide.pdf
15. Maxwell, J.: On the calculation of the equilibrium and stiffness of frames. Phil. Mag. **27**(182), 294–299 (1864)
16. Mosses, P.: Denotational semantics, In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science B: Formal Models and Semantics, pp. 575–631. Elsevier, Amsterdam (1990)
17. Mucherino, A., Lavor, C., Liberti, L.: The discretizable distance geometry problem. Optimization Letters, Springer: **6**(8), 1671–1686 (2012)
18. Saviotti, C.: Nouvelles méthodes pour le calcul des travures réticulaires In: Appendix to Cremona, L., "Les figures réciproques en statique graphique", pp. 37–100. Gauthier-Villars, Paris (1885)
19. Saxe, J.: Embeddability of weighted graphs in *k*-space is strongly **NP**-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
20. Servatius, B., Servatius, H.: Generic and abstract rigidity, In: Thorpe, M., Duxbury, P. (eds.) Rigidity Theory and Applications, Fundamental Materials Research, pp. 1–19. Springer, New York (2002) DOI: 10.1007/0-306-47089-6_1
21. So, A.M.C., Ye, Y.: Theory of semidefinite programming for sensor network localization. Math. Program. B **109**, 367–384 (2007)
22. Stewart, I.: Galois Theory, 2nd edn. Chapman and Hall, London (1989)
23. Tay, T.S., Whiteley, W.: Generating isostatic frameworks. Structural Topology **11**, 21–69 (1985)
24. Varignon, P.: Nouvelle Mecanique. Claude Jombert, Paris (1725)

# Chapter 6
# Solving Spatial Constraints with Generalized Distance Geometry

**Lu Yang**

**Abstract** This is a short survey about how to use a generalization of distance geometry to solve spatial constraints. Described first are definitions and some basic theorems in the generalized distance geometry, and a systematic approach for the solution of spatial constraints follows, making use of results of the previous section. An intrinsic coordinate system based on geometric invariants, named distance coordinate system, is established for simplification and algorithmization to the process of constraint solving. A short program is proposed, which implements the algorithm producing automatically a complete set of constraint equations for a given point-plane configuration, and the point-line-plane configurations are converted into point-plane ones beforehand. The so-called nontypical constraint problem is considered in the last section and illustrated by an example, which the previous method seems unable to help.

**Keywords** Generalized distance geometry • Distance coordinate • Spatial constraint solving • Point-line-plane configuration

## 6.1 Introduction

Presented is a short survey to understand the approaches the author involved for spatial constraint solving which make use of the generalized distance geometry.

In a seminar many years ago, colleagues tried to make a comparison of efficiency between distance geometry method and Cartesian coordinate method for computing and constraint solving. We found that the classical distance geometry would not be very convenient in case of the constraints involving not only points but also lines,

L. Yang (✉)
Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China
e-mail: luyang@casit.ac.cn; lyang@sei.ecnu.edu.cn

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_6,

planes and spheres. Especially, the classical embedding theory always requires too much data available that may be impossible or unnecessary in many instances.

The so-called *elementary configuration* was proposed in 1994 [31] that is a configuration $\mathscr{P}$ of points, $(n-1)$-dimensional hyperplanes and $(n-1)$-dimensional hyperspheres in $n$-dimensional Euclidean space. An *abstract distance* is defined pairwise over the configuration, $\mathscr{P} \times \mathscr{P} \to \mathrm{R}$. It was shown in [31] a criterion to determine whether or not an elementary configuration can be realized with all the abstract distances prescribed. This result generalizes the classical distance geometry; however, how it can be efficiently made use of for constraint solving in practice?

In general case, we are given only a part of the distance data while other ones are to be found. So, we need to establish an equation system connecting the distance data known or unknown, by means of the extended Cayley–Menger determinants of the point-plane configurations. Meanwhile if there are some lines involved, we need to supplement some new points or planes to replace the lines such that the extended point-plane configuration keeps all the data. And then, solve the constraint equation system by symbolic or numeric methods with programs, if any.

Since distance geometry often provides too much data more than the required, which ones should be chosen to establish the constraint equation system? To avoid this hesitation, a scheme of returning to *distance coordinate* is taken. That is an intrinsic coordinate, quite different from Cartesian.

The content of this chapter is arranged as follows. The next section introduces a generalization of distance geometry and provides a proof to a fundamental theorem for the point-plane configuration in $n$-dimensional Euclidean space. Section 6.3 introduces a systematic approach making use of distance geometry to solve spatial constraints that demonstrates how to create the constraint equations by means of a relevant distance coordinate system. A short program is made (in Maple) which implements the algorithm producing automatically a complete set of constraint equations for a given point-plane configuration. The point-line-plane configurations are converted into point-plane ones beforehand. Section 6.4 shows a nontypical problem of constraint solving which the distance coordinate method seems unable to help.

## 6.2 Generalization of Distance Geometry

The classical distance geometry studies metric relations based on the single kind of geometric invariant: *distance* between points. To meet the requirement of geometric computing and reasoning, a generalized frame was developed by the author and his collaborators in 1980s–1990s [24–26, 31]. In this frame, an *abstract distance* is defined over a configuration of points, hyperplanes and hyperspheres, and then an extension of the Cayley–Menger algebra is introduced. The parallel approaches to non-Euclidean constant-curvature spaces have been made in the beginning of 1980s as well; see [24].

The following notation describes the metric relation over a point-plane configuration. Given an $n$-tuple of points and oriented hyperplanes in a Euclidean space, $P = (p_1, p_2, \cdots, p_n)$, we define a mapping $g : P \times P \to \mathrm{R}$ by letting

- $g(p_i, p_j)$ be the square of the distance between $p_i$ and $p_j$ if both are points.
- $g(p_i, p_j)$ be the signed distance from $p_i$ to $p_j$, if one is a point and the other an oriented hyperplane.
- $g(p_i, p_j)$ be $-\frac{1}{2}\cos(p_i, p_j)$, if both are oriented hyperplanes.

By $g_{ij}$ denote $g(p_i, p_j)$ and $G$ denote the matrix $(g_{ij})_{n\times n}$, and let

$$\delta = (\delta_1, \delta_2, \cdots, \delta_n),$$

where

$$\delta_i = \begin{cases} 1, & \text{if } p_i \text{ is a point,} \\ 0, & \text{if } p_i \text{ is a hyperplane,} \end{cases}$$

where $i = 1, \ldots, n$. Then, let

$$M(p_1, p_2, \cdots, p_n) = \begin{pmatrix} G & \delta^T \\ \delta & 0 \end{pmatrix},$$

which is called the Cayley–Menger matrix of $P$, and let

$$D(p_1, p_2, \cdots, p_n) = \begin{vmatrix} G & \delta^T \\ \delta & 0 \end{vmatrix},$$

which is called the Cayley–Menger determinant of $P$. The following theorem is an extension of the classical result on Cayley–Menger determinant.

**Theorem 6.1.** *Let $D(p_1, p_2, \cdots, p_n)$ be the Cayley–Menger determinant of an n-tuple of points and oriented hyperplanes in d-dimensional space. If $n \geq d+2$, then*

$$D(p_1, p_2, \cdots, p_n) = 0. \tag{6.1}$$

This theorem is essentially the Theorem 1.1 of [26] and can also be found in earlier articles [24, 25].

*Proof.* Assume there is at least one point among $p_1, p_2, \cdots, p_n$; otherwise, the Cayley–Menger matrix is obviously singular since the entries of the last row all are zero. Without loss of generality, we let $p_1, p_2, \cdots, p_s$ be oriented hyperplanes (where $s < d$) and $p_{s+1}, \ldots, p_n$ be points. Take $p_n$ as the origin of coordinates. By $\alpha_1, \cdots, \alpha_s$ denote the unit normal vectors of oriented hyperplanes $p_1, \cdots, p_s$, by $\alpha_{s+1}, \cdots, \alpha_n$ denote the position vectors of points $p_{s+1}, \ldots, p_n$, and $\beta_1, \cdots, \beta_s$ denote the position vectors of the feet of the perpendiculars to hyperplanes $p_1, \cdots, p_s$ from the origin, respectively.

Then, $g_{ij}$, all the entries of the Cayley–Menger matrix $G$, can be represented in a vector format as follows:

$$g_{ij} = g(p_i, p_j) = \begin{cases} -\dfrac{1}{2}\alpha_i\alpha_j & (1 \le i \le s, 1 \le j \le s), \\ \alpha_i(\alpha_j - \beta_i) & (1 \le i \le s, s < j \le n), \\ (\alpha_i - \alpha_j)^2 & (s < i \le n, s < j \le n). \end{cases} \tag{6.2}$$

In this case, clearly,

$$\delta_i = \begin{cases} 1, & (1 \le i \le s), \\ 0, & (s < i \le n). \end{cases} \tag{6.3}$$

Now, do a series of elementary transformations to matrix $G$ step by step:

1. Multiply the $i$th row by $-2$, for $i = 1, \ldots, s$.
2. Multiply the $j$th column by $-\frac{1}{2}$, for $j = s+1, \ldots, n$.
3. Multiply the last row (i.e. the $(n+1)$th row) by $-2$.
4. Add the last row times $\alpha_i\beta_i$ to the $i$th row, for $i = 1, \ldots, s$.
5. Add the last column times $\alpha_j\beta_j$ to the $j$th column, for $j = 1, \ldots, s$.
6. Add the last row times $\frac{1}{2}\alpha_i^2$ to the $i$th row, for $i = s+1, \ldots, n$.
7. Add the last column times $\frac{1}{2}\alpha_j^2$ to the $j$th column, for $j = s+1, \ldots, n$.

The procedure results in a matrix $L$,

$$L = \begin{pmatrix} F & \delta^T \\ \delta & 0 \end{pmatrix}, \quad \text{where} \quad F = (f_{ij})_{n\times n}$$

with $f_{ij} = \alpha_i\alpha_j$ for $i, j = 1, \ldots, n$ and $\delta = (\delta_1, \delta_2, \cdots, \delta_n)$ defined as (6.3).

By $|F|_{n,n}$ denote the $(n,n)$th cofactor of determinant $|F|$, i.e. the determinant of the matrix $(f_{ij})_{(n-1)\times(n-1)}$. Since $\alpha_n$ is a null vector, we have $f_{ni} = \alpha_n\alpha_i = 0$, $f_{in} = \alpha_i\alpha_n = 0$, for $i = 1, \ldots, n$. Thus

$$|L| = \begin{vmatrix} F & \delta^T \\ \delta & 0 \end{vmatrix} = -|F|_{n,n}.$$

Noting that $|F|_{n,n}$ is Gram determinant of $d$-dimensional vectors $\alpha_1, \cdots, \alpha_{n-1}$ and $n-1 > d$, it is vanishing, so is $|L|$, and so is $D(p_1, p_2, \cdots, p_n)$ because $L$ is a result of some elementary transformations from $G$. This completes the proof. □

By $A_{i,j}$ denote the $(i,j)$th cofactor of a determinant $A$, i.e., $(-1)^{i+j}$ times the subdeterminant of $A$ in which the $i$th row and the $j$th column have been removed. As a corollary of Theorem 6.1, we have

**Corollary 6.1.** *Let $D$ be the Cayley–Menger determinant of an $n$-tuple of points and oriented hyperplanes in $d$-dimensional space. If $n \geq d+3$, then*

$$D_{i,j} = 0 \qquad (\textit{for } i,j = 1,\ldots,n+1). \tag{6.4}$$

*Proof.* This depends on a lemma on determinants: Given a determinant $A$, it holds for $i < j$ that

$$A_{i,i}A_{j,j} - A_{i,j}A_{j,i} = (A_{j,j})_{i,i}A. \tag{6.5}$$

Here $(A_{j,j})_{i,i}$ is the $(i,i)$th cofactor of $A_{j,j}$, while $A_{j,j}$ stands for the $(j,j)$th cofactor of $A$. The identity (6.5) was used in Blumenthal's book [1] several times without proof; maybe he thought it well known; see the pages 100, 102, etc. In fact, identity (6.5) is a simple instance of Jacobi's Theorem[8, 9, 19, 21]. A simpler proof of Jacobi's Theorem can be found in [26] or [25]. Applying (6.5) to the Cayley–Menger determinant $D$,

$$D_{i,i}D_{j,j} - D_{i,j}D_{j,i} = (D_{j,j})_{i,i}D,$$

where clearly $D_{i,i}$ is the Cayley–Menger determinant of an $(n-1)$-tuple; by Theorem 6.1, $D_{i,i} = 0, D = 0$ because $n \geq d+3$. Noting $D$ is symmetric, we have $D_{i,j} = 0$. Corollary 6.1 thus holds. □

A proof of Corollary 6.1 in the case that all elements are points can also be found in [21].

## 6.3 Solving Spatial Constraints with Distance Coordinates

Distance geometry has been used in molecular conformation for many years [13,14]. The classical embedding algorithm requires the full set of distances between all atoms; see [4]. This is usually impossible or unnecessary in practice. When some of the distances are not provided, two major heuristic approaches have been used to solve the *sparse* distance geometry problem. Recently, another heuristic approach was presented in [5] that requires provided at least $4n-10$ distances and rather severe conditions, e.g. a full set of distances between some four atoms must be given at first. The method was illustrated in [5] by a 7-atom conformation as Fig. 6.1. Solving a spatial constraint means locating a configuration in three-dimensional Euclidean space. The usual practice is to determine the coordinates of the points with respect to a certain coordinate system, Cartesian or others. Usually, $3n-6$ data are enough for solving a spatial constraint on $n$ points if these data are independent. In that case the problem is called well constrained. A data may be a distance, may be an equation which connects some distances. A well-constrained problem usually has just finitely many solutions. There are 18 distances given in Fig. 6.1, a 7-point-configuration, so the problem is over-constrained.

In the present section, a distance-based global coordinate method is proposed in a succinct formulation for solving well-constrained problems without redundant data.

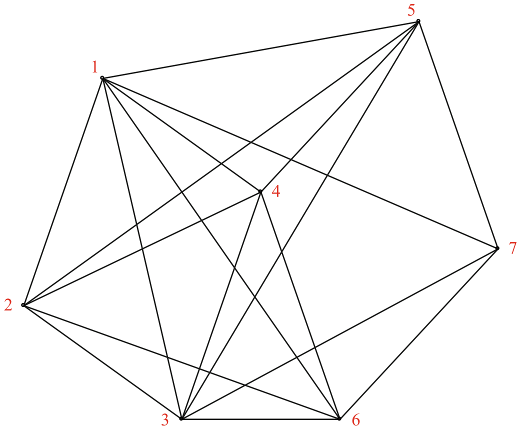


**Fig. 6.1** A 7-atom conformation with 18 distance data

### *6.3.1 Creating Constraint Equations with Distance Coordinates*

The concept of "distance coordinates" is not new that appeared in earlier literature, say, [21] where it was called "Cayley–Menger coordinates" which mean the squared distances to reference points from a point. What we investigate here are configurations consisting of points and planes both, so we need to generalize the concept accordingly.

There are different ways to create a distance coordinate system on a given point-plane configuration. Two kinds of coordinate systems, quartuple's and triple's, and how to use them to create constraint equations in a succinct formulation for spatial geometric constraint solving will be described in this section.

#### Quartuple Coordinate System in $\mathbf{E}^3$

Choose 4 points or oriented planes as a *reference tetrahedron*, say, $\{p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}\}$. For any point or oriented plane $p_j$, take

$$(\,g(p_j, p_{i_1}),\, g(p_j, p_{i_2}),\, g(p_j, p_{i_3}),\, g(p_j, p_{i_4})\,)$$

as the coordinates of $p_j$.

Here the choice for the reference tetrahedron may be arbitrarily but usually requires the Cayley–Menger determinant of $p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}$ to be nonvanishing, that is,

$$D(p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}) \neq 0.$$

To solve a constraint problem on a point-plane configuration $\{p_1, \cdots, p_n\}$, we need only find the distance coordinates of every $p_i$. For any couple $(p_i, p_j)$ with

$i < j$, if $g(p_i, p_j)$ is given as a constraint and $\{p_i, p_j\} \cap \{p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}\} = \emptyset$, then we take

$$D_{5,6}(p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}, p_i, p_j) = 0 \tag{6.6}$$

as a constraint equation, where $D_{5,6}$ stands for the (5,6)th cofactor of $D$, as defined formerly. Let $l$ be the number of these equations. Furthermore, for every $p_j \notin \{p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}\}$, take

$$D(p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}, p_j) = 0 \tag{6.7}$$

as a constraint equation, we obtain $n-4$ equations. A complete set of constraint equations with $l+n-4$ members is established in this way.

On the other hand, let

$$S_2 = \{(p_i, p_j) \mid 1 \leq i < j \leq n, \{p_i, p_j\} \cap \{p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}\} \neq \emptyset\}, \tag{6.8}$$

then the cardinal number of set $S_2$ is $4n-10$.

Assume this geometric constraint problem is well constrained. Then, the number of independent constraints should be $3n-6$ since this is a point-plane configuration. It was known from above argument that exactly $l$ constraints do not concern $p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}$, so exactly $3n-6-l$ constraints concern $p_{i_1}, p_{i_2}, p_{i_3}, p_{i_4}$. Therefore, among $4n-10$ distances (or angles) on $S_2$, there are exactly $3n-6-l$ items that are given as constraints so that $l+n-4$ items are unknown. Thus, the number of unknowns equals that of constraint equations.

As one of the advantages, the quartuple coordinates can uniquely and easily determine the Cartesian coordinates when the reference tetrahedron is fixed.

### Triple Coordinate System in $\mathbf{E}^3$

Choose 3 points or oriented planes as a *reference triangle*, say, $\{p_{i_1}, p_{i_2}, p_{i_3}\}$. For any point or oriented plane $p_j$, take

$$(g(p_j, p_{i_1}), g(p_j, p_{i_2}), g(p_j, p_{i_3}))$$

as the coordinates of $p_j$, where function $g$ was well defined in last section.

Here the choice for the reference triangle may be arbitrarily but usually requires the Cayley–Menger determinant of $p_{i_1}, p_{i_2}, p_{i_3}$ to be nonvanishing, that is,

$$D(p_{i_1}, p_{i_2}, p_{i_3}) \neq 0.$$

To solve a constraint problem on a point-plane configuration $\{p_1, \cdots, p_n\}$, we need only find the distance coordinates of every $p_i$. For any couple $(p_i, p_j)$ with $i < j$, if $g(p_i, p_j)$ is given as a constraint and $\{p_i, p_j\} \cap \{p_{i_1}, p_{i_2}, p_{i_3}\} = \emptyset$, then we take

$$D(p_{i_1}, p_{i_2}, p_{i_3}, p_i, p_j) = 0$$

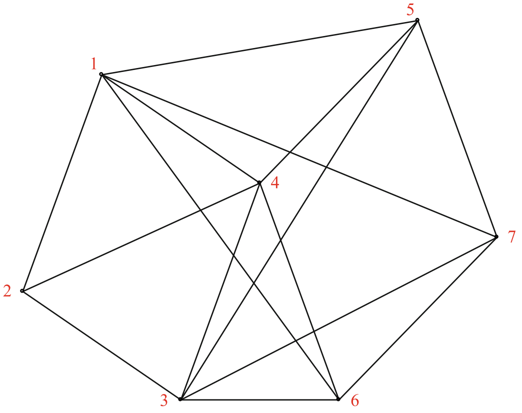


**Fig. 6.2** A 7-atom conformation with 15 distance data

as a constraint equation. A complete set of constraint equations is established in this way. Let $k$ be the number of these equations. On the other hand, let

$$S_1 = \{(p_i, p_j) \,|\, 1 \le i < j \le n, \{p_i, p_j\} \cap \{p_{i_1}, p_{i_2}, p_{i_3}\} \ne \emptyset\}, \tag{6.9}$$

then the cardinal number of set $S_1$ is $3n-6$.

Assume this geometric constraint problem is well constrained. Then, the number of independent constraints should be $3n-6$ since this is a point-plane configuration. It was known from above argument that exactly $k$ constraints do not concern $p_{i_1}, p_{i_2}, p_{i_3}$, so exactly $3n-6-k$ constraints concern $p_{i_1}, p_{i_2}, p_{i_3}$. Therefore, among $3n-6$ distances (or angles) on $S_1$, there are exactly $3n-6-k$ items that are given as constraints so that $k$ items are unknown. Thus, the number of unknowns equals that of constraint equations.

### *6.3.2 Examples Using Quartuple Distance Coordinates*

As mentioned in a survey paper [10], the Cayley–Menger determinant was first used for geometric constraint solving in a web document by D. Michelucci, but it did not follow a systematic approach. A revision of that was presented in a conference later [18].

Now, let us illustrate the method proposed in last section with several examples; that method automatically produces a complete set of constraint equations in a succinct formulation for a well-constrained problem, though these examples could be solved in other way.

*The 7-Atom Conformation.* For the first example, consider the 7-point conformation shown in Fig. 6.1 but remove three segments, say, $p_1p_3, p_2p_5, p_2p_6$, and then get a well-constrained one as Fig. 6.2. The problem cannot be solved by the method of [5].

To solve this, choose a quartuple, say $\{p_1, p_3, p_4, p_7\}$, as the reference tetrahedron. This coordinate system yields a complete set of constraint equations consisting of three equations,

$$\begin{cases} D(p_1, p_3, p_4, p_7, p_2) = 0, \\ D(p_1, p_3, p_4, p_7, p_5) = 0, \\ D(p_1, p_3, p_4, p_7, p_6) = 0, \end{cases}$$

with three unknowns, $g(p_1, p_3), g(p_4, p_7)$ and $g(p_2, p_7)$.

We may, of course, choose another quartuple as the reference tetrahedron; then a different number of equations and unknowns would result.

In fact, we use a short program to yield the constraint equations automatically. Setting $x = g(p_1, p_3), y = g(p_4, p_7), z = g(p_2, p_7)$, the computer gives

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{12} & 1 \\ x & 0 & g_{34} & g_{37} & g_{23} & 1 \\ g_{14} & g_{34} & 0 & y & g_{24} & 1 \\ g_{17} & g_{37} & y & 0 & z & 1 \\ g_{12} & g_{23} & g_{24} & z & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{15} & 1 \\ x & 0 & g_{34} & g_{37} & g_{35} & 1 \\ g_{14} & g_{34} & 0 & y & g_{45} & 1 \\ g_{17} & g_{37} & y & 0 & g_{57} & 1 \\ g_{15} & g_{35} & g_{45} & g_{57} & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{16} & 1 \\ x & 0 & g_{34} & g_{37} & g_{36} & 1 \\ g_{14} & g_{34} & 0 & y & g_{46} & 1 \\ g_{17} & g_{37} & y & 0 & g_{67} & 1 \\ g_{16} & g_{36} & g_{46} & g_{67} & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

where $g_{ij}$ stand for $g(p_i, p_j)$ as defined before. These equations are briefly written as:

$$c_0 + c_1 y + c_2 z + c_3 yz + c_4 y^2 + c_5 z^2 = 0,$$

$$a_0 + a_1 y + a_2 y^2 = 0,$$

$$b_0 + b_1 y + b_2 y^2 = 0,$$

where all the $a_i, b_j, c_k$ are polynomials in $x$ of degree 2 or less. The system can be easily converted to a triangular set:

$$b_2{}^2\, a_0{}^2 - 2\, b_2\, a_0\, a_2\, b_0 + a_2{}^2\, b_0{}^2 - b_1\, b_2\, a_1\, a_0 - b_1\, a_1\, a_2\, b_0 + a_2\, b_1{}^2\, a_0 + b_0\, b_2\, a_1{}^2 = 0,$$
$$(b_2\, a_1 - a_2\, b_1)\, y + b_2\, a_0 - a_2\, b_0 = 0,$$
$$c_0 + c_1\, y + c_2\, z + c_3\, y\, z + c_4\, y^2 + c_5\, z^2 = 0.$$

The first equation is of degree 8 in $x$ since all the $a_i, b_j$ are polynomials in $x$ of degree 2 or less. The second equation is linear in $y$ and the third is quadratic in $z$, so the problem has 16 isolated solutions for $\{x, y, z\}$ at most.

Furthermore, it is not necessarily the seven elements all to be points; some of them may be planes; for instance, $p_2, p_5, p_6$ are oriented planes and $p_1, p_3, p_4, p_7$ points. Still choose $\{p_1, p_3, p_4, p_7\}$ as the reference tetrahedron; our program produces the following constraint equations automatically:

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{12} & 1 \\ x & 0 & g_{34} & g_{37} & g_{23} & 1 \\ g_{14} & g_{34} & 0 & y & g_{24} & 1 \\ g_{17} & g_{37} & y & 0 & z & 1 \\ g_{12} & g_{23} & g_{24} & z & -\frac{1}{2} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{15} & 1 \\ x & 0 & g_{34} & g_{37} & g_{35} & 1 \\ g_{14} & g_{34} & 0 & y & g_{45} & 1 \\ g_{17} & g_{37} & y & 0 & g_{57} & 1 \\ g_{15} & g_{35} & g_{45} & g_{57} & -\frac{1}{2} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & x & g_{14} & g_{17} & g_{16} & 1 \\ x & 0 & g_{34} & g_{37} & g_{36} & 1 \\ g_{14} & g_{34} & 0 & y & g_{46} & 1 \\ g_{17} & g_{37} & y & 0 & g_{67} & 1 \\ g_{16} & g_{36} & g_{46} & g_{67} & -\frac{1}{2} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

which are somewhat different from those for the 7-point conformation . By an analogous but more careful argument, the system of equations has 12 isolated solutions for $\{x, y, z\}$ at most.

*Solving a 2-2-2 Wire-Based Tracking Device.* The second example originated from solving a 2-2-2 configuration for wire-based tracking device as proposed in [11]; also see [23].

Given six points $p_4$, $p_5$, $p_6$, $p_7$, $p_8$, $p_9$ fixed on the bottom plane, the problem is how to locate a triangular platform $p_1p_2p_3$ provided the lengths of the six legs $p_1p_4$, $p_1p_7$, $p_2p_5$, $p_2p_8$, $p_3p_6$, $p_3p_9$ which connect the platform and the bottom.

Let us make some simplification to this problem at first. The configuration includes nine points. It had been better for us to manage to locate the platform $p_1p_2p_3$ by means of a simpler configuration which involves fewer points, say, $p_1,p_2,p_3,p_4,p_5,p_6$, six points only. The three constraints, the lengths of $p_1p_7$, $p_2p_8$, $p_3p_9$, from $p_7,p_8,p_9$ can be converted to three additional constraints among only $p_1,p_2,p_3,p_4,p_5,p_6$ beforehand . Now we give the conversion as follows.

The previous assumption implies that

$$D(p_1,p_4,p_5,p_6,p_7)=0, \quad D(p_4,p_5,p_6,p_7)=0.$$

Setting $A=D(p_1,p_4,p_5,p_6,p_7)$ and $A_{i,j}$ stands for its $(i,j)$th cofactor for $i,j=1..6$, we still make use of Jacobi's Theorem:

$$A_{1,1}A_{6,6}-A_{1,6}A_{6,1}=(A_{6,6})_{1,1}A.$$

Noting $A=0$ and $A_{1,1}=D(p_4,p_5,p_6,p_7)=0$, we have $A_{1,6}=0$, that is,

$$\begin{vmatrix} g_{14} & 0 & g_{45} & g_{46} & g_{47} \\ g_{15} & g_{45} & 0 & g_{56} & g_{57} \\ g_{16} & g_{46} & g_{56} & 0 & g_{67} \\ g_{17} & g_{47} & g_{57} & g_{67} & 0 \\ 1 & 1 & 1 & 1 & 1 \end{vmatrix} = 0.$$

This is a linear equation on $\{g_{15},g_{16}\}$ since the other entries all are known, so we may let $g_{16}=m_1g_{15}+n_1$, and by the analogous argument, $g_{24}=m_2g_{26}+n_2$, $g_{35}=m_3g_{34}+n_3$, where $m_i,n_i$ are known already. These three linear equations we take as the supplementary data.

Now, consider the simplified configuration consisting of only six points (where the points $p_7$, $p_8$ and $p_9$ are absent from) as shown in Fig. 6.3.

Two triangles $p_1p_2p_3$ and $p_4p_5p_6$ are not coplanar. The lengths of segments $p_1p_4,p_2p_5,p_3p_6$ and all the sides of the triangles are provided. Let

$$x=g(p_1,p_5), \quad y=g(p_2,p_6), \quad z=g(p_3,p_4),$$
$$x_1=g(p_1,p_6), \quad y_1=g(p_2,p_4), \quad z_1=g(p_3,p_5).$$

Then the 3 supplementary data we obtained above,

$$x_1=m_1x+n_1, \quad y_1=m_2y+n_2, \quad z_1=m_3z+n_3,$$

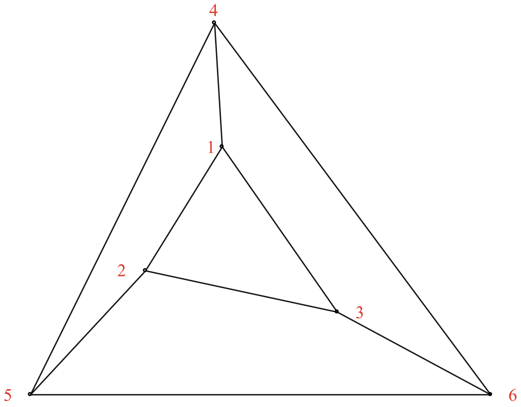


**Fig. 6.3** A simplified configuration from a parallel wire mechanism

where $m_i, n_i$ are already known, combining with the nine distances given in Fig. 6.3, results a well-constrained problem because we have had required 12 data, even though of which only nine data appear as distances.

To solve this, choose a quartuple, say $\{p_1, p_4, p_5, p_6\}$, as the reference tetrahedron. This coordinate system yields a complete set of constraint equations consisting of three equations,

$$\begin{cases} D(p_1, p_4, p_5, p_6, p_2) = 0, \\ D(p_1, p_4, p_5, p_6, p_3) = 0, \\ D_{5,6}(p_1, p_4, p_5, p_6, p_2, p_3) = 0, \end{cases}$$

with three unknowns, $x = g(p_1, p_5), y = g(p_2, p_6)$ and $z = g(p_3, p_4)$. Here $D_{5,6}(\cdots)$ stands for the $(5,6)$th cofactor of determinant $D(\cdots)$, as introduced formerly. The computer writes them in detail:

$$\begin{vmatrix} 0 & g_{14} & x & x_1 & g_{12} & 1 \\ g_{14} & 0 & g_{45} & g_{46} & y_1 & 1 \\ x & g_{45} & 0 & g_{56} & g_{25} & 1 \\ x_1 & g_{46} & g_{56} & 0 & y & 1 \\ g_{12} & y_1 & g_{25} & y & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & g_{14} & x & x_1 & g_{13} & 1 \\ g_{14} & 0 & g_{45} & g_{46} & z & 1 \\ x & g_{45} & 0 & g_{56} & z_1 & 1 \\ x_1 & g_{46} & g_{56} & 0 & g_{36} & 1 \\ g_{13} & z & z_1 & g_{36} & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & g_{14} & x & x_1 & g_{12} & 1 \\ g_{14} & 0 & g_{45} & g_{46} & y_1 & 1 \\ x & g_{45} & 0 & g_{56} & g_{25} & 1 \\ x_1 & g_{46} & g_{56} & 0 & y & 1 \\ g_{13} & z & z_1 & g_{36} & g_{23} & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0.$$

Recalling $x_1 = m_1x + n_1$, $y_1 = m_2y + n_2$, $z_1 = m_3z + n_3$, these equations are briefly written as:

$$\begin{aligned} &a_0 + a_1y + a_2y^2 = 0, \\ &b_0 + b_1z + b_2z^2 = 0, \\ &c_0 + c_1y + c_2z + c_3yz = 0, \end{aligned}$$

where all the $a_i, b_j, c_k$ are polynomials in $x$ of degree 2 or less. By a routine method of symbolic elimination, the system can be converted to a triangular set:

$$\begin{aligned} &{b_2}^2 {a_2}^2 {c_0}^4 - 2 {a_2}^2 c_0 b_0 {c_2}^3 b_1 + {b_2}^2 {a_0}^2 {c_1}^4 + {a_2}^2 {b_0}^2 {c_2}^4 + {c_3}^4 {b_0}^2 {a_0}^2 \\ &+ 8 b_2 a_2 c_3 c_0 b_0 a_0 c_1 c_2 - 2 b_2 a_2 b_0 a_0 {c_1}^2 {c_2}^2 - 2 {b_2}^2 a_2 {c_0}^3 a_1 c_1 \\ &+ 2 {b_2}^2 a_2 {c_0}^2 {c_1}^2 a_0 + {c_3}^2 {b_0}^2 {c_2}^2 {a_1}^2 + {c_3}^2 {a_0}^2 {c_1}^2 {b_1}^2 + {a_2}^2 {c_0}^2 {b_1}^2 {c_2}^2 \\ &+ {b_2}^2 {c_0}^2 {a_1}^2 {c_1}^2 + a_2 {c_3}^2 {c_0}^2 a_0 {b_1}^2 + b_2 {c_3}^2 {c_0}^2 b_0 {a_1}^2 + a_2 {b_1}^2 {c_2}^2 {c_1}^2 a_0 \\ &+ b_2 b_0 {c_2}^2 {a_1}^2 {c_1}^2 - c_3 b_0 c_1 {c_2}^2 {a_1}^2 b_1 + a_2 b_0 {c_2}^3 c_1 b_1 a_1 + b_2 a_2 c_3 {c_0}^3 a_1 b_1 \\ &+ b_2 a_0 {c_1}^3 c_2 b_1 a_1 - a_2 c_0 a_1 {c_2}^2 c_1 {b_1}^2 - {c_3}^2 c_0 b_0 c_2 {a_1}^2 b_1 + {c_3}^3 c_0 b_0 a_0 a_1 b_1 \\ &- b_2 c_0 {c_1}^2 c_2 {a_1}^2 b_1 - a_2 c_3 {c_0}^2 c_2 a_1 {b_1}^2 - b_2 c_3 {c_0}^2 {a_1}^2 c_1 b_1 - c_3 {c_1}^2 c_2 a_0 {b_1}^2 a_1 \\ &+ c_3 c_0 c_2 {a_1}^2 {b_1}^2 c_1 - {c_3}^2 c_0 a_0 c_1 {b_1}^2 a_1 - 2 {b_2}^2 c_0 a_1 {c_1}^3 a_0 + 2 b_2 {a_2}^2 {c_0}^2 b_0 {c_2}^2 \\ &- 2 a_2 c_3 {b_0}^2 {c_2}^3 a_1 - 2 {c_3}^3 {b_0}^2 c_2 a_0 a_1 + 2 a_2 {c_3}^2 {b_0}^2 {c_2}^2 a_0 - 2 {c_3}^3 b_0 {a_0}^2 b_1 c_1 \\ &- 2 b_2 c_3 {a_0}^2 {c_1}^3 b_1 + 2 b_2 {c_3}^2 b_0 {a_0}^2 {c_1}^2 - 2 b_2 {a_2}^2 {c_0}^3 c_2 b_1 + 3 a_2 c_3 c_0 b_0 {c_2}^2 a_1 b_1 \\ &- 2 a_2 {c_3}^2 c_0 b_0 c_2 a_0 b_1 + 3 b_2 c_3 c_0 a_1 {c_1}^2 b_1 a_0 - 2 b_2 a_2 c_0 a_0 b_1 {c_1}^2 c_2 \\ &- 2 b_2 {c_3}^2 c_0 b_0 a_1 a_0 c_1 + 3 b_2 a_2 {c_0}^2 c_1 b_1 a_1 c_2 - 2 b_2 a_2 c_3 {c_0}^2 a_0 b_1 c_1 \\ &- 2 b_2 a_2 c_3 {c_0}^2 b_0 a_1 c_2 + 3 {c_3}^2 b_0 c_2 a_0 c_1 b_1 a_1 - 2 a_2 c_3 b_0 c_1 {c_2}^2 a_0 b_1 \\ &- 2 b_2 a_2 c_0 b_0 a_1 {c_2}^2 c_1 - 2 b_2 c_3 b_0 {c_1}^2 c_2 a_1 a_0 - 2 b_2 a_2 {c_3}^2 {c_0}^2 b_0 a_0 = 0, \end{aligned}$$

$$\begin{aligned} &(-a_2 b_2 {c_0}^2 c_1 - {c_3}^2 b_0 a_0 c_1 + a_2 b_0 {c_2}^2 c_1 - b_2 {c_1}^3 a_0 + {c_3}^2 a_1 c_0 b_0 + b_2 {c_1}^2 c_0 a_1 \\ &- 2 c_3 a_2 c_0 b_0 c_2 + c_3 a_2 {c_0}^2 b_1 + c_3 {c_1}^2 b_1 a_0 - c_3 c_1 b_1 c_0 a_1) y + a_0 {c_3}^2 c_0 b_0 \\ &- b_2 c_0 {c_1}^2 a_0 + b_0 a_1 {c_2}^2 c_1 + b_2 {c_0}^2 c_1 a_1 + a_0 b_1 {c_1}^2 c_2 - a_1 b_1 c_0 c_1 c_2 \\ &+ a_2 c_2 {c_0}^2 b_1 - 2 b_0 c_3 c_2 a_0 c_1 - a_2 b_2 {c_0}^3 - a_2 {c_2}^2 c_0 b_0 = 0, \end{aligned}$$

$$\begin{aligned} &(-b_0 {c_3}^2 + c_1 b_1 c_3 - b_2 {c_1}^2) y + (b_2 c_0 c_3 - b_2 c_1 c_2) z - b_2 c_1 c_0 - b_0 c_2 c_3 \\ &\qquad + c_3 b_1 c_0 = 0. \end{aligned}$$

The first equation is of degree 16 in $x$ since all the $a_i, b_j, c_k$ are polynomials in $x$ of degree 2 or less. The second equation is linear in $y$, and the third is linear in $z$, so the problem has 16 isolated solutions for $\{x, y, z\}$ at most. This coincides with that mentioned in [23].

*Rotation Around a Fixed Point.* Let $\{p_1, p_2, p_3\}$ be a triangle on a rigid body $B$. When $B$ rotates around a fixed point, the triangle goes to a new position, as $p_1 \mapsto p_4$, $p_2 \mapsto p_5$, $p_3 \mapsto p_6$. Our question is how to locate the position of the fixed point, provided all the distances between $p_i$ and$p_j$ for $1 \le i < j \le 6$. This originates from a problem proposed by S. H. Xia (Private Communication, 2007) for locating human joint position.

Let $p_7$ stand for the fixed point, and accordingly set

$$\begin{aligned} x &= g(p_1, p_7) = g(p_4, p_7), \\ y &= g(p_2, p_7) = g(p_5, p_7), \\ z &= g(p_3, p_7) = g(p_6, p_7), \end{aligned}$$

where $x$, $y$, $z$ are to be determined.

Take quartuple $\{p_1, p_2, p_3, p_4\}$ as the reference tetrahedron. This coordinate system yields a complete set of constraint equations consisting of 3 equations,

$$\begin{cases} D_{5,6}(p_1, p_2, p_3, p_4, p_7, p_5) = 0, \\ D_{5,6}(p_1, p_2, p_3, p_4, p_7, p_6) = 0, \\ D(p_1, p_2, p_3, p_4, p_7) = 0, \end{cases}$$

that is,

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x & 1 \\ g_{12} & 0 & g_{23} & g_{24} & y & 1 \\ g_{13} & g_{23} & 0 & g_{34} & z & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x & 1 \\ g_{15} & g_{25} & g_{35} & g_{45} & y & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0, \tag{6.10}$$

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x & 1 \\ g_{12} & 0 & g_{23} & g_{24} & y & 1 \\ g_{13} & g_{23} & 0 & g_{34} & z & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x & 1 \\ g_{16} & g_{26} & g_{36} & g_{46} & z & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0, \tag{6.11}$$

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x & 1 \\ g_{12} & 0 & g_{23} & g_{24} & y & 1 \\ g_{13} & g_{23} & 0 & g_{34} & z & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x & 1 \\ x & y & z & x & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0. \tag{6.12}$$

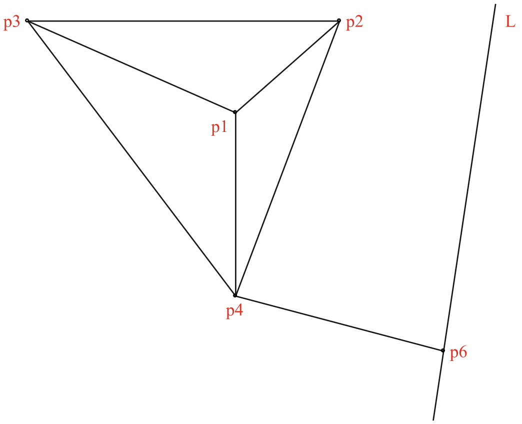


**Fig. 6.4** Convert the 4p1L problem into a point-plane configuration

Equations (6.10) and (6.11) are linear on $x$, $y$, $z$, while Eq. (6.12) is quadratic. The Bezout bound [3] is 2, so the system has two solutions at most. However, there is still possibility to improve the bound. Solving (6.10) and (6.11) for $\{y,z\}$, obtain $\{y = l_1(x),\ z = l_2(x)\}$ where $l_1$, $l_2$ are linear on $x$. Substituting that into Eq. (6.12) for $y$, $z$,

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x & 1 \\ g_{12} & 0 & g_{23} & g_{24} & l_1 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & l_2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x & 1 \\ x & l_1 & l_2 & x & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0, \tag{6.13}$$

we obtained an equation in single unknown $x$. By expanding it, we find Eq. (6.13) to be linear on $x$ actually, as it happens. So, the system $\{$(6.10), (6.11), (6.12)$\}$ has an unique isolated solution at most.

*The 4p1L Problem.* Consider a spatial configuration of points $p_1,\cdots,p_4$ and a straight line $L$, given the distances between the points and the distances from every $p_i$ to $L$, distance$(p_i,L) = d_i$, for $i = 1,\ldots,4$. In other words, draw lines tangent to 4 given spheres; see [10, 12, 16]. Since $(p_1,\cdots,p_4,L)$ is not a point-plane configuration, we consider a new one instead. Through point $p_4$ draw a plane $p_5$ perpendicular to $L$, and assume $p_5$ intersects $L$ at point $p_6$. Now we have a point-plane configuration, $p_1,\cdots,p_6$ (Fig. 6.4). By $x_1,x_2,x_3$ denote the unknown signed distances from $p_1,p_2,p_3$ to plane $p_5$, respectively, that is,

$$x_1 = g(p_1,p_5),\ x_2 = g(p_2,p_5),\ x_3 = g(p_3,p_5).$$

It follows by Pythagorean Theorem that

$$g(p_1,p_6) = x_1^2 + d_1^2, \quad g(p_2,p_6) = x_2^2 + d_2^2, \quad g(p_3,p_6) = x_3^2 + d_3^2.$$

The quartuple coordinate system $\{p_1, p_2, p_3, p_4\}$ yields a complete set of constraints consisting of three equations:

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & 0 & 1 \\ x_1 & x_2 & x_3 & 0 & -\frac{1}{2} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2+d_1^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2+d_2^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2+d_3^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & d_4^2 & 1 \\ x_1^2+d_1^2 & x_2^2+d_2^2 & x_3^2+d_3^2 & d_4^2 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & 0 & 1 \\ x_1^2+d_1^2 & x_2^2+d_2^2 & x_3^2+d_3^2 & d_4^2 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0.$$

The Bezout bound of the system is $2 \times 4 \times 3 = 24$. Observing that if $\{x_1, x_2, x_3\}$ is a solution, then, so is $\{-x_1, -x_2, -x_3\}$; the problem has essentially at most 12 isolated solutions. This coincides with Macdonald, Pach and Theobald's result [16] that shows the bound to be tight.

The numerical solution of this problem was computed by a new method [20] employing a redundant system of ten equations in six unknowns. The redundant equations are used to speed up the resolution process. Contrastively, the present method usually receives fewer equations and fewer unknowns: in this example, three equations in three unknowns only.

*Draw Lines with Equal Distances to 5 Points.* Given points $p_1, p_2, p_3, p_4, p_5$ in $E^3$, let $g_{ij}$ or $g(p_i, p_j)$ be the squared distances between $p_i$ and $p_j$ for $i, j = 1, \ldots, 5$ as defined formerly, and $d$ the unknown distance from every $p_i$ to a line L which is to be determined. In other words, draw a circular cylinder through given five points; see [2].

Since $(p_1, \cdots, p_5, L)$ is not a point-plane configuration, we consider a new one instead. Through point $p_5$ draw a plane $p_6$ perpendicular to $L$, and assume $p_6$ intersects $L$ at point $p_7$. Now we have a point-plane configuration, $p_1, \cdots, p_7$.

By $x_1, x_2, x_3, x_4$ denote the unknown signed distances from $p_1, p_2, p_3, p_4$ to plane $p_6$, respectively, that is,

$$x_1 = g(p_1, p_6), x_2 = g(p_2, p_6), x_3 = g(p_3, p_6), x_4 = g(p_4, p_6).$$

It follows by Pythagorean Theorem that

$$\begin{aligned} g(p_1, p_7) &= x_1^2 + d^2, g(p_2, p_7) = x_2^2 + d^2, \\ g(p_3, p_7) &= x_3^2 + d^2, g(p_4, p_7) = x_4^2 + d^2. \end{aligned}$$

The quartuple coordinate system $\{p_1, p_2, p_3, p_4\}$ yields a set of five equations, $\{f_1 = 0, \cdots, f_5 = 0\}$, where

$$f_1 = D_{5,6}(p_1, p_2, p_3, p_4, p_5, p_6) = \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & g_{15} & 1 \\ g_{12} & 0 & g_{23} & g_{24} & g_{25} & 1 \\ g_{13} & g_{23} & 0 & g_{34} & g_{35} & 1 \\ g_{14} & g_{24} & g_{34} & 0 & g_{45} & 1 \\ x_1 & x_2 & x_3 & x_4 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix},$$

$$\begin{aligned} f_2 &= D_{5,6}(p_1, p_2, p_3, p_4, p_7, p_5) \\ &= \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2 + d^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2 + d^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2 + d^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4^2 + d^2 & 1 \\ g_{15} & g_{25} & g_{35} & g_{45} & d^2 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4^2 & 1 \\ g_{15} & g_{25} & g_{35} & g_{45} & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix}, \end{aligned}$$

$$\begin{aligned} f_3 &= D_{5,6}(p_1, p_2, p_3, p_4, p_7, p_6) \\ &= \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2 + d^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2 + d^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2 + d^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4^2 + d^2 & 1 \\ x_1 & x_2 & x_3 & x_4 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix} = \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4^2 & 1 \\ x_1 & x_2 & x_3 & x_4 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix}, \end{aligned}$$

$$f_4 = D(p_1, p_2, p_3, p_4, p_6) = \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4 & 1 \\ x_1 & x_2 & x_3 & x_4 & -\frac{1}{2} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix},$$

$$f_5 = D(p_1,p_2,p_3,p_4,p_7) = \begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & x_1^2+d^2 & 1 \\ g_{12} & 0 & g_{23} & g_{24} & x_2^2+d^2 & 1 \\ g_{13} & g_{23} & 0 & g_{34} & x_3^2+d^2 & 1 \\ g_{14} & g_{24} & g_{34} & 0 & x_4^2+d^2 & 1 \\ x_1^2+d^2 & x_2^2+d^2 & x_3^2+d^2 & x_4^2+d^2 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{vmatrix}.$$

The system $\mathscr{F} = \{f_1 = 0, f_2 = 0, f_3 = 0, f_4 = 0\}$ is enough for determining the unknowns $\{x_1,x_2,x_3,x_4\}$, and $d$ can be uniquely determined by the values of $x_1^2$, $x_2^2$, $x_3^2$, $x_4^2$ as $f_5 = 0$. The Bezout bound of $\mathscr{F}$ is $1 \times 2 \times 3 \times 2 = 12$. Observing that if $\{x_1,x_2,x_3,x_4\}$ is a solution, then, so is $\{-x_1,-x_2,-x_3,-x_4\}$; the problem has essentially at most six isolated solutions. This coincides with Daniel Lichtblau's result [22] that shows the bound to be tight.

*Construct a Spatial Quadrilateral.* For a spatial quadrilateral, $p_1p_2p_3p_4$, by $\mu$, denote the distance between lines $p_1p_2$ and $p_3p_4$, and $\nu$ denote the distance between lines $p_1p_4$ and $p_2p_3$. The problem is how to construct the quadrilateral with data $\mu$, $\nu$ and the edge lengths $p_1p_2$, $p_2p_3$, $p_3p_4$, $p_1p_4$.

As the same as in case of the last two examples, $\mu$ and $\nu$ are not standard distances between points or planes. The strategy we take is to supplement some geometric elements such that $\mu$ and $\nu$ become the standard distances on a certain point-plane configuration.

To do in this way, draw two planes $p_5$ and $p_6$ such that $p_5$ passes through points $p_3,p_4$ and parallels to line $p_1p_2$ and $p_6$ passes through points $p_2,p_3$ and parallels to line $p_1p_4$.

Consider the configuration $p_1,p_2,p_3,p_4,p_5,p_6$. The quartuple coordinate system $\{p_1,p_2,p_3,p_4\}$ yields a complete set of constraint equations,

$$\begin{cases} D(p_1,p_2,p_3,p_4,p_5) = 0, \\ D(p_1,p_2,p_3,p_4,p_6) = 0, \\ D_{5,6}(p_1,p_2,p_3,p_4,p_5,p_6) = 0. \end{cases}$$

Let $x = g(p_1,p_3),\ y = g(p_2,p_4)$. Observing that

$$g(p_1,p_5) = g(p_2,p_5) = \mu, \quad g(p_3,p_5) = g(p_4,p_5) = 0,$$
$$g(p_1,p_6) = g(p_4,p_6) = \nu, \quad g(p_2,p_6) = g(p_3,p_6) = 0,$$

we obtain that the first two are Cayley–Menger determinants,

$$\begin{vmatrix} 0 & g_{12} & x & g_{14} & \mu & 1 \\ g_{12} & 0 & g_{23} & y & \mu & 1 \\ x & g_{23} & 0 & g_{34} & 0 & 1 \\ g_{14} & y & g_{34} & 0 & 0 & 1 \\ \mu & \mu & 0 & 0 & -1/2 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

$$\begin{vmatrix} 0 & g_{12} & x & g_{14} & \nu & 1 \\ g_{12} & 0 & g_{23} & y & 0 & 1 \\ x & g_{23} & 0 & g_{34} & 0 & 1 \\ g_{14} & y & g_{34} & 0 & \nu & 1 \\ \nu & 0 & 0 & \nu & -1/2 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{vmatrix} = 0,$$

which are enough for determining the values of $x$ and $y$, so the third equation is no longer necessary for our purpose.

Making use of computer algebra to compute the characteristic set of that system, it appears the problem has 4 isolated solutions at most.

A further approach to the construction of tetrahedra can be found in [30] where prescribed are only the widths and heights, without any edge length provided.

Most of the material in this section is taken from [29]. See [27] for some more examples; some of which are investigated in [10] and [6] with different methods.

### 6.3.3 Coordinate Transformation

Given are two quartuple coordinate systems with reference tetrahedra $\{p_1, p_2, p_3, p_4\}$ and $\{p_5, p_6, p_7, p_8\}$, respectively, and $p_9$ a point or a oriented plane. If the coordinates of $p_5, p_6, p_7, p_8$ and $p_9$ in the coordinate system $\{p_1, p_2, p_3, p_4\}$ are provided, then the coordinate of $p_9$ in system $\{p_5, p_6, p_7, p_8\}$, that is, $(g(p_5, p_9), g(p_6, p_9), g(p_7, p_9), g(p_8, p_9))$, is well determined by solving the following linear equations:

$$D_{5,6}(p_1, p_2, p_3, p_4, p_5, p_9) = 0,$$
$$D_{5,6}(p_1, p_2, p_3, p_4, p_6, p_9) = 0,$$
$$D_{5,6}(p_1, p_2, p_3, p_4, p_7, p_9) = 0,$$
$$D_{5,6}(p_1, p_2, p_3, p_4, p_8, p_9) = 0,$$

respectively.

The coordinate transformation would be helpful to solving larger configuration which we need to divide into parts.

## 6.4 Nontypical Problems of Geometric Constraint Solving

In a typical geometric constraint problem we are asked to construct a point-line-plane configuration which satisfies a set of pairwise constraints between these points, lines and planes. A pairwise constraint may be one of the following items: point-point distance, line-line angle, line-line distance, plane-plane angle, point-line distance, point-plane distance, line-plane angle and line-plane distance (in case of parallel).

A nontypical problem we refer to that whereof the constraints include more kinds of geometric invariants: area, volume, circumradius and so on. It seems that the distance coordinates can rarely help to nontypical case, but in a coordinate-free manner, distance geometry may still be made use of.

In [17], a question proposed by M. Mazur asked whether or not a tetrahedron is uniquely determined by its volume, circumradius and face areas. P. Lisoněk and R. B. Israel [15] gave a negative answer to this question by constructing two or more tetrahedra that share the same volume, circumradius and facet areas and suggested to discuss whether, for *any* positive real constants $V,R,A_1,A_2,A_3,A_4$, there are *finitely many* tetrahedra, all having these values as their respective metric invariants.

Using the previous notation, $D(p_1,p_2,p_3,p_4)$ stands for the Cayley–Menger determinant of tetrahedron $p_1p_2p_3p_4$,

$$D(p_1,p_2,p_3,p_4)=\begin{vmatrix} 0 & g_{12} & g_{13} & g_{14} & 1\\ g_{12} & 0 & g_{23} & g_{24} & 1\\ g_{13} & g_{23} & 0 & g_{34} & 1\\ g_{14} & g_{24} & g_{34} & 0 & 1\\ 1 & 1 & 1 & 1 & 0 \end{vmatrix},$$

where $g_{ij}=g(p_i,p_j)=p_ip_j^2$ for $1\le i<j\le 4$.

It is well known in distance geometry that the formulae [7] connecting the volume $V$, the circumradius $R$ and the facet areas $A_1,A_2,A_3,A_4$ of a tetrahedron are given by

$$\begin{aligned} V^2 &= \frac{1}{288}D(p_1,p_2,p_3,p_4),\\ R^2 &= -\frac{D_{5,5}(p_1,p_2,p_3,p_4)}{2D(p_1,p_2,p_3,p_4)},\\ A_i^2 &= -\frac{1}{16}D_{i,i}(p_1,p_2,p_3,p_4),\ \text{for } i=1,2,3,4, \end{aligned}$$

where $D_{i,j}(p_1,p_2,p_3,p_4)$ stands for the $(i,j)$th cofactor of $D(p_1,p_2,p_3,p_4)$ as before. Set

$$\begin{aligned} f_1 &= D(p_1,p_2,p_3,p_4)-288V^2,\\ f_2 &= 2R^2D(p_1,p_2,p_3,p_4)+D_{5,5}(p_1,p_2,p_3,p_4), \end{aligned}$$

$$\begin{aligned}
f_3 &= D_{1,1}(p_1,p_2,p_3,p_4)+16A_1^2,\\
f_4 &= D_{2,2}(p_1,p_2,p_3,p_4)+16A_2^2,\\
f_5 &= D_{3,3}(p_1,p_2,p_3,p_4)+16A_3^2,\\
f_6 &= D_{4,4}(p_1,p_2,p_3,p_4)+16A_4^2.
\end{aligned}$$

Now, Lisoněk and Israel's question is equivalent to whether the polynomial equation system $\mathscr{F} = \{f_1 = 0, \cdots, f_6 = 0\}$ has always finitely many real solutions for the unknowns $\{g_{12}, g_{13}, g_{14}, g_{23}, g_{24}, g_{34}\}$?

A negative answer was given in [28] by showing the following assertion:

**Proposition 6.1.** *The system $\mathscr{F}$ has a manifold solution of positive dimension if*

$$V = 441,\ \ R = \frac{43\sqrt{3}}{6},\ \ A_1 = 84\sqrt{3},\ \ A_2 = A_3 = A_4 = 63\sqrt{3}.$$

Define several polynomials

$$\begin{aligned}
H(x,y) &= 3\,(1-x)(17-18y)(1+3x+3y)-9x^2-3x-37,\\
G_{12}(x,y) &= 324\,(1-y)\,(1+y),\\
G_{13}(x,y) &= 324\,(1-x)\,(1+x),\\
G_{14}(x,y) &= (29-18x-18y)\,(7+18x+18y),\\
G_{23}(x,y) &= 36\,(7-3x-3y)\,(1+3x+3y),\\
G_{24}(x,y) &= (17-18x)\,(31+18x),\\
G_{34}(x,y) &= (17-18y)\,(31+18y),
\end{aligned}$$

and a semi-algebraic set

$$H_0 = \{\,(x,y) \in \mathrm{R}^2 \mid H(x,y) = 0,\ |x| < 1,\ |y| < 1\}.$$

One can see that the polynomial $H(x,y)$ is symmetric in $x$ and $y$ by expanding it. The shape of the semi-algebraic set $H_0$ looks like a UFO, as shown in Fig. 6.5. Proposition 6.1 means that there is a family of infinitely many tetrahedra which share the same volume, circumradius and face areas. This presents a negative answer to Lisoněk and Israel's question. To prove the proposition, we need to verify the following lemmas.

**Lemma 6.1.** *It holds for all $(\xi, \eta) \in H_0$ that*

$$-\frac{1}{2} < \xi < \frac{3}{4},\ -\frac{1}{2} < \eta < \frac{3}{4},$$

*where $H_0$ is defined as above.*

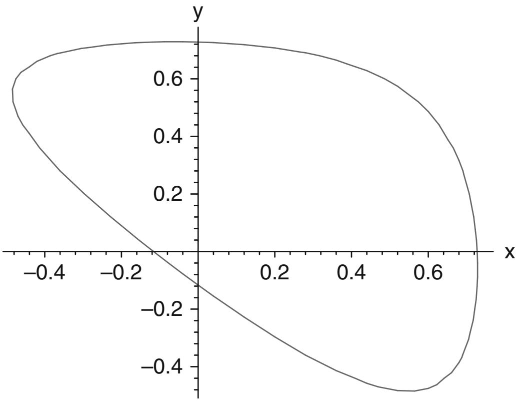


**Fig. 6.5** The shape of the semi-algebraic set $H_0$ looks like a UFO

*Proof.* The set $H_0$ is not empty since

$$\left(\frac{1}{4}+\frac{\sqrt{2}}{3},\ \frac{1}{4}-\frac{\sqrt{2}}{3}\right)\in H_0.$$

The curve $H(x,y)=0$ does not intersect either of the lines $x=1, x=-1, y=1, y=-1$ because none of $H(1,y), H(-1,y), H(x,1), H(x,-1)$ have a real zero. So $H_0$ is compact. For all $(\xi,\eta)\in H_0$, the maximum and minimum of $\xi$ both are real zeros of the polynomial obtained by eliminating $\eta$ from $H(\xi,\eta)$ and $\frac{\partial}{\partial\eta}H(\xi,\eta)$, that is,

$$324\,\xi^3+576\,\xi^2-275\,\xi-233,$$

which has only two real zeros in $(-1,\,1)$ as follows:

$$-0.48681\cdots,\quad 0.73069\cdots.$$

So we have $-\frac{1}{2}<\xi<\frac{3}{4}$; analogously, $-\frac{1}{2}<\eta<\frac{3}{4}$. □

**Lemma 6.2.** *It holds for all $(\xi,\eta)\in H_0$ that*

$$G_{ij}(\xi,\eta)>0,\quad (1\le i<j\le 4)$$

*where $H_0$ and $G_{ij}$ are defined as above.*

*Proof.* The first two inequalities

$$G_{12}(\xi,\eta)=324\,(1-\eta)\,(1+\eta)>0,$$
$$G_{13}(\xi,\eta)=324\,(1-\xi)\,(1+\xi)>0$$

are trivial since $|\xi|<1$ and $|\eta|<1$. The last two,

$$G_{24}(\xi,\eta)=(17-18\,\xi)\,(31+18\,\xi)>0,$$
$$G_{34}(\xi,\eta)=(17-18\,\eta)\,(31+18\,\eta)>0,$$

also hold because $|\xi| < \frac{3}{4}$, $|\eta| < \frac{3}{4}$, by Lemma 6.1. Next, observing that $-9x^2 - 3x - 37$, the sum of the last three terms of $H(x,y)$ is always negative; whenever $H(x,y) = 0$, the following inequality holds:

$$3\,(1-x)(17-18y)(1+3x+3y) > 0.$$

We have known $(1-\xi)(17-18\eta) > 0$ for $(\xi,\eta) \in H_0$; hence $1+3\xi+3\eta > 0$, and then

$$G_{23}(\xi,\eta) = 36\,(7-3\xi-3\eta)(1+3\xi+3\eta) > 0.$$

Furthermore, $1+3\xi+3\eta > 0$ implies $7+18\xi+18\eta > 0$ and that $|\xi| < \frac{3}{4}$ and $|\eta| < \frac{3}{4}$ imply $29-18\xi-18\eta > 0$, so we have

$$G_{14}(\xi,\eta) = (29-18\xi-18\eta)(7+18\xi+18\eta) > 0.$$

This completes the proof of Lemma 6.2. □

**Lemma 6.3.** *Given* $V = 441$, $R = \frac{43\sqrt{3}}{6}$, $A_1 = 84\sqrt{3}$, $A_2 = A_3 = A_4 = 63\sqrt{3}$, *the assignment* $\{g_{ij} = G_{ij}(\xi,\eta),\ (1 \le i < j \le 4)\}$ *solves the system* $\mathscr{F}$ *for every* $(\xi,\eta) \in H_0$.

This is simple, just recall $H(\xi,\eta) = 0$ on doing substitution.

*Proof of Proposition 6.1.* Denote the Cartesian coordinates to be determined of the vertices $p_1, p_2, p_3, p_4$ by

$$(0,0,0),\ (x_1,0,0),\ (x_2,x_3,0),\ (x_4,x_5,x_6),$$

respectively. Assigned ${p_ip_j}^2 = g_{ij} = G_{ij}(\xi,\eta)$, we have

$$\begin{aligned}
&x_1^2 = 324\,(1-\eta)(1+\eta)\\
&x_2^2 + x_3^2 = 324\,(1-\xi)(1+\xi)\\
&x_4^2 + x_5^2 + x_6^2 = (29-18\xi-18\eta)(7+18\xi+18\eta)\\
&(x_1-x_2)^2 + x_3^2 = 36\,(7-3\xi-3\eta)(1+3\xi+3\eta)\\
&(x_1-x_4)^2 + x_5^2 + x_6^2 = (17-18\xi)(31+18\xi)\\
&(x_2-x_4)^2 + (x_3-x_5)^2 + x_6^2 = (17-18\eta)(31+18\eta).
\end{aligned}$$

Solve the equation system for $\{x_1, x_2, \cdots, x_6\}$ and receive a manifold solution:

$$\begin{aligned}
x_1 &= 18\sqrt{(1-\eta)(1+\eta)},\\
x_2 &= \frac{11-18\xi-18\eta+18\xi\eta}{\sqrt{(1-\eta)(1+\eta)}},
\end{aligned}$$

$$\begin{aligned}
x_3 &= \frac{7\sqrt{3}}{\sqrt{(1-\eta)(1+\eta)}},\\
x_4 &= \frac{18\xi + 11\eta - 18\xi\eta - 18\eta^2}{\sqrt{(1-\eta)(1+\eta)}},\\
x_5 &= \frac{7\sqrt{3}\,\eta}{\sqrt{(1-\eta)(1+\eta)}},\\
x_6 &= 7\sqrt{3},
\end{aligned}$$

where $(\xi, \eta)$ ranges over

$$H_0 = \{\,(x,y) \in \mathrm{R}^2 \mid H(x,y) = 0,\ |x| < 1,\ |y| < 1\}.$$

Thus, we obtain a family of tetrahedra $T_{(\xi,\eta)}$ with vertices:

$$\begin{aligned}
p_1 &= (0,0,0),\\
p_2 &= (18\sqrt{(1-\eta)(1+\eta)}, 0, 0),\\
p_3 &= \left(\frac{11 - 18\xi - 18\eta + 18\xi\eta}{\sqrt{(1-\eta)(1+\eta)}}, \frac{7\sqrt{3}}{\sqrt{(1-\eta)(1+\eta)}}, 0\right),\\
p_4 &= \left(\frac{18\xi + 11\eta - 18\xi\eta - 18\eta^2}{\sqrt{(1-\eta)(1+\eta)}}, \frac{7\sqrt{3}\,\eta}{\sqrt{(1-\eta)(1+\eta)}}, 7\sqrt{3}\right),
\end{aligned}$$

which share the same volume, circumradius and face areas,

$$441,\ \frac{43\sqrt{3}}{6},\ 84\sqrt{3},\ 63\sqrt{3},\ 63\sqrt{3},\ 63\sqrt{3},$$

according to Lemma 6.3. Now, Proposition 6.1 is proved. □

A conjecture was proposed [28] that for given six positive numbers $V, R, A_1, A_2, A_3, A_4$ where $A_1, A_2, A_3, A_4$ are pairwise distinct, there are at most eight tetrahedra, all having these values as their volume, circumradius and four facet areas, respectively.

## References

1. Blumenthal, L.M.: Theory and Applications of Distance Geometry, 2nd edn. Chelsea, New York (1970)
2. Bottema, O., Veldkamp, G.R.: On the lines in space with equal distances to $n$ given points. Geometriae Dedicata **6**, 121–129 (1977)

3. Cox, D.A., Little, J.B., O'Shea, D.: Ideals, varieties, and algorithms: An introduction to computational algebraic geometry and commutative algebra. Springer (1997)
4. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
5. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global. Optim. **26**, 321–333 (2003)
6. Gao, X.S., Hoffmann, C.M., Yang, W.Q.: Solving spatial basic geometric constraint configurations with locus intersection. Comput. Aided. Des. **36**(2), 111–122 (2004)
7. Gonzalez, O., Maddocks, J.H., Smutny, J.: Curves, circles, and spheres. Contemp. Math. (AMS) **304**, 195–215 (2002)
8. Gröbner, W.: Matrizenrechnung, Bibliographisches Institut AG, Mannh-eim, F.R.G., 119–126 (1966)
9. Hodge, W.V., Pedoe, D.: Methods of Algebraic Geometry, vol. 1. Cambridge Univ Press, Cambridge (1953)
10. Hoffmann, C.M., Yuan, B.: On spatial constraint approaches. In: Richter-Gebert, J., Wang, D. (eds.) Automated Deduction in Geometry 2000, LNAI 2061, Springer, 1–15 (2001)
11. Jeong, J.W., Kim, S.H., Kwak, Y.K.: Kinematics and workspace analysis of a parallel wire mechanism for measuring a robot pose. Mech. Mach. Theor. **34**(6), 825–841 (1999)
12. Larman, D.: Problem posed in the Problem Session of the DIMACS Workshop on Arrangements. Rutgers University, New Brunswick (1990)
13. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2011)
14. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. arXiv:1205.0349v1 (2012)
15. Lisoněk, P., Israel, R.B.: Metric invariants of tetrahedra via polynomial elimination. In: ACM Conference Proceedings, 2000 International Symposium on Symbolic and Algebraic Computation (ISSAC 2000), New York, 217–219 (2000)
16. Macdonald, I.G., Pach, J., Theobald, T.: Common tangents to four unit balls in $R^3$. Discrete Comput. Geom. **26**(1), 1–17 (2001)
17. Mazur, M.: Problem 10717. The Am. Math. Mon. **106**(2), 167 (1999)
18. Michelucci, D., Foufou, S.: Using Cayley-Menger determinants for geometric constraint solving. In: ACM Symposium on Solid Modelling and Applications (2004)
19. Muir, T.: A treatise on the theory of determinants, revised by W.H. Metzler, Longmans, Green, New York, 166–170 (1933)
20. Porta, J.M., Ros, L., Thomas, F., Torras, C.: A Branch-and-Prune solver for distance constraints. IEEE Trans. Robot. **21**(2), 176–187 (2005)
21. Sippl, M.J., Scheraga, H.A.: Cayley-Menger coordinates. In: Proceedings of the National Academy of Sciences of the USA, vol. 83, pp. 2283–2287 (1986)
22. Sottile, F.: An excursion from enumerative geometry to solving systems of polynomial equations with Macaulay 2. In: Eisenbud, D. (eds.) Computations in Algebraic Geometry with Macaulay 2, pp. 101–129, Springer (2001)
23. Thomas, F., Ottaviano, E., Ros, L., Ceccarelli, M.: Coordinate-free formulation of a 3-2-1 wire-based tracking device using Cayley-Menger determinants. IEEE International Conference on Robotics and Automation (ICRA03), 355–561 (2003)
24. Yang, L., Zhang, J.Z. : The concept of the rank of an abstract distance space (in Chinese). J. China Univ. Sci. Tech. **10**(4), 52–65, MR 84j:52019 (1980)
25. Yang, L., Zhang, J.Z.: A class of geometric inequalities on finite points (in Chinese). Acta Math. Sin. **23**(5), 740–749, MR 83f:51027 (1980)
26. Yang, L., Zhang, J.Z.: Metric equations in geometry and their applications. Research Report IC/89/281. International Centre for Theoretical Physics, Trieste (1989)
27. Yang, L.: Distance coordinates used in geometric constraint solving. In: Winkler, F. (ed.) "Automated Deduction in Geometry 2002", LNAI 2930, Springer, 216–229 (2004)
28. Yang, L., Zeng, Z.B.: An open problem on metric invariants of tetrahedra. In: ACM Conference Proceedings, ISSAC 2005, New York, 362–364 (2005)

29. Yang, L.: Solving spatial constraints with global distance coordinate system. Int. J. Comput. Geom. Appl. **16**(5–6), 533–548 (2006)
30. Yang, L., Zeng, Z.B.: Constructing a tetrahedron with prescribed heights and widths. Automated Deduction in Geometry 2006, LNAI 4869, pp. 203–211, Springer (2007)
31. Zhang, J.Z., Yang, L., Yang, X.C.: The realization of elementary configurations in Euclidean space. Science China A**37**(1), 15–26 (1994)

# Chapter 7
# A Topological Interpretation of the Walk Distances

**Pavel Chebotarev and Michel Deza**

**Abstract** The walk distances in graphs have no direct interpretation in terms of walk weights, since they are introduced via the *logarithms* of walk weights. Only in the limiting cases where the logarithms vanish such representations follow straightforwardly. The interpretation proposed in this chapter rests on the identity $\ln\det B = \operatorname{tr}\ln B$ applied to the cofactors of the matrix $I-tA$, where $A$ is the weighted adjacency matrix of a weighted multigraph and $t$ is a sufficiently small positive parameter. In addition, this interpretation is based on the power series expansion of the logarithm of a matrix. Kasteleyn (Graph theory and crystal physics. In: Harary, F. (ed.) Graph Theory and Theoretical Physics. Academic Press, London, 1967) was probably the first to apply the foregoing approach to expanding the determinant of $I-A$. We show that using a certain linear transformation the same approach can be extended to the cofactors of $I-tA$, which provides a topological interpretation of the walk distances.

**Keywords** Graph distances • Walk distances • Transitional measure • Network

P. Chebotarev (✉)
Institute of Control Sciences of the Russian Academy of Sciences,
65 Profsoyuznaya Street, Moscow 117997, Russia
e-mail: chv@member.ams.org

M. Deza
Laboratoire de Geometrie Appliquee, LIGA, Ecole Normale Superieure,
45, rue d'Ulm, F-75230, Paris, Cedex 05, France
e-mail: Michel.Deza@ens.fr

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_7,

## 7.1 Introduction

The walk distances for graph vertices were proposed in [4] and studied in [5]. Along with their modifications they generalize [5] the logarithmic forest distances [3], resistance distance, shortest path distance, and the weighted shortest path distance. The walk distances are graph-geodetic: for a distance[1] $d(i,j)$ in a graph $G$ this means that $d(i,j)+d(j,k)=d(i,k)$ if and only if every path in $G$ connecting $i$ and $k$ visits $j$.

It is well known that the resistance distance between two adjacent vertices in a tree is equal to 1. In contrast to this, the walk distances take into account the centrality of vertices. For example, any walk distance between two central adjacent vertices in a path turns out [5] to be less than that between two peripheral adjacent vertices. This property may be desirable in some applications including machine learning, mathematical chemistry, the analysis of social and biological networks, etc.

In the chapter, we obtain a topological interpretation of the simplest walk distances. Such an interpretation is not immediate from the definition, since the walk distances are introduced via the *logarithms* of walk weights. Only in the limiting cases where the logarithms vanish such representations follow straightforwardly [5]. The interpretation we propose rests on the identity $\ln\det B = \operatorname{tr}\ln B$ applied to the cofactors of the matrix $I-tA$, where $A$ is the weighted adjacency matrix of a weighted multigraph and $t$ is a sufficiently small positive parameter. In addition, it is based on the power series expansion of the logarithm of a matrix. We do not employ these identities explicitly; instead, we make use of a remarkable result by Kasteleyn [13] based on them. More specifically, Kasteleyn obtained an expansion of the determinant of $I-A$ and the logarithm of this determinant. We show that using a certain linear transformation the same approach can be extended to the cofactors of $I-tA$, which provides a topological interpretation of the walk distances.

## 7.2 Notation

In the graph definitions we mainly follow [10]. Let $G$ be a weighted multigraph (a weighted graph where multiple edges are allowed) with vertex set $V(G)=V$, $|V|=n>2$, and edge set $E(G)$. Loops are allowed; we assume that $G$ is connected. For brevity, we will call $G$ a *graph*. For $i,j\in V(G)$, let $n_{ij}\in\{0,1,\ldots\}$ be the number

[1] In this chapter, a *distance* is assumed to satisfy the axioms of metric.

of edges incident to both $i$ and $j$ in $G$; for every $q \in \{1,\dots,n_{ij}\}$, $w_{ij}^q > 0$ is the *weight* of the $q$th edge of this type. Let

$$a_{ij} = \sum_{q=1}^{n_{ij}} w_{ij}^q \tag{7.1}$$

(if $n_{ij} = 0$, we set $a_{ij} = 0$) and $A = (a_{ij})_{n\times n}$; $A$ is the symmetric *weighted adjacency matrix* of $G$. In what follows, all matrix entries are indexed by the vertices of $G$. This remark is essential when submatrices are considered: say, "the $i$th column" of a submatrix of $A$ means "the column corresponding to the vertex $i$ of $G$" rather than just the "column number $i$."

By the *weight* of a graph $G$, $w(G)$, we mean the product of the weights of all its edges. If $G$ has no edges, then $w(G) = 1$. The weight of a set $\mathscr{S}$ of graphs, $w(\mathscr{S})$, is the total weight (the sum of the weights) of its elements; $w(\varnothing) = 0$.

For $v_0, v_m \in V(G)$, a $v_0 \to v_m$ *walk* in $G$ is an arbitrary alternating sequence of vertices and edges $v_0, \mathrm{e}_1, v_1, \dots, \mathrm{e}_m, v_m$ where each $\mathrm{e}_i$ is a $(v_{i-1}, v_i)$ edge. The *length* of a walk is the number $m$ of its edges (including loops and repeated edges). The *weight* of a walk is the product of the $m$ weights of its edges. The weight of a set of walks is the total weight of its elements. By definition, for any vertex $v_0$, there is one $v_0 \to v_0$ walk $v_0$ of length 0; its weight is 1.

We will need some special types of walks. A *hitting* $v_0 \to v_m$ *walk* is a $v_0 \to v_m$ walk containing only one occurrence of $v_m$. A $v_0 \to v_m$ walk is called *closed* if $v_m = v_0$ and *open* otherwise. The *multiplicity* of a closed walk is the maximum $\mu$ such that the walk is a $\mu$-fold repetition of some walk.

We say that two closed walks of nonzero length are *phase twins* if the edge sequence $\mathrm{e}_1, \mathrm{e}_2, ,\dots, \mathrm{e}_m$ of the first walk can be obtained from the edge sequence $\mathrm{e}'_1, \mathrm{e}'_2, ,\dots, \mathrm{e}'_m$ of the second one by a cyclic shift. For example, the walks $v_0$, $\mathrm{e}_1$, $v_1$, $\mathrm{e}_2$, $v_2$, $\mathrm{e}_3$, $v_0$ and $v_2$, $\mathrm{e}_3$, $v_0$, $\mathrm{e}_1$, $v_1$, $\mathrm{e}_2$, $v_2$ are phase twins. A *circuit* [11, 13] in $G$ is any complete equivalence class of phase twins. The *multiplicity* of a circuit is the multiplicity of any closed walk it contains (all such walks obviously have the same multiplicity). A walk (circuit) whose multiplicity exceeds 1 is *periodic*.

Let $r_{ij}$ be the weight of the set $\mathscr{R}^{ij}$ of all $i \to j$ walks in $G$ provided that this weight is finite. $R = R(G) = (r_{ij})_{n\times n} \in \mathbb{R}^{n\times n}$ will be referred to as the *matrix of the walk weights* of $G$.

It was shown in [4] that if $R$ exists then it *determines a transitional measure in* $G$, that is, (i) it satisfies the transition inequality

$$r_{ij}\, r_{jk} \le r_{ik}\, r_{jj}, \quad i, j, k = 1, \dots, n \tag{7.2}$$

and (ii) $r_{ij}\, r_{jk} = r_{ik}\, r_{jj}$ if and only if every path from $i$ to $k$ visits $j$.

## 7.3 The Walk Distances

For any $t > 0$, consider the graph $tG$ obtained from $G$ by multiplying all edge weights by $t$. If the matrix of the walk weights of $tG$, $R_t = R(tG) = (r_{ij}(t))_{n\times n}$, exists, then[2]

$$R_t = \sum_{k=0}^{\infty}(tA)^k = (I - tA)^{-1}, \tag{7.3}$$

where $I$ denotes the identity matrix of appropriate dimension.

By assumption, $G$ is connected, while its edge weights are positive, so $R_t$ is also positive. Apply the logarithmic transformation to the entries of $R_t$, namely, consider the matrix

$$H_t = \overrightarrow{\ln R_t}, \tag{7.4}$$

where $\overrightarrow{\varphi(S)}$ stands for elementwise operations, i.e., operations applied to each entry of a matrix $S$ separately. Finally, consider the matrix

$$D_t = \frac{1}{2}(h_t\mathbf{1}^{\mathrm{T}} + \mathbf{1}h_t^{\mathrm{T}} - H_t - H_t^{\mathrm{T}}), \tag{7.5}$$

where $h_t$ is the column vector containing the diagonal entries of $H_t$, $\mathbf{1}$ is the vector of ones of appropriate dimension, and $H_t^{\mathrm{T}}$, $h_t^{\mathrm{T}}$, and $\mathbf{1}^{\mathrm{T}}$ are the transposes of $H_t$, $h_t$, and $\mathbf{1}$. An alternative form of Eq. (7.5) is $D_t = (U_t + U_t^{\mathrm{T}})/2$, where $U_t = h_t\mathbf{1}^{\mathrm{T}} - H_t$, and its elementwise form is

$$d_{ij}(t) = \frac{1}{2}(h_{ii}(t) + h_{jj}(t) - h_{ij}(t) - h_{ji}(t)), \quad i, j \in V(G), \tag{7.6}$$

where $H_t = (h_{ij}(t))$ and $D_t = (d_{ij}(t))$. This is a standard transformation used to obtain a distance from a proximity measure (cf. the inverse covariance mapping in [8, Section 5.2] and the cosine law in [6]).

In the rest of this section, we present several known facts (lemmas) which will be of use in what follows, one simple example, and two remarks.

**Lemma 7.1 ([4]).** *For any connected $G$, if $R_t = (r_{ij}(t))$ exists, then the matrix $D_t = (d_{ij}(t))$ defined by Eqs. (7.3)–(7.5) determines a graph-geodetic distance $d_t(i,j) = d_{ij}(t)$ on $V(G)$.*

This enables one to give the following definition.

**Definition 7.1.** For a connected graph $G$, the *walk distances* on $V(G)$ are the functions $d_t(i,j)\colon V(G)\times V(G) \to \mathbb{R}$ and the functions, $d_t^{\mathrm{W}}(i,j)$, positively proportional to them, where $d_t(i,j) = d_{ij}(t)$ and $D_t = (d_{ij}(t))$ are defined by Eqs. (7.3)–(7.5).

[2] In the more general case of weighted *digraphs*, the $ij$-entry of the matrix $R_t - I$ is called the *Katz similarity* between vertices $i$ and $j$. Katz [14] proposed it to evaluate the social status taking into account all $i \to j$ paths. Among many other papers, this index was studied in [13, 23].

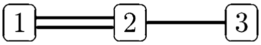


**Fig. 7.1** A multigraph $G$ on 3 vertices

*Example 7.1. For the multigraph G shown in Fig. 7.1, the weighted adjacency matrix is*

$$A = \begin{bmatrix} 0 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

*the matrix $R_{\frac{1}{3}}$ of the walk weights of $\frac{1}{3}G$ exists and has the form*

$$R_{\frac{1}{3}} = R\left(\frac{1}{3}G\right) = \left(r_{ij}\left(\frac{1}{3}\right)\right) = \frac{1}{4}\begin{bmatrix} 8 & 6 & 2 \\ 6 & 9 & 3 \\ 2 & 3 & 5 \end{bmatrix},$$

*and the computation (7.5) of the walk distances $d_t(i,j)$ with parameter $t = \frac{1}{3}$ yields*

$$D_{\frac{1}{3}} = \left(d_{ij}\left(\frac{1}{3}\right)\right) = \frac{1}{2}\begin{bmatrix} 0 & \ln 2 & \ln 10 \\ \ln 2 & 0 & \ln 5 \\ \ln 10 & \ln 5 & 0 \end{bmatrix} \approx \begin{bmatrix} 0 & 0.35 & 1.15 \\ 0.35 & 0 & 0.80 \\ 1.15 & 0.80 & 0 \end{bmatrix}.$$

*Since the walk distances are graph-geodetic (Lemma 7.1) and all paths from 1 to 3 visit 2, $d_{\frac{1}{3}}(1,2) + d_{\frac{1}{3}}(2,3) = d_{\frac{1}{3}}(1,3)$ holds.*

Regarding the existence of $R_t$, since for a connected graph, $A$ is irreducible, the Perron-Frobenius theory of nonnegative matrices provides the following result (cf. [23, Theorem 4]).

**Lemma 7.2.** *For any weighted adjacency matrix $A$ of a connected graph $G$, the series $R_t = \sum_{k=0}^{\infty}(tA)^k$ with $t > 0$ converges to $(I - tA)^{-1}$ if and only if $t < \rho^{-1}$, where $\rho = \rho(A)$ is the spectral radius of $A$. Moreover, $\rho$ is an eigenvalue of $A$; as such $\rho$ has multiplicity 1 and a positive eigenvector.*

Observe that for the graph $G$ of Example 7.1, $\rho = \sqrt{5}$, so $\frac{1}{3} = t < \rho^{-1}$ is satisfied.

**Lemma 7.3.** *For any vertices $i, j \in V(G)$ and $0 < t < \rho^{-1}$,*

$$d_t(i,j) = -\ln\left(\frac{r_{ij}(t)}{\sqrt{r_{ii}(t)\, r_{jj}(t)}}\right). \tag{7.7}$$

Lemma 7.3 is a corollary of Eqs. (7.4), (7.5), and Lemma 7.2.

On the basis of Lemma 7.3, the walk distances can be given the following short definition: $d_t(i,j) = -\ln r'_{ij}(t)$, where $r'_{ij}(t) = \frac{r_{ij}(t)}{\sqrt{r_{ii}(t)\,r_{jj}(t)}}$ and $R_t = (r_{ij}(t))_{n\times n}$ are defined by Eq. (7.3).

*Remark 7.1.* Consider another transformation of the correlation-like index $r'_{ij}(t) = \frac{r_{ij}(t)}{\sqrt{r_{ii}(t)\,r_{jj}(t)}}$:

$$d'_t(i,j) = 1 - \frac{r_{ij}(t)}{\sqrt{r_{ii}(t)\,r_{jj}(t)}}. \tag{7.8}$$

Is $d'_t(i,j)$ a metric? It follows from Definition 7.1 and Eqs. (7.7) and (7.8) that for any walk distance $d^{\mathrm{W}}_t(i,j)$, there exists $\lambda > 0$ such that

$$d'_t(i,j) = 1 - e^{-\lambda d^{\mathrm{W}}_t(i,j)}. \tag{7.9}$$

Equation (7.9) is the Schoenberg transform [21, 22] (see also [8, Section 9.1] and [1, 15]). As mentioned in [7], an arbitrary function $\tilde{d}(i,j)$ is the result of the Schoenberg transform of some metric if and only if $\tilde{d}(i,j)$ is a *P-metric*, i.e., a metric with values in $[0,\,1]$ that satisfies the *correlation triangle inequality*

$$1 - \tilde{d}(i,k) \ge (1 - \tilde{d}(i,j))(1 - \tilde{d}(j,k)), \tag{7.10}$$

which can be rewritten as $\tilde{d}(i,k) \le \tilde{d}(i,j) + \tilde{d}(j,k) - \tilde{d}(i,j)\tilde{d}(j,k)$.

This fact implies that Eq. (7.8) defines a P-metric. It is easily seen that the correlation triangle inequality for $d'_t(i,j)$ reduces to the transition inequality (7.2); obviously, it can be given a probabilistic interpretation.

For the graph $G$ of Example 7.1, the P-metric $d'_t(i,j)$ with $t = \frac{1}{3}$ is given by the matrix

$$D'_{\frac{1}{3}} = \left(d'_{ij}\left(\tfrac{1}{3}\right)\right) = \begin{bmatrix} 0 & 1-\sqrt{0.5} & 1-\sqrt{0.1} \\ 1-\sqrt{0.5} & 0 & 1-\sqrt{0.2} \\ 1-\sqrt{0.1} & 1-\sqrt{0.2} & 0 \end{bmatrix} \approx \begin{bmatrix} 0 & 0.29 & 0.68 \\ 0.29 & 0 & 0.55 \\ 0.68 & 0.55 & 0 \end{bmatrix}.$$

Obviously, this metric is not graph-geodetic.

*Remark 7.2.* It can be noted that the Nei standard genetic distance [17] and the Jiang-Conrath semantic distance [12] have a form similar to Eq. (7.7). Moreover, the transformation $-\ln(r(i,j))$ where $r(i,j)$ is a similarity measure between objects $i$ and $j$ was used in the construction of the Bhattacharyya distance between probability distributions [2] and the Tomiuk-Loeschcke genetic distance [24] (see also the Leacock-Chodorow similarity [16] and the Resnik similarity [19]). These and other distances and similarities are surveyed in [7].

## 7.4 An Interpretation of the Walk Distances

For a fixed $t\colon 0 < t < \rho^{-1}$, where $\rho = \rho(A)$ let us use the notation

$$B = I - tA. \tag{7.11}$$

Assume that $i$ and $j \neq i$ are also fixed and that $i + j$ is even; otherwise this can be achieved by renumbering the vertices. Hence, using Eq. (7.3)–(7.6), the positivity of $R_t = (I - tA)^{-1}$, and the determinant representation of the inverse matrix we obtain

$$d_t(i,j) = 0.5(\ln\det B_{\overline{\imath\imath}} + \ln\det B_{\overline{\jmath\jmath}} - \ln\det B_{\overline{\imath\jmath}} - \ln\det B_{\overline{\jmath\imath}}), \tag{7.12}$$

where $B_{\overline{\imath\jmath}}$ is $B$ with row $i$ and column $j$ removed.

### 7.4.1 *Logarithms of the Cofactors: Expressions in Terms of Circuits*

To obtain an interpretation of the right-hand side of Eq. (7.12), we need the following remarkable result due to Kasteleyn.

**Lemma 7.4 (Kasteleyn [13]).** *For a digraph $\Gamma$ with a weighted adjacency matrix $\tilde{A}$,*

$$\det(I - \tilde{A}) = \exp\left(-\sum_{c\in\mathscr{C}} \frac{w(c)}{\mu(c)}\right) \tag{7.13}$$

$$= \prod_{c\in\mathscr{C}_1} (1 - w(c)), \tag{7.14}$$

*where $\mathscr{C}$ and $\mathscr{C}_1$ are the sets of all circuits and of all nonperiodic circuits in $\Gamma$, $w(c)$ and $\mu(c)$ being the weight and the multiplicity of the circuit $c$.*

The representation (7.13) was obtained by considering the generating function of walks in $\Gamma$. Basically, the sum $\sum\limits_{c\in\mathscr{C}} \frac{w(c)}{\mu(c)}$ is a formal counting series in abstract *weight variables* (cf. [20, p. 19]). However, as soon as the weights are real and thus the generating function is a function in real counting variables, the issue of convergence arises. Since Eq. (7.13) is based on the power expansion $-\ln(I - \tilde{A}) = \sum\limits_{k=1}^{\infty} k^{-1}\tilde{A}^k$, a necessary condition of its validity in the real-valued setting is $\rho(\tilde{A}) < 1$.

When the arc weights are nonnegative, the same condition is sufficient. However, if some vertices $i$ and $j$ are connected by parallel $i \to j$ arcs carrying weights of different signs (we will encounter this case), then the problem of conditional convergence arises. Namely, if the absolute values of such weights are large enough, then, even though $\rho(\tilde{A}) < 1$, by choosing the order of summands in the right-hand side of Eq. (7.13), the sum can be made divergent or equal to any given number.

To preserve Eq. (7.13) in the latter case, the order of summands must be adjusted with an arbitrary order of items in $\sum_{k=1}^{\infty} k^{-1}\tilde{A}^k$. Hence it suffices to rewrite (7.13) in the form

$$\det(I-\tilde{A}) = \exp\left(-\sum_{k=1}^{\infty}\sum_{c\in\mathscr{C}_k}\frac{w(c)}{\mu(c)}\right), \tag{7.15}$$

where $\mathscr{C}_k$ is the set of all circuits that involve $k$ arcs in $\Gamma$.

Lemma 7.4 is also applicable to undirected graphs. To verify this, it is sufficient to replace an arbitrary undirected graph $G$ with its *directed version*, i.e., the digraph obtained from $G$ by replacing every edge by two opposite arcs carrying the weight of that edge.

Since by Eq. (7.11), $B_{\overline{\imath\imath}} = I-(tA)_{\overline{\imath\imath}}$, Lemma 7.4 can be used to evaluate $\ln\det B_{\overline{\imath\imath}}$ in Eq. (7.12). Let $G_{\bar{\imath}}$ $(G_{\overline{\imath\jmath}})$ be $G$ with vertex $i$ (vertices $i$ and $j$) and all edges incident to $i$ ($i$ and $j$) removed.

**Corollary 7.1.** *Let $B$ be defined by Eq.* (7.11)*. Then*

$$-\ln\det B_{\overline{\imath\imath}} = \sum_{c\in\mathscr{C}^{\bar{\imath}}}\frac{w(c)}{\mu(c)} = \sum_{c\in\mathscr{C}^{\overline{\imath\jmath}}\cup\mathscr{C}^{j\bar{\imath}}}\frac{w(c)}{\mu(c)},$$

*where*

- $\mathscr{C}^{\bar{\imath}}$ *is the set of circuits in* $tG_{\bar{\imath}}$ *,*
- $\mathscr{C}^{\overline{\imath\jmath}}$ *is the set of circuits in* $tG_{\overline{\imath\jmath}}$ *,*
- $\mathscr{C}^{j\bar{\imath}}$ *is the set of circuits visiting* $j$*, but not* $i$ *in* $tG$*,*

$w(c)$ *and* $\mu(c)$ *being the weight and the multiplicity of* $c$*.*

*Proof.* By assumption, $0 < t < \rho^{-1}(A)$; $B_{\overline{\imath\imath}} = I - tA_{\overline{\imath\imath}}$. Since $A$ is irreducible, $\rho(tA_{\overline{\imath\imath}}) < \rho(tA) < 1$ [9, Ch. III, § 3.4]. Moreover, the edge weights in $G$ are positive by assumption. Therefore, the expansion (7.13) holds for $B_{\overline{\imath\imath}}$, which yields the desired statement. □

To interpret (7.12), we also need an expansion of $\ln\det B_{\overline{\imath\jmath}}$ $(j\neq i)$. Convergence in such an expansion provided by Lemma 7.4 can be achieved by applying a suitable linear transformation of $B_{\overline{\imath\jmath}}$.

For the fixed $i$ and $j\neq i$, consider the matrix

$$T_{ij} = I(j,i)_{\overline{\imath\jmath}}, \tag{7.16}$$

where $I(j,i)$ differs from $I_{n\times n}$ by the $ji$-entry: $I(j,i)_{ji} = -1$.

The reader can easily construct examples of $T_{ij}$ and verify the following properties.

**Lemma 7.5.**

1. *The columns of $T_{ij}$ form an orthonormal set, i.e., $T_{ij}$ is orthogonal: $T_{ij}^{\mathrm{T}}T_{ij} = I$.*
2. *If $i+j$ is even (as assumed), then* $\det T_{ij} = 1$.
3. $T_{ij}^{\mathrm{T}} = T_{ji}$.
4. *For any $M_{n\times n}$, $M_{\overline{ij}}T_{ij}^{-1}$ is obtained from $M$ by:* (i) *deleting row $i$,* (ii) *multiplying column $i$ by $-1$, and* (iii) *moving it into the position of column $j$.*

The proof of Lemma 7.5 is straightforward.

**Corollary 7.2.** *1. $I_{\overline{ij}}T_{ij}^{-1}$ is obtained from $I_{(n-1)\times(n-1)}$ by replacing the kk-entry with* 0*, where*

$$k = \begin{cases} j, & j < i, \\ j-1, & j > i. \end{cases} \tag{7.17}$$

*2. $I_{\overline{ij}}T_{ij}^{-1}I_{\overline{ij}} = I_{\overline{ij}}$, i.e., $T_{ij}^{-1}$ is a g-inverse* [18] *of $I_{\overline{ij}}$.*

Since $\det T_{ji} = 1$ (Lemma 7.5), we have

$$\det B_{\overline{ij}} = \det(B_{\overline{ij}}T_{ji}). \tag{7.18}$$

Now we apply Kasteleyn's Lemma 7.4 to $B_{\overline{ij}}T_{ji}$ by considering a (multi)digraph $\Gamma$ whose weighted adjacency matrix is

$$\mathscr{A} = I - B_{\overline{ij}}T_{ji}, \tag{7.19}$$

where $B$ is defined by Eq. (7.11). Namely, Lemma 7.4 in the form Eq. (7.15) along with Eq. (7.18) yields

**Lemma 7.6.**

$$-\ln\det B_{\overline{ij}} = \sum_{k=1}^{\infty}\sum_{c\in\mathscr{C}'_k}\frac{w(c)}{\mu(c)}, \tag{7.20}$$

*where $\mathscr{C}'_k$ is the set of all circuits that involve $k$ arcs in a digraph $\Gamma$ whose weighted adjacency matrix is $\mathscr{A}$, while $w(c)$ and $\mu(c)$ are the weight and the multiplicity of the circuit $c$.*

As well as (7.13), Eq. (7.20) is applicable to the case of formal counting series. However, in Eq. (7.11), $t$ is a real weight variable. In this case, a necessary and sufficient condition of the convergence in Eq. (7.20) is $\rho(\mathscr{A}) < 1$.

Let us clarify the relation of $\Gamma$ and its circuits with $G$ and its topology. This is done in the following section.

### *7.4.2 The Walk Distances: An Expression in Terms of Walks*

To elucidate the structure of the digraph $\Gamma$ introduced in Lemma 7.6, an algorithmic description of the matrix $\mathscr{A}$ is useful.

**Lemma 7.7.** *$\mathscr{A}$ can be obtained from $tA$ by: replacing $ta_{ji}$ with $ta_{ji}-1$, deleting row $i$, multiplying column $i$ by $-1$, and moving it into the position of column $j$.*

*Proof.* By Eq. (7.19), items 1 and 3 of Lemma 7.5, and Eqs. (7.16) and (7.11), we have

$$\mathscr{A}=(T_{ij}-B_{\overline{\imath\jmath}})T_{ij}^{-1}=(I(j,i)-I+tA)_{\overline{\imath\jmath}}T_{ij}^{-1}.$$

Now the result follows from item 4 of Lemma 7.5. □

Let us reformulate Lemma 7.7 in terms of $G$ and $\Gamma$. Recall that a digraph is the *directed version* of a graph if it is obtained by replacing every edge in the graph by two opposite arcs carrying the weight of that edge.

**Corollary 7.3.** *A digraph $\Gamma$ with weighted adjacency matrix $\mathscr{A}$ can be obtained from $tG$ by:*

- *taking the directed version of the restriction of $tG$ to $V(G)\smallsetminus\{i,j\}$,*
- *adding a vertex "$ij$" with two loops of weights $1$ and $-ta_{ji}$ (negative; if $a_{ji}=0$, then this loop is omitted), weights $ta_{jm}$ of outgoing arcs, and weights $-ta_{mi}$ of incoming arcs, where $m\in V(G)\smallsetminus\{i,j\}$*

*Vertex ij is represented in $\mathscr{A}$ by row and column $k$, where $k$ is given by* (7.17).

In what follows, $\Gamma$ denotes the digraph defined in Corollary 7.3. The *jump* in $\Gamma$ is the loop of weight 1 at $ij$. The walk in $\Gamma$ that consists of one jump is called the *jump walk* (at $ij$).

To interpret $\ln\det B_{\overline{\imath\jmath}}$ in terms of $G$, we need the following notation.

**Definition 7.2.** A *walk with $i,j$ jumps in $G$* is any walk in the graph $G'$ obtained from $G$ by attaching two additional loops of weight 1: one adjacent to vertex $i$ and one adjacent to $j$. These loops are called *jumps.* A walk with $i,j$ jumps (in $G$) only consisting of one jump is called a *jump walk* (at $i$ or $j$).

**Definition 7.3.** A *$j\to i$ alternating walk with jumps* is any $j\to i$ walk $\mathrm{w}$ with $j,i$ jumps such that (a) any $j\dots j$ subwalk of $\mathrm{w}$ either visits $i$ or contains no edges except for jumps and (b) any $i\dots i$ subwalk of $\mathrm{w}$ either visits $j$ or contains no edges except for jumps.

A *$j\to i\to j$ alternating walk with jumps* is defined similarly: the only difference is that the endpoint of such a walk is $j$.

To introduce some additional notation, observe that any $j\to i$ alternating walk $\mathrm{w}$ with jumps can be uniquely partitioned into a sequence of subwalks $(\mathrm{w}_1,\dots,\mathrm{w}_t)$ such that every two neighboring subwalks share one terminal vertex and each $\mathrm{w}_k$ is a jump walk or is a $j\to i$ or an $i\to j$ hitting walk without jumps. For every $k\in\{1,\dots,t\}$, consider the set $p_k=\{\mathrm{w}_k,\tilde{\mathrm{w}}_k\}$, where $\tilde{\mathrm{w}}_k$ is $\underline{\text{either}}$ $\mathrm{w}_k$ written from end to beginning (reversed[3]) when $\mathrm{w}_k$ is a hitting walk without jumps, or a jump walk at $i$ ($j$) when $\mathrm{w}_k$ is a jump walk at $j$ (resp., $i$). The sequence $(p_1,\ldots,p_t)=p(\mathrm{w})$ will be called the *route partition of* w. We say that two $j\to i$ alternating walks with jumps, w and w$'$, are *equipartite* if the route partition of w$'$ can be obtained from that of w by a cyclic shift. Finally, any equivalence class of equipartite $j\to i$ alternating walks with jumps will be called an *alternating* $j\to i$ *route with jumps.* If r is such a route, then its *length* and *weight* are defined as the common length and weight of all walks with jumps it includes, respectively. If a route partition $p(\mathrm{w})=(p_1,\ldots,p_t)$ has period (the length of the elementary repeating part) $y$, then the *multiplicity* of the alternating $j\to i$ route with jumps that corresponds to $p(\mathrm{w})$ is defined to be $t/y$.

Completely the same construction can be applied to define *alternating* $j\to i\to j$ *route with jumps* (starting with the above definition of a $j\to i\to j$ alternating walk with jumps). A notable difference is that there are alternating $j\to i\to j$ routes with jumps that do not visit $i$: these consist of jumps at $j$. The weight of such a route with jumps is 1 and its multiplicity is the number of jumps.

**Lemma 7.8.** *There is a one-to-one correspondence between the set of circuits in $\Gamma$ that contain vertex $ij$ and have odd (even) numbers of negatively weighted arcs and the set of alternating $j\to i$ routes (alternating $j\to i\to j$ routes) with jumps in $G$. The circuit in $\Gamma$ and route with jumps in $G$ that correspond to each other have the same length, weight, and multiplicity.*

*Proof.* Every circuit containing vertex $ij$ in $\Gamma$ can be uniquely represented by a cyclic sequence[4] of walks each of which either is an $ij\to ij$ walk including exactly one negatively weighted arc, or is the jump walk at $ij$. Such a cyclic sequence uniquely determines an alternating $j\to i$ or $j\to i\to j$ route with jumps in $G$ (if the number of negatively weighted arcs involved in the circuit is odd or even, respectively).

On the other hand, every set $p_k=\{\mathrm{w}_k,\tilde{\mathrm{w}}_k\}$ involved in an alternating $j\to i$ or $j\to i\to j$ route with jumps in $G$ uniquely determines either an $ij\to ij$ walk containing exactly one negatively weighted arc, or the jump walk at $ij$ in $\Gamma$. Thereby, every alternating route with jumps under consideration uniquely determines a circuit in $\Gamma$. Furthermore, the two correspondences described above are inverse to each other. Thus, these determine a one-to-one correspondence.

Finally, it is easily seen that the corresponding circuits and alternating routes with jumps share the same length, weight, and multiplicity. □

*Remark 7.3.* It can be noted that the multiplicity of an alternating $j\to i$ route with jumps in $G$ can only be odd.

Both circuits and alternating routes will be called *figures*. Lemmas 7.6 and 7.8 enable one to express $\ln\det B_{\overline{ij}}$ in terms of figures in $tG$ and $tG_{\overline{ij}}$.

[3] Cf. “dihedral equivalence” in [11].

[4] A *cyclic sequence* is a set $X=\{x_1,\ldots,x_N\}$ with the relation “*next*” $\eta=\{(x_2,x_1),\ldots,(x_N,x_{N-1}),(x_1,x_N)\}$.

**Lemma 7.9.**

$$-\ln\det B_{\overline{\imath\jmath}} \;=\; \sum_{k=1}^{\infty}\;\sum_{c\in(\mathscr{C}^{\overline{\imath\jmath}}\cup\mathscr{C}^{j\to i\to j}\cup\mathscr{C}^{j\to i})\cap\mathscr{C}_k}(-1)^{\zeta(c)}\frac{w(c)}{\mu(c)},$$

*where*

- $\mathscr{C}^{\overline{\imath\jmath}}$ *is the set of circuits in* $tG_{\overline{\imath\jmath}}$*,*
- $\mathscr{C}^{j\to i\to j}$ *is the set of alternating* $j\to i\to j$ *routes with jumps in* $tG$*,*
- $\mathscr{C}^{j\to i}$ *is the set of alternating* $j\to i$ *routes with jumps in* $tG$*,*
- $\mathscr{C}_k$ *is the set of figures* $(in\ tG\ or\ tG_{\overline{\imath\jmath}})$ *that involve* $k$ *arcs,*

$$\zeta(c)=\begin{cases}0, & c\in\mathscr{C}^{\overline{\imath\jmath}}\cup\mathscr{C}^{j\to i\to j},\\ 1, & c\in\mathscr{C}^{j\to i},\end{cases}$$

*while* $w(c)$ *and* $\mu(c)$ *are the weight and the multiplicity of* $c$*.*

Similarly, we can express $\ln\det B_{\overline{\jmath\imath}}$ in terms of the sets $\mathscr{C}^{\overline{\imath\jmath}}$, $\mathscr{C}^{i\to j\to i}$, and $\mathscr{C}^{i\to j}$. There exist natural bijections between $\mathscr{C}^{j\to i\to j}$ and $\mathscr{C}^{i\to j\to i}$ and between $\mathscr{C}^{j\to i}$ and $\mathscr{C}^{i\to j}$. Namely, to obtain an element of $\mathscr{C}^{i\to j\to i}$ from $c\in\mathscr{C}^{j\to i\to j}$ (or an element of $\mathscr{C}^{i\to j}$ from $c\in\mathscr{C}^{j\to i}$), it suffices to reverse all $j\to i$ and $i\to j$ hitting walks without jumps in $c$ and to replace every jump walk at $j$ with the jump walk at $i$ and vice versa.

On the other hand, the sets $\mathscr{C}^{i\rightleftarrows j}\overset{\text{def}}{=}\mathscr{C}^{j\to i\to j}\cup\mathscr{C}^{i\to j\to i}$ and $\mathscr{C}^{i-j}\overset{\text{def}}{=}\mathscr{C}^{j\to i}\cup\mathscr{C}^{i\to j}$ also make sense. Specifically, they are useful for expressing $d_t(i,j)$. Such an expression is the main result of this chapter. It follows by combining (7.12), Corollary 7.1, and Lemma 7.9.

**Theorem 7.1.**

$$d_t(i,j) \;=\; \frac{1}{2}\sum_{k=1}^{\infty}\;\sum_{c\in(\mathscr{C}^{i\,\bar\jmath}\cup\mathscr{C}^{\bar\imath j}\cup\mathscr{C}^{i\rightleftarrows j}\cup\mathscr{C}^{i-j})\cap\mathscr{C}_k}(-1)^{\zeta(c)}\frac{w(c)}{\mu(c)},$$

*where the* *<u>sets</u>* *of figures in* $tG$ *are denoted by*:

- $\mathscr{C}^{i\,\bar\jmath}$: *of circuits visiting* $i$, *but not* $j$*,*
- $\mathscr{C}^{\bar\imath j}$: *of circuits visiting* $j$, *but not* $i$*,*
- $\mathscr{C}^{i\rightleftarrows j}$: *of alternating* $j\to i\to j$ *and* $i\to j\to i$ *routes with jumps,*
- $\mathscr{C}^{i-j}$: *of alternating* $j\to i$ *and* $i\to j$ *routes with jumps,*
- $\mathscr{C}_k$: *of figures that involve* $k$ *arcs,*

$$\zeta(c)=\begin{cases}0, & c\in\mathscr{C}^{i\rightleftarrows j},\\ 1, & c\in\mathscr{C}^{i\,\bar\jmath}\cup\mathscr{C}^{\bar\imath j}\cup\mathscr{C}^{i-j},\end{cases}$$

*while* $w(c)$ *and* $\mu(c)$ *are the weight and the multiplicity of* $c$*.*

**Table 7.1** The figures forming the leading terms in the expansion of $d_{\frac{1}{3}}(1,3)$ in Example 7.2

| $\cap$ | $\mathscr{C}^{1\bar{3}}\cup\mathscr{C}^{\bar{1}3}$ | $\mathscr{C}^{1\rightleftarrows 3}$ | $\mathscr{C}^{1-3}$ |
|---|---|---|---|
| $\mathscr{C}_1$ | $\varnothing$ | (11), (33) | $\varnothing$ |
| $\mathscr{C}_2$ | 4(121), (323) | $\frac{1}{2}(111), \frac{1}{2}(333)$ | 2(123), 2(321) |
| $\mathscr{C}_3$ | $\varnothing$ | $\frac{1}{3}(1111), \frac{1}{3}(3333)$ | 2(1123), 2(3321) |
| $\mathscr{C}_4$ | $\frac{4}{2}(12121), 6(12121),$<br>$\frac{1}{2}(32323)$ | $\frac{1}{4}(11111), \frac{1}{4}(33333),$<br>$\frac{2}{2}(12321), (12321), \frac{2}{2}(32123), (32123)$ | 2(11123), 2(33321) |
| $\mathscr{C}_5$ | $\varnothing$ | $\frac{1}{5}(111111), \frac{1}{5}(333333), 4(112321), 4(332123)$ | 2(111123), 2(333321) |

In more general terms, Theorem 7.1 can be interpreted as follows. The value of the walk distance between $i$ and $j$ is reduced by $j \to i$ and $i \to j$ walks (see $\mathscr{C}^{i-j}$), connections of $i$ with other vertices avoiding $j$ ($\mathscr{C}^{i\bar{j}}$), and connections of $j$ avoiding $i$ ($\mathscr{C}^{\bar{i}j}$). The set $\mathscr{C}^{i\rightleftarrows j}$ supplies all positive terms in the expansion of $d_t(i,j)$. It comprises constantly jumping walks along with closed walks involving $i$ and $j$ whose positive weights compensate the negative overweight of $j \to i$ and $i \to j$ routes with extra jumps.

Note that Theorem 7.1 supports the observation in the Introduction that the high centrality of $i$ and $j$ reduces, ceteris paribus, the walk distance between them. Indeed, the elements of $\mathscr{C}^{i\bar{j}}\cup\mathscr{C}^{\bar{i}j}$ which account for the centrality of $i$ and $j$ make a negative contribution to the distance.

The following example may provide some additional insight into Theorem 7.1.

*Example 7.2. For the graph G of Example 7.1, let us approximate* $d_{\frac{1}{3}}(1,3) = \frac{1}{2}\ln 10 \approx 1.15$ *using Theorem 7.1. Due to Eq. (7.19),* $\mathscr{A} = \frac{1}{3}\begin{bmatrix} 0 & -2 \\ 1 & 3 \end{bmatrix}$. *As* $\rho(\mathscr{A}) = 2/3 < 1$, *convergence holds in Eq. (7.20) and thus in Theorem 7.1. The leading terms of the expansion Theorem 7.1 provides for* $d_{\frac{1}{3}}(1,3)$ *are presented in Table 7.1. In this table,* $\frac{k}{\mu}(v_0\cdots v_m)$ *is the denotation of a collection of figures where each figure has multiplicity* $\mu$ *and contains some walk (or walk with jumps) whose sequence of vertices is* $v_0,\ldots,v_m$; *k is the cardinality of the collection. If* $\mu = 1$, *then* $\mu$ *is omitted; if* $\mu = k = 1$, *then* $\mu$ *and k are omitted.*

*The first terms of the series Theorem 7.1 provides are:*

$$\begin{aligned} d_{\frac{1}{3}}(1,3) = \frac{1}{2}\Bigg[&(2\cdot 1) + \left(-\frac{4}{9} - \frac{1}{9} + 2\cdot\frac{1}{2} - 2\cdot\frac{2}{9}\right) + \left(2\cdot\frac{1}{3} - 2\cdot\frac{2}{9}\right) \\ &+ \left(-\frac{2+6}{81} - \frac{1}{2}\cdot\frac{1}{81} + 2\left(\frac{1}{4} + \frac{1+1}{81}\right) - 2\cdot\frac{2}{9}\right) \\ &+ \left(2\left(\frac{1}{5} + \frac{4}{81}\right) - 2\cdot\frac{2}{9}\right) + \ldots\Bigg] \end{aligned}$$

$$= \frac{461}{405} + \dots,$$

*where* $\frac{461}{405} \approx 1.1383$.

*In the above expression, the sum (with signs) of the weights of figures that involve k edges is* 0 *whenever k is even. Thus, the above expansion reduces to*

$$d_{\frac{1}{3}}(1,3) = \frac{1}{2}\left[(2\cdot 1) + \left(2\cdot\frac{1}{3} - 2\cdot\frac{2}{9}\right) + \left(2\left(\frac{1}{5}+\frac{4}{81}\right) - 2\cdot\frac{2}{9}\right) + \dots\right].$$

*The relative error of this approximation is* 1.1%.

In some cases, the convergence of such expansions is extremely slow. On the other hand, the meaning of Theorem 7.1 is to clarify the concept of walk distance by representing it as the sum of route/circuit weights rather than to provide an effective algorithm for computing it.

## References

1. Bavaud, F.: On the Schoenberg transformations in data analysis: Theory and illustrations. J. Classification **28**(3), 297–314 (2011)
2. Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by their probability distributions. Bulletin of the Calcutta Mathematical Society **35**, 99–109 (1943)
3. Chebotarev, P.: A class of graph-geodetic distances generalizing the shortest-path and the resistance distances. Discrete Appl. Math.**159**(5), 295–302 (2011)
4. Chebotarev, P.: The graph bottleneck identity. Adv. Appl. Math. **47**(3), 403–413 (2011)
5. Chebotarev, P.: The walk distances in graphs. Discrete Appl. Math. **160**(10–11), 1484–1500 (2012)
6. Critchley, F.: On certain linear mappings between inner-product and squared-distance matrices. Lin. Algebra. Appl. **105**, 91–107 (1988)
7. Deza, M.M., Deza, E.: Encyclopedia of Distances. Springer, Berlin (2009)
8. Deza, M.M., Laurent, M.: Geometry of Cuts and Metrics. Algorithms and Combinatorics, vol. 15. Springer, Berlin (1997)
9. Gantmacher, F.R.: Applications of the Theory of Matrices. Interscience, New York (1959)
10. Harary, F.: Graph Theory. Addison-Wesley, Reading, MA (1969)
11. Harary, F., Schwenk, A.: The spectral approach to determining the number of walks in a graph. Pac. J. Math. **80**(2), 443–449 (1979)
12. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. In: Proceedings of International Conference on Research in Computational Linguistics (ROCLING X), Taiwan, 19–33 (1997)
13. Kasteleyn, P.W.: Graph theory and crystal physics. In: Harary, F. (ed.) Graph Theory and Theoretical Physics, pp. 43–110. Academic Press, London (1967)
14. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
15. Laurent, M.: A connection between positive semidefinite and Euclidean distance matrix completion problems. Lin. Algebra. Appl. **273**(1–3), 9–22 (1988)

16. Leacock, C., Chodorow, M.: Combining local context and WordNet similarity for word sense identification. In: Fellbaum, C. (ed.) WordNet. An Electronic Lexical Database, pp. 265–283, chap. 11. MIT Press, Cambridge, MA (1998)
17. Nei, M.: Genetic distance between populations. Am. Nat. **106**(949), 283–292 (1972)
18. Rao, C.R., Mitra, S.K.: Generalized Inverse of Matrices and its Applications. Wiley, New York (1971)
19. Resnik, P.: Using information content to evaluate semantic similarity. In: Proceedings of the $14^{th}$ International Joint Conference on Artificial Intelligence (IJCAI'95), vol. 1, Morgan Kaufmann Publishers, San Francisco, CA (1995)
20. Riordan, J.: An Introduction to Combinatorial Analysis. Wiley, New York (1958)
21. Schoenberg, I.J.: Remarks to M. Fréchet's article "Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert". Ann. Math. **36**, 724–732 (1935)
22. Schoenberg, I.J.: Metric spaces and positive definite functions. Trans. Am. Math. Soc. **44**, 522–536 (1938)
23. Thompson, G.L.: Lectures on Game Theory, Markov Chains and Related Topics, Monograph SCR–11, Sandia Corporation, Albuquerque, NM (1958)
24. Tomiuk, J., Loeschcke, V.: A new measure of genetic identity between populations of sexual and asexual species. Evolution **45**, 1685–1694 (1991)

# Part II
# Methods

# Chapter 8
# Distance Geometry Methods for Protein Structure Determination

**Zachary Voller and Zhijun Wu**

**Abstract** We review some recent advances for solving the distance geometry (DG) problems for protein structure determination. We focus on the development of a geometric buildup approach to the problems with sparse but exact distances and on the formulation of a generalized DG problem for the determination of structural ensembles with inexact distances or distance bounds. We describe the novel ideas of these approaches, show their potentials for the solution of large-scale problems in practice, and discuss their possible future developments. For some historical background, we also provide a brief introduction to the classical matrix decomposition method, the embedding algorithm, and the global smoothing algorithm for the solution of the DG problems with exact and inexact distances. Many other methods have been developed. We will not cover them all, but refer the readers to a list of papers, hoping to provide the readers with a relatively complete knowledge of the field.

## 8.1 Introduction

Proteins are biological molecules that are encoded in genes and produced by the ribosome in a process called genetic translation. These molecules are used in almost all biological processes and they often play a critical role in the regulation of these processes. Proteins are composed of linear chains of amino acids which fold into stable conformations. The conformation of a protein is a critical factor in

Z. Voller
Department of Mathematics, Iowa State University, Ames, IA 50011, USA
e-mail: zvoller@iastate.edu

Z. Wu (✉)
Department of Mathematics, Iowa State University, Ames, IA 50011, USA
e-mail: zhijun@iastate.edu

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_8,

determining its functionality. Therefore, protein structure determination becomes a necessary component in the study of proteins and their interactions in biological pathways.

Nuclear magnetic resonance spectroscopy (NMR) is a major experimental technique used to determine protein structures [3, 32]. These experiments yield a set of data corresponding to the distances between hydrogen atoms. Let $n$ denote the number of atoms in a protein and $x_1, x_2, \ldots, x_n$ be the coordinate vectors corresponding to atoms $1, 2, \ldots, n$, where $x_i = (x_{i1}, x_{i2}, x_{i3})^T \in \mathbb{R}^3$. Let $d_{i,j}$ denote the distance between $i$th and $j$th atoms. This naturally leads to the distance geometry (DG) problem of determining a set of coordinates $x_1, x_2, \ldots, x_n$ that satisfy all the distances $d_{i,j} = \| x_i - x_j \|$ from the experimental data set, where $\| \cdot \|$ denotes the usual Euclidean norm [3, 32].

In this chapter, we review some recent advances for solving the DG problems for protein structure determination. We focus on the development of a geometric buildup approach to the problems with sparse but exact distances [4, 30] and on the formulation of a generalized DG problem for the determination of structural ensembles with inexact distances or distance bounds [25, 26]. We describe the novel ideas of these approaches, show their potentials for the solution of large-scale problems in practice, and discuss their possible future developments.

For some historical background, we also provide a brief introduction to the classical matrix decomposition method [2,27], the embedding algorithm [3,10], and the global smoothing algorithm [19, 20] for the solution of the DG problems with exact and inexact distances. Many other methods have been developed. We will not cover them all, but refer the readers to a list of papers cited herein [1,7–9,11–14,28].

## 8.2 Historical Development

The DG problem has been studied for several different cases including the problem with a full set of exact (error-free) distances, the problem with a sparse set of exact distances, and the problem with a sparse set of inexact distances (or distance bounds). The first type of problem can be solved efficiently in polynomial time by using, for example, a matrix decomposition method [2, 27]. The second type is computationally intractable and has been proven to be NP-hard [24]. An approximate solution to this type of problem may be obtained by using a global optimization method such as the global smoothing algorithm [19,20]. The third type is of particular interest in NMR structural determination. An embedding algorithm has been developed and used widely in NMR modeling software [3, 10].

### 8.2.1 Matrix Decomposition Method

A general DG problem can be defined in any metric space, but for protein structure determination, we will only consider the problem in 3D real space. First, a DG problem with a full set of exact distances can be defined formally as follows.

**Definition 8.1.** Given a full set of exact distances,

$$d_{i,j} = \| x_i - x_j \| \qquad i,j = 1,2,\ldots,n, \tag{8.1}$$

a DG problem with a full set of exact distances determines the coordinate vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ satisfying the above distance equations, where $\| \centerdot \|$ denotes the Euclidean norm.

A classical method for solving the DG problem with a full set of exact distances is the matrix decomposition method. The idea of this method came as early as 1953 in Blumenthal's studies on DG [2] and in the study of multidimensional scaling for statistics by Torgerson in 1958 [27]. First, since a protein structure is invariant under rotations and translations, the last atom can be fixed at the origin, $x_n = (0,0,0)^T$, and used as a reference point for the determination of the rest of the structure. Then, Eq. (8.1) yields,

$$\begin{aligned} d_{i,j}^2 &= \| x_i - x_j \|^2 \\ &= \| x_i \|^2 - 2x_i^T x_j + \| x_j \|^2 \\ &= \| x_i - x_n \|^2 - 2x_i^T x_j + \| x_j - x_n \|^2, \end{aligned}$$

and the DG problem with a full set of exact distances is transformed into the following set of distance constraints:

$$d_{ij}^2 = d_{i,n}^2 - 2x_i^T x_j + d_{j,n}^2 \qquad i,j = 1,2,\ldots,n-1. \tag{8.2}$$

These constraints form a matrix equation $XX^T = D$, where

$$X = (x_1, x_2, \ldots, x_{n-1})^T$$

and

$$D = \left\{ \left( d_{i,n}^2 - d_{i,j}^2 + d_{j,n}^2 \right) / 2 : \quad i,j = 1,2,\ldots,n-1 \right\}.$$

Its solution is then obtained through the singular-value decomposition $D = U\Sigma U^T$, where $U$ is an orthogonal matrix and $\Sigma$ a diagonal matrix with the singular values of $D$ as the diagonal elements. If the matrix $D$ is of rank 3 (as it should be), then $X = V\Lambda^{1/2}$ solves the matrix equation $XX^T = D$, where $V = U(:,1:3)$ and $\Lambda = \Sigma(1:3,1:3)$.

The matrix decomposition method requires only order of $n^2$ calculations, if a proper singular-value decomposition algorithm is employed (to obtain only the three largest singular values and corresponding singular vectors of $D$). If the distances that form $D$ are not exact (or are inconsistent), then the rank of $D$ can be larger than 3 and $XX^T$ obtained from this method is still a good approximation to $D$ in a least-squares sense. This can be justified via the low-rank matrix approximation theory established long ago by Eckart and Young in 1936 [6]:

**Theorem 8.1 (Eckart and Young 1936).** *Let $M$ be an $n \times n$ symmetric matrix. Let $M = U\Sigma U^T$ be the singular value decomposition of $M$. Then, $N = U(:,1:r)\Sigma(1:r,1:r)U(:,1:r)^T$ minimizes $\|N - M\|_F$ for all $n \times n$ symmetric matrices $N$ of rank less than or equal to $r$.*

While the matrix decomposition method can only be applied to the DG problem with a full set of exact distances, only a sparse set of inexact distances or distance bounds are typically available. Nonetheless, a practical DG problem is often transformed to a group of DG subproblems with full sets of exact distances, which can be solved efficiently by using the matrix decomposition method, as we will see in the embedding algorithm [3, 10] and the geometric buildup algorithm [25, 26] for solving general DG problems.

### *8.2.2 The Embedding Algorithm*

One of the limitations of NMR experiments is that they can only determine inter-atomic distances over short intervals, generally no more than 5Å in length [3, 32]. Thus, a more realistic version of the DG problem utilizes a sparse set of exact distances:

**Definition 8.2.** Given a subset $S$ of the full set of exact distances,

$$d_{i,j} = \| x_i - x_j \|, \qquad d_{i,j} \in S, \tag{8.3}$$

a DG problem with a sparse set of exact distances determines the coordinate vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ satisfying the above distance equations derived from $S$.

Another limitation of NMR experiments is the low accuracy of the generated data: The distance data derived from NMR experiments is typically estimated in ranges corresponding to the atomic fluctuations [3, 32]. As such, an even more general formulation of the DG problem is required for inexact distances or distance bounds:

**Definition 8.3.** Given a subset $S$ of the full set of distance ranges, a DG problem with a sparse set of distance bounds seeks the coordinates $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ satisfying

$$l_{i,j} \leq \| x_i - x_j \| \leq u_{i,j}, \qquad d_{i,j} \in S \tag{8.4}$$

where $l_{i,j}$ and $u_{i,j}$ denote the lower and upper bound for the range of the distance $d_{i,j} = \| x_i - x_j \|$.

The above problem provides an accurate description for the DG problem that arises in NMR structure determination: First, the coordinates of the atoms need to be determined so that the interatomic distances are within the experimental distance ranges. Second, such a set of coordinates, corresponding to one solution to the problem, is not unique. In fact, there may be infinitely many such solutions. Third, while it is not necessary to obtain the whole set of solutions, a subset of solutions instead of a single solution is always interesting in practice because they demonstrate the natural fluctuations of the structure [3, 32].

Crippen and Havel are two of the pioneers that formulated DG problems for NMR structure determination. They have considered the DG problem with distance bounds and developed a so-called EMBED algorithm, which has been adopted widely in NMR modeling practices. The algorithm is named EMBED because the solution to a DG problem can be viewed as an embedding of a distance graph into a Euclidean space [3]. We will therefore adopt their convention and call the algorithm the embedding algorithm.

The EMBED algorithm stores the given set of lower and upper distance bounds, and then implements a bound-smoothing process in order to further refine and complete the given distance data. The bound-smoothing process considers sets of three and four atoms and uses the triangle and tetrangle inequalities to complete the missing bounds while refining the given bounds. This refinement yields a more restrictive set of distance constraints by lowering the upper bounds and raising the lower bounds [3, 10].

The next step of the algorithm uses the distance intervals defined by the smoothed bounds to randomly form a trial distance matrix, $\{d_{i,j} \in [l_{i,j}, u_{i,j}] : \; i, j = 1, \ldots, n\}$. Even though this matrix is symmetric, unfortunately, it is not generally a three-dimensional Euclidean distance matrix.

**Definition 8.4.** A matrix $\{d_{i,j} : \; i, j = 1, \ldots, n\}$ is a three-dimensional Euclidean distance matrix if:

- $d_{i,j} = d_{j,i}$ for $1 \le i, j \le n$.
- $d_{i,i} = 0$ for $i = 1, \ldots, n$.
- There exists coordinates $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ such that $d_{i,j} = \| x_i - x_j \|^2$ for $1 \le i, j \le n$

In any case, the EMBED algorithm first applies the matrix decomposition algorithm described in Sect. 8.2.1 to the randomly generated distances $\{d_{i,j} : \; i, j = 1, \ldots, n\}$, in order to obtain a set of coordinate vectors, $x_1, x_2, \ldots, x_n$. The algorithm then proceeds as follows: First, set $x_n = (0,0,0)^T$. Then, solve an equation $XX^T = D$, where

$$X = (x_1, x_2, \ldots, x_{n-1})^T$$

and

$$D = \left\{ \left(d_{i,n}^2 - d_{i,j}^2 + d_{j,n}^2\right)/2 : \quad i, j = 1, 2, \ldots, n-1 \right\}.$$

Let $D = U\Sigma U^T$ be the singular-value decomposition of $D$. Then, $X = V\Lambda^{1/2}$ yields an approximate solution to $XX^T = D$, where $V = U(:,1:3)$ and $\Lambda = \Sigma(1:3,1:3)$. This represents the best possible 3D coordinate vectors, $x_1, x_2, \ldots, x_n$, for $\{d_{i,j} : i,j = 1,\ldots,n\}$, even if it is not a 3D Euclidean distance matrix [3, 10].

The obtained coordinate vectors, $x_1, x_2, \ldots, x_n$, may not necessarily satisfy the conditions: $l_{i,j} \leq \|x_i - x_j\| \leq u_{i,j},\ i,j = 1,\ldots,n$. Therefore, the final step of the EMBED algorithm invokes an optimization procedure to further minimize the distance violations if there are any, using the coordinate vectors obtained in the previous step as an initial guess. A typical error function to be minimized is given as follows:

$$f(x_1, x_2, \ldots, x_n) = \sum_{i,j=1}^{n} (l_{i,j} - \|x_i - x_j\|)_+^2 + (\|x_i - x_j\| - u_{i,j})_+^2, \qquad (8.5)$$

where $g_+ = g$ if $g > 0$ and $g_+ = 0$ otherwise. The error function is typically nonconvex. Hence, even though the EMBED algorithm converges to a solution, the solution may not be the global minimizer of the error function. It is therefore necessary to execute multiple simulations using distinct trial distance matrices to ensure an acceptable solution has been determined [3, 10].

### *8.2.3 Global Smoothing Algorithm*

The DG problem with a sparse or full set of exact distances or distance bounds can also be formulated as an optimization problem. A general form of the problem can be given as follows:

$$\min f(x_1, x_2, \ldots, x_n) = \sum_{d_{i,j} \in S} (l_{i,j} - \|x_i - x_j\|)_+^2 + (\|x_i - x_j\| - u_{i,j})_+^2, \qquad (8.6)$$

where $S$ is a given set of distances. If $S$ contains a full set of distances, the objective function is exactly the error function (8.5) in the EMBED algorithm. Otherwise, it is essentially a least-squares problem for the DG problem with distance bounds Eq. (8.4), with the problem composed of exact distances Eq. (8.3) as a special case, when the lower and upper bounds are equal. Many approaches to the problem (8.6) have been proposed and investigated by researchers in optimization community [1, 7–9, 13, 14, 28]. The objective function defined by Eq. (8.2.3) is highly nonconvex and contains numerous local minima. Thus, the greatest challenge in solving the problem is obtaining the required global minimum as its solution.

Moré and Wu [19, 20] proposed a global smoothing algorithm for the solution of the problem (8.6). The idea came originally from the diffusion equation method proposed by Scheraga and co workers [15, 23] for protein potential energy minimization. Wu [31] performed a theoretical analysis on this method and called it an effective energy transformation method. In this method, the objective function is considered to be some physical quantity to be diffused in time. The diffused function, as the solution of a diffusion equation, is a convolution of the objective function with a kernel function. It depends on a time parameter: The longer the time, the more diffused the function. In addition, if the objective function has many local minima, it diffuses to a smoother function with fewer local minima after a sufficiently long time [15, 23].

Thus, the diffusion equation method, instead of directly minimizing an objective function with many local minima, first transforms the objective function to a new function. The transformed function, which depends on a smoothing parameter, starts from the original objective function when the parameter value equals zero. Furthermore, the transformation yields smoother and smoother functions, which contain fewer and fewer local minima, as the parameter is increased to larger and larger values. Therefore, at certain point, an optimization procedure can be applied to the smoothed function, to find its global minimum (which is presumably easier than that for the original objective function). The optimization procedure can be applied repeatedly to the smoothed function starting from the obtained global minimum, as the parameter value is gradually decreased. Hopefully, in the end, it can find the global minimum of the original objective function when the parameter value is decreased to zero.

The above process is called by Moré and Wu [19, 20] the global smoothing algorithm. Essential to this algorithm is the transformation function: Given a function $f:\ \mathbb{R}^n \to \mathbb{R}$, a transformed function $< f >_\lambda:\ \mathbb{R}^n \to \mathbb{R}$ is defined such that for any $x \in \mathbb{R}^n$,

$$< f >_\lambda (x) = \int_{y \in \mathbb{R}^n} f(y) p_\lambda (y - x) dy, \tag{8.7}$$

where $p_\lambda(x) = c_\lambda exp(\|x\|^2/\lambda^2)$ is a Gaussian kernel and $c_\lambda$ is a normalization constant.

Moré and Wu [19, 20] showed that the objective function from the optimization formulation of the DG problem (8.6) is easily transformed into such a function. A global optimization algorithm can then be applied to the "smoothed" function in order to obtain its global minimum relatively more efficiently, at least for small, to medium-sized problems, with several hundreds of variables.

## 8.3 The Geometric Buildup Approach

A series of works have been done in recent years by Wu and various collaborators [4, 25, 26, 30] in order to develop a more geometrically based method for obtaining the solution of the DG problem. This approach is based on basic DG principles for the determination of a point by using a set of distances in a given space [2]. For example, in one-dimensional Euclidean space, two distances are required to determine a point uniquely, in two-dimensional, three distances, and in three-dimensional, four distances, etc. Therefore, a buildup procedure can be developed to determine a given set of points (or atoms in structure determination), one at a time, by using any available distances for the point.

There are several advantages of using a geometric buildup method: First, the determination of each point does not require much calculation, and in the ideal case when the required distances are always available for each point, the total calculation for determining all the points will be proportional to the number of points. Second, the method does not require a full set of distances. A lower bound for determining $n$ points in $\mathbb{R}^k$ is $(k+1)n$. Finally, the method determines as many points as possible, while the points without sufficient distance supports can be identified (for further examination). In the following sections, we describe this method as it is applied to different types of distance data.

### 8.3.1 Geometric Buildup with a Full Set of Exact Distances

Recall, the DG problem for a full set of exact distances $\{d_{i,j} : \ i,j = 1,2,\ldots,n\}$ is to determine the coordinate vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ for $n$ atoms satisfying

$$d_{i,j} = \parallel x_i - x_j \parallel \qquad\qquad i,j = 1,2,\ldots,n. \tag{8.8}$$

The geometric buildup algorithm first finds four atoms not in the same plane. The coordinates of the four atoms can then be determined with the distances among them uniquely except for rotations or translations. Suppose that the coordinate for $x_1, x_2, x_3,$ and $x_4$ are given by

$$\begin{aligned} x_1 &= (x_{11}, x_{12}, x_{13})^T \\ x_2 &= (x_{21}, x_{22}, x_{23})^T \\ x_3 &= (x_{31}, x_{32}, x_{33})^T \\ x_4 &= (x_{41}, x_{42}, x_{43})^T . \end{aligned}$$

With each successive iteration, the buildup algorithm determines the coordinates of $x_j = \left(x_{j1}, x_{j2}, x_{j3}\right)$ using the distances

$$\begin{aligned} d_{1,j} &= \parallel x_j - x_1 \parallel \\ d_{2,j} &= \parallel x_j - x_2 \parallel \\ d_{3,j} &= \parallel x_j - x_3 \parallel \\ d_{4,j} &= \parallel x_j - x_4 \parallel . \end{aligned}$$

These equations can be transformed into

$$d_{i,j}^2 = \| x_i \|^2 - 2x_i^T x_j + \| x_j \|^2 \qquad i = 1,2,3,4. \tag{8.9}$$

Subtracting each equation from the following one and expanding the inner product yields the system $Ax_j = b$ where

$$A = 2\begin{pmatrix} x_{11} - x_{21} & x_{12} - x_{22} & x_{13} - x_{23} \\ x_{21} - x_{31} & x_{22} - x_{32} & x_{23} - x_{33} \\ x_{31} - x_{41} & x_{32} - x_{42} & x_{33} - x_{43} \end{pmatrix} \tag{8.10}$$

and

$$b = \begin{pmatrix} \left(\| x_1 \|^2 - \| x_2 \|^2\right) - \left(d_{1,j}^2 - d_{2,j}^2\right) \\ \left(\| x_2 \|^2 - \| x_3 \|^2\right) - \left(d_{2,j}^2 - d_{3,j}^2\right) \\ \left(\| x_3 \|^2 - \| x_4 \|^2\right) - \left(d_{3,j}^2 - d_{4,j}^2\right) \end{pmatrix}. \tag{8.11}$$

Since $x_1, x_2, x_3,$ and $x_4$ are not in the same plane, $A$ must be nonsingular and $x_j$ can be determined uniquely. This process then continues until all the atoms are determined. The resulting solution yields a structure for the given protein. Dong and Wu [4] verified that the whole process finishes in time proportional to the number of atoms in the protein.

This technique has the advantage of being able to determine a protein structure efficiently in $\mathscr{O}(n)$ versus $\mathscr{O}\left(n^2\right)$ floating point operations by the matrix decomposition algorithm. However, the assumption on the availability of a full set of exact interatomic distances is not realistic in practice due to the limitations of NMR experiments as discussed above.

### 8.3.2 Geometric Buildup with a Sparse Set of Exact Distances

Recall, the DG problem for a sparse set $S$ of exact distances is to determine the coordinate vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ for $n$ atoms satisfying the distance equations

$$d_{i,j} = \| x_i - x_j \|, \qquad d_{i,j} \in S. \tag{8.12}$$

Wu and Wu [30] extended the work of Dong and Wu [4] and showed that a full set of exact interatomic distances is not needed in order to uniquely determine all the atoms. Instead, in each iteration, when an atom is to be determined, only four distances from this atom to four already determined atoms are required. Let $x_j$ denote the coordinates of the atom to be determined in the current iteration, and let $x_{i_1}, x_{i_2}, x_{i_3}, x_{i_4}$ denote four atoms not in the same plane such that $d_{i_1,j}, d_{i_2,j}, d_{i_3,j}, d_{i_4,j} \in S$. Thus, by a similar derivation in previous section,

$$d_{i_k,j}^2 = \| x_{i_k} \|^2 - 2x_{i_k}^T x_j + \| x_j \|^2, \qquad k = 1,2,3,4.$$

The coordinate $x_j$ is then obtained as the solution of the system $Ax_j = b$ where

$$A = -2 \begin{pmatrix} (x_{i_2} - x_{i_1})^T \\ (x_{i_3} - x_{i_2})^T \\ (x_{i_4} - x_{i_3})^T \end{pmatrix} \tag{8.13}$$

and

$$b = \begin{pmatrix} \left(d_{i_2,j}^2 - d_{i_1,j}^2\right) - (\| x_{i_2} \|^2 - \| x_{i_1} \|^2) \\ \left(d_{i_3,j}^2 - d_{i_2,j}^2\right) - (\| x_{i_3} \|^2 - \| x_{i_2} \|^2) \\ \left(d_{i_4,j}^2 - d_{i_3,j}^2\right) - (\| x_{i_4} \|^2 - \| x_{i_3} \|^2) \end{pmatrix}. \tag{8.14}$$

The above algorithm has the advantage of using any set of four atoms with four distances to the atom to be determined. This allows for the selection of atoms which are closer to the atom of interest. Therefore, the algorithm may use distances like those provided by the NMR experiments. In ideal cases, if the required distances are always available, only $4n$ distances are needed for the determination of all the atoms.

Wu and Wu [30] noted that their algorithm is still subject to error propagation, but they provide an updating scheme to minimize the effects of error propagation on the determined structure. The updating scheme takes advantage of the fact that the coordinates of any four atoms can be determined if the full set of distances among them is available in the given distance data using, for example, the matrix decomposition method.

Let $X = (x_1, x_2, x_3, x_4)^T$ denote the coordinate matrix for atoms 1, 2, 3, and 4 determined by using the given distance data among the atoms, and $Y = (y_1, y_2, y_3, y_4)^T$ denote the coordinate matrix for these atoms determined in the geometric buildup algorithm. The coordinate vectors derived from the given distance data are then aligned to their positions in the protein structure using a translation and rotation: The geometric centers

$$x_c^T = \sum_{i=1}^{4} X(i,:)/4, \qquad y_c^T = \sum_{i=1}^{4} Y(i,:)/4 \tag{8.15}$$

of both sets of coordinates are calculated. Then the translation

$$X = X + e\,(y_c - x_c)^T, \tag{8.16}$$

where $e = (1,1,1,1)^T$, is performed to align the geometric centers. Finally, a rotation is performed in order to finish aligning the two structures. The rotation matrix $Q$ is the solution of the optimization problem,

$$\begin{aligned} &\min_{Q} \| Y - XQ \|_F \\ &\text{s.t.} \quad QQ^T = I, \end{aligned} \tag{8.17}$$

which can be obtained by using the formula $Q = UV^T$, where $U$ and $V$ are the orthogonal matrices in the singular-value decomposition $Y^TX = U\Sigma V^T$.

The above algorithm represents a significant improvement over previous attempts to use a geometric buildup algorithm for protein structure determination, but it is still based on the assumption of exact interatomic distance data. Unfortunately, protein structure fluctuations limit the accuracy of the distance measurements generated by NMR experiments [3, 32].

### 8.3.3 Geometric Buildup with a Sparse Set of Inexact Distances

The assumption on the exactness of the distances is not realistic for the real distance data is often subject to all kinds of errors. The errors may cause the distances to be inconsistent, that is, the distances may not even satisfy the triangle inequality. This is especially problematic for the geometric buildup algorithm since different sets of four available distances to an atom may determine its position differently. Suppose in a particular iteration of the geometric buildup algorithm that there are $l$ distances available from $l$ determined atoms $x_{i_1}, x_{i_2}, \ldots, x_{i_l}$ to atom $x_j$. Then, $x_j$ must satisfy all these distance constraints, i.e.,

$$d_{i_k,j}^2 = \| x_{i_k} \|^2 - 2x_{i_k}^T x_j + \| x_j \|^2, \qquad k = 1, \ldots, l$$

Assume that the distances are consistent. The above system of equations can be reduced to a linear system $Ax_j = b$ where

$$A = -2 \begin{pmatrix} (x_{i_2} - x_{i_1})^T \\ (x_{i_3} - x_{i_2})^T \\ \vdots \\ (x_{i_l} - x_{i_{l-1}})^T \end{pmatrix}, \tag{8.18}$$

and

$$b = \begin{pmatrix} \left(d_{i_2,j}^2 - d_{i_1,j}^2\right) - \left(\| x_{i_2} \|^2 - \| x_{i_1} \|^2\right) \\ \left(d_{i_3,j}^2 - d_{i_2,j}^2\right) - \left(\| x_{i_3} \|^2 - \| x_{i_2} \|^2\right) \\ \vdots \\ \left(d_{i_l,j}^2 - d_{i_{l-1},j}^2\right) - \left(\| x_{i_l} \|^2 - \| x_{i_{l-1}} \|^2\right) \end{pmatrix}. \tag{8.19}$$

The above system is overdetermined since $l \geq 4$, and a solution may fail to exist if the distances are inconsistent. An approximate solution may be obtained by solving a least-squares problem **min** $\| b - Ax_j \|$. However, the least-squares solution may contain large errors, which will be propagated in the buildup process, resulting in an incorrect structure.

Sit and Wu [25] developed a new buildup algorithm for solving a DG problem with a sparse set of inexact distances. In each buildup iteration, instead of obtaining an approximate solution to the above overdetermined linear system, the new algorithm tries to determine the coordinate vector $x_j$ as well as the coordinate vectors $x_{i_1}, \dots, x_{i_l}$ all together, using the matrix decomposition method: First, $x_j$ is set to the origin. Then, a distance matrix $D = \{(d_{i_r,j}^2 - d_{i_r,i_s}^2 + d_{i_s,j}^2)/2 : \ r,s = 1,\dots,l\}$ is formed. Let $X = \{x_{i_k,m} : \ k = 1,\dots,l; \ m = 1,2,3\}$ be the coordinate matrix of atoms $i_1,\dots,i_l$. Let $D = U\Sigma U^T$ be the singular-value decomposition of $D$. Then, $X = V\Lambda^{1/2}$, where $V = U(:,1:3)$ and $\Lambda = \Sigma(1:3,1:3)$.

This new algorithm has the following distinctive features: First, in each buildup iteration, it uses all $l$ determined atoms that have distances with atom $j$. The latter is determined to satisfy all $l$ distance constraints. Second, the distances among the $l$ determined atoms and atom $j$ are all used to determine its position. Third, since the matrix decomposition method is used, the coordinate vectors for atom $j$ as well as the $l$ determined atoms are all determined (or redetermined). Finally, the coordinate vectors are determined in a separate reference system with atom $j$ located at the origin. Therefore, the algorithm needs to translate and rotate the newly determined $l+1$ coordinate vectors so the $l$ determined atoms are aligned with their old positions as best as possible (in terms of their coordinate root-mean-squared deviations (RMSD) [26]).

In any case, in each buildup iteration, the new algorithm can find an approximate solution to the system of distance equations for the atoms to be determined. The solution is the best possible approximation in a least-squares sense, as described in the matrix decomposition method. Therefore, the algorithm can tolerate small errors (or in other words, small inconsistencies) in the distances. Yet, the algorithm can prevent the rounding errors from building up for in every iteration, the coordinate vectors for the already determined atoms are recalculated using available distances and their errors due to previous calculations are "corrected." This algorithm is thus far the best buildup algorithm for solving a DG problem with a sparse set of inexact distances [25].

## 8.4 The Discretizable Molecular Distance Geometry Problem

Recall the DG problem for a sparse set $S$ of exact distances Eq. (8.3.1). Lavor, Liberti, Mucherino, and various collaborators [16–18] have recently published a series of works that have demonstrated a significant improvement over numerous aspects of Wu and Wu's [30] geometric buildup algorithm. Their approach to the DG problem for a sparse set of distance data, called the discretizable molecular distance geometry problem (DMDGP), is based on the fact that a consistent set of sparse distance data which meets certain conditions can only have a finite number of feasible protein conformations as its solutions [16].

**Definition 8.5.** Suppose there exists a sequence of atoms, $\{1,2,\ldots,n\}$ and a consistent subset $S$ of the distance data such that:

1. The distances $d_{1,2}, d_{1,3}$, and $d_{2,3} \in S$.
2. For every atom $i \geq 4$, the distances $d_{i-3,i}, d_{i-2,i}$, and $d_{i-1,i} \in S$.
3. For every atom $i \geq 4$, the distances $d_{i-3,i}, d_{i-2,i}$, and $d_{i-1,i} \in S$ satisfy the strict triangle inequality: $d_{i-3,i-1} < d_{i-3,i-2} + d_{i-2,i-1}$.

Then the DMDGP is to determine all possible conformations which satisfy the distances in $S$ [17].

The DMDGP has two main advantages over the DG problem for a sparse set of exact distances. First, the DMDGP determines a set of feasible structures instead of the single solution returned by Wu and Wu's formulation of the problem. Second, the DMDGP requires only three distances, instead of the four-distance requirement of the DG problem for a sparse set of exact distances. This is a significant development over previous work since the four-distance restriction of geometric buildup algorithms lead to instances in which they fail to find a valid conformation from a consistent set of distances [16]. However, the relaxation on the number of required distance leads to instances in which two valid positions exist for multiple atoms within the structure [17]. Thus, any algorithm implemented to solve the DMDGP must be able to account for this possibility. To this end, Liberti et al. [18] have developed a recursive algorithm, called the branch-and-prune (BP) algorithm, which utilizes the given atomic ordering to represent the solutions of the DMDGP as directed binary tree.

The initial step of the process uses the distances among the first three atoms to determine their positions, denoted $x_1, x_2$, and $x_3$, prior to the first call to the BP algorithm. Following the derivation of Liberti et al. [18], each recursive call to the BP algorithm determines up to two possibilities for the coordinate $x_i = (x_{i,1},\ x_{i,2},\ x_{i,3})\, i = 4,5,\ldots,n$ using the distances

$$\begin{aligned} \| x_{i-1} - x_i \|^2 &= d_{i-1,i}^2 \\ \| x_{i-2} - x_i \|^2 &= d_{i-2,i}^2 \\ \| x_{i-3} - x_i \|^2 &= d_{i-3,i}^2. \end{aligned}$$

Subtracting the third equation from the first and second equations yields

$$\begin{aligned} 2\,(x_{i-1} - x_{i-3}) \cdot x_i &= \left(\| x_{i-1} \|^2 - \quad d_{i-1,i}^2\right) - \left(\| x_{i-3} \|^2 - \quad d_{i-3,i}^2\right) \\ 2\,(x_{i-2} - x_{i-3}) \cdot x_i &= \left(\| x_{i-2} \|^2 - \quad d_{i-2,i}^2\right) - \left(\| x_{i-3} \|^2 - \quad d_{i-3,i}^2\right) \\ \| x_{i-3} - x_i \|^2 &= d_{i-3,i}^2. \end{aligned}$$

The first two equations can be represented as the system $Ax_i = b$ with

$$A = 2\begin{pmatrix} x_{i-1,1} - x_{i-3,1} & x_{i-1,2} - x_{i-3,2} & x_{i-1,3} - x_{i-3,3} \\ x_{i-2,1} - x_{i-3,1} & x_{i-2,2} - x_{i-3,2} & x_{i-2,3} - x_{i-3,3} \end{pmatrix} \tag{8.20}$$

and

$$b = \begin{pmatrix} \left(\| x_{i-1} \|^2 - \quad d_{i-1,i}^2\right) - \left(\| x_{i-3} \|^2 - \quad d_{i-3,i}^2\right) \\ \left(\| x_{i-2} \|^2 - \quad d_{i-2,i}^2\right) - \left(\| x_{i-3} \|^2 - \quad d_{i-3,i}^2\right) \end{pmatrix} \tag{8.21}$$

where the matrix $A \in \mathbb{R}^{2\times 3}$ is of full rank by the strict triangle inequality [18]. Thus, the matrix can be partitioned into two basic columns, denoted $A_B$, and a nonbasic column, denoted $A_N$, with corresponding partitioning of the variables into $x_{i_B}$ and $x_{i_N}$. The coordinates for the basic variables are then determined by the matrix equation $x_{i_B} = A_B^{-1}(b - A_N x_{i_N})$. Plugging these values into $\| x_{i-3} - x_i \|^2 = d_{i-3,i}^2$ yields a quadratic equation in $x_{i_N}$ with either one or current node. The solutions are then added to the binary tree, as the children of the node called the BP algorithm. The above process continues, with any resulting children recursively performing instances of BP algorithm, until children are placed at the $n^{th}$ level of the tree.

While it may be the case that there are no addition distances in $S$, the complex nature of protein structures generally yield additional distance information that can be used to prune infeasible branches from the BP search tree [16]. Thus, consider the set of distance $\{d_{j,i} \in S : 1 \le j \le i-4\}$ for an instance of the BP algorithm on the $i$th atom, and let $x_i$ denote a possible position returned as the solutions of the above derivation. While numerous pruning devices exist for pruning infeasible branches out of the BP search tree [16], we restrict our discussion to the direct distance feasibility (DDF) pruning device. The DDF pruning device deems a position $x_i$ infeasible if $| \, \| x_j - x_i \| - \quad d_{j,i}| > \varepsilon$ for a constant tolerance $\varepsilon > 0$ [17]. The DDF pruning device is implemented at each instance of the BP algorithm, yielding a significant reduction in the amount of CPU time required to execute the BP algorithm [16].

The above algorithm represents a significant improvement over previous work aimed at obtaining feasible protein conformations from a sparse set of exact distance data. Unfortunately, NMR experiments yield interval distance data, which reflect natural fluctuations in protein structures. Thus, the assumption of exact interatomic distance data represents a significant limitation. Fortunately, the authors freely admit and are actively pursuing mechanisms which resolve this issue [16, 18].

## 8.5 The Generalized DG Problem

In NMR structure determination, due to the fluctuation of the structure, the interatomic distances can only be estimated in certain ranges. Given a set $S$ of distance ranges, a more practical DG problem is to find the coordinate vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ for $n$ atoms satisfying

$$l_{i,j} \le \| x_i - x_j \| \le u_{i,j}, \qquad d_{i,j} \in S \tag{8.22}$$

where $l_{i,j}$ and $u_{i,j}$ denote the lower and upper bounds for the range of the distance $d_{i,j} = \| x_i - x_j \|$.

Several approaches have been proposed to the above problem, such as the embedding algorithm by Crippen and Havel [3, 10] and the global optimization method by Moré and Wu [19, 20], as we have discussed in previous sections. Unfortunately, the arbitrariness of the positions of the atoms makes it difficult to implement a geometric buildup algorithm for this problem. These difficulties are due to the fact that any incorrect placement of atoms in the early buildup stages may result in infeasible or unrealizable atoms in later stages [26].

In order to determine a meaningful set of solutions to the DG problem with distance bounds, Sit and Wu [26] formulated another set of DG problems called the generalized DG problem. Instead of solving a system of inequalities (8.22), a generalized DG problem determines an equilibrium position as well as a maximum possible fluctuation radius for each atom as long as the distances among the atoms are within the given ranges. Therefore, if the distance ranges are not flexible, i.e., their lower and upper bounds are equal, then the generalized DG problem is equivalent to a regular DG problem with exact distances. If a true set of distance bounds is given, an equilibrium structure and its possible fluctuations within the given distance bounds can be found by solving a generalized DG problem.

**Definition 8.6.** A generalized DG problem is to determine a set of coordinate vectors $x_1, x_2, \ldots, x_n$ for $n$ atoms with maximum fluctuation radii $r_1, r_2, \ldots, r_n$ such that all possible atomic locations do not violate any of the given distance bounds constraints.

**Definition 8.7.** A generalized DG problem can be represented as the constrained optimization problem:

$$
\begin{array}{ll}
\max\limits_{x_i, r_i} \sum\limits_{i=1}^{n} r_i & \\
\text{s.t.} & \\
\| x_i - x_j \| + r_i + r_j \le u_{i,j} & \\
\| x_i - x_j \| - r_i - r_j \ge l_{i,j}, & d_{i,j} \in S \\
r_i \ge 0, \qquad i = 1, 2, \ldots, n &
\end{array}
$$

where $l_{i,j}$ and $u_{i,j}$ denote the lower and upper bounds for the ranges of the distances $d_{i,j} = \| x_i - x_j \|$ in some distance set $S$.

The generalized DG problem is a better defined problem than the DG problem with distance bounds. The latter has an ensemble of solutions which is almost impossible to determine, while the generalized DG problem requires only a single solution. Yet, with the maximum possible fluctuation radii for all the atoms, this solution can provide a sufficient description for the fluctuations of the structure, as usually demonstrated by a set of samples in the ensemble of solutions to the DG problem with distance bounds. This property makes the generalized DG problem a better formulation for the protein structure determination problem with NMR distance bounds. In addition, it can represent an NMR structure in a similar way as an X-ray crystal structure with a single structural file with the atomic fluctuation radii listed as B-factors [5].

The solution to the generalized DG problem (8.23) can be obtained by solving a constrained optimization problem, using available optimization methods [22]. On the other hand, the DG problem with distance bounds Eq. (8.22) requires the solution to a system of nonlinear inequalities, which is computationally intractable in general, even if the inequalities are linear [21]. The generalized DG problem has a large set of variables and constraints and is by no mean trivial to solve. For example, suppose that a protein has 1,000 atoms. Then the corresponding optimization problem will have 4,000 variables, 1,000 for the atomic fluctuation radii and 3,000 for the coordinates of the atoms, and up to 999,000 constraints if all of the pairwise interatomic distances are known. Solving such a large constrained optimization problem is still very challenging.

### 8.5.1 Geometric Buildup Approach

In order to avoid directly solving a large and complex optimization problem as given in Eq. (8.23), Sit and Wu [26] developed a geometric buildup algorithm for the problem. Their algorithm first finds four atoms not in the same plane among which the interatomic distances are given. A coordinate system for the problem is then established using these four atoms Each of the remaining atoms are then determined, one at a time, by solving the following subproblem:

$$\begin{aligned}
&\max_{x_{l+1},r_{l+1}} \; r_{l+1} \\
&\text{s.t.} \\
&\| \, x_i - x_{l+1} \, \| + r_i + r_{l+1} \le u_{i,l+1} \\
&\| \, x_i - x_{l+1} \, \| - r_i - r_{l+1} \ge l_{i,l+1} \\
&r_i \ge 0, \qquad\qquad i = 1,2,\ldots,l
\end{aligned} \tag{8.23}$$

where $(x_{l+1}, r_{l+1})$ are the coordinate vector and fluctuation radius to be determined for a new atom, and $(x_i, r_i)$, $i = 1,\ldots,l$ are assumed to be the coordinate vectors and fluctuation radii of $l$ determined atoms, $l \ge 4$, from which there are given distance bounds to the new atom. This subproblem has only four variables and $2l$ constraints, and is easy to solve by calling a conventional constrained optimization routine.

Sit and Wu [26] applied the above buildup procedure to a dozen of model protein problems. Each of these problems is generated from a known protein structure determined by X-ray crystallography utilizing interatomic distances that are less than or equal to 5Å in the known structure. A pair of lower and upper bounds is computed for each distance by subtracting and adding the atomic fluctuation radii derived from the atomic B-factors. A solution to the corresponding DG problem, based on the distance bounds described above, is determined by solving a sequence of generalized DG subproblems. Sit and Wu [26] showed that the solutions yield a

**Table 8.1** RMSD error of calculated protein structures using the buildup algorithm for the generalized DG problem[a]

| Unperturbed distance bounds ($\leq$ 5Å) | | | | | | |
|---|---|---|---|---|---|---|
| Protein ID | 1PTQ | 1HOE | 1LFB | 1PHT | 1POA | 1AX8 |
| RMSD | 1.0e-13 | 7.1e-14 | 4.6e-12 | 2.5e-13 | 3.2e-12 | 2.1e-11 |
| B-factor corr | 0.9857 | 0.9692 | 0.9960 | 0.9903 | 0.9592 | 0.9940 |
| Protein ID | 4MBA | 1F39 | 1RGS | 1BPM | 1HMV | |
| RMSD | 7.8e-12 | 1.3e-12 | 1.0e-09 | 1.3e-11 | 1.7e-04 | |
| B-factor corr | 0.9815 | 0.9976 | 0.9786 | 0.9781 | 0.9904 | |
| 0.3% Perturbed distance bounds ($\leq$ 5Å) | | | | | | |
| Protein ID | 1PTQ | 1HOE | 1LFB | 1PHT | 1POA | 1AX8 |
| RMSD | 5.0e-02 | 2.0e-01 | 1.4e-01 | 3.5e+00 | 4.0e-01 | 3.0e+00 |
| B-factor corr | 0.7396 | 0.6810 | 0.5698 | 0.7330 | 0.8348 | 0.6649 |
| Protein ID | 4MBA | 1F39 | 1RGS | 1BPM | 1HMV | |
| RMSD | 6.2e-02 | 1.6e+01 | 1.4e+01 | 7.4e-01 | 1.3e+01 | |
| B-factor corr | 0.8274 | 0.5683 | 0.9052 | 0.8770 | 0.7770 | |

[a] Data taken from Sit and Wu [26]

set of coordinate vectors which almost exactly match the original structures. They also demonstrated that the atomic fluctuation radii for each protein also correlate well with the atomic B-factors of the original structure.

In some sense, however, the above model problems were generated carefully so that there always exists an equilibrium structure for any of these sets of distance bounds. The distance bounds are also assumed to be the distances plus or minus the atomic fluctuation radii derived from the atomic B-factors. These conditions make the corresponding generalized DG problems relatively easy to solve, but they cannot always be guaranteed in practice. In order to make the problems more realistic (or more like those from NMR experiments), Sit and Wu [26] also made some random perturbations on the generated distance bounds. The algorithm still succeeded for small bound perturbations, but failed to converge when the perturbations are large (see Table 8.1).

### *8.5.2 Direct Optimization Approach*

Due to the limitation of memory storage and computational power, a divide and conquer approach is necessary to determine the structures of large proteins. The geometric buildup algorithms are one implementation of such an idea. While it is successful in avoiding large computations, its efficiency comes at a high cost: All geometric buildup algorithms suffer from the accumulation of rounding errors. These errors prevent the accurate calculations of large protein structures. The algorithms may be improved if their sequential nature can be reduced, for example, by determining multiple atoms in each buildup stage whenever possible. An alternative approach would be solving the generalized DG problem directly using an optimization algorithm.

**Table 8.2** RMSD error of calculated protein structures using the direct optimization approach to the generalized DG problem[a]

| Unperturbed distance bounds (≤ 5Å) | | | | | | |
|---|---|---|---|---|---|---|
| Protein ID | 1PTQ | 1HOE | 1LFB | 1PHT | 1POA | 1AX8 |
| Total atoms | 402 | 558 | 641 | 809 | 914 | 1003 |
| RMSD | 8.4e-03 | 3.2e-02 | 1.7e-02 | 7.3e-02 | 7.1e-02 | 1.5e-01 |
| B-factor corr | 0.9978 | 0.9853 | 0.9904 | 0.9559 | 0.9721 | 0.8968 |
| Protein ID | 4MBA | 1F39 | 1RGS | 1BPM | 1HMV | |
| Total atoms | 1083 | 1534 | 2010 | 3669 | 7389 | |
| RMSD | 1.9e-02 | 2.2e-02 | 9.2e-02 | 1.9e-02 | 4.6e-01 | |
| B-factor corr | 0.9948 | 0.9851 | 0.9930 | 0.9954 | 0.9386 | |
| 5% Perturbed distance bounds (≤ 5Å) | | | | | | |
| Protein ID | 1PTQ | 1HOE | 1LFB | 1PHT | 1POA | 1AX8 |
| Total atoms | 402 | 558 | 641 | 809 | 914 | 1003 |
| RMSD | 1.4e-01 | 1.9e-01 | 2.2e-01 | 2.5e-01 | 2.2e-01 | 3.5e-01 |
| B-factor corr | 0.9832 | 0.9801 | 0.9673 | 0.9349 | 0.9827 | 0.8442 |
| Protein ID | 4MBA | 1F39 | 1RGS | 1BPM | 1HMV | |
| Total atoms | 1083 | 1534 | 2010 | 3669 | 7389 | |
| RMSD | 1.8e-01 | 2.4e-01 | 2.3e-01 | 1.7e-01 | 7.1e-01 | |
| B-factor corr | 0.9841 | 0.9510 | 0.9764 | 0.9852 | 0.7856 | |

[a] Data taken from Voller and Wu [29]

As we have mentioned before, the optimization problem related to the generalized DG problem is not trivial to solve directly. It may have tens of thousands of variables and constraints for regular-sized proteins, which can be difficult even for obtaining a local optimal solution. Nonetheless, Voller and Wu [29] have recently implemented an optimization algorithm in Matlab for the solution of the generalized DG problem. They have tested this algorithm on the same problem set in [26] using up to the first 1,000 atoms in each structure. The results showed that the algorithm was able to solve the problems successfully. In particular, because it does not have large error accumulations, the algorithm can successfully solve the problems with large perturbations on the distance bounds [29].

The results from using the direct optimization method are shown in Table 8.2. The full structures were not considered for some cases when the total number of atoms exceeded 1,000, for otherwise, the problem size was too large to handle by the initial implementation of the algorithm. The computation was conducted using Matlab R2010b on an Ubuntu 10.04 LTS Linux machine. The optimization was performed using the function fmincon() from Matlab's Optimization Toolbox. The gradients for the objective function and constraints were defined with particular attentions paid to memory saving techniques, and the Hessian matrix was approximated using Matlab's built-in quasi-Newton approximation.

The results in Table 8.2 show that for unperturbed distance bounds, the computed structures are reasonable approximations to the original structures, based on their RMSD values, and the computed atomic fluctuation radii correlated well with those derived from the B-factors of the original structures. Most importantly, for perturbed distance bounds, even with the perturbations as large as 5%, the RMSD values of the computed structures and the original structures are reasonably small, and the computed atomic fluctuation radii still correlate well with those derived from the B-factors of the original structures. These results are certainly more stable than the geometric buildup method for the latter fails when the perturbation is beyond 0.3%.

The advantage of using the direct optimization method, as a more stable algorithm than the geometric buildup algorithm, is now evidenced in these preliminary test results: The direct optimization method searches for the solution to the problem in the entire variable space, guided by the first and second-order information on the objective function and the constraints. It therefore has a stable convergence property. But the geometric buildup algorithm is more or less like a coordinate direction search algorithm: It fixes the values for the variables in sequence, one at a time. Without repeating or, in other words, backtracking, the algorithm is not guaranteed to converge. The direct optimization method, in contrast, is very expensive to implement. For each of the test problems in Table 8.2, the method runs typically more than one hour on the testing machine. Therefore, when a real-sized problem is solved, the required computation may not be affordable on a regular laptop or desktop computer. However, by employing more efficient optimization algorithms and more powerful computers such as parallel high-performance computers, there is a hope to obtain a stable as well as affordable solution to the generalized DG problem using a direct optimization approach.

## 8.6 Concluding Remarks

The solution to the DG problem, in whatever form, has been a central issue in NMR protein structure determination [3]. It has been a critical part of the Nobel Prize winning work on NMR protein structure determination done by Wuthrich and coworkers [33]. Yet, the problem remains not clearly defined and not well solved. The main reason is of course the difficult nature of the problem in many of its forms. But, it may also be due to the lack of the codevelopment of the solution methods with their applications to NMR modeling practices: In fact, there have not been any significantly more effective methods developed in the past twenty years. The NMR modeling community has not been actively adopting any new computational advances, either. Nonetheless, with the further expansion of the NMR technology for structure determination and the development of more efficient and accurate methods, it is foreseeable that in next ten or twenty years, a meaningful solution to the DG problem will be obtained, with which a great impact will be made in future NMR protein structure determination.

In this chapter, we have reviewed some recent advances in solving the DG problem in its various forms. We have focused on the development of a geometric buildup approach to the problem with sparse but exact distances and on the formulation of a generalized DG problem for the determination of structural ensembles with inexact distances or distance bounds. We have described the novel ideas of these approaches, showed their potentials for the solution of large-scale problems in practice, and discussed their possible future developments. For some historical background, we have also provided a brief introduction to the classical matrix decomposition method, the embedding algorithm, and the global smoothing algorithm for the solution of the DG problems with exact and inexact distances. Many other methods have been developed. We have not covered them all, but referred the readers to a list of papers, hoping to provide the readers with a relatively complete knowledge of the field.

## References

1. Biswas, P., Toh, K.C., Ye, Y.: A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. SIAM J. Sci. Comput. **30**, 1251–1277 (2008)
2. Blumenthal, L.M.: Theory and Applications of Distance Geometry. Oxford Clarendon Press, London (1953)
3. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
4. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. J. Global. Optim. **22**, 365–375 (2002)
5. Drenth, J.: Principals of Protein X-ray Crystallography. Springer, Berlin (2006)
6. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika **1**, 211–218 (1936)
7. Glunt, W., Hayden, T.L., Hong, S., Wells, J.: An alternating projection algorithm for computing the nearest Euclidean distance matrix. SIAM J. Matrix. Anal. Appl. **11**, 589–600 (1990)
8. Glunt, W., Hayden, T.L., Liu, W.: The embedding problem for predistance matrices. Bulletin of Mathematical Biology **53**, 769–796 (1991)
9. Grosso, A., Locatelli, M., Schoen, F.: Solving molecular distance geometry problems by global optimization algorithms. Comput. Optim. Appl. **43**, 23–27 (2009)
10. Havel, T., Distance geometry. In: Grant, D., Harris, R. (eds.) Encyclopedia of Nuclear Magnetic Resonance, pp. 1701–1710. Wiley, New York (1995)
11. Hendrickson, B.: Conditions for unique realizations. SIAM J. Comput. **21**, 65–84 (1992)
12. Hendrickson, B.: The molecule problem: Exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
13. Hoai An, L.T., Tao, P.D.: Large scale molecular optimization from distance matrices by D.C. optimization approach. SIAM J. Optim. **14**, 77–114 (2003)
14. Kearsly, A., Tapia, R., Trosset, M.: Solution of the metric STRESS and SSTRESS problems in multidimensional scaling by Newton's method. Comput. Stat. **13**, 369–396 (1998)
15. Kostrowicki, J., Piela, L.: Diffusion equation method of global optimization: performance for standard test functions. J. Optim. Theor. Appl. **69**, 269–284 (1991)
16. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
17. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
18. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2011)
19. Moré, J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**, 814–836 (1997)

20. Moré, J., Wu, Z.: Distance geometry optimization for protein structures. J. Global Optim. **15**, 219–234 (1999)
21. Neumaier, A.: Interval Methods for Systems of Equations. Cambridge University Press, New York (1990)
22. Nocedal, J., Wright, S.: Numerical Optimization. Springer, New York (2002)
23. Piela, L., Kostrowicki, J., Scheraga, H.A.: The multiple-minima problem in the conformational analysis: deformation of the potential energy hyper-surface by the diffusion equation method. J. Phys. Chem. **93**, 3339–3346 (1989)
24. Saxe, J.: Embeddability of weighted graphs in k-space is strongly NP-hard. In: Proceedings of the 17th Allerton Conference in Communications, Control, and Computing, pp. 480–489 (1979)
25. Sit, A., Wu, Z., Yuan, Y.: A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation. Bulletin of Mathematical Biology **71**, 1914–1933 (2009)
26. Sit, A., Wu, Z.: Solving a generalized distance geometry problem for protein structure determination. Bulletin of Mathematical Biology **73**(8), 1932–1951 (2011)
27. Torgerson, W.S.: Theory and Method of Scaling. Wiley, New York (1958)
28. Trosset, M.: Applications of multidimensional scaling to molecular conformation. Comput. Sci. Stat. **29**, 148–152 (1998)
29. Voller, Z., Wu, Z.: Direct optimization approach to the generalized distance geometry problem, to be submitted (2012)
30. Wu, D., Wu, Z.: An updated geometric buildup algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global Optim. **37**, 321–333 (2007)
31. Wu, Z.: The effective energy transformation scheme as a special continuation approach to global optimization with application in molecular conformation. SIAM J. Optim. **6**, 748–768 (1996)
32. Wüthrich, K.: NMR in Structural Biology. World Scientific, New York (1995)
33. Wüthrich, K.: NMR studies of structure and function of biological macromolecules. Nobel Lectures, Nobel Organizations (2002)

# Chapter 9
# Solving the Discretizable Molecular Distance Geometry Problem by Multiple Realization Trees

**Pedro Nucci, Loana Tito Nogueira, and Carlile Lavor**

**Abstract** The discretizable molecular distance geometry problem (DMDGP) is a subclass of the MDGP, where instances can be solved using a discrete algorithm called branch-and-prune (BP). We present an initial study showing that the BP algorithm can be used differently from its original form, where the initial atoms are fixed and the branches of the BP tree are generated until the last atom is reached. Particularly, we show that the use of multiple BP trees may explore the search space faster than the original BP.

**Keywords** Distance geometry • Branch-and-prune • Realization tree

## 9.1 Introduction

The molecular distance geometry problem (MDGP) basically consists of obtaining all feasible three-dimensional structures for a molecule when some of its interatomic distances are given [2,4,6,8]. For the case when all interatomic distances are known, the problem can be solved in linear time [3]. Otherwise, the problem is classified as NP-hard [11].

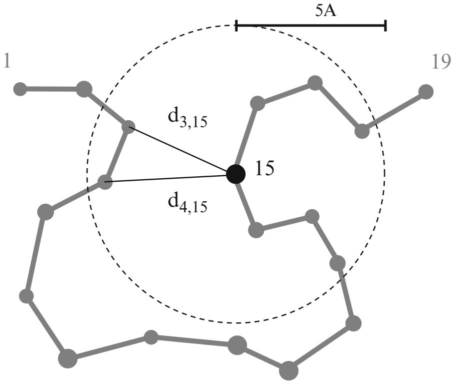
P. Nucci (✉)
Navy Arsenal of Rio de Janeiro, Brazilian Navy, Brazil
e-mail: pedro.nucci@amrj.mar.mil.br

L.T. Nogueira
Instituto de Computação, Universidade Federal Fluminense, Rio de Janeiro, Brazil
e-mail: loana@ic.uff.br

C. Lavor
IMECC-UNICAMP, Campinas-SP, Brazil
e-mail: clavor@ime.unicamp.br

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_9,

**Fig. 9.1** *F*-type distances



Formally, the MDGP can be described as follows. Given an atomic sequence $1, 2, \ldots, n$, and a set $S$ of all pairs of atoms $(i, j)$ such that the distance $d_{ij}$ is known, find the feasible Cartesian coordinates $x_1, \ldots, x_n \in \mathbb{R}^3$ of the atomic sequence (which can be seen as a linear sequence of bonded atoms) so that

$$\|x_i - x_j\| = d_{ij}, \quad \forall (i, j) \in S. \tag{9.1}$$

By supposing the validity of some properties for the known interatomic distance set (usually compatible with proteins, a very important class of macromolecules), the problem has a discrete search space and is called discretizable molecular distance geometry problem (DMDGP) [5].

The following assumptions turn the MDGP into a combinatorial problem (DMDGP), for a given atomic ordering:

1. $d_{ij}$ is known for all pairs of atoms $(i, j)$, with $1 \le j - i \le 3$
2. Angles between vectors $(x_{i+2} - x_{i+1})$ and $(x_{i+1} - x_i)$, where $1 \le i \le n-2$, are never a multiple of $\pi$.

The set $S$ of pair of atoms with known distances may be partitioned in two subsets: the set $E$, corresponding to all pairs of atoms $(i, j)$, where $1 \le j - i \le 3$, and the set $F$ of all pairs of atoms $(i, j)$, where $j - i > 3$ (see Fig. 9.1).

In [7], a branch-and-prune (BP) algorithm has been proposed for solving the DMDGP. In this chapter, we provide an alternative way for solving this problem, making use of the BP algorithm.

The main idea in the BP algorithm is to explore the search space by using torsion matrices of each atom and to eliminate the infeasible positions as soon as possible. The torsion matrix $B_i$ related to the atom $i$ can be calculated as follows:

$$B_i = \begin{bmatrix} -\cos\theta_{i-2,i} & -\sin\theta_{i-2,i} & 0 & -d_{i-1,i}\cos\theta_{i-2,i} \\ \sin\theta_{i-2,i}\cos\omega_{i-3,i} & -\cos\theta_{i-2,i}\cos\omega_{i-3,i} & -\sin\omega_{i-3,i} & d_{i-1,i}\sin\theta_{i-2,i}\cos\omega_{i-3,i} \\ \sin\theta_{i-2,i}\sin\omega_{i-3,i} & -\cos\theta_{i-2,i}\sin\omega_{i-3,i} & \cos\omega_{i-3,i} & d_{i-1,i}\sin\theta_{i-2,i}\sin\omega_{i-3,i} \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

for $i \geq 4$, where $\omega_{i-3,i}$ is a torsion angle, $\theta_{i-2,i}$ is a bond angle, and $d_{i-1,i}$ is a bond length. Matrices $B_1$, $B_2$, and $B_3$ are given by

$$B_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B_2 = \begin{bmatrix} -1 & 0 & 0 & -d_{1,2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$B_3 = \begin{bmatrix} -\cos\theta_3 & -\sin\theta_3 & 0 & -d_{2,3}\cos\theta_3 \\ \sin\theta_3 & -\cos\theta_3 & 0 & d_{2,3}\cos\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{9.2}$$

With the product $B_1 B_2 \dots B_i$, one can easily obtain the positions for the atom $i$ which is consistent with all E-type distances. Each atom $i \geq 4$ has two possible torsion matrices $B_i$ (calculated by using $\sin\omega_{i-3,i} = +\sqrt{(1-\cos^2\omega_{i-3,i})}$) or $B_i'$ (calculated by using $\sin\omega_{i-3,i} = -\sqrt{(1-\cos^2\omega_{i-3,i})}$). The first three atoms have only one torsion matrix, which implies that there are $2^{i-3}$ feasible coordinates, a priori, for each atom $i > 3$. The BP algorithm behaves like a tree search algorithm (such as depth or breadth-first search): at each level $i$, each torsion matrix $B_i$ and $B_i'$ is multiplied by the previous matrix product $B_1 \dots B_{i-1}$, thus providing us with two positions $x_i$ and $x_i'$ (*branching*), and positions that are not consistent (according to a constant error tolerance $\varepsilon$) with the related F-type distances are discarded (*pruning*).

We introduce now some definitions. Let $M$ be a molecule defined by a sequence of $n$ atoms $1, 2, \dots, n$. An *interval* $[a,b]$ of $M$ is any subsequence $\{a, \dots, b\}$ of atoms of $M$, with $1 \leq a \leq b \leq n$. The size of $[a,b]$ is defined by $b-a$. A *realization* $R_{a,b}$ is a function $R_{a,b} : [a,b] \mapsto \mathbb{R}^3$ that associates each atom of an interval to a point in $\mathbb{R}^3$. We say that $R_{a,b}$ is *infeasible* for a given instance of the DMDGP when $\exists (i,j) \in S$ such that $i, j \in [a,b]$ and $d_{ij} \neq ||R_{a,b}(j) - R_{a,b}(i)||$; otherwise, it is *feasible*. $R_{a,b}$ is *complete* if $[a,b] = [1,n]$ otherwise, it is *partial*. The idea of working with partial realizations was previously exploited in [10], for investigating a conjecture on the DMDGP, and in [9], for the development of a parallel version of the BP algorithm, even though no formal definitions were given in the latter.

A *realization tree* $T_{a,b}$ is a rooted tree with two properties:

1. Each level $k$ of $T_{a,b}$ corresponds to one atom of the interval $[a,b]$, given by $atom(k)$.
2. Each node at level $k$ contains a coordinate vector for $atom(k)$ corresponding to the atom of the root node.

We use $\left\|T_{a,b}\right\|$ to denote the number of nodes in $T_{a,b}$. We also use $T_{a,b}^{+}$ and $T_{a,b}^{-}$ to denote, respectively, the tree growth direction from left to right and from right to left. As it can be seen, the BP algorithm (given in Algorithm 2) yields a realization tree containing all nodes visited by the algorithm.

**Algorithm 2** The Branch-and-Prune algorithm

```
Branch-And-Prune($T, node$)
$j \leftarrow$ Level($n_0$)
if ($j < n$) then
    Compute torsion matrices $\mathbf{B_j}$ and $\mathbf{B'_j}$, by using 9.1
    Obtain the accumulative torsion matrix $\mathbf{C_{j-1}}$ of Father(node)
    $C_j \leftarrow C_{j-1}B_j;\ C'_j \leftarrow C_{j-1}B'_j$
    $x_j \leftarrow C_j\ [0,0,0,1]^T; x'_j \leftarrow C'_j[0,0,0,1]^T$
    // test $F$-type distances related to $j$
    valid $\leftarrow$ true; valid$'$ $\leftarrow$ true
    for $((i,j) \in F)$ do
        if $((\|x_j - x_i\|^2 - d^2_{i,j})^2 > \varepsilon)$ then
            valid $\leftarrow$ false
        end if
        if $((\|x_j{}' - x_i\|^2 - d^2_{i,j})^2 > \varepsilon)$ then
            valid’ $\leftarrow$ false
        end if
    end for
    // create node with valid positions
    if (valid) then
        create node $z$ containing $C_j$ and $x_j$
        mark $z$ as son of node
        mark node as father of $z$
        $T = T \cup \{z\}$
        $BranchAndPrune(T, z)$
    end if
    if (valid’) then
        create node $z'$ obtaining $C_j{}'$ and $x_j$
        mark $z'$ as son of node
        mark node as father of $z'$
        $T = T \cup \{z'\}$
        $BranchAndPrune(T, z')$
    end if
else
    for (each leaf node of $T$) do
        solution stored in parent nodes from level n to 1, output by back-traversal
    end for
end if
```

In the next sections, we will use more than one realization tree to solve the DMDGP, showing that this strategy can improve the BP algorithm performance. The rest of this work is structured as follows. In Sect. 9.2, we motivate alternative uses of the BP algorithm through one simple theoretical example. In sects. 9.3 and 9.4, we present a technique for merging realization trees, which is the main contribution of this work. Section 9.5 provides a heuristic method that controls the growth of the realization trees. In Sect. 9.6, we describe a methodology and show some computational experiments where the presented techniques are considered. Section 9.7 provides our conclusions.

## 9.2 BP May Be Used in Different Ways

One must notice that the BP algorithm may be used to explore the search space by other means than the original procedure where the initial atoms of the sequence are fixed and branching on the tree is performed until the last atom is positioned. We are going to show that the BP algorithm presented in [7] actually provides a framework for solving DMDGP instances in various ways. Despite DMDGP is already proven to be NP-hard [5], it is still important to know how to solve its instances as fast as possible, even if its asymptotic behavior does not change.

Let us first use a simple example as motivation, showing that we can use BP in two ways with different performances for the same DMDGP instance. Let us consider a DMDGP instance where $n = 6$ and $F = \{(2,6)\}$. When we execute the BP as described in [7], the algorithm starts placing atoms 1, 2, and 3. Then, it branches two possible positions for atom 4, and so on, until it reaches the last atom. At this level, it must branch all eight possibilities in order to check the feasibility of each one through the information provided by $d_{2,6}$. Only at the last level the algorithm is able to reject some atomic positions.

However, we could solve the considered instance in the opposite direction along the sequence of atoms (for implementation purposes, without loss of generality, this could also be seen as executing the same BP algorithm for an "inverted" instance, where each atom label $i$ is swapped by $7 - i$). In this alternative approach, BP starts fixing atoms 6, 5, and 4, then branching two positions for atom 3. When atom 2 is reached, four positions are computed, and the known distance $d_{2,6}$ can be tested for each of them. In case a node is pruned, this node will not have any child nodes on the subsequent level. In this way, BP explores fewer nodes if compared to the classical approach. In other words, the knowledge about $d_{2,6}$ allows the second approach to restrict its search space one level before the first approach, thus making the search faster.

The presented example makes it clear that solving an instance with BP by the usual way is not always the best approach—it mainly depends on the $F$ distance set. The same analysis may be applied to the concept of interval, introduced earlier, since this can be seen as DMDGP instance as well. Each interval may be solved separately by the BP algorithm, yielding multiple partial realization trees to be combined later, forming complete realizations. Therefore, it is important to study the different ways of solving the DMDGP instances with BP by dividing instances in intervals and, then, by solving intervals in different directions.

## 9.3 Merging Two Partial Realizations

In order to solve DMDGP instances by using many intervals, we need to be able to combine solutions associated to each interval. If $R_{a,x}$ and $R_{b,y}$ ($a < b < x < y$) are two feasible realizations sharing three non-colinear atoms (the existence of these atoms implies that $x - b \geq 3$), then we can combine them in order to obtain a single realization $R_{a,y}$.

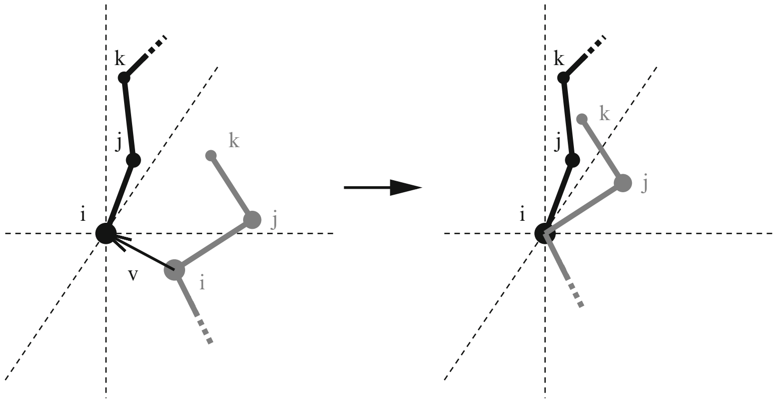


**Fig. 9.2** Translation of $R'_{b,y}$ for aligning atom $i$

Arbitrarily we choose $R_{a,x}$ as a basis for constructing $R_{a,y}$. Thus, both realizations will have the same reference system, and $R_{a,y}$ will inherit all coordinates of $R_{a,x}$, that is, $\forall k \in [a,x]$, $R_{a,y}(k) = R_{a,x}(k)$. In order to complete the coordinate sequence of $R_{a,y}$, we still need to fill the remaining interval $[x+1,y]$, which will be done by applying Euclidean transformations over the coordinates of $R_{b,y}$.

Let $i$, $j$, and $k$ be three atoms that belong to both realizations $R_{a,x}$ and $R_{b,y}$. In order to make the coordinates of interval $[x+1,y]$ satisfy all E-type distances, we must align $R_{b,y}$ to $R_{a,x}$, that is, to find $R^{'}_{b,y}$ such that

$$\begin{cases} R_{b,y}(i) = R_{a,x}(i), \\ R_{b,y}(j) = R_{a,x}(j), \\ R_{b,y}(k) = R_{a,x}(k). \end{cases} \tag{9.3}$$

Initially, we consider $R^{'}_{b,y}$ as a copy of $R_{b,y}$. Then, the first equation in Eq. (9.3) is achieved by applying a simple translation over $R^{'}_{b,y}$ (Fig. 9.2), whose translation vector $v$ is given by

$$v = R_{a,x}(i) - R_{b,y}(i).$$

In order to satisfy the second equation in Eq. (9.3), after the translation, we need to apply a rotation around the axis perpendicular to the two vectors connecting the atoms $i$ and $j$ in each realization ($R_{a,x}$ and $R^{'}_{b,y}$) (Fig. 9.3). These vectors are:

$$L_j = R_{a,x}(j) - R_{a,x}(i)$$

and

$$L^{'}_j = R^{'}_{b,y}(j) - R^{'}_{b,y}(i).$$

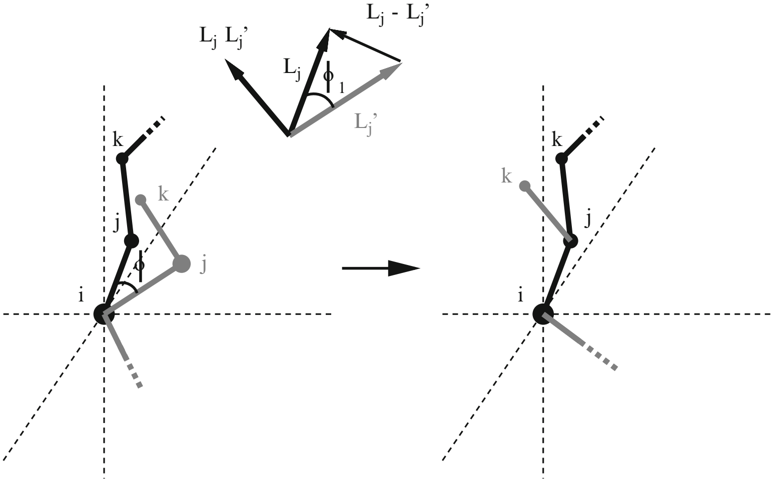


**Fig. 9.3** Rotation of $R'_{by}$ for aligning atom $j$

The rotation axis can be obtained through the cross product $L_j \times L_j^{'}$. The rotation angle is the one between the two vectors, and can be obtained by using the cosine law:

$$\phi_1 = \cos^{-1}\left(\frac{L_j^2 + L_j^{\prime 2} - \left|L_j - L_j^{'}\right|^2}{2L_jL_j^{'}}\right).$$

For aligning the atom $k$ (satisfying the last equation in Eq. (9.3)) we need another rotation. Atoms $i$ and $j$—already aligned—determine the only possible rotation axis for $R_{b,y}^{'}$ in order to continue satisfying the first two equations. The rotation angle around this axis is calculated by using the two vectors connecting the atoms $j$ and $k$ in each realization, as follows:

$$L_k = R_{a,x}(k) - R_{a,x}(j)$$

and

$$L_k^{'} = R_{b,y}^{'}(k) - R_{b,y}^{'}(j).$$

However, we are not interested anymore in the angle formed by these two vectors, as in the previous case. Now, what matters is the angle between their projections over the perpendicular plane to the rotation axis (Fig. 9.4). For calculating these projections, we use the projection matrix $M$, oriented by vector $L_j$:

$$M = L_jL_j^T.$$

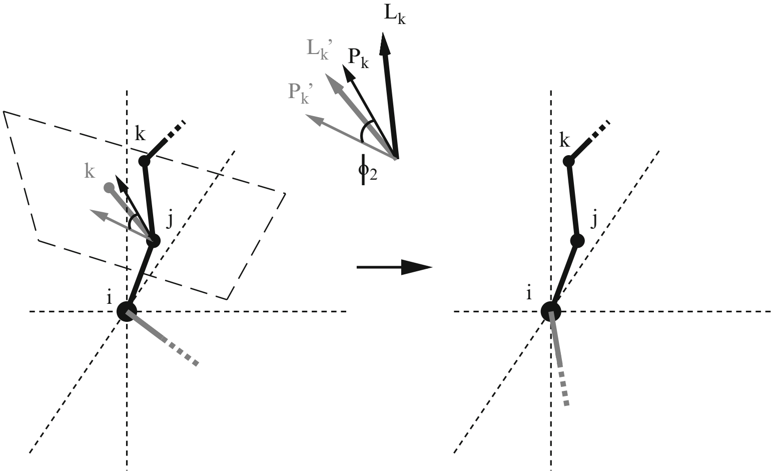


**Fig. 9.4** Rotation of $R'_{by}$ for aligning atom $k$

The projected vectors are given by:

$$P_k = ML_k$$

and

$$P_k^{'} = ML_k^{'},$$

and the angle between them may also be calculated by the cosine law:

$$\phi_2 = \cos^{-1}\left(\frac{P_k^2 + P_k^{'2} - \left|P_k - P_k^{'}\right|^2}{2P_kP_k^{'}}\right).$$

## 9.4 Merging Two Realization Trees

Once we know how to combine two feasible realizations sharing three non-colinear atoms, we can combine two realization trees sharing three atoms, which cannot be colinear by definition of DMDGP. If $T_{a,x}$ and $T_{b,y}$ are two realization trees sharing at least three atoms, the realizations generated by their combination will fill the interval $[a,y]$.

According to the growth direction of the trees, different kinds of merging may occur, described below. Algorithm 3, presented next, provides a way for merging trees $T_{a,x}$ and $T_{b,y}$, so yielding realizations for the interval $[a,y]$, and is applicable to all three kinds of merging.

**Algorithm 3** MergeTrees

```
MergeTrees($T_{a,x}$,$T_{b,y}$)
create an empty realization list $L_{a,y} = \{\}$
for (each realization $R_{a,x}$ in $T_{a,x}$) do
    for (each realization $R_{b,y}$ in $T_{b,y}$) do
        align $R_{b,y}$ to $R_{a,x}$ as described in Sect. 9.3
        fail = false
        for (each $(i,j) \in F$ such that $a \le i \le x$ and $b \le j \le y$) do
            if ($\left| \left\| R_{b,y}(j) - R_{a,x}(i) \right\| - \varepsilon \right| > d_{i,j}$) then
                fail = true
                exit loop
            end if
        end for
        if (not fail) then
            create a realization $R_{a,y}$ such that
                $R_{a,y}(i) = R_{a,x}(i)$ if $i \in [a,x]$,
                $R_{a,y}(i) = R_{b,y}(i)$ otherwise
            insert $R_{a,y}$ into $L_{a,y}$
        end if
    end for
end for
return $L_{a,y}$
```

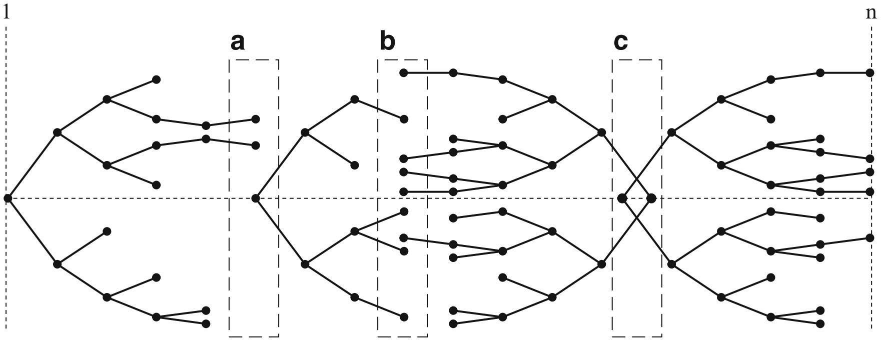


**Fig. 9.5** Realization trees merging

**Root–Root Merging.** When two trees $T^{+}_{a,x}$ and $T^{-}_{b,y}$ grow in opposite direction and overlay their roots, satisfying $x - b \geq 3$, they can be combined from their initialization (Fig. 9.5c).

**Leaf–Root Merging.** This kind of merging occurs between trees $T_1$ and $T_2$ growing in the same direction, when $T_1$ reaches the atom related to the root of $T_2$ (Fig. 9.5a). In order to be able to merge them, it is necessary to expand $T_1$ into two levels, so that we have $T_1 = T_{a,x}$ and $T_2 = T_{b,y}$, $x - b \geq 3$.

**Leaf–Leaf Merging.** This case happens when two trees grow one in direction of the other (Fig. 9.5b). Let us suppose that at some point, two trees (one negative, the other positive) reach the same atom $i$. In order to share three atoms, they need to grow at most two levels more. This can be done in three ways: the positive tree growing two levels, the negative tree growing two levels, or both growing one level. Thus, we will have $T^{-}_{a,x}$ and $T^{+}_{b,y}$, such that $x - b \geq 3$. From the performance point of view, the tree that is about to undergo more prunings should have higher growth priority. Even though two levels may seem to not be significant, it is worth to emphasize that, for big amounts of leafs, growing one level may be very expensive (in the order of the total amount of nodes in the tree until the previous level).

In Algorithm 3, we initially combine each realization in $T_{a,x}$ with each realization in $T_{b,y}$. Clearly, the total amount of combinations is the product of the amount of leaves in each tree. Considering that both trees share exactly three atoms (this condition is enough for merging them), by naming $n$ as $y - a$, the amount of leaves for $T_{a,x}$ and $T_{b,y}$ is $O(2^{x-a})$ and $O(2^{n-(x-a)+3})$, respectively. Thus, the total amount of combined realizations is $O(2^n)$. Then, each combined realization is verified according to the $F$ distance set, whose size is $O(n^2)$. Finally, the complexity of the algorithm for combining $T_{a,x}$ and $T_{b,y}$ is $O(n^2 2^n)$, which is greater than the exponential complexity of the original BP algorithm. In Sect. 9.6, we will see that, according to our computational results, this worst-case analysis does not seem to entail practical significance.

## 9.5 Growth Control

When we solve one instance by using multiple realization trees $T_1, T_2, \ldots, T_x$, each pair of subsequent trees will undergo one of the three kinds of merging described in Sect. 9.4. In case of pairs $(T_i, T_{i+1})$ that undergo root–root merging, there is only one way for performing the merging. The same happens in the case of pairs undergoing root–leaf merging, since only the tree that undergoes the merging on its leaves can grow inside the interval delimited by the roots (the tree which undergoes merging on its leaves has to grow until it reaches the root of the other tree). However, in leaf–leaf merging cases, the atom in which the merging will occur is not previously known, and it depends on the growth of both trees.

Aiming at minimizing the algorithm's execution time and the total amount of nodes (of both trees), we may consider the following heuristic: we give growth priority to the tree with fewer leaves. In other words, at each step, we verify which tree has fewer leaves, and we let it grow by one level (changing its amount of leaves for the next step). This procedure is repeated until their leaves reach the same atom. However, this is a greedy method, which does not consider the possibility of allowing the other tree to grow first. For example, it might be more convenient to let a tree grow if it is about to apply a large pruning in a few steps. Algorithm 4 summarizes this approach.

**Algorithm 4** GrowthControl

```
GrowthControl($I$)
initialize trees $T^{+}_{a,x}$ and $T^{-}_{b,y}$
while ($atom(x) \neq atom(b)$) do
    if ($T^{+}_{a,x}$ has more leaves than $T^{-}_{b,y}$) then
        make $T^{+}_{a,x}$ to grow one level
    else
        make $T^{-}_{b,y}$ to grow one level
    end if
    grow more 2 levels in $T^{+}_{a,x}$ or $T^{-}_{b,y}$ to enable merging
end while
return MergeTrees($T^{+}_{a,x}$,$T^{-}_{b,y}$)
```

## 9.6 Computational Experiments

In this section, we will consider some artificial instances, automatically generated by computer programs, and real instances, produced from protein structures obtained from the protein data bank (PDB) [1]. PDB is a public database where three-dimensional conformations of proteins and nucleic acids are stored. We have selected only structures generated by NMR. Our goal is to study in which cases the use of multiple realization trees is more efficient than the original BP algorithm. For accomplishing this, we have implemented two methods which use two realization trees in a primitive way (without any previous analysis of the F set for determining in which atom the trees start and in which directions they grow). Then, we have compared both methods to the original BP algorithm, in positive and negative directions. Our analysis did not consider the quality of the solutions which are found, since all methods fully explore the search space of instances, thus reaching the same set of solutions.

All algorithms were implemented in C++, using the standard template library (STL). Experiments were executed on an Intel Core2Duo 2.2GHz, with 2GB RAM.

We introduce a graphical representation which allows us to view how the $F$ set pairs are distributed along the molecule. We do this through the plot of $x \times P(x)$, where $x$ is an atom of the molecule and $P(x)$ is the function expressing the sum of the interval lengths related to $F$-set pairs whose last atom to be reached by BP is $x$. Considering $F^{+}_{x} = \{(i,x) \in F\}$ and $F^{-}_{x} = \{(x,j) \in F\}$, $P(x)$ is defined, for each direction, by the following formulae:

$$P^{+}(x) = \sum_{(i,x) \in F^{+}_{x}} (x - i)$$

and

$$P^{-}(x) = \sum_{(x,j) \in F^{-}_{x}} (j - x).$$

Figure 9.6 shows the $F$-set of tested artificial instances, described through an arc representation (each pair $(i,j) \in F$ is represented by an arc that connects atoms $i$ and $j$), their respective plots of $P^{+}(x)$ and $P^{-}(x)$, and the execution time for the following methods (also listed in Table 9.1):

*Method 1* One positive tree $T^{+}_{1,n}$ (original BP), implemented with breadth-first tree search;

*Method 2* One positive tree $T^{+}_{1,n}$ (original BP), implemented with depth-first tree search;

*Method 3* One negative tree $T^{-}_{1,n}$ (original BP), implemented with breadth-first tree search;

*Method 4* One negative tree $T^{-}_{1,n}$ (original BP), implemented with depth-first tree search;

*Method 5* Two trees in opposite directions, growing from their extremities towards the center, with growth control (leaf–leaf merging);

*Method 6* Two trees in opposite directions, growing from the center towards their extremities (root–root merging).

From the test results with artificial instances, we can observe some facts. The variability of instances has showed that different cases require different approaches, where the direction and the amount of trees play an important role (see Fig. 9.6). The bad performance of methods with two trees for instances (b) and (c) is not due to the growth of trees, but to the merging process, since both trees have many leaves in their merging point.

Method 5, which uses the growth control heuristic for two trees, behaves in a versatile manner, allowing the tree with fewer leaves to traverse a greater part of the molecule, and is not so sensitive to less uniform distributions of $F$-type distances as the BP algorithm is, thus obtaining good performances for instances such as (f) and (a).

The same methods have been tested in instances produced from real protein data. For this, we created a DMDGP instance from a PDB file that contains a known protein structure, by taking all atomic coordinates from the main chain (protein backbone) for determining those inter-atomic distances that are inside a cut-off radius of 5 Å. Figure 9.7 and Table 9.2 provide further details about the used instances and the execution time of our methods (for instances generated from real data, the arc representation is not clear, due to the big amount of atoms and $F$-type distances, so it was not used).

In these tests, as in tests with artificial instances, the use of two trees has been efficient in certain cases and has showed some advantages over the original BP. However, in order to justify the use of more than one tree, due to its computational cost, it is necessary that the trees are placed in strategic positions along the molecule (as it happened for method 5 for instance 1SFV), so that $F$-type distances can be used as soon as possible, causing prunings and having yielding few leaves in the moment of merging. As it has been showed for artificial instances, the cases where molecules have some interval with low amount of $F$-type distances to be

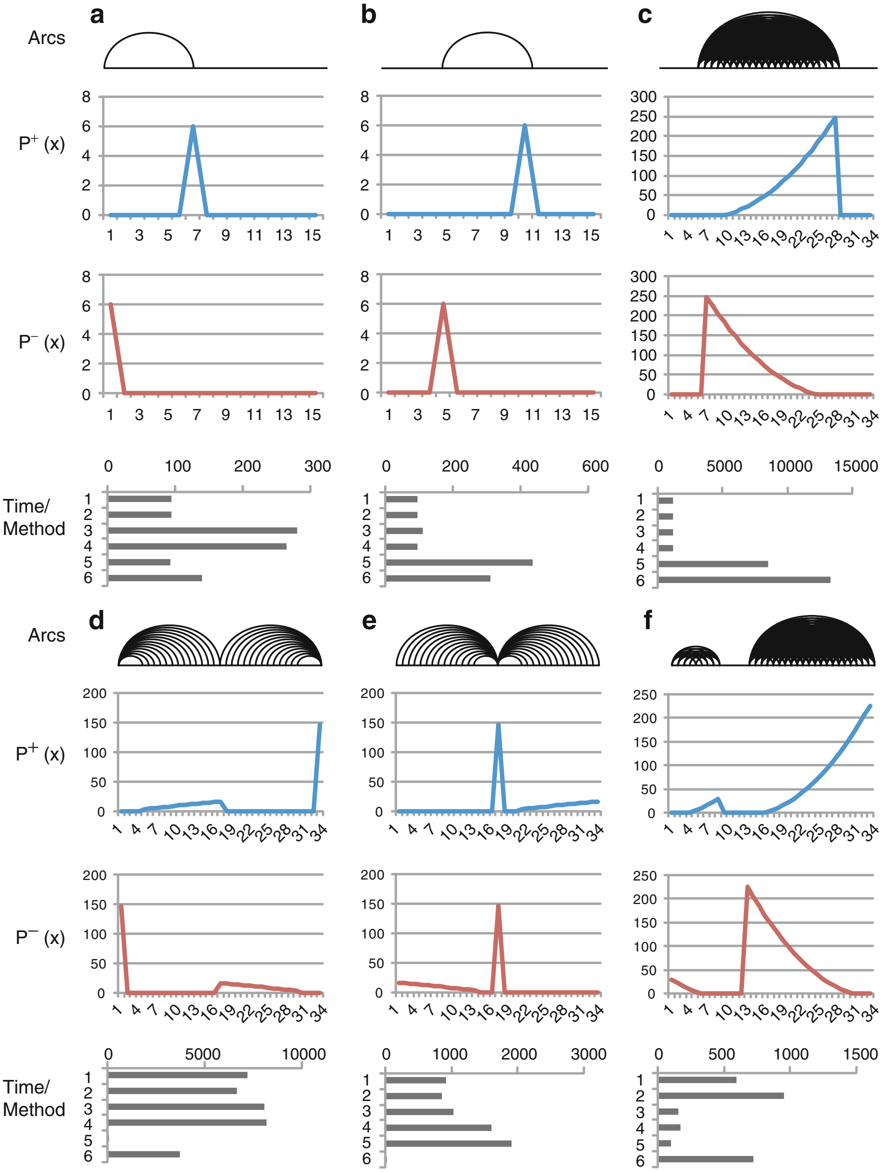


**Fig. 9.6** Tests with artificial instances

**Table 9.1** Tests with artificial instances

| | | Execution time of each method (ms) | | | | | |
|---|---|---|---|---|---|---|---|
| Instance | n | 1 | 2 | 3 | 4 | 5 | 6 |
| a | 16 | 94 | 94 | 281 | 266 | 93 | 140 |
| b | 16 | 94 | 94 | 109 | 94 | 438 | 312 |
| c | 35 | 1,141 | 1,171 | 1,125 | 1,172 | 8,485 | 13,391 |
| d | 35 | 7,203 | 6,672 | 8,079 | 8,203 | 16 | 3,719 |
| e | 35 | 922 | 844 | 1,031 | 1,609 | 1,906 | 15 |
| f | 35 | 594 | 953 | 156 | 172 | 93 | 718 |

traversed (as instances (1DFN-a) and (1DFN-b)) give greater growth priority for one of the trees, what cannot be foreseen by the original BP. Method 5, whose trees have controlled growth, has showed that it can deal better with this kind of $F$-set topology.

## 9.7 Conclusions

We have presented some initial studies that enable solving the DMDGP with multiple realization trees. Through Euclidean transformations as translations and rotations, we discussed about possible ways for combining realizations of distinct intervals that share at least three atoms. We studied the three possible cases for merging realization trees, and, by using this technique, we presented an algorithm that deals with these three cases. This algorithm is what actually allows the use of multiple trees for solving the DMDGP. Moreover, we presented a heuristic for the multiple tree strategy, consisting of regulating the growth of trees that will undergo leaf–leaf merging.

We have made tests with artificial instances and instances generated from real protein data, by comparing the BP algorithm, which produces one realization tree, with two primitive methods using two realizations trees. For both artificial instances and instances generated from proteins, the use of two trees (despite the simplicity of the implemented methods) has showed good performance in comparison to the original BP algorithm. For each instance, depending on the topology of the $F$ set, the methods that use only one tree had a high performance variability according to the direction of growth along the molecule. However, the heuristic method that consider two trees was not so sensitive to the $F$-set topology, having, most of the time, a performance which is similar to the method of one tree in its most efficient direction (with no need to detect which is the direction).

The results presented here reinforce the interest about studying alternative solving approaches for the DMDGP. Our intention here was twofold: (1) to show that the BP algorithm itself does not assure the best performance, depending on

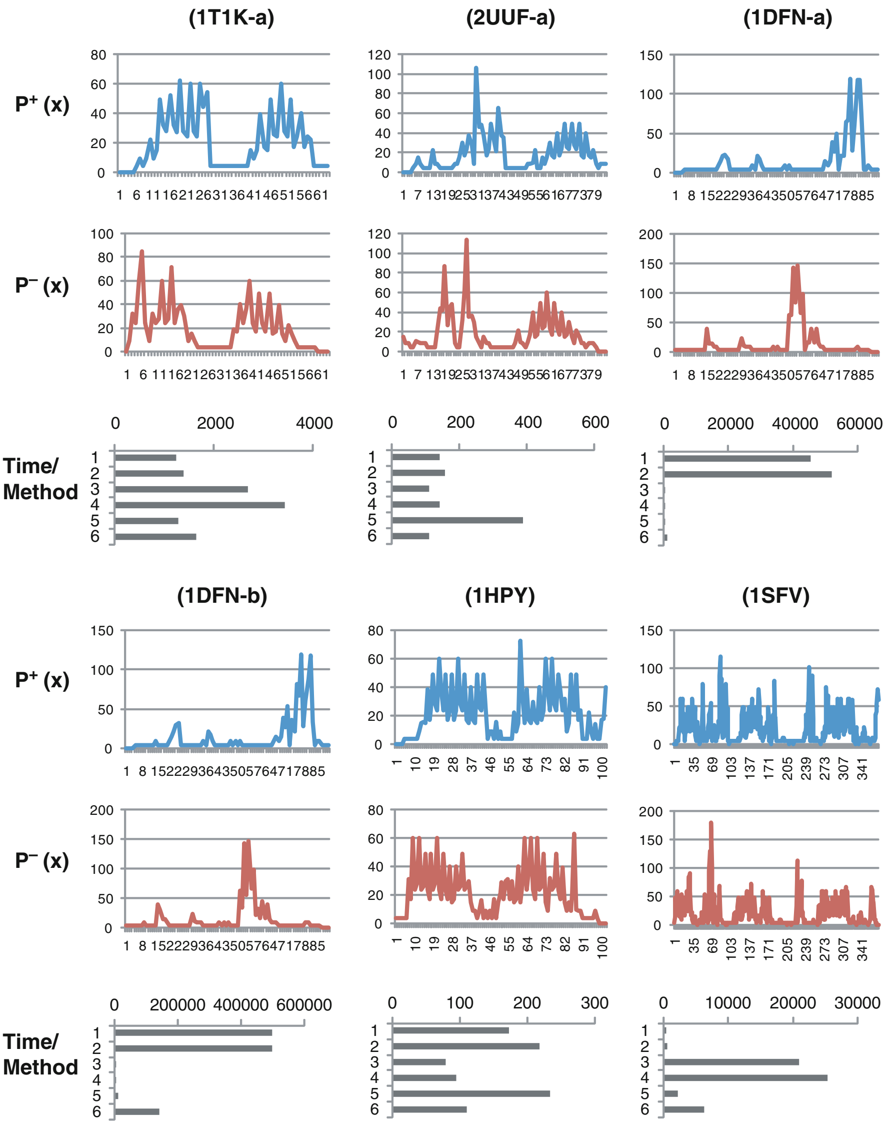


**Fig. 9.7** Tests with instances generated from real protein data

the direction of its growth along the molecule and (2) it may be used by more complex strategies, as solving the molecule with multiple realization trees, which provide advantages over the traditional approach. The multiple trees strategy lets us think about performance (not only completeness and correctness) when solving the DMDGP, stimulating investigation of heuristics for the DMDGP.

**Table 9.2** Tests with instances generated from real protein data

| | | Execution time for each method (ms) | | | | | |
|---|---|---|---|---|---|---|---|
| Instance | n | 1 | 2 | 3 | 4 | 5 | 6 |
| 1T1K-a (Insulin) | 63 | 1,250 | 1,391 | 2,688 | 3,437 | 1,281 | 1,641 |
| 2UUF-a (Thrombin) | 84 | 141 | 156 | 109 | 141 | 391 | 109 |
| 1DFN-a (Human defensin) | 90 | 45,500 | 52,031 | 78 | 94 | 328 | 922 |
| 1DFN-b (Human defensin) | 90 | * | * | 969 | 1,172 | 8,453 | 141,672 |
| 1HPY (PTH hormone ) | 102 | 172 | 219 | 78 | 94 | 234 | 110 |
| 1SFV (Phospholipase A2) | 372 | 343 | 453 | 20,985 | 25,328 | 2,109 | 6,234 |

* not concluded, due to memory allocation demands

**Acknowledgments** The authors would like to thank the Brazilian research agencies FAPESP, FAPERJ, and CNPq, for the financial support.

## References

1. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Res. **28**, 235–242 (2000)
2. Crippen, G., Havel, T.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
3. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact interatomic distances. J. Global Optim. **22**, 365–375 (2002)
4. Lavor, C., Liberti, L., Maculan, N.: Molecular distance geometry problem. Encyclopedia of Optimization, pp. 2305–2311, 2nd edn. Springer, New York (2009)
5. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: The discretizable molecular distance geometry problem. Comput. Optim. Appl. **52**, 115–146 (2012)
6. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
7. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
8. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
9. Mucherino, A., Lavor, C., Liberti, L., Talbi, E.-G.: A parallel version of the branch and prune algorithm for the molecular distance geometry problem. In: IEEE Conference Proceedings, ACS/IEEE International Conference on Computer Systems and Applications (AICCSA10), pp. 1–6. Hammamet, Tunisia (2010)
10. Nucci, P.: Heuristicas para o problema molecular de geometria de distancias aplicado a proteinas. Undergraduate Dissertation, Fluminense Federal University
11. Saxe, J.B.: Embeddability of weighted graphs in *k*-space is strongly NP-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489. Monticello, IL (1979)

# Chapter 10
# ASAP: An Eigenvector Synchronization Algorithm for the Graph Realization Problem

**Mihai Cucuringu**

**Abstract** We review a recent algorithm for localization of points in Euclidean space from a sparse and noisy subset of their pairwise distances. Our approach starts by extracting and embedding uniquely realizable subsets of neighboring sensors called patches. In the noise-free case, each patch agrees with its global positioning up to an unknown rigid motion of translation, rotation, and possibly reflection. The reflections and rotations are estimated using the recently developed eigenvector synchronization algorithm, while the translations are estimated by solving an overdetermined linear system. In other words, to every patch, there corresponds an element of the Euclidean group Euc(3) of rigid transformations in $\mathbb{R}^3$, and the goal is to estimate the group elements that will properly align all the patches in a globally consistent way. The algorithm is scalable as the number of nodes increases, and can be implemented in a distributed fashion. Extensive numerical experiments show that it compares favorably to other existing algorithms in terms of robustness to noise, sparse connectivity and running time.

**Keywords** Graph realization problem • Sensor networks • Molecule problem • Distance geometry • Eigenvectors • Synchronization • Rigidity theory • Spectral graph theory

## 10.1 Introduction

The graph realization problem has attracted significant attention in recent years, especially in the context of localization of sensor networks and three-dimensional structuring of molecules [30, 31]. The problem falls naturally under the large

M. Cucuringu (✉)
PACM, Princeton University, Fine Hall, Washington Road, Princeton,
NJ 08544-1000, USA
e-mail: mcucurin@math.princeton.edu

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_10,

umbrella of distance geometry problems and has received growing interest from researchers across a variety of fields, including computer science, engineering, and mathematics. In this chapter we review a recently proposed two dimensional sensor network localization (SNL) algorithm introduced in [16]. Very recently, we have extended our approach to three dimensions and have added several improvements to the algorithm specific to the molecule problem from structural biology [17].

Given a set of $|V| = n$ nodes and $|E| = m$ edges, defining the graph $G = (V,E)$, together with a distance measurement associated with each edge, the graph realization problem is to assign to each vertex coordinates in $\mathbb{R}^d$ such that the Euclidean distance between any two adjacent nodes matches the prescribed distance associated to that edge. In other words, for any edge $(i, j) \in E$ of the measurement graph $G$, one is given the distance $d_{ij} = d_{ji}$, with the goal of finding a $d$-dimensional embedding $p_1, p_2, \ldots, p_n \in \mathbb{R}^d$ such that $\|p_i - p_j\| = d_{ij}$, for all $(i, j) \in E$. In this chapter, we focus on the two-dimensional case, although the approach is applicable to higher dimensions $d > 2$ as well. The graph realization problem comes up naturally in a variety of settings such as wireless sensor networks [9, 45], structural biology [24], environmental monitoring [1], and multidimensional scaling (MDS) [15]. In such real-world applications, it is typically the case that the available distances $d_{ij}$ between the nodes are very noisy, with $d_{ij} = \|p_i - p_j\| + \varepsilon_{ij}$ where $\varepsilon_{ij}$ represents the added noise, and the goal is to find an embedding that matches all available distances $d_{ij}$ as best as possible. Classical multidimensional scaling successfully solves the localization problem as long as all $n(n-1)/2$ pairwise distances are available, which unfortunately is rarely the case in practical applications.

We assume that the graph realization problem has a unique solution in other words, the underlying graph is globally rigid, and note that applying a rigid transformation (composition of rotation, translation, and possibly reflection) to a graph realization results in another graph realization, as rigid transformations preserve distances. Whenever an embedding is possible, it is unique (up to rigid transformations) only if there are enough distance constraints, in which case the graph is said to be globally rigid (see, e.g., [23]). From a computational perspective, the graph realization problem has been shown to be very difficult. Saxe has shown it is strongly NP-complete in one dimension and strongly NP-hard for higher dimensions [35, 47]. A popular model for the SNL problem is that of a disc graph model, where two sensors communicate with each other if and only if they are within sensing radius $\rho$ of each other, i.e., $(i, j) \in E \iff d_{ij} \leq \rho$. The SNL problem is NP-hard also under the disc graph model [4]. Despite its difficulty, the problem has received a great deal of attention in the networking and distributed computation communities, and numerous heuristic algorithms exist that approximate its solution. In the context of sensor networks [2, 4, 5, 27], there are many algorithms that solve the graph realization problem, and they include methods such as global optimization [12], semidefinite programming (SDP) [9–11, 42, 43, 49], and local to global approaches [29, 32, 36, 37, 48], some of which we briefly review in Sect. 10.2.

The algorithm we review in this chapter follows a local to global divide-and-conquer approach, integrating local distance information into a global structure determination. Locally, we identify for every sensor, the globally rigid subgraphs of its 1-hop neighborhood, which we call patches. Once the 1-hop neighborhood has been decomposed into patches, we separately localize each such patch in a coordinate system of its own using either the stress minimization method of [21] or SDP. When all available distances are noiseless, the computed coordinates of the sensors in each patch will agree with the ground truth solution up to some unknown rigid motion, i.e., a combination of a translation, rotation, and possibly reflection, which is precisely what out proposed algorithm is estimating. In other words, to every existing patch, there corresponds an element of the Euclidean group Euc(2) of rigid transformations in the plane, and the goal is to estimate these unknown group elements that will properly align all the patches in a globally consistent framework. In this process, the only available information we make use of is the set of pairwise alignments between any two patches that overlap in sufficiently many nodes. In other words, by finding the optimal alignment between pairs of patches whose intersection is large enough, we obtain measurements for the ratios of the unknown corresponding group elements. Finding group elements from noisy measurements of their ratios is also known as the synchronization problem [20, 28], which we discuss in Sect. 10.6. Intuitively, we use the eigenvector method for the compact part of the group, when we synchronize over the groups $\mathbb{Z}_2$ and SO(2) to recover the reflections and rotations, and solve by least squares an overdetermined linear system in $\mathbb{R}^2$ for the translations. We consider the performance of our algorithm with respect to several criteria, including robustness to noise and sparsity of the distance measurements, and scalability to large networks with tens or hundreds of thousands of nodes.

This chapter is organized as follows: Section 10.2 is a survey of existing methods for solving the two-dimensional graph realization problem, with a focus on localization of planar sensor networks. Section 10.3 gives an overview of the 2D-as-synchronized-as-possible (ASAP) algorithm described in this chapter. In Sect. 10.4, we motivate our divide-and-conquer approach and explain how to break up the initial measurement graph and how to localize the resulting patches. Section 10.5 explains the synchronization algorithm that aligns all patches in a globally consistent structure. In Sect. 10.6 we give a brief self-contained introduction to the group synchronization problem and the references therein. In Sect. 10.7, we report on numerical simulations where we tested the performance of 2D-ASAP in comparison to existing state-of-the-art algorithms. Finally, Sect. 10.8 is a summary and discussion of the extension of our algorithm to the molecule problem in structural biology.

## 10.2 Related Work

Numerous algorithms across different communities have been proposed for the SNL problem, with the goal of finding an approximate embedding $p_1, \dots, p_n \in \mathbb{R}^2$ that preserves the measured (noisy) distances $d_{ij}, (i,j) \in E$ as best as possible.

Approaches coming from the SDP community [8–11, 49] propose minimizing a variety of error functions, such as

$$f(p_1,\ldots,p_n) = \sum_{(i,j)\in E} \left(\|p_i - p_j\|^2 - d_{ij}^2\right)^2 \tag{10.1}$$

$$g(p_1,\ldots,p_n) = \sum_{(i,j)\in E} \left|\| p_i - p_j \|^2 - d_{ij}^2\right| \tag{10.2}$$

$$Stress(p_1,\ldots p_n) = \sum_{(i,j)\in E} (\| p_i - p_j \| - d_{ij})^2 . \tag{10.3}$$

Unfortunately, all the above functions are not convex over the constraint set, and the search for the global minimum is prone to getting stuck at a local minima. Their relaxations to an SDP [9] are computationally expensive and not very robust to noise, as the solutions belong to a higher dimensional Euclidean space, and the projection to the plane often results in large errors for the estimation of the coordinates. The stress majorization algorithm (also known as SMACOF [12]), was originally introduced by Leeuw [18] as a variant of the gradient descent approach for minimizing the stress function in Eq. (10.3).

Maximum variance unfolding (MVU) is a non-linear dimensionality reduction algorithm proposed by Weinberger et al.[46], which became very popular within the machine-learning community. The algorithm produces a low-dimensional representation of the data by maximizing the variance of its embedding while preserving the original local distance constraints. MVU builds on the SDP approach and addresses the issue of the possibly high-dimensional solution to the SDP problem by maximizing the variance of the embedding (also known as the maximum trace heuristic). The main contribution of the FAST-MVU algorithm in [46] is the approximation of the $x$ and $y$ coordinate vectors of the sensors by just the first few (e.g., 10) low-oscillatory eigenvectors of the graph Laplacian. This allows one to replace the original and possibly large-scale SDP by a much smaller SDP, which leads to a significant reduction in running time. The locally rigid embedding (LRE) algorithm [37] is reminiscent of the locally linear embedding (LLE) [34] technique used in machine learning for dimensionality reduction. LRE tries to preserve, in a global coordinate system, the local affine relationships present within patches. Each sensor contributes with a linear equation relating its location to those of its neighboring nodes, thus altogether setting up a global linear system. LRE builds up a specially designed sparse matrix whose eigenvectors give an embedding of all sensors, from which a global affine transformation must be removed.

The recent as-rigid-as-possible (ARAP) algorithm in [48] is along the lines of PATCHWORK [29] and LRE, and starts off by localizing small patches in a similar manner, but instead of finding a global embedding via affine mappings, they use rigid mappings. Again, the patch overlaps impose constraints on the mappings however, the usage of rigid mappings has the advantage of better preserving the local relationships between patches. This comes at the price of resulting in a non linear optimization problem, which is solved efficiently using a two-phase alternating least-squares method. The initial guess required by the nonlinear optimization is obtained by as-affine-as-possible (AAAP), an improved version of the LRE and PATCHWORK algorithms.

Very recently, Javanmard and Montanari [26] proposed a localization algorithm based on SDP, for which they provide a theoretical analysis in terms of robustness to noise for the random geometric graph model and uniformly bounded adversarial noise. For noiseless data, they provide a lower bound for the radius beyond which the algorithm is guaranteed to recover the original solution, up to a rigid transformation. On a related note, we also report on recent and ongoing work of Ozyesil and Singer on SyncContract [33], an optimization algorithm able to synchronize over the non-compact special Euclidean group $\mathrm{SE}(k)$ by solving an analogous problem on a compact group, of which $\mathrm{SE}(k)$ is a Lie group contraction. They provide experimental results for synthetic data and for the SNL problem and a robustness analysis of their algorithm for the complete and the sparse Erdös-Rényi graph models.

## 10.3 Overview of the 2D-ASAP Algorithm

This section provides an overview of the 2D-ASAP algorithm reviewed in this chapter. We follow a divide-and-conquer approach that breaks up the large graph into many smaller overlapping subgraphs, that we call patches, and "stitches" them together concurrently and consistently in a global coordinate system with the purpose of localizing the entire initial measurement graph. To avoid foldovers in the final solution, each such patch needs to be globally rigid and the entire measurement graphs needs to be globally rigid as well.

We break up the initial graph into patches in the following way. For every node $i$ we let $V(i) = \{j : (i,j) \in E\} \cup \{i\}$ the set of its 1-hop neighbors together with the node itself, and $E(i) = \{(i,j) \in E | \{i,j\} \in V(i)\}$, and denote by $G(i) = (V(i), E(i))$ its subgraph of 1-hop neighbors. If $G(i)$ is a globally rigid graph, we embed it in $\mathbb{R}^2$, otherwise we break it into maximally globally rigid subgraphs that we call patches, and embed each patch in $\mathbb{R}^2$. The embedding of every patch in $\mathbb{R}^2$ is given in its own local frame. We defer to Sect. 10.4 the specific details of breaking up the measured graph into smaller maximally globally rigid subgraphs. Let $N$ denote the number of patches obtained in the above decomposition of the measurement graph, and note that it may be different from $n$, the number of nodes in $G$, since the neighborhood graph of a node may contribute several patches or none.

For the embedding of local patches we usually use the stress majorization algorithm as described in [21]. Once each patch is embedded in its own coordinate system, one must find the reflections, rotations, and translations that will stitch all patches together in a consistent manner, a process to which we refer as *synchronization*. To every embedded patch $P_i$ there corresponds an element $e_i \in$ Euc(2), where Euc(2) is the Euclidean group of rigid motions in the plane, i.e., reflections, rotations, and translations. The rigid motion $e_i$ moves patch $P_i$ to its correct position with respect to the global coordinate system. Our goal is to estimate simultaneously the rigid motions $e_1,\ldots,e_N$ (up to a global rigid motion) that will properly align all the patches in a globally consistent way. To achieve this goal, we first estimate the alignment between any pair of patches $P_i$ and $P_j$ that have enough nodes in common. We describe one such alignment method in Sect. 10.4 and refer the reader to Sect. 6 of [16] for additional robust alignment methods. The alignment of patches $P_i$ and $P_j$ provides a (usually noisy) measurement for the ratio $e_i e_j^{-1}$ in Euc(2). We solve the resulting synchronization problem in a globally consistent manner, such that information from local alignments propagates to pairs of nonoverlapping patches. Ideally, we would like to be able to solve the synchronization problem over Euc(2); however, the non-compactness of the group makes the problem significantly harder. Only very recently, the authors of [33] introduced several optimization-based algorithms that are able to synchronize over the non-compact special Euclidean group SE$(k)$ by solving an analogous problem on a compact group.

As an alternative, we replace the synchronization problem over Euc(2) by three different consecutive synchronization problems. In the first one, we find the reflections of all the patches using the eigenvector synchronization algorithm over the group $\mathbb{Z}_2$. After we have estimated the reflections, we use the same eigenvector synchronization method, but this time over the group SO(2) to estimate the rotations of all patches. Once both reflections and rotations have been computed, we estimate the translations by solving an overdetermined linear system. To summarize, we simultaneously integrate all the available local information into a global coordinate system over three steps, using the eigenvector synchronization algorithm and the least-squares method over the isometries of the Euclidean plane. As we shall see, the main advantage of the eigenvector method is that it can recover the reflections and rotations even when many of the pairwise alignments are incorrect. A complete summary of the 2D-ASAP algorithm is given in Table 10.1.

## 10.4 Finding and Localizing Globally Rigid Patches

Next we describe how to identify and localize patches, a crucial step of our divide-and-conquer algorithm. Unlike most other localization algorithms, we choose to build patches that are globally rigid, and provide an efficient and theoretically motivated method for doing so. Previous localization algorithms that use a local to global approach, such as PATCHWORK, LRE, and ARAP, simply define patches by associating with every node $i$ its entire 1-hop neighborhood $G(i)$, which usually leads to patches which are not globally rigid and have more than one possible realization in the plane. Therefore, whenever $G(i)$ is not globally rigid, we find its maximally globally rigid components, which we call patches. Note that the number of resulting patches can be 0, 1, or greater than 1. The novelty of our approach is that breaking up the 1-hop neighborhood subgraph $G(i)$ is much easier than breaking up a general graph, by utilizing recent results of [14] about the global rigidity property of cone graphs.

**Table 10.1** Overview of the 2D-ASAP algorithm

| Input | $G=(V,E)$, $\lvert V\rvert=n$, $\lvert E\rvert=m$, $d_{ij}$ for $(i,j)\in E$ |
|---|---|
| Pre-processing<br>step | 1. Break the measurement graph $G$ into $N$ globally rigid patches $P_1,\ldots,P_N$<br>2. Embed each patch $P_i$ separately using the embedding method of choice (e.g., stress majorization or SDP) |
| Step 1<br>Estimating<br>reflections | 1. Align all pairs of patches $(P_i,P_j)$ that have enough nodes in common<br>2. Estimate their relative reflection $z_{ij}\in\{-1,+1\}$<br>3. Build a sparse $N\times N$ symmetric matrix $Z=(z_{ij})$ as defined in Eq. (10.4)<br>4. Define $\mathscr{Z}=D^{-1}Z$, where $D$ is a diagonal matrix with $D_{ii}=deg(i)$<br>5. Compute the top eigenvector $v_1^{\mathscr{Z}}$ of $\mathscr{Z}$ which satisfies $\mathscr{Z}v_1^{\mathscr{Z}}=\lambda_1^{\mathscr{Z}}v_1^{\mathscr{Z}}$<br>6. Estimate the global reflection of patch $P_i$ by $\hat{z}_i=\text{sign}(v_1^{\mathscr{Z}}(i))=\frac{v_1^{\mathscr{Z}}(i)}{\lvert v_1^{\mathscr{Z}}(i)\rvert}$<br>7. Replace the embedding patch $P_i$ with its mirrored image whenever $\hat{z}_i=-1$ |
| Step 2<br>Estimating<br>rotations | 1. Align all pairs of patches $(P_i,P_j)$ that have enough nodes in common<br>2. Estimate their relative rotation angle $\theta_{ij}\in[0,2\pi)$ and set $r_{ij}=e^{\iota\theta_{ij}}$<br>3. Build a sparse $N\times N$ Hermitian matrix $R=(r_{ij})$ as defined in Eq. (10.5)<br>4. Define $\mathscr{R}=D^{-1}R$<br>5. Compute the top eigenvector $v_1^{\mathscr{R}}$ of $\mathscr{R}$ corresponding to $\mathscr{R}v_1^{\mathscr{R}}=\lambda_1^{\mathscr{R}}v_1^{\mathscr{R}}$<br>6. Estimate the global rotation angle $\hat{\theta}_i$ of patch $P_i$ using $e^{\iota\hat{\theta}_i}=\frac{v_1^{\mathscr{R}}(i)}{\lvert v_1^{\mathscr{R}}(i)\rvert}$<br>7. Rotate the embedding of patch $P_i$ by the angle $\theta_i$ |
| Step 3<br>Estimating<br>translations | 1. Build an $m\times n$ overdetermined system of linear equations<br>2. Include the anchors information (if available) into the linear system<br>3. Compute the least squares solution for the $x$-axis and $y$-axis coordinates |
| Output | Estimated coordinates $\hat{p}_1,\ldots,\hat{p}_n$ |

We call a *star graph* a graph which contains at least one vertex that is connected to all remaining nodes. Note that for each node $i$, the local graph $G(i)$ composed of the central node $i$ and all its neighbors takes the form of a star graph. We make use of this structural property of the graph, and propose a simple efficient algorithm that break up non-globally rigid star graph into smaller globally rigid star subgraphs, each of which is of maximal size.

**Proposition 10.1.** *A star graph is generically globally rigid in $\mathbb{R}^2$ iff it is three-vertex-connected.*

In light of Proposition 10.1 [16], we propose the following simple algorithm for breaking up a star graph into maximally globally rigid components. We start by removing all vertices of degree one, since no globally rigid subgraph can contain such a vertex. Note that a vertex of degree two can be only be contained in a triangle, provided its two neighbors are connected. Next, we search for the (maximal) three-connected components in the graph, taking advantage of its structure as a star graph.

Once we have divided the original graph into many overlapping globally rigid patches, the next step is to find a two-dimensional embedding of each one of them. Localizing a small globally rigid subgraph is significantly easier in terms of speed and accuracy than localizing the whole measurement graph. First, the size of a patch is significantly smaller than the size of the whole network. Also, another advantage of embedding locally is that we are no longer constrained to a distributed computation that can impose additional challenges due to intersensor communication. Since each node in the patch is connected to a central node, all the information can be passed on to this node which will perform the computation in a centralized manner. Finally, under the assumptions of the disc graph model, it is likely that 1-hop neighbors of the central node will also be interconnected, rendering a relatively high density of edges for the patches.

After extensively experimenting with different localization algorithms, our method of choice for embedding the patches was the three-stage procedure described in [21], due to its relatively low running time and its robustness to noise for small patches. When used for small patches (e.g., of size 20–30) rather than the entire network, the stress minimization is more reliable and less sensitive to local minima. Compared to an anchor-free SDP localization algorithm like SNL-SDP,[1] it produces similar results in terms of the localization error, but with lower running times. To the best of our knowledge, the SDP-based approaches (in particular those of [9–11,42,43,49]) have not been analyzed in the context of the disc graph model, and the SDP localization theory is built only on the known distances, without any additional lower and upper bounds that can be inferred from the disc graph assumption. Note that we restrict the size of the patches to some maximal prescribed size to avoid inaccurate patch embeddings.

A crucial step in the synchronization algorithm is the accurate alignment of pairs of patches for building the matrices $Z$ and $R$ of pairwise reflections and rotations. For any two patches $P_i$ and $P_j$ embedded in their own coordinate system, we are interested in estimating their relative reflection rotation. Clearly, any two patches that are far apart and have no common nodes cannot be aligned thus, there must be enough overlapping nodes to make the alignment possible. The problem of aligning two labeled sets of nodes is known as the registration problem, for which a closed form solution for any dimension was proposed by [25], where the best rigid transformation between two sets of points is obtained by various matrix manipulations and eigenvalue/eigenvector decomposition. We refer the reader to Sect. 6 of [16] for a thorough discussion of several methods for aligning patches.

## 10.5 Synchronization over $\mathbb{Z}_2$, SO(2), and $\mathbb{R}^2$

This section details Steps 1, 2, and 3 of the 2D-ASAP algorithm. We use the eigenvector method for the group synchronization problems over the groups $\mathbb{Z}_2$ and SO(2), to estimate the reflections and rotations of the $N$ patches. In the last step, we synchronize over $\mathbb{R}^2$ by solving an overdetermined system of linear equations to recover the translations of the patches and provide a final estimate for the sensor coordinates.

[1] We used the SNL-SDP code of [44].

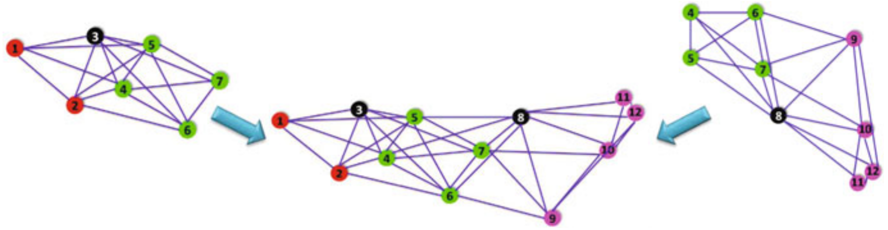


**Fig. 10.1** Optimal alignment of two patches that overlap in four nodes. The alignment provides a measurement for the ratio of the two group elements in Euc(2). In this example we see that a reflection was required to properly align the patches

### *10.5.1 Step 1: Synchronization over $\mathbb{Z}_2$ to Estimate Reflections*

In the first step of the algorithm, we identify which patches need to be reflected with respect to the global coordinate system. We denote the reflection of patch $P_i$ by $z_i \in \{-1,1\}$, a $-1$, indicating that the patch requires a reflection, and $+1$ otherwise. Note that all reflections are defined up to a global reflection (global sign). The alignment of every pair of patches $P_i$ and $P_j$ whose intersection is sufficiently large, provides a measurement $z_{ij}$ for the ratio $z_i z_j^{-1}$ (in the case of $\mathbb{Z}_2$ this is simply the product $z_i z_j$, since an element is its own inverse). When the initial distance measurements are noisy (and hence the patch embeddings) many ratio measurements can be corrupted, i.e., have their sign flipped. We denote by $G^P = (V^P, E^P)$ the patch graph whose vertices $V^P$ are the patches $P_1, \ldots, P_N$, and two patches $P_i$ and $P_j$ are adjacent, $(P_i, P_j) \in E^P$, iff they have enough[2] vertices in common to be aligned such that the ratio $z_i z_j^{-1}$ can be estimated (Fig. 10.1).

We solve this synchronization problem over $\mathbb{Z}_2$ using the eigenvector method, which starts off by building the following $N \times N$ sparse symmetric matrix $Z = (z_{ij})$:

$$z_{ij} = \begin{cases} 1 & \text{aligning } P_i \text{ with } P_j \text{ did not require reflection} \\ -1 & \text{aligning } P_i \text{ with } P_j \text{ required reflection of one of them} \\ 0 & (i,j) \notin E^P \ (P_i \text{ and } P_j \text{ cannot be aligned}) \end{cases} \tag{10.4}$$

Prior to computing the top eigenvector of the matrix $Z$, as done in [38], we choose the following normalization that increases the robustness to noise and numerical stability. Let $D$ be an $N \times N$ diagonal matrix,[3] whose entries are given by $D_{ii} = \sum_{j=1}^{N} |z_{ij}|$. In other words, $D_{ii} = deg(i)$, where $deg(i)$ is the node degree of patch $P_i$ in $G^P$, i.e., the number of other patches that can be aligned with it. We define the

[2] For example three common vertices, although the precise definition of "enough" will be given later.

[3] The diagonal matrix $D$ should not be confused with the partial distance matrix.

matrix $\mathscr{Z}$ as $\mathscr{Z} = D^{-1}Z$, and note that, although not necessarily symmetric, it is similar to the symmetric matrix $D^{-1/2}ZD^{-1/2}$ through

$$\mathscr{Z} = D^{-1/2}(D^{-1/2}ZD^{-1/2})D^{1/2}.$$

Therefore, the matrix $\mathscr{Z}$ has $N$ real eigenvalues $\lambda_1^{\mathscr{Z}} > \lambda_2^{\mathscr{Z}} \geq \cdots \geq \lambda_N^{\mathscr{Z}}$ and $N$ orthonormal eigenvectors $v_1^{\mathscr{Z}}, \ldots, v_N^{\mathscr{Z}}$, satisfying $\mathscr{Z} v_i^{\mathscr{Z}} = \lambda_i^{\mathscr{Z}} v_i^{\mathscr{Z}}$. In the eigenvector method, we compute the top eigenvector $v_1^{\mathscr{Z}} \in \mathbb{R}^N$ of $\mathscr{Z}$ and use it to obtain estimators $\hat{z}_1, \ldots, \hat{z}_N$ for the reflections of the patches, in the following way: $\hat{z}_i = \text{sign}(v_1^{\mathscr{Z}}(i)) = \frac{v_1^{\mathscr{Z}}(i)}{|v_1^{\mathscr{Z}}(i)|}, \quad i = 1, 2, \ldots, N$. After estimating the reflection of all patches (up to a global sign), we replace the embedding of patch $P_i$ by its mirrored image whenever $\hat{z}_i = -1$.

### 10.5.2 Step 2: Synchronization over SO(2) to Estimate Rotations

After having estimated the appropriate reflections, next we estimate the rotations of all patches. To each patch we associate an element $r_i \in \text{SO}(2)$, $i = 1, \ldots, N$ that we represent as a point on the unit circle in the complex plane $r_i = e^{\imath\theta_i} = \cos\theta_i + \imath \sin\theta_i$. We repeat the alignment process from Step 1 to estimate the angle $\theta_{ij}$ between two overlapping patches, i.e., the angle by which one needs to rotate patch $P_i$ to align it with patch $P_j$. When the aligned patches contain corrupted distance measurements, $\theta_{ij}$ is a noisy measurement of their offset $\theta_i - \theta_j \mod 2\pi$. Following a similar approach to Step 1, we build the $N \times N$ sparse symmetric matrix $R = (r_{ij})$ whose elements are either 0 or points on the unit circle in the complex plane:

$$r_{ij} = \begin{cases} e^{\imath\theta_{ij}} & \text{if } (i,j) \in E^P \\ 0 & \text{if } (i,j) \notin E^P \end{cases}. \tag{10.5}$$

Since $\theta_{ij} = -\theta_{ji} \mod 2\pi$, it follows that $R$ is a Hermitian matrix, i.e., $R_{ij} = \bar{R}_{ji}$, where for any complex number $w = a + \imath b$ we denote by $\bar{w} = a - \imath b$ its complex conjugate. As in Step 1, we choose to normalize $R$ using the diagonal matrix $D$, whose diagonal elements are also given by $D_{ii} = \sum_{j=1}^N |r_{ij}|$. Next, we define the matrix $\mathscr{R} = D^{-1}R$, which is similar to the Hermitian matrix $D^{-1/2}RD^{-1/2}$ through

$$\mathscr{R} = D^{-1/2}(D^{-1/2}RD^{-1/2})D^{1/2}.$$

We define the estimated rotation angles (up to an additive phase) $\hat{\theta}_1, \ldots, \hat{\theta}_N$ and their corresponding elements in SO(2), $\hat{r}_1, \ldots, \hat{r}_N$ using the top eigenvector $v_1^{\mathscr{R}}$ as

$$\hat{r}_i = e^{\imath\hat{\theta}_i} = \frac{v_1^R(i)}{|v_1^R(i)|}, \quad i = 1, 2, \ldots, N. \tag{10.6}$$

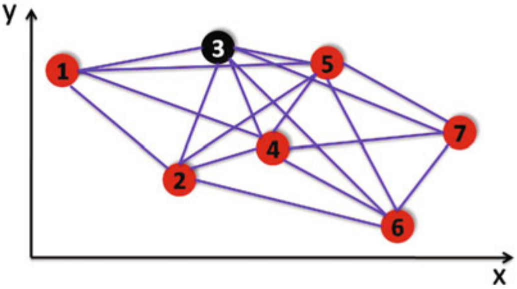


**Fig. 10.2** Embedding patch $P_k$ in its local coordinate frame after it was appropriately reflected and rotated. In the noise-free case, the coordinates $p_i^{(k)} = (x_i^{(k)}, y_i^{(k)})^T$ agree with the global positioning $p_i = (x_i, y_i)^T$ up to some translation $t^{(k)}$ (unique to all $i$ in $V_k$)

### *10.5.3 Step 3: Synchronization over $\mathbb{R}^d$ to Estimate Translations*

In the last step of the 2D-ASAP algorithm we compute the global translations of all patches and obtain the final estimates for the coordinates. For each patch $P_k$, we denote by $G_k = (V_k, E_k)$[4] the graph associated to patch $P_k$, where $V_k$ is the set of nodes in $P_k$, and $E_k$ is the set of edges induced by $V_k$ in the measurement graph $G = (V, E)$. We denote by $p_i^{(k)} = (x_i^{(k)}, y_i^{(k)})^T$ the known local frame coordinates of node $i \in V_k$ in the embedding of patch $P_k$ (see Fig. 10.2). Since each patch $P_k$ has been properly reflected and rotated so that the local frame coordinates are consistent with the global coordinates , up to a translation $t^{(k)} \in \mathbb{R}^2$, in the noise-free case, it holds that

$$p_i = p_i^{(k)} + t^{(k)}, \quad i \in V_k, \quad k = 1, \ldots, N. \tag{10.7}$$

We estimate the global coordinates $p_1, .., p_n$ as the least-squares solution to the overdetermined system of linear equation (10.7), while ignoring the by-product translations $t^{(1)}, \ldots, t^{(N)}$. In practice, we write a linear system for the displacement vectors $p_i - p_j$ for which the translations have been eliminated. From Eq. (10.7) it follows that each edge $(i, j) \in E_k$ contributes a linear equation of the form

$$p_i - p_j = p_i^{(k)} - p_j^{(k)}, \quad (i, j) \in E_k, \quad k = 1, \ldots, N. \tag{10.8}$$

We separate these constraints along the $x$ and $y$ global coordinates of nodes $i$ and $j$, and solve (independently) each of the two resulting linear systems using the ordinary linear regression.

[4] Not to be confused with $G(i) = (V(i), E(i))$ defined in the beginning of this section.

## 10.6 The Eigenvector Method for the Group Synchronization Problem

In general, the synchronization problem can be applied in such settings where the underlying problem exhibits a group structure and one has available noisy measurements of ratios of group elements. We have already seen in the previous sections two such instances of the group synchronization problem. The eigenvector and SDP-based methods for solving angular synchronization problem for the group SO(2) were originally introduced by Singer in [38]. There, one is asked to estimate $N$ unknown angles $\theta_1, \ldots, \theta_N \in [0, 2\pi)$ given $M$ noisy measurements $\delta_{ij}$ of their offsets $\theta_i - \theta_j \mod 2\pi$. The difficulty of the problem is amplified on one hand by the amount of noise in the offset measurements, and on the other hand by the fact that $M << \binom{N}{2}$, i.e., only a small subset of all possible offsets are measured. In general, one may consider any group $\mathscr{G}$ other than SO(2), for which there are available noisy measurements $g_{ij}$ of ratios between group elements

$$g_{ij} = g_i g_j^{-1}, g_i, g_j \in \mathscr{G}.$$

As long as the group $\mathscr{G}$ is compact and has a real or complex representation, one may construct a real or Hermitian matrix (which may be a matrix of matrices) where the element in the position $\{ij\}$ is the matrix representation of the measurement $g_{ij}$ (possibly a matrix of size $1 \times 1$) or the zero matrix if there is no direct measurement for the ratio of $g_i$ and $g_j$. For example, the rotation group SO(3) has a real representation using $3 \times 3$ rotation matrices, and the rotation group SO(2) has a complex representation as points on unit circle $e^{\imath\theta_i} = \cos\theta_i + \imath \sin\theta_i$. One may now make use of the top eigenvectors of this matrix to estimate the unknown group elements. Alternatively, one may use this matrix to formulate an SDP program and extract the unknown group elements. The set $E$ of pairs $\{ij\}$ for which a ratio of group elements is available can be realized as the edge set of a graph $G^P = (V^P, E^P)$, $|V^P| = N, |E^P| = M$ with vertices corresponding to the group elements $g_1, \ldots, g_N$ and edges corresponding to the available measurements $g_{ij} = g_i g_j^{-1}$. Note that we use the superscript to denote the patch graph, as introduced in the previous section. Two vertices $i$ and $j$ of the graph $G^P$ are connected, i.e., $\{ij\} \in E^P$, if and only if their corresponding patches $P_i$ and $P_j$ have enough points in common and can be pairwise aligned.

In Sect. 4 of [16] we give an analysis of the eigenvector method for the group synchronization problem in the noiseless case, using the fact that the eigenvalues of the normalized matrices $\mathscr{Z}$ and $\mathscr{R}$ are related to the those of the discrete normalized graph Laplacian of the underlying patch graph. An analysis of the eigenvector synchronization method in the presence of noise was first explored by Singer [38], in the case of the group SO(2), and uses tools from random matrix theory that allow for a precise matrix perturbation analysis that quantifies the robustness to noise of the method under a certain random noise model. Furthermore, it provides an information theoretic analysis showing that the eigenvector method is asymptotically nearly optimal and achieves the information theoretic Shannon bound up to a multiplicative factor that depends only on the discretization error of the measurements. In very recent work, Bandeira, Singer, and Spielman [6] proved a Cheeger-type inequality via the graph connection Laplacian operator, providing a deterministic worst case performance guarantee for the synchronization problem over the group O(d) of orthogonal transformations.

The eigenvector synchronization method has proven extremely useful in a variety of applications other than the SNL problem. In particular, Hadani et al.[22, 39, 41] demonstrated its usefulness in solving the "class averaging" problem in cryo-electron microscopy [19] and showed its mathematical connection to the parallel transport and the connection Laplacian operators from differential geometry. Other applications include the 3D structure from motion problem in computer vision [3] and the analysis of high-dimensional data point clouds [40], specifically, to robustly compute Laplacian eigenmaps and diffusion maps [7, 13] that are popular methods for dimensionality reduction and spectral clustering.

## 10.7 Experimental Results

We have implemented our 2D-ASAP algorithm and performed numerical simulations, comparing its performance with other methods across a variety of measurement graphs. In this section we report such results for three data sets, and refer the reader to Sect. 8 of [16] for additional numerical experiments. We use multiplicative and uniform noise, meaning that to each true distance measurement $l_{ij} = \parallel p_i - p_j \parallel$, we add random independent noise $\varepsilon_{ij}$ in the range $[-\eta l_{ij}, \eta l_{ij}]$, i.e., $d_{ij} = l_{ij} + \varepsilon_{ij}$ where $\varepsilon_{ij} \sim Uniform([-\eta l_{ij}, \eta l_{ij}])$. The percentage noise added is $100\eta$ (e.g., $\eta = 0.1$ corresponds to 10% noise).

In terms of time complexity, 2D-ASAP scales almost linearly in the size of the network, number of nodes $n$, and edges $m$. We refer the reader to Sect. 7 of [16] for a detailed complexity analysis of each step of the algorithm. We augment this theoretical analysis with the running times of numerical simulations for the localization of networks of increasing sizes $n = 10^3, 10^4, 10^5$, as detailed in Table 10.2.

**Table 10.2** Running times (in seconds) of the ASAP algorithm on the SQUARE graph with $n = \{10^3, 10^4, 10^5\}$ nodes inside the unit square, $\eta = 0\%$ and $deg \approx 12, 13$

| Stage \# of nodes $n$ | 1,000 | 10,000 | 100,000 |
|---|---|---|---|
| Break $G$ into patches | 41 | 901 | 52,180 |
| Embedding patches | 414 | 4,325 | 37,140 |
| Patch intersections | 2 | 132 | 58,134 |
| Build $\mathscr{X}$ | 8.7 | 90 | 2,237 |
| Compute $v_1^{\mathscr{X}}$ | 0.8 | 13 | 926 |
| Build $\mathscr{R}$ | 4.6 | 49 | 3,414 |
| Compute $v_1^{\mathscr{R}}$ | 0.2 | 7 | 522 |
| Step 3 | 6 | 88 | 4,772 |
| Total time (s) | 477 | 5,605 | 159,325 |

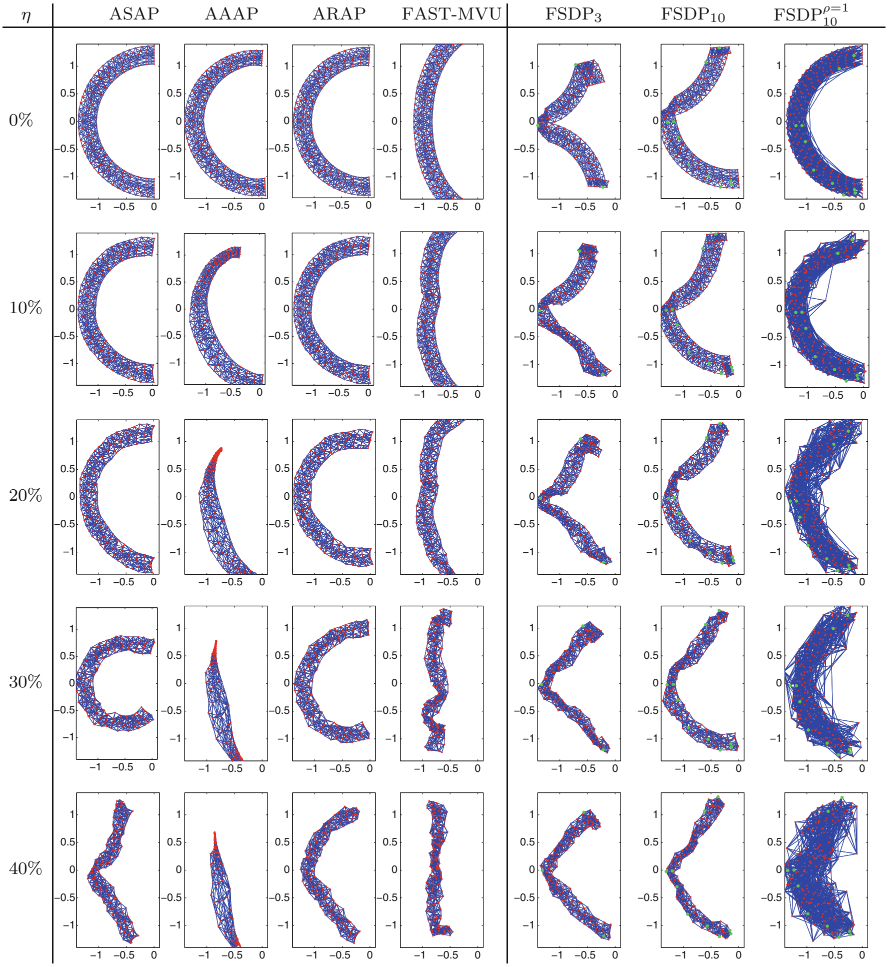


**Fig. 10.3** Reconstructions of the dense C graph with $n = 200$ nodes, $\rho = 0.28$, and $\eta = 35\%, 40\%, 50\%, 60\%$, and $70\%$

The C-shape, graphs in Fig. 10.3 have $n = 200$ nodes, sensing radius $\rho = 0.28$, the average degrees are between 20 and 28, and the noise levels are $\eta = 35\%, 40\%, 50\%, 60\%$, and $70\%$. In addition to ARAP and FAST-MVU, we compare our results against the FULL-SDP algorithm [11] in three different scenarios. In the first two, we run FULL-SDP on the same measurement graph used by the other algorithms, but provide FULL-SDP with additional 3 and 10 anchors placed at random, that are not provided to the other algorithms. We choose the anchors at random from the set of all sensors. In the third scenario, we use a measurement graph of (approximately) the same average degree *deg* as the one used by the other

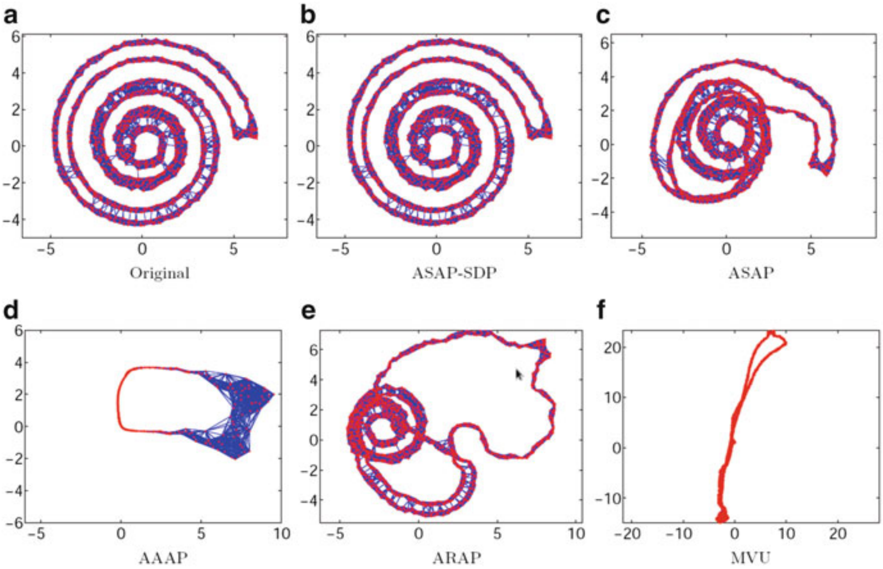


**Fig. 10.4** Reconstructions of the SPIRAL graph with $n = 2259$ nodes, $\rho = 0.47$ and $\eta = 0\%$. ASAP-SDP is a version of ASAP where we used SDP for the localization of the patches, instead of SMACOF

algorithms, but allow FULL-SDP to use a much larger sensing radius $\rho = 1$, while keeping the average degree constant. These experiments show that FULL-SDP is somewhat sensitive to the sensing radius and the number of anchors used.

A second graph we report on is the SPIRAL graph shown in Fig. 10.4a. This graph contains $n = 2259$ nodes that are spread near a spiral curve that starts at the origin, and once it gets to its outermost loop it traces back towards the origin. The perturbation of the sensors from the curve ensures that the 1-hop neighborhoods are not too close to being collinear. The sensing radius for this graph is $\rho = 0.47$. Despite the fact that the measured distances are noise-free, the localizations obtained by both ASAP, AAAP, ARAP, and MVU (Fig. 10.4c, e, f) deviate from the true positioning. The failure of ASAP to find the original embedding in this noise-free case is due to a failure of the SMACOF procedure to localize a small number of patches. Although there is no noise in the distance measurements, the stress minimization algorithm sometimes converges to a local minimum, resulting in patches that are incorrectly localized. Since the topology of this graph is that of a closed curve, such bad patches lead to incorrect twists and turns in our computed embedding. Although ASAP and ARAP are using the same algorithm to localize the patches, it is clear that the incorrectly localized patches are less harmful to ASAP as they are to ARAP. Figure 10.4b shows the accurate embedding obtained by ASAP when SNL-SDP was used to localize the patches, denoted ASAP-SDP.

Finally, in order to illustrate the scaling behavior of ASAP and compare its running time to that of the other algorithms, we experimented with random graphs

**Table 10.3** Comparison of the running times (in seconds) of different algorithms for the SQUARE graph with $n = \{10^3, 10^4\}$ nodes and $\eta = 0\%$

| Algorithm | $n = 1{,}000$ | $n = 10{,}000$ |
|---|---|---|
| ASAP | 477 | 5,605 |
| AAAP | 1,170 | $> 48$ h |
| ARAP | 1,201 | $> 48$ h |
| FAST-MVU | 2.7 | 10.8 |
| FULLSDP$_{20}$ | 5,250 | – |

with $n = \{10^3, 10^4, 10^5\}$ nodes distributed uniformly at random in the unit square, with average degree close to 13. Table 10.2 details the running times of the various steps of the ASAP algorithm for three graphs, and Table 10.3 compares the running times of ASAP, AAAP, ARAP, FAST-MVU, and FULLSDP$_{20}$ for a random graph on $n = 10^3$ nodes.

## 10.8 Summary and Discussion

In this chapter we reviewed 2D-ASAP, a novel non-incremental non-iterative anchor-free algorithm for solving ab initio the SNL problem. Our extensive numerical simulations show that 2D-ASAP is very robust to high levels of noise in the measured distances and to sparse connectivity in the measurement graph. It compares favorably to some of the current state-of-the-art graph localization algorithms both in terms of robustness to noise and running time. Following a "divide and conquer" philosophy, we start with local coordinate computations based only on the 1-hop neighborhood information of individual nodes, but unlike previous incremental methods, we synchronize all such local information in a noise robust global optimization process using an efficient eigenvector computation.

In very recent work [17], we consider the three-dimensional version of the localization problem, and formulate it as a synchronization problem over Euc(3). The problem can be similarly solved in three steps: an eigenvector synchronization for the reflections over $\mathbb{Z}_2$, an eigenvector synchronization for the rotations over SO(3), and a least-squares solution for the translations in $\mathbb{R}^3$. In the second step of the algorithm, the optimal rotations between pairs of patches will be represented by $3 \times 3$ rotation matrices, and the elements of SO(3) will be obtained from the top three eigenvectors. Furthermore, we build on the approach used in 2D-ASAP to accommodate for the additional challenges posed by rigidity theory in $\mathbb{R}^3$ as opposed to $\mathbb{R}^2$. In particular, we extract patches that are not only globally rigid, but also weakly uniquely localizable, a notion that is based on the recent unique localizability of So and Ye [43]. In addition, we also increase the robustness to noise of the algorithm by using a median-based denoising algorithm in the preprocessing step by combining into one step the methods for computing the reflections and rotations and thus doing synchronization over O(3) $= \mathbb{Z}_2\times$ SO(3) rather than individually over $\mathbb{Z}_2$ followed by SO(3). Of equal importance is the possibility to integrate prior available information. As it is often the case in real applications (such as NMR), one has readily available structural information on various parts of the network that we are trying to localize. For example, in the NMR application, there are often subsets of atoms whose relative coordinates are known a priori, and thus it is desirable to be able to incorporate such information in the reconstruction process.

## References

1. Akyildiz, I.F., Su, W., Cayirci, Y.S.E.: Wireless sensor networks: A survey. Comput. Network. **38**, 393–422 (2002)
2. Anderson, B.D.O., Belhumeur, P.N., Eren, T., Goldenberg, D.K., Morse, A.S., Whiteley, W.: Graphical properties of easily localizable networks. Wireless Network. **15**, 177–191 (2009)
3. Arie-Nachimson, M., Basri, R., Singer, A.: in preparation
4. Aspnes, J., Eren, T., Goldenberg, D.K., Morse, A.S., Whiteley, W., Yang, Y.R., Anderson, B.D.O., Belhumeur, P.N.: A theory of network localization. IEEE Trans. Mobile. Comput. **5**, 1663–1678 (2006)
5. Aspnes, J., Goldenberg, D.K., Yang, Y.R.: On the computational complexity of sensor network localization. In: Proceedings of Algorithmic Aspects of Wireless Sensor Networks: First International Workshop (ALGOSENSORS), Lecture Notes in Computer Science, pp. 32–44. Springer (2004)
6. Bandeira, A.S., Singer, A., Spielman, D.A.: A cheeger inequality for the graph connection laplacian, arXiv.12043873
7. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
8. Biswas, P., Aghajan, H., Ye, Y.: Semidefinite programming algorithms for sensor network localization using angle of arrival information. In: Proceedings of the 39th Annual Asilomar Conference on Signals, Systems, and Computers, pp. 220–224 (2005)
9. Biswas, P., Lian, T.C., Wang, T.C., Ye, Y.: Semidefinite programming based algorithms for sensor network localization. ACM Transactions on Sensor Networks **2**, 188–220 (2006) a
10. Biswas, P., Liang, T., Toh, K., Ye, Y., Wang, T.: Semidefinite programming approaches for sensor network localization with noisy distance measurements. IEEE Transactions on Automation Science and Engineering **3**, 360–371 (2006) b
11. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: ACM Conference Proceedings, Third International Symposium on Information Processing in Sensor Networks, pp. 46–54. New York (2004)
12. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications. Springer, New York (2005)
13. Coifman, R.R., Lafon, S.: Diffusion maps. Applied and Computational Harmonic Analysis **21**, 5–30 (2006)
14. Connelly, R., Whiteley, W.J.: Global rigidity: The effect of coning. Discrete Comput. Geom. **43**, 717–735 (2010)
15. Cox, T.F., Cox, M.A.A.: Multidimensional scaling. Monographs on Statistics and Applied Probability 88. Chapman & Hall/CRC, Boca Raton (2001)
16. Cucuringu, M., Lipman, Y., Singer, A.: Sensor network localization by eigenvector synchronization over the Euclidean group. ACM Tran. Sen. Net. **8**(3), 1–42 (2011)
17. Cucuringu, M., Singer, A., Cowburn, D.: Synchronization, graph rigidity and the molecule problem. submitted (2011)

18. De Leeuw, J.: Applications of convex analysis to multidimensional scaling. In: Barra, J.R., Brodeau, F., Romierand, G., Cutsem, B.V. (eds.) Recent Developments in Statistics, pp. 133–146. North Holland Publishing Company, Amsterdam (1977)
19. Frank, J.: Three-dimensional Electron Microscopy of Macromolecular Assemblies: Visualization of Biological Molecules in Their Native State, 2nd edn. Oxford University Press, USA (2006)
20. Giridhar, A., Kumar, P.R.: Distributed clock synchronization over wireless networks: algorithms and analysis. In: IEEE Conference Proceedings, 45th IEEE Conference on Decision and Control, pp. 4915–4920 (2006)
21. Gotsman, C., Koren, Y.: Distributed graph layout for sensor networks. In: Proceedings of the International Symposium on Graph Drawing, pp. 273–284 (2004)
22. Hadani, R., Singer, Representation theoretic patterns in three dimensional Cryo-Electron Microscopy I – the intrinsic reconstitution algorithm, Ann. Math. **174**(2), pp. 1219–1241 (2011).
23. Hendrickson, B.: Conditions for unique graph realizations. SIAM J. Comput. **21**, 65–84 (1992)
24. Hendrickson, B.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
25. Horn, B., Hilden, H., Negahdaripour, S.: Closed-form solution of absolute orientation using orthonormal matrices. J. Opt. Soc. Am. A **5**, 1127–1135 (1988)
26. Javanmard, A., Montanari, A.: Localization from incomplete noisy distance measurements, submitted
27. Ji, X., Zha, H.: Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In: Proceedings of INFOCOM, vol. 4, pp. 2652–2661 (2004)
28. Karp, R., Elson, J., Estrin, D., Shenker, S.: Optimal and global time synchronization in sensornets. Tech. Report, Center for Embedded Networked Sensing, University of California, Los Angeles (2003)
29. Koren, Y., Gotsman, C., Ben-Chen, M.: PATCHWORK: Efficient localization for sensor networks by distributed global optimization. Tech. Report (2005)
30. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. Tech. Report, arXiv.12050349 (2012)
31. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: From continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2011)
32. Moore, D., Leonard, J., Rus, D., Teller, S.: Robust distributed network localization with noisy range measurements. In: Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems, pp. 50–61 (2004)
33. Ozyesil, O., Singer, A.: Synchronization in non-compact groups: Special euclidean group case, submitted
34. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290** 2323–2326 (2000)
35. Saxe, J.B.: Embeddability of weighted graphs in k-space is strongly NP-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
36. Shang, Y., Ruml, W.: Improved MDS-based localization. In: Proceedings of IEEE INFOCOM, vol. 23, pp. 2640–2651. Hong Kong, China (2004)
37. Singer, A.: A remark on global positioning from local distances. Proc. Natl. Acad. Sci. **105**, 9507–9511 (2008)
38. Singer, A.: Angular synchronization by eigenvectors and semidefinite programming. Applied and Computational Harmonic Analysis **30**, 20–36 (2011)
39. Singer, A., Shkolnisky, Three-Dimensional Structure Determination from Common Lines in Cryo-EM by Eigenvectors and Semidefinite Programming. *SIAM* J. Imag. Sci. **4**(2), pp. 543–572 (2011).
40. Singer, A., Wu, H.-T.: Vector diffusion maps and the connection Laplacian, Commun. Pur. Appl. Math. *(CPAM)*, **65**(8), pp. 1067–1144 (2012)
41. Singer, A., Zhao, Z., Shkolnisky, Y., Hadani, R.: Viewing angle classification of cryo-electron microscopy images using eigenvectors. SIAM J. Imag. Sci. **4**, 543–572 (2011)

42. So, A.M.C.: A semidefinite programming approach to the graph realization problem: theory, applications and extensions. Ph.D. Thesis (2007)
43. So, A.M.C., Ye, Y.: Theory of semidefinite programming for sensor network localization. In: Proceedings of the 17th Annual ACM–SIAM Symposium on Discrete Algorithm (SODA), pp. 405–414 (2005)
44. Toh, K., Biswas, P., Ye, Y.: SNLSDP version 0 – a MATLAB software for sensor network localization, October 2008 `http://www.math.nus.edu.sg/ mattohkc/SNLSDP.html`
45. Tubaishat, M., Madria, S.: Sensor networks: an overview. IEEE Potentials **22**, 20–23 (2002)
46. Weinberger, K.Q., Sha, F., Zhu, Q., Saul, L.K.: Graph laplacian regularization for large-scale semidefinite programming. In: Schoolkopf, B., Platt, J., Hofmann, T. (eds.) Advances in neural information processing systems (NIPS), MIT Press, Cambridge (2007)
47. Yemini, Y.: Some theoretical aspects of location-location problems. In: Proceedings of the IEEE Annual Symposium on Foundations of Computer Science, pp. 1–8 (1979)
48. Zhang, L., Liu, L., Gotsman, C., Gortler, S.J.: An As-Rigid-As-Possible approach to sensor network localization. ACM Tran. Sen. Net. **6**(4), 1–21 (2010)
49. Zhu, Z., So, A.M.C., Ye, Y.: Universal rigidity: towards accurate and efficient localization of wireless networks. In: Proceedings of the IEEE INFOCOM, pp. 1–9 (2010)

# Chapter 11
# Global Optimization for Atomic Cluster Distance Geometry Problems

**Marco Locatelli and Fabio Schoen**

**Abstract** This chapter is devoted to a survey of global optimization methods suitable for the reconstruction of the three-dimensional conformation of atomic clusters based on the possibly noisy and imprecise knowledge of a sparse subset of pairwise distances. The problem we address is that of finding the geometry of a three-dimensional object without making use of any structural knowledge, but relying only on a subset of measured pairwise distances. The techniques we present are based on global optimization methods applied to different formulations of the problem. The methods are based on the use of standard local searches within a global optimization (GO) method which is based on local perturbation moves. Different definitions of these perturbations lead to different methods, whose performances are compared. Both sequential and population-based variants of the methods are reviewed in this chapter and some relevant numerical results are presented. From the evidence reported, it can be concluded that, when no additional information is available, such as, e.g., information about a linear order which allows for using a build-up technique, the methods proposed in this chapter represent an effective tool for difficult distance geometry problems (DGPs).

M. Locatelli
Università di Parma, viale G.P. Usberti 181/A, 43100 Parma, Italy
e-mail: locatelli@ce.unipr.it

F. Schoen (✉)
Università di Firenze, via di S. Marta 3, 50139 Firenze, Italy
e-mail: fabio.schoen@unifi.it

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_11,

## 11.1 Introduction

In a *distance geometry problem* (DGP in what follows), given an incomplete subset of possibly noisy distance measures between $N$ points, the aim is to reconstruct the configuration of the $N$ points in the three-dimensional Euclidean space. More formally, we aim at determining the positions for $N$ points $x^1, x^2, \dots, x^N \in \mathbb{R}^3$ so that, for a given subset of pairs $\mathscr{D}$ and given bounds $\ell_{ij}$, $u_{ij}$, the following distance constraints are satisfied:

$$\ell_{ij} \leq ||x^i - x^j|| \leq u_{ij} \qquad \text{for all } \{i,j\} \in \mathscr{D}. \tag{11.1}$$

Note that if $x^{1*}, x^{2*}, \dots, x^{N*}$ is a solution to the problem, any configuration obtained by translating and/or rotating this solution also solves the problem. Therefore, from now on, we will always refer to solutions, modulo translations and rotations. Exact bounds (i.e., $\ell_{ij} = u_{ij}$ for all $\{i,j\} \in \mathscr{D}$) usually make the problem simpler. In fact, if exact bounds are given and $\mathscr{D}$ is made up by all possible pairs of points, the problem is solvable in polynomial time $\mathscr{O}(N^3)$ through eigenvalue decomposition of the distance matrix. However, if not all distances are known and/or lower and upper bounds differ, the problem has been proved to be strongly NP-hard (see [6, 23]).

We would like to stress the fact that in this chapter we are assuming that no information is available, except for the partial list of distance measurements. Thus, even if for some numerical examples we got our data from fragments of proteins, we assume that no information on the linear ordering of residues in the protein is included. In other words, from the point of view of the approaches discussed here, all points (atom centers) are considered as indistinguishable one from the other. In addition, the methods we present here do not rely on pre-processing or other tools aimed at discovering partial structures with known geometry. As an example, when a linear ordering of atoms is known—this situation happens, e.g., when studying the problem applied to proteins—if, given the sequence of atoms, all pairwise distances are exactly known among all consecutive four-tuples of atoms, then it is easy to see that the problem, apart from a few degenerate cases which never happen in nature, is trivially solvable by elementary geometric arguments.

Thus the approaches we present here are quite more general than those based on geometric build-up strategies (see, e.g., [7]) and can be applied to even widely different conformational problems. On the other hand, when compared with methods which are strongly based on such a knowledge, our methods clearly lag behind.

Many geometrical problems where points have to be placed in the two- or three-dimensional space in such a way that some constraints are satisfied and/or some objective function is minimized or maximized, have been tackled by global optimization (GO in what follows) techniques. They include, for instance, molecular conformation problems (such as Lennard–Jones and Morse clusters, see, e.g., [8, 9, 15, 16, 18, 25]) or packing problems (see, e.g., [1]).

The quite promising results obtained for such problems suggest that also the DGP can be effectively tackled by similar GO approaches. For this reason, in this chapter, we will focus our attention on GO approaches for the DGP. We point out, however, that the DGP has been tackled also by many other different approaches in the literature as it can be seen browsing the different chapters of this volume—thus we do not review them in this chapter, referring the reader to the other sections in this volume.

Problem (11.1) turns out to be a nonlinear, non-convex feasibility problem. Given $X = (x^1, x^2, \ldots, x^N) \in \mathbb{R}^{3N}$, we can define the relative error function

$$\begin{aligned} E_r(X) = \sum_{\{i,j\}\in\mathscr{D}} & \left[ \max^2\left( \frac{\ell_{ij}^2 - ||x^i - x^j||^2}{\ell_{ij}^2}, 0 \right) \right. \\ & \left. + \max^2\left( \frac{||x^i - x^j||^2 - u_{ij}^2}{u_{ij}^2}, 0 \right) \right]. \end{aligned} \tag{11.2}$$

Solutions to Eq. (11.1) (if any) coincide with global minimizers of the unconstrained GO problem whose objective function is Eq. (11.2) (the objective function value in any solution is 0). Similarly, one can use the absolute error function

$$\begin{aligned} E_a(X) = \sum_{\{i,j\}\in\mathscr{D}} & \left[ \max^2\left( \ell_{ij}^2 - ||x^i - x^j||^2, 0 \right) \right. \\ & \left. + \max^2\left( ||x^i - x^j||^2 - u_{ij}^2, 0 \right) \right]. \end{aligned} \tag{11.3}$$

We remark that both functions $E_a$ and $E_r$ are smooth, which allows for the use of efficient local minimization procedures in search algorithms.

This chapter is structured as follows. In Sect. 11.2 we will outline solution approaches based on multiple local searches; in Sect. 11.3 we introduce the test instances which will be employed throughout the chapter; in Sect. 11.4 we will make some preliminary observations based on some computational experiments. In Sect. 11.5 the basic approach will be extended to a population-based version. Finally, in Sect. 11.6, we will draw some conclusions and indicate some possible directions for future research.

## 11.2 GO Approaches: DGSOL and Basin Hopping

An interesting approach based on the GO reformulation of the DGP was the DGSOL algorithm proposed by Moré and Wu [21, 22]. DGSOL is based on *global* *continuation* techniques, and works by successive minimizations of smoothed functions $\langle E_a(X)\rangle_\lambda$, where

$$\langle E_a(X)\rangle_\lambda = \frac{1}{\pi^{3N/2}\lambda^{3N}}\int_{\mathbb{R}^{3N}} E_a(Y)\exp\left(-\frac{||Y-X||}{\lambda^2}\right)\mathrm{d}Y$$

is obtained by convolution with a Gaussian function for decreasing values of the parameter $\lambda \geq 0$ ($\langle E_a(X)\rangle_\lambda$ converges to $E_a$). The effect of a large $\lambda$ value is that of filtering the oscillations of the original function $E_a$ out, thus making the resulting function a convex one. Then, the $\lambda$ value is slowly reduced, and the oscillations are gradually restored until we are back to the original function when $\lambda = 0$. By decreasing $\lambda$ and starting a local search from the previously detected local minimizer, we may hope to jump over the oscillations separating the local minimizers until we reach the global minimizer.

The idea of filtering the oscillations out is also the basis of the approach proposed here. Following an approach already employed in the field of molecular conformation problems (see, e.g., [24]) we can modify the original function $E_a$ as follows:

$$E'_a(X) = E_a(\mathscr{L}\mathscr{S}(X)),$$

where $\mathscr{L}\mathscr{S}$ is a local search procedure. The role played by the parameter $\lambda$ in DGSOL (filtering the oscillations out) is played here by the local search procedure. However, here we need to specify how to escape from a local minimizer. Indeed, each local minimizer lies in a flat region of the function $E'_a$, and we need a way to escape from this region and reach a different local minimizer. In the approach proposed in this chapter, this is accomplished through the definition of a neighborhood structure between local minimizers. Let us denote such structure by $\mathscr{N}$. Then, following the terminology in [6], (see also [19]) a *funnel* $\mathscr{F}$ can be defined as a maximal set of local minimizers such that for each $X \in \mathscr{F}$, there exists at least one decreasing sequence of neighbor local minimizers

$$X_0 = X \to X_1 \to \cdots \to X_t = X^*$$

$$E_a(X_i) < E_a(X_{i-1}),\ \ X_i \in \mathscr{N}(X_{i-1})\ i = 1,\ldots,t$$

starting at $X$ and ending at a common local minimizer $X^*$, called the *funnel bottom*. Note that the global minimizer is always a funnel bottom, independently of the neighborhood structure $\mathscr{N}$. The neighborhood must possess two properties (sometimes conflicting with each other):

- It should have a manageable size (in particular, it should be much smaller than the whole set of local minimizers).
- It should be such that the corresponding number of funnel bottoms is much lower than the overall number of local minimizers (in the easiest cases, the function $E_a$ has a single funnel, whose funnel bottom is the global minimizer, in spite of the huge number of local minimizers).

**Algorithm 5** The MBH algorithm with local move $\Phi$

```
Let $X_0$ be an initial local minimizer, $k = 0$
while a stopping rule is not satisfied do
    Let $Y_{k+1} = \Phi(X_k)$
    if $E_a(Y_{k+1}) < E_a(X_k)$ then
        set $X_{k+1} = Y_{k+1}$
    end if
    set $k = k + 1$
end while
```

Once the notions of neighborhood structure, funnel and funnel bottom have been introduced, we need some effective algorithm to explore funnels, and detect funnel bottoms. The algorithm that we are going to describe here is called monotonic basin hopping (MBH) algorithm, introduced in [16, 24] in the context of molecular conformation problems. The key operation of this algorithm is the *local move* $\Phi$ which, for a given local minimizer $X$, returns a local minimizer in its neighborhood, i.e.,

$$\Phi(X) \in \mathscr{N}(X).$$

Algorithm 5 is a sketch of the algorithm. The algorithm stops either when $E_a(X_k) = 0$ (in such case a solution has been found), or when $X_k$ has not changed for a prefixed number of iterations denoted by MAXITER. The choice of the local move, and hence the choice of the neighborhood it explores, is essential for the performance of MBH (all the ingenuity of the method, which is otherwise extremely simple, lies in the choice of the local move). An indicative rule for its choice is that the local move should generate a new local minimizer but without completely disrupting the structure of the current one $X$ (the principle is very similar to that of iterated local searches in combinatorial optimization problems, see [20]). In case the function has more than one funnel bottom, it may be necessary to run MBH more than once (in a multistart fashion) before detecting the global minimizer. For the DGP we propose here two different local moves. One move is quite general and the other more problem-specific.

### 11.2.1 Local Move $\Phi_{1,\Delta}$

The first local move that we tested is an extremely simple one and is basically the same employed for molecular conformation problems (see [8, 16, 18, 24]). Given a local minimizer $X$, a local search procedure $\mathscr{LS}$, and some $\Delta > 0$, we set

$$\Phi_{1,\Delta}(X) = \mathscr{LS}(X + \delta),$$

where $\delta$ is randomly sampled within the box $[-\Delta,\Delta]^{3N}$. Basically, we randomly perturb the current local minimizer, by controlling the size of the perturbation through $\Delta$, and then we start a local search from the perturbed point. The choice of the parameter $\Delta$ is not a trivial one. Driven by the observation for which the local move should, at least partially, keep the structure of the original configuration, we first tried to consider small values for $\Delta$. In particular, in the experiments reported later, we tried to set $\Delta = 0.05D$, where $D$ represents the length of the edges of a box surely containing the final configuration (in all the experiments, $D$ was chosen to be 100). However, the experiments revealed that such value is too small: when starting a local search from the point returned by this perturbation, very often the result was the original configuration, i.e., such perturbation often turned out to belong to the region of attraction of the current minimizer and, in fact, the local move did not make any move at all. For this reason, we increased the value up to $\Delta = 0.5D$. At a first glance, such perturbation appears to be too large (recall that all molecules in our test instances lie within a box with edge size equal to $D = 100$). In order to verify the behavior of the perturbations, we produced the histograms representing the distances between corresponding points (after appropriate rotations and translations) in both the initial configuration $X$ and the perturbed one $\Phi_{1,\Delta}(X)$. Such histograms were often concentrated around small distances. Therefore, it seems that, in spite of the large $\Delta$ value, the local search procedure is able to drive back the perturbed point towards a local minimizer close to the original one. For the sake of completeness we also tested a larger perturbation size $\Delta = 5D$. As expected, in this case, the histograms were usually not concentrated around small distances but more spread, i.e., the final configuration $\Phi_{1,\Delta}(X)$ appeared to be only mildly related with the initial one $X$.

As a final remark, it is important to point out that, while we selected a priori a single value for $\Delta$, a more reasonable approach is that of using some adaptive rule to update the value of $\Delta$ according to the outcomes of the different iterations of the algorithm, so that the dependency of such value from the problem at hand can be taken into account.

### *11.2.2 Local Move $\Phi_{2,\Gamma}$*

In the move described in Sect. 11.2.1, the only way to include some a priori knowledge about the problem and some information collected by the algorithm is through the parameter $\Delta$. In fact, it may be a good idea to exploit more both information. This can be accomplished through a problem-specific local move, as the one that we are going to discuss. While defining a perturbation on the current configuration, we are supposed to take into account which of the distances in $\mathscr{D}$ are currently violated. Therefore, one possible idea is to consider for each point $r$ the most violated distance in $\mathscr{D}$ within the current configuration $X_k$, and then to move the point $r$ in a direction which should reduce the violation of such distance (the point $r$ is not moved at all if no violation involving the point $r$ occurs within the current configuration). Even though quite reasonable, this strategy did not deliver good results. However, a strategy based on the same idea turned out to be very effective. In the new strategy, we considered the current configuration $X_k$, as well as another configuration, and the perturbation is based on both. Instead of considering the distances of a point $r$ within the current configuration with respect to the other points in the same configuration, we considered the distances of a point $r$ with respect to points in the other configuration. More precisely, we first introduced the local minimizer $Z_k$

$$Z_k = \arg\max\{E_a(X_{k-1}), E_a(Y_k)\}, \tag{11.4}$$

i.e., $Z_k$ is equal to the last discarded point, which is $Y_k = \Phi(X_{k-1})$, if $Y_k$ has been rejected at iteration $k-1$, otherwise it is equal to $X_{k-1}$. Then, as previously commented, for each point $r = 1, \dots, N$, we take the point $s(r)$ which mostly violates one of the distances in $\mathscr{D}$ involving the point $r$, when the point $r$ is taken from $X_k$, while the other points $s$ are taken from $Z_k$, i.e.,

$$s(r) \in \arg \max_{s:(r,s)\in\mathscr{D}} \{0, \|x_k^r - z_k^s\| - u_{rs}, \ell_{rs} - \|x_k^r - z_k^s\|\}.$$

The corresponding maximum distance is denoted by $\eta(r)$. Then, we try to move each point $r$ in such a way that the value $\eta(r)$ is reduced by moving $r$ towards the point $s(r)$ in $Z_k$ (if we have a violation of the upper bound) and far from $s(r)$ in $Z_k$ (if we have a violation of the lower bound). More formally, we introduce a direction vector $D_k = (d_k^1, \dots, d_k^N)$ whose $r$th coordinate is defined as follows:

$$d_k^r = \begin{cases} 0 & \text{if } \eta(r) = 0 \\ -(x_k^r - z_k^{s(r)}) & \text{if } \eta(r) = \|x_k^r - z_k^{s(r)}\| - u_{rs(r)}. \\ (x_k^r - z_k^{s(r)}) & \text{otherwise} \end{cases}$$

Finally, we perturb the current configuration along direction $D_k$ and start a local search from the perturbed point, i.e., we define the local move as follows:

$$\Phi_{2,\Gamma}(X) = \mathscr{L}\mathscr{S}(X + \gamma D_k),$$

where $\gamma$ is randomly sampled in the interval $[0, \Gamma]$ (in our tests we fixed $\Gamma = 4$).

## 11.3 Test Instances

One of the contexts from which DGPs arise is molecular chemistry, where one aims at determining the geometric structures of large molecules (e.g., proteins) from NMR experiments, which usually only deliver a subset of pairwise atom distances, and from X-ray diffraction measurements on crystallized proteins. In such a context, the problem is known as molecular DGP (MDGP), and each point in $\{1, \dots, N\}$

represents (the center of) an atom. All test instances in this chapter have been generated from data from the protein data bank (PDB, see [2]); we considered two classes of instances. First of all, we considered some instances from the work by Moré and Wu [22]; such instances have been generated from the `1GPV` molecule in the PDB. Moré and Wu took 100-atom and 200-atom fragments, and generated distance data for pairs in successive residues $R_k$, $R_{k+1}$:

$$\mathscr{D} = \{\{i,j\} : i \in R_k, j \in R_{k+1}\},$$

setting the bounds to

$$\ell_{ij} = (1-\varepsilon)d_{ij}, \qquad u_{ij} = (1+\varepsilon)d_{ij}, \qquad \{i,j\} \in \mathscr{D},$$

with $\varepsilon$ set to values from 0.04 to 0.16, with $\varepsilon = 0.04$ yielding the hardest instances. We considered two 200-atom instances—here called `DNA200a` and `DNA200b`—with $\varepsilon = 0.04$ provided with the DGSOL package.[1]

We then generated instances of the MDGP by a technique similar to that used in [3] for the exact MDGP. We selected a number of molecules from the PDB for various sizes $N$, and for each of them we kept in $\mathscr{D}$ only the pairs with interatomic distances $d_{ij}$ below the cutoff value $R = 6$ Å; then we set

$$\ell_{ij} = (1-\varepsilon)d_{ij}, \qquad u_{ij} = (1+\varepsilon)d_{ij}, \qquad \{i,j\} \in \mathscr{D}$$

with $\varepsilon$ randomly set in the interval $(0, 0.04)$. The choice of the interval is related to the observations given in [22], where bounds $\ell_{ij} = (1-0.04)d_{ij}$, $u_{ij} = (1+0.04)d_{ij}$ are claimed to give difficult instances—indeed, computational experience not discussed in the remainder of this chapter confirms that for larger $\varepsilon$ the resulting instances are quite easy. It is important to remark that usually, if the sequence of residuals is explicitly taken into account, test problems obtained by using a cutoff distance of 6 Å are relatively easy to solve by geometrical arguments or by the approaches proposed by [7, 14]. Here we stress the fact that our approach does not use information on the linear sequence of amino acids. One might argue that disregarding this information makes the problem unnecessarily hard; however, from one side there are many cases in which no linear ordering is given, e.g., the case of clusters of atoms. Also, there may be situations in which the protein being analyzed has an unknown primary structure, and the aim of the NMR observation is exactly that of obtaining both the primary and the tertiary (i.e., 3D) conformation of the molecule. In these cases atoms are considered as undistinguishable, and a GO approach seems to be the best alternative available. Some properties of the tested

[1] Available at www.mcs.anl.gov/~more/dgsol.

**Table 11.1** Instances used for testing

| Name | $N$ | Density (%) |
|---|---|---|
| DNA200a | 200 | 1.7 |
| DNA200b | 200 | 16.5 |
| 1PTQ | 402 | 8.8 |
| 1HOE | 558 | 6.5 |
| 1LFB | 641 | 5.6 |
| 1PHT | 814 | 5.3 |
| 1POA | 914 | 4.1 |
| 1AX8 | 1,003 | 3.7 |

instances are reported in Table 11.1, where the density parameter is the fraction of known distances with respect to all distances, i.e., it is equal to

$$2|\mathscr{D}|/[N(N-1)].$$

## 11.4 Preliminary Observations on Numerical Experiments

In our first set of preliminary computational experiments, we aimed at testing the behavior of multiple local searches started from randomly generated initial points. Basically, we ran a multistart approach, where a given number of local searches (1,000 in our experiments) is started from randomly sampled points over a $3N$-dimensional boxes, i.e., $D = [-R,R]^{3N}$, for some $R > 0$. We point out that, here and in what follows, local searches have always been performed through limited memory BFGS [17]. Of course, multistart is a rather inefficient approach for highly multimodal GO problems, but its simplicity allows to analyze the behavior of local searches without being influenced by other components of the solution approach. Two interesting observations can be drawn from the experiments.

The first observation is that a relevant role is played by the size of the initial box. When using the objective function (11.3), the number of times a solution was detected for the instances presented in Table 11.1 over 1,000 local searches was clearly superior for a large $R$ value (namely, $R = 5{,}000$) with respect to a small $R$ value (namely, $R = 50$).[2] A possible explanation for this fact is the following. When we generate the initial configuration in a very large box, most bounds for the distances in $\mathscr{D}$ are strongly violated. Therefore, in function (11.3), strong *attractive* forces come into play, pushing the points, initially spread around, much closer to each other. According to the experiments, it seems that the attractive forces are able to drive the local search procedure in such a way that it jumps over local minimizers with high function values and reaches more

[2] Note that, from the set of known distances, it is possible to guarantee that a box of edge size equal to 100 is able to enclose all the molecules for the tested instances.

easily local minimizers with low function value (if not even the global minimizer). An analogous phenomenon has been observed in [15] for so-called Big-Bang algorithm for Lennard–Jones clusters. That algorithm works by generating an initial configuration of atoms in a Lennard–Jones cluster in a very narrow region and then starting a local search from such configuration. The strong *repulsive* forces which are active within the narrow region allow to spread the atoms during the local search in such a way that the detection of good local minimizers is considerably simplified.

The second interesting observation is related to the choice of the objective function. From the theoretical point of view, there is no difference between the functions (11.2) and (11.3): in both cases, a global minimizer with function value equal to 0 is a solution of our problem. However, there are clear differences from the computational point of view. By considering the same $R$ value for both functions, the number of successes with the objective function (11.2) is clearly inferior with respect to the number of successes with the objective function (11.3). Just to cite a single significative result, with $R = 5{,}000$ and test PTQ, we have 49 successes out of 1,000 local searches with the function (11.3) and 0 successes with the function (11.2). We point out that the superiority of the function (11.3) has not only been observed with the multistart algorithm but also with all the other methods tested in this chapter. Thus, from now on we will always employ the function (11.3). A possible explanation for the superiority of the absolute value function with respect to the relative one is the following. Obviously, the larger the deviation from the given bounds, the deeper is the required modification of the current configuration. This is taken into account in the absolute error function but might not emerge in the relative error function. Indeed, if a large deviation occurs for a distance whose bounds are also large, the corresponding term in the relative error function is of comparable size with respect to a small deviation for a distance whose bounds are small ones. Thus, decreasing the second error has the same impact, from the point of view of the relative error function, with respect to decreasing the first error, but, from the point of view of the final configuration, the modification in the configuration when trying to reduce the second error is considerably less relevant with respect to the modification when trying to reduce the first error.

### *11.4.1 Computational Results for MBH*

In Table 11.2, we report the experimental results over the test instances for MBH with the two local moves described above ($\Phi_{1,\Delta}$ with $\Delta = 50$, and $\Phi_{2,\Gamma}$ with $\Gamma = 4$). The results are taken from [11] and reported here for sake of completeness. The observation of the results in the table allows us to draw some interesting conclusions. The first one is that MBH with both local moves has a high number of successes over 10 runs (see columns SUCC). This means that both neighborhood structures

**Table 11.2** Number of successes (columns SUCC) over 10 runs, overall average number of local searches per success (columns NSavg), and average number of local searches in successful runs (columns NSavg-succ) for MBH with the two local moves $\Phi_{1,\Delta}$ and $\Phi_{2,\Gamma}$

| | $\Phi_{1,\Delta}$ | | | $\Phi_{2,\Gamma}$ | | |
|---|---|---|---|---|---|---|
| Instance | Succ | NSavg | NSavg-succ | Succ | NSavg | NSavg-succ |
| DNA200a | 10 | 151 | 151 | 10 | 39 | 39 |
| DNA200b | 10 | 24 | 24 | 10 | 20 | 20 |
| PTQ | 10 | 15 | 15 | 7 | 217 | 3 |
| HOE | 10 | 17 | 17 | 8 | 139 | 14 |
| LFB | 10 | 55 | 55 | 8 | 192 | 67 |
| PHT | 10 | 55 | 55 | 8 | 136 | 11 |
| POA | 9 | 180 | 124 | 8 | 152 | 27 |
| AX8 | 10 | 107 | 107 | 9 | 74 | 18 |
| GPV | 4 | 888 | 138 | 6 | 470 | 137 |

have a quite limited number of funnels (or, at least, the funnel corresponding to the global minimizer is extremely large with respect to other funnels). Another interesting observation arises from the comparison of the average number of local searches per success (columns NSavg), and the average number of local searches in successful runs (columns NSavg-succ). We could expect that the problem-specific local move $\Phi_{2,\Gamma}$ dominates the general local move $\Phi_{1,\Delta}$. In fact, this is not always true. Especially over the smaller instances, the number of successes and the average number of local searches per success for the local move $\Phi_{1,\Delta}$ are better than that of $\Phi_{2,\Gamma}$. But if we look at the average number of local searches only in the successful runs we observe that $\Phi_{2,\Gamma}$ is almost always better (and usually much better) than $\Phi_{1,\Delta}$. Basically, the problem-specific local move $\Phi_{2,\Gamma}$ reaches more easily a funnel bottom, but most likely such a funnel bottom is not a global minimizer. In such case, before stopping MBH, we need to pay the MAXITER local searches (500 in our tests) which have to be added each time a failure occurs. On the other hand, when starting within the funnel corresponding to the global minimizer, the local move $\Phi_{2,\Gamma}$ usually reaches the global minimizer (much) faster than the local move $\Phi_{1,\Delta}$. Of course, one could overcome this difficulty if we could choose MAXITER as small as possible. Unfortunately, the selection of an appropriate value for MAXITER is not a trivial task: as suggested by the large variability of the NSavg-succ values in Table 11.2, the appropriate value might be quite different from instance to instance.

## 11.5 Population Basin Hopping

The experimental results discussed above for MBH show that it would be important to recognize as soon as possible when a run of MBH leads to a failure. This is true in general and is especially true for $\Phi_{2,\Gamma}$, where the NSavg-succ values reveal that, when starting from a good point, the effort to reach a global minimizer by MBH is quite limited. As commented above, we should choose `MAXITER` as small as possible, but it is not clear at all how to choose it: if the value is too large, it causes an undesired computational waste, otherwise it could prejudice the ability of reaching the funnel bottom even when starting in the funnel corresponding to the global minimizer.

A possible way to overcome this difficulty is to substitute $K$ trajectories which are independently and sequentially followed by $K$ runs of MBH with $K$ different trajectories which are followed in parallel (and, possibly, not independently but with some information exchange) within a population-based approach. This obviously increases not only the effort per iteration (by a factor of $K$) but also the probability of success, and, above all, the total number of iterations to reach the global minimizer is determined by the shortest of the $K$ trajectories, which partially counterbalances the larger effort per iteration. Note that we still have to choose a parameter (the number $K$ of trajectories), but a good choice for it appears to be less variable from instance to instance (in our tests, we always fixed $K = 10$).

The easiest way to follow $K$ trajectories is to run $K$ parallel independent runs of MBH with no information exchange between them. However, some more ingenuity can be introduced in the algorithm by allowing for some kind of information exchange. This results in the population basin hopping (PBH) algorithm already tested in [9] on cluster optimization problems. The main elements of this algorithm are a local move $\Phi$ as in MBH, and a new relevant component, the dissimilarity measure $d$ between local minimizers (see Algorithm 6 for a sketch).

**Algorithm 6** The PBH algorithm with local move $\Phi$ and dissimilarity measure $d$

```
Let $\{X_0^1,\dots,X_0^N\}$ be an initial set of $N$ local minimizers, $k=0$
while a stopping rule is not satisfied do
    Let $Y_i = \Phi(X_k^i)$, $i=1,\dots,N$
    for $j \leftarrow 1$ to $N$ do
        Let $i(j) \in \arg\min_{i=1,\dots,N} d(Y_j, X_k^i)$
        if $E_a(Y_j) < E_a(X_k^{i(j)})$ then
            set $X_{k+1}^{i(j)} = Y_j$
        end if
    end for
    set $k = k+1$
end while
```

Once again, we stop either when $E_a(X_k^i) = 0$ for some member of the population, or when we reach a prefixed number of iterations (in fact, in our tests only the former rule came into play). In the PBH algorithm, at each iteration $k$, the population $\{X_k^1,\dots,X_k^N\}$ is available. A set $\{Y_1,\dots,Y_N\}$ of candidate points, where $Y_j = \Phi(X_k^j)$ for $j = 1,\dots,N$, is generated. Then, the candidate point $Y_j$ is not necessarily compared with $X_k^j$ but with the member $X_k^{i(j)}$ of the population which is less dissimilar to $Y_j$. Dissimilarity is measured by some function $d$. If we define, e.g.,

$d(Y_j, X_k^i) = |j - i|$, then obviously we always have $i(j) = j$, and Algorithm 6 is equivalent to $N$ independent runs of Algorithm 5. However, other choices for $d$ are possible. Ideally, $d$ should be very close to 0 if two local minimizers have a high probability of lying in the same funnel, while it should increase if such probability decreases. If this were the case, members of a funnel would tend to be compared only with members of the same funnel, so that $d$ acts as a diversification tool which allows to explore different funnels and avoids the multiple exploration of the same one. As extensively discussed in [4, 9, 10], dissimilarity measures allow for communication and collaboration among members of the population. When no communication and collaboration occurs, PBH simply follows $K$ independent trajectories, and the only difference with respect to $K$ independent MBH runs is that the latter are run sequentially, while the former are followed in parallel. While this is usually a minor difference for the DGP, such difference has a relevant effect. Indeed, in the DGP we know in advance the global optimum value (which is equal to 0) so that, as soon as a configuration with objective function value equal to 0 is reached, we can stop the search along *all* trajectories. In some sense, this is a form of communication and collaboration because as soon as one trajectory reaches a global minimizer, it immediately communicates it to the other ones, thus avoiding unnecessary computational waste. But if we use more sophisticated dissimilarity measures, also other advantages may come into play. In general, a good dissimilarity measure allows to counterbalance the greedy tendency of MBH (too fast convergence to a funnel bottom not corresponding to a global minimizer) leading to a much greater efficiency with respect to the case of $K$ independent trajectories, as observed in [9] and in the related experimental analysis in [10]. This usually happens when many funnel bottoms exist and/or the funnel bottom corresponding to the global minimizer can be reached through trajectories of considerably different lengths. In fact, for the DGP we cannot claim at the moment that we have found a dissimilarity measure which brings the above-mentioned advantages. After testing a few dissimilarity measures, we restricted our attention to a very general and simple one, the absolute value of the difference between function values:

$$d(X, Y) = \mid E_a(X) - E_a(Y) \mid . \tag{11.5}$$

The results we obtained with this measure (reported in Table 11.3) are good ones but not much better than those which can be obtained by following independent trajectories in parallel. However, a positive effect of collaboration and communication between members of the population will be discussed at the end of the section.

Before discussing the results, we just remark a small difference in the definition of $Z_k$ with respect to Eq. (11.4) for the local move $\Phi_{2,\Gamma}$. In PBH we define a point $Z_k^j$ for each member $j$ of the population as follows:

$$Z_k^j = \arg\max\{E_a(X_{k-1}^{i(j)}), E_a(Y_k^j)\},$$

**Table 11.3** Number of successes (columns SUCC) over 10 runs and average number of local searches per success (columns NSavg) for PBH with the two local moves $\Phi_{1,\Delta}$ and $\Phi_{2,\Gamma}$

| | $\Phi_{1,\Delta}$ | | $\Phi_{2,\Gamma}$ | |
|---|---|---|---|---|
| Instance | Succ | NSavg | Succ | NSavg |
| DNA200a | 10 | 116 | 10 | 143 |
| DNA200b | 10 | 30 | 10 | 23 |
| PTQ | 10 | 57 | 10 | 33 |
| HOE | 10 | 67 | 10 | 53 |
| LFB | 10 | 192 | 10 | 89 |
| PHT | 10 | 162 | 10 | 63 |
| POA | 10 | 234 | 10 | 88 |
| AX8 | 10 | 263 | 10 | 87 |
| GPV | 10 | 949 | 10 | 312 |

i.e., $Z_k^j$ is the loser of the last competition between the new candidate point $Y_k^j = \Phi(X_{k-1}^j)$ and its competitor $X_{k-1}^{i(j)}$, the member of the population at iteration $k-1$ less dissimilar to $Y_k^j$. The results with both local moves $\Phi_{1,\Delta}$ and $\Phi_{2,\Gamma}$ are reported in Table 11.3. As expected from the results with MBH, a population size $K = 10$ was already enough to reach 100 % of successes on all test instances. Moreover, as expected from the values in Columns NSavg-succ in Table 11.2, we observe that better results are usually obtained with the problem-specific move $\Phi_{2,\Gamma}$ with respect to $\Phi_{1,\Delta}$.

As previously mentioned, while the dissimilarity measures tested up to now do not give impressive improvements with respect to following in parallel $K$ independent trajectories, we could in fact observe a positive effect of the dissimilarity measure. We worked in order to identify a proper value for the parameter $\Gamma$ in the local move $\Phi_{2,\Gamma}$. At the beginning, we tried $\Gamma = 2$, and this choice turned out to be a bad one. Indeed, with the neighborhood structure corresponding to this choice, a large number of funnels were created. Both MBH and PBH with $K = 10$ had a very large number of failures in this case. We could get back to 100% successes only after enlarging the population size to $K = 40$. We analyzed the behavior of PBH during these runs with population size $K = 40$, and we realized that the phenomenon of survival, one of the two phenomena (the other is backtracking) which, according to the analysis in [9,10], is responsible for the success of PBH, had a great positive impact on the final results. Survival occurs when a member $X_k^j$ of the population generates a better child $Y_{k+1}^j$, i.e., $E_a(Y_{k+1}^j) < E_a(X_k^j)$, but $i(j) \neq j$ so that $X_k^j$ will still survive in the next population. This phenomenon counterbalances the greedy tendency of MBH (in MBH $X_k^j$ would be simply replaced by $Y_{k+1}^j$). Such tendency might cause too fast convergence to funnel bottoms not corresponding to the global minimizer, while survival allows to keep within the population local minimizers from which it is still possible to reach the global minimizer. Therefore, although overall the performance of PBH with $\Gamma = 2$ and $K = 40$ is inferior to that with $\Gamma = 4$ and $K = 10$, the collaboration and communication between members of the population which takes place through the dissimilarity measure is able to considerably reduce the negative effects of a wrong choice of the parameter $\Gamma$ defining the local move.

## 11.6 Possible Directions for Future Research

In this chapter, we considered a reformulation of the DGP as a GO problem and tackled it with some approaches (MBH and its population-based version PBH) which already turned out to be particularly suitable for geometrical problems which can be reformulated as GO ones, such as molecular conformation and packing problems. The results of our computational experiments suggest that these are indeed promising approaches to tackle DGPs. However, in this field, there are still many directions which could be explored. Here we mention a few of them.

- We reformulated the DGP as an unconstrained GO problem with objective function (11.2) or (11.3). In Sect. 11.4 we observed that the performance of any GO algorithm can be considerably different when using Eq. (11.2) or (11.3). In particular, Eq. (11.3) appears to be much better (and we gave a tentative explanation of this fact). We might wonder if other (theoretically) equivalent reformulations of the DGP are possible which turn out to be more efficient than Eq. (11.3). For instance, for any increasing one-dimensional function $f$, the DGP is equivalent to the unconstrained minimization of $f(E_a(X))$.
- When dealing with very large instances, the CPU time required by local searches may be very large. A good idea in this case could be that of decomposing the problem: different and mildly correlated (within the subset $\mathscr{D}$ of distances) parts are separately optimized, and then some technique is employed to merge together the different parts (the idea of decomposing solutions was also exploited in the ABBIE approach [12] and in the SDP approach proposed in [3]).
- In this chapter, we have only proposed two local moves, a general and a problem-specific one. Surely, the geometrical nature of the problem might suggest further local moves. For instance, one could think about moves like the surface-repair ones employed in [5] or atom relocation techniques like those used in [13] for molecular conformation problems. For DGP, one might try to localize the moves in such a way that only points which are involved in violated distances in $\mathscr{D}$ are moved (in fact, this is already in the spirit of the local move $\Phi_{2,\Gamma}$).
- There is space to find more effective dissimilarity measures. At the moment, we have not found any which really enhances the performance of the proposed PBH approach (although, as we observed, the proposed one might be helpful in order to counterbalance bad choices of the parameters).

## References

1. Addis, B., Locatelli, M., Schoen, F.: Efficiently packing unequal disks in a circle. Oper. Res. Lett. **36**, 37–42 (2008)
2. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Res. **28**, 235–242 (2000)

3. Biswas, P., Toh, K.-C., Ye, Y.: A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. SIAM J. Sci. Comput. **30** 1251–1277 (2008)
4. Cassioli, A., Locatelli, M., Schoen, F.: Dissimilarity measures for population-based global optimization algorithms. Comput. Optim. Appl. **45**, 257–281 (2010)
5. Cheng, L., Feng, Y., Yang, J., Yang, J.: Funnel hopping: Searching the cluster potential energy surface over the funnels. J. Chem. Phys. **130**(21), 214112 (2009)
6. Crippen, G., Havel, T.: Distance Geometry and Molecular Conformation. Wiley (1988)
7. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global. Optim. **26**, 321–333 (2003)
8. Doye, J.P.K., Leary, R.H., Locatelli, M., Schoen, F.: The global optimization of Morse clusters by potential transformations. INFORMS Journal on Computing **16**, 371–379 (2004)
9. Grosso, A., Locatelli, M., Schoen, F.: A population based approach for hard global optimization problems based on dissimilarity measures. Math. Program. **110**, 373–404 (2007)
10. Grosso, A., Locatelli, M., Schoen, F.: An experimental analysis of population based approach for global optimization. Comput. Optim. Appl. **38**, 351–370 (2007)
11. Grosso, A., Locatelli, M., Schoen, F.: Solving molecular distance geometry problems by global optimization algorithms. Comput. Optim. Appl. **43**, 23–37 (2009)
12. Hendrickson, B.A.: The molecular problem: determining conformation from pairwise distances. Ph.D. Thesis, Cornell University (1991)
13. Lai, X., Xu, R., Huang, W.: Prediction of the lowest energy configuration for Lennard-Jones clusters. Sci. China Chem. **54**, 985–991 (2011)
14. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
15. Leary, R.H.: Global optima of Lennard-Jones clusters. J. Global. Optim. **11**, 35–53 (1997)
16. Leary, R.H.: Global optimization on funneling landscapes. J. Global. Optim. **18**, 367–383, 2000.
17. Liu, D.C., Nocedal, J.: On the Limited Memory Method for Large Scale Optimization. Math. Program. **B45**, 503–528 (1989)
18. Locatelli, M., Schoen, F.: Efficient algorithms for large scale global optimization: Lennard-Jones clusters. Comput. Optim. Appl. **26**, 173–190 (2003)
19. Locatelli, M.: On the multilevel structure of global optimization problems. Comput. Optim. Appl. **30**, 5–22 (2005)
20. Lourenço, H.R., Martin, O., Stützle, T.: Iterated Local Search. In: Gendreau, M., Potvin, J.-Y. (eds.) Handbook of Metaheuristics, pp. 320–353. Springer (2003)
21. Moré, J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**, 814–836 (1997)
22. Moré, J., Wu, Z.: Distance geometry optimization for protein structures. J. Global. Optim. **15**, 219–234 (1999)
23. Saxe, J.B.: Embeddability of graphs in $k-$space is strongly NP-hard. In: Proceedings of the 17th Allerton Conference in Communication, Control and Computing, pp. 480–489 (1979)
24. Wales, D.J., Doye, J.P.K.: Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. J. Phys. Chem. **A101**, 5111–5116 (1997)
25. Wales, D.J.: Energy Landscapes with Applications to Clusters, Biomolecules and Glasses. Cambridge University Press, Cambridge (2003)

# Chapter 12
# Solving Molecular Distance Geometry Problems Using a Continuous Optimization Approach

**Rodrigo S. Lima and J.M. Martínez**

**Abstract** The molecular distance geometry problem consists in finding the positions in $\mathbb{R}^3$ of atoms of a molecule, given some inter-atomic distances. In this work we formulate this problem as a nonlinear optimization problem and solve some instances using a continuous optimization routine. For each proposed experiment, we compare the numerical solution obtained with the true structure. This comparison is performed by solving a Procrustes problem.

**Keywords** Molecular distances • Nonlinear programming • Numerical experiments

## 12.1 Introduction

In this work we propose and solve some computational experiments involving instances of the molecular distance geometry problem [11]. We employ a continuous optimization software to find numerical solutions to the problem. Our objective is to reconstruct three-dimensional structures of proteins using only the distances between their atoms. To attain this goal, we need to determine a set of $n$ points $\{x^1, x^2, \ldots, x^n\} \subset \mathbb{R}^3$ such that $\|x^i - x^j\| = \hat{d}_{ij}$, where $\hat{d}_{ij}$ is the Euclidean distance between the atoms $i$ and $j$. We can formulate this task as a continuous optimization problem as follows:

R.S. Lima
Department of Mathematics and Computation, ICE-UNIFEI, Federal University of Itajubá,
37500-903 Itajubá MG, Brazil
e-mail: rodlima@unifei.edu.br

J.M. Martínez
Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas,
13081-970 Campinas SP, Brazil
e-mail: martinez@ime.unicamp.br

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_12,

$$\begin{aligned}\text{minimize } & \sum_{i,j} (\|x^i - x^j\| - \hat{d}_{ij})^2, \\ \text{subject to } & x^i \in \mathbb{R}^3, i = 1, 2, \ldots, n.\end{aligned} \tag{12.1}$$

The variables in Eq. (12.1) are the coordinates of points $x^i \in \mathbb{R}^3$, and the objective function is not differentiable when $x^i = x^j$, for some $i, j$. However, as the distances between atoms are always positive real numbers, we can apply a minimization algorithm that uses first derivatives to solve the problem (12.1). Then, if $\hat{d}_{ij} > 0$ for all $i, j$, the local minimizers of Eq. (12.1) are configurations that do not contain coincident points. This result was proved by Jan de Leeuw in [4]. In the computational experiments, we solve some instances of the molecular distance geometry problem using *GENCAN* [2]. This routine, available at

www.ime.usp.br/~egbirgin/tango

is able to find approximate solutions to minimization problems with box constraints. For each considered instance, the numerical solutions obtained by *GENCAN* were compared to the true structure of the analyzed protein. The comparison was carried out as follows: given the true configuration of the protein and a numerical solution, we determine a transformation that superimposes both structures in some optimal manner. This problem is known as the Procrustes problem [6, 8].

The Procrustes problem consists in finding an orthogonal matrix $Q \in \mathbb{R}^{3\times 3}$ that minimizes the function

$$g(Q) = \|M_0 - M_1 Q\|_F, \tag{12.2}$$

where $M_0$ and $M_1$ are matrices in $\mathbb{R}^{n\times 3}$ and $\|.\|_F$ is the Frobenius norm. The orthogonal matrix $Q$ that minimizes Eq. (12.2) has a closed-form expression. In the book of Golub and Van Loan [5], a singular value decomposition is employed to determine $Q$. This way of solving Eq. (12.2) does not ensure that the orthogonal matrix $Q$ is a rotation matrix. There are cases where $Q$ is the composition of a rotation and of a reflection. Kearsley in [9] uses unitary quaternions to find $Q$, and, as a result, he always obtains a rotation matrix. More references about quaternions and rotations can be found in [3, 7, 10, 12]. We desire to investigate if the numerical solutions obtained by *GENCAN* to the problem (12.1) differ from the original configurations by transformations involving a pure rotation or a rotation followed by a reflection. For this, we solve the Procrustes problem applying both proposed techniques: singular value decomposition and unitary quaternions.

## 12.2 Numerical Experiments

We selected some proteins from protein data bank [1] and we considered only the alpha carbon coordinates ($C_\alpha$) of each structure. The selected proteins and the number of atoms ($n_{C_\alpha}$) are indicated in Table 12.1. We initially propose three sets

**Table 12.1** Proteins used in the computational experiments

| Protein | $n_{C_\alpha}$ |
|---|---|
| 1AMU | 509 |
| 1OOH | 126 |
| 2O12 | 407 |
| 3RAT | 124 |
| 6PAX | 133 |
| 1lmO | 3,535 |

**Table 12.2** First set of experiments: all the distances are known

| Protein | ndist | nvar | iter | evalf | $f(x^*)$ | $t(s)$ | pure rot. | rot. + ref. |
|---|---|---|---|---|---|---|---|---|
| 1AMU | 129,286 | 1,527 | 20 | 39 | 1.21E−19 | 1.76 | 13 | 7 |
| 1OOH | 7,875 | 378 | 14 | 30 | 3.61E−19 | 0.08 | 12 | 8 |
| 2O12 | 82,621 | 1,221 | 17 | 36 | 1.10E−19 | 0.99 | 12 | 8 |
| 3RAT | 7,626 | 372 | 17 | 27 | 3.84E−19 | 0.11 | 13 | 7 |
| 6PAX | 8,778 | 399 | 18 | 42 | 7.27E−19 | 0.21 | 6 | 8 |
| 1lmO | 6,246,345 | 10,605 | 26 | 68 | 5.59E−21 | 120.23 | 101 | 99 |

of tests with these proteins; the details are discussed below. All experiments in this work have been carried out on a single core of an Intel Core 2 CPU 2.4GHz with 2GB RAM, running MAC OS X 10.5.

### *12.2.1 Solving Problems Using All Distances Between Atoms*

In this set of experiments we suppose that all the distances $\hat{d}_{ij}$ between the atoms are known. With the first five proteins of Table 12.1, we solve the problem (12.1) using *GENCAN* twenty times whereas the problem with 1lmO protein was solved two hundred times, where each run corresponds to a different starting point. Table 12.2 shows the results of runs for which *GENCAN* reached the lowest objective function value. The columns in this table have the following meaning: *ndist* is the total number of distances between pairs of atoms, *nvar* is the number of variables, *iter* and *evalf* are, respectively, the total number of iterations and evaluations of the objective function, $f(x^*)$ is the final objective function value, and $t(s)$ is the CPU time in seconds. The column *pure rot.* shows the quantity of runs in which the optimization routine obtained a solution that differs from the true structure by a transformation involving a pure rotation. The column *rot.* + *ref.* indicates the total of rounds in which the numerical solution differs from the true structure by a transformation involving a rotation followed by a reflection. The stopping criterion of *GENCAN* in all tests required that the gradient norm had to be smaller than $10^{-4}$.

The final values of the objective function show that *GENCAN* finds configurations of points in $\mathbb{R}^3$ that fit all the distances. However, we noted in six tests related to the protein 6PAX that the routine obtains configurations with $f(x^*) \approx 10^3$ and gradient norm smaller than $10^{-4}$. These configurations are certainly local minimizers.

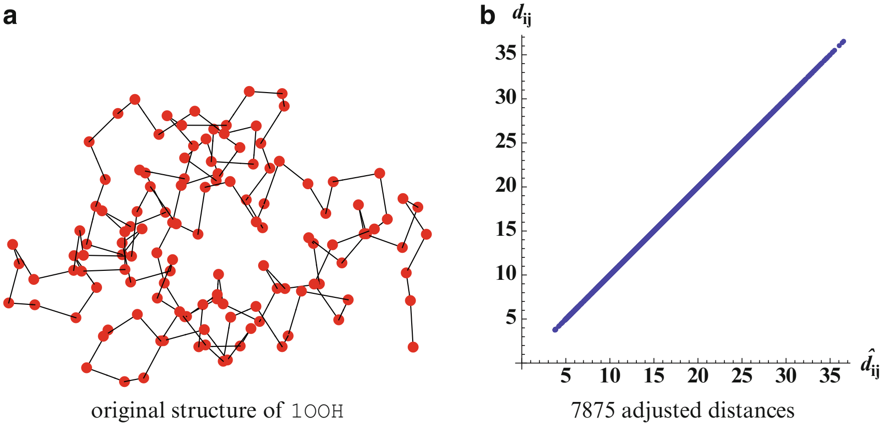


**Fig. 12.1** Experiments with 1OOH protein

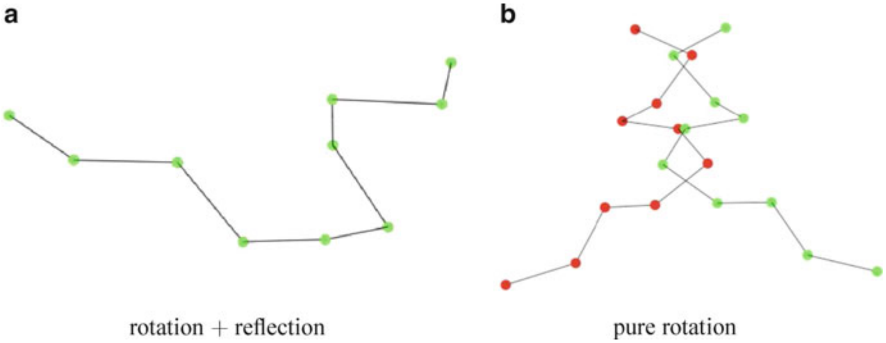


**Fig. 12.2** 1OOH protein: solving procrustes problem to compare structures

We chose the protein 1OOH to illustrate a test where *GENCAN* obtains a solution that differs of the true configuration by a transformation involving reflection. Figure 12.1a shows the true structure of 1OOH with 126 alpha carbons. Each atom is represented by a point in $\mathbb{R}^3$ and consecutive points are joined by lines. Figure 12.1b compares the original distances between pairs of atoms ($\hat{d}_{ij}$ axis) to the distances obtained numerically (axis $d_{ij}$).

The analysis with the Procrustes problem is shown in Fig. 12.2. To construct this figure we use only ten first consecutive $C_\alpha$ atoms of 1OOH. We solve the Procrustes problem (12.2) using the two formulations discussed above. Figure 12.2a shows the optimal superimposition of the true and numerical structures. The numerical solution (red points) does not appear in this image because it is superimposed by the true structure (green points). The transformation matrix obtained in this case involves a reflection and was obtained solving Eq. (12.2) with the strategy proposed

**Table 12.3** Results of procrustes problem with 1OOH protein

| Procrustes: Golub and Van Loan’s strategy | |
|---|---|
| $Q = \begin{pmatrix} 0.545563 & 0.463014 & -0.698555 \\ 0.835074 & -0.229917 & 0.49979 \\ -0.0708004 & 0.856012 & 0.512085 \end{pmatrix},$ | $g(Q) = 2.87132\text{E}{-}10,$ |
| Procrustes: Kearsley’s strategy | |
| $Q = \begin{pmatrix} 0.583586 & 0.810877 & 0.0436581 \\ 0.798626 & -0.563372 & -0.211681 \\ -0.147052 & 0.158401 & -0.976363 \end{pmatrix},$ | $g(Q) = 1.37579\text{E}{+}02.$ |

by G. Golub and C. Van Loan. Figure 12.2b shows the superimposition obtained by applying the strategy proposed by Kearsley. In this case, the orthogonal matrix describes a pure rotation. We note in Fig. 12.2b that the numerical configuration (red points) is the reflected image of the original one (green points). In this case, it is not possible to determine a rotation that superimposes both structures. The results obtained for the Procrustes problem are reported in Table 12.3.

### *12.2.2 Simulating Errors*

In this set of experiments, we consider the first five proteins of Table 12.1 and we suppose that all distances between pairs of atoms were obtained with errors. To simulate this situation, we add to each value $\hat{d}_{ij}$ a random number created in the interval $[-\rho, \rho]$, with $|\rho| \leq 1$. Then, for each protein and each fixed value $\rho$, we solve Eq. (12.1) twenty times with a different starting point in each run. All tables below show only the results corresponding to the tests in which *GENCAN* reached the lowest final value of objective function. Table 12.4 indicates the final values of objective function attained by *GENCAN* and Table 12.5 shows the performance of the routine for solving each instance in terms of iterations, evaluations of the objective function, and CPU time. According to Table 12.4, when we increase the parameter $\rho$, the final values of the objective function also increase by a factor of $10^2$. The performance of *GENCAN* in these problems was quite similar to the results obtained in correspondence with problems considered in the first set of experiments.

We built some figures for an experiment related to the protein 3RAT, where we fixed $\rho = 1$ (fifth line and last column of Table 12.4). Figure 12.3a shows the true structure of 3RAT with 124 alpha carbons, and Fig. 12.3b indicates the result of superimposing both structures (true and numerical) by a transformation involving a pure rotation. For building this figure, we use only the first ten atoms of structures: true (green points) and numerical (red points). In this case, we obtained the same

**Table 12.4** Final values of objective function reached by *GENCAN*

| Protein | $\rho = 10^{-5}$ | $\rho = 10^{-4}$ | $\rho = 10^{-3}$ | $\rho = 10^{-2}$ | $\rho = 10^{-1}$ | $\rho = 1$ |
|---|---|---|---|---|---|---|
| 1AMU | 4.263864E−06 | 4.263864E−04 | 4.263864E−02 | 4.263864E+00 | 4.263864E+02 | 4.263871E+04 |
| 1OOH | 2.527256E−07 | 2.527256E−05 | 2.527256E−03 | 2.527253E−01 | 2.527222E+01 | 2.526905E+03 |
| 2O12 | 2.709833E−06 | 2.709833E−04 | 2.709833E−02 | 2.709833E+00 | 2.709808E+02 | 2.709746E+04 |
| 3RAT | 2.453024E−07 | 2.453024E−05 | 2.453024E−03 | 2.453025E−01 | 2.453041E+01 | 2.453186E+03 |
| 6PAX | 2.829028E−07 | 2.829028E−05 | 2.829028E−03 | 2.829023E−01 | 2.828973E+01 | 2.828446E+04 |

**Table 12.5** Performance of *GENCAN* in tests with errors

| | $\rho = 10^{-5}$ | | | $\rho = 10^{-4}$ | | | $\rho = 10^{-3}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Protein | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ |
| 1AMU | 18 | 33 | 1.75 | 17 | 35 | 1.60 | 17 | 34 | 1.50 |
| 1OOH | 16 | 40 | 0.10 | 20 | 38 | 0.11 | 17 | 30 | 0.10 |
| 2O12 | 21 | 52 | 1.31 | 21 | 45 | 1.53 | 18 | 34 | 1.25 |
| 3RAT | 16 | 29 | 0.11 | 17 | 34 | 0.13 | 17 | 35 | 0.13 |
| 6PAX | 16 | 34 | 0.17 | 22 | 48 | 0.26 | 19 | 50 | 0.15 |
| | $\rho = 10^{-2}$ | | | $\rho = 10^{-1}$ | | | $\rho = 1$ | | |
| Protein | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ |
| 1AMU | 20 | 34 | 1.79 | 24 | 63 | 1.76 | 18 | 40 | 1.84 |
| 1OOH | 15 | 33 | 0.09 | 13 | 27 | 0.08 | 16 | 36 | 0.09 |
| 2O12 | 22 | 51 | 1.50 | 18 | 27 | 1.27 | 22 | 45 | 1.76 |
| 3RAT | 15 | 19 | 0.11 | 16 | 30 | 0.12 | 15 | 28 | 0.10 |
| 6PAX | 17 | 41 | 0.17 | 18 | 41 | 0.19 | 21 | 56 | 0.22 |

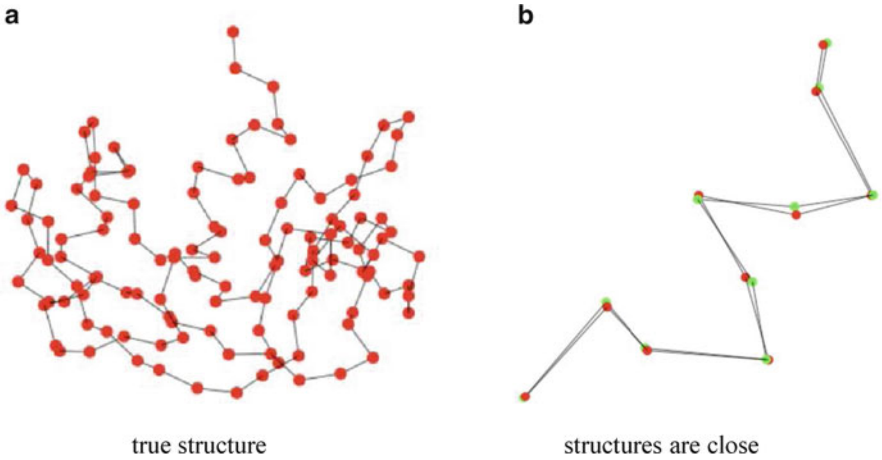


**Fig. 12.3** Test with 3RAT protein

orthogonal matrix by solving the Procrustes problem with the two approaches described above:

$$Q = \begin{pmatrix} -0.728719 & -0.651336 & 0.211497 \\ 0.141985 & -0.44583 & -0.883785 \\ 0.669932 & -0.614001 & 0.417364 \end{pmatrix}, \quad g(Q) = 2.02778.$$

Figure 12.4 shows a graph where the $x$-and $y$-axes represent, respectively, the perturbed distances and the distances obtained by solving Eq. (12.1) with *GENCAN*. Although the final value of objective function is not small, the points are close to the line $y = x$.

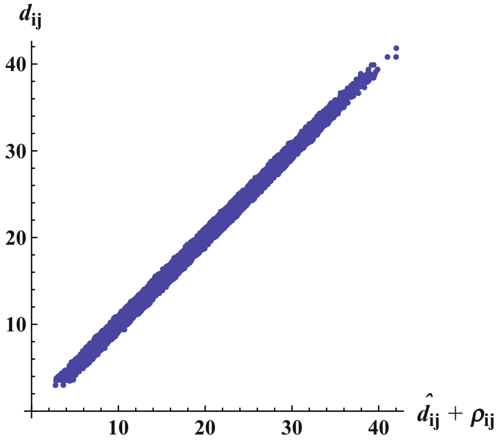


**Fig. 12.4** Simulating 7,626 distances with errors

### 12.2.3 Solving Problems Using a Subset of Interatomic Distances

In these experiments, we use the same proteins reported in Table 12.1. However, we try here to recover the true structure using only distances not greater than a fixed parameter $d_{\text{fix}}$. Assuming that the distances are known exactly, we varied the parameter value $d_{\text{fix}}$ and we analyzed the obtained results using the Procrustes technique. To each protein and each value $d_{\text{fix}}$, we solve the problem (12.1) fifty times with a multistart strategy: a different initial point was used in each run. In all the cases, *GENCAN* stopped when the gradient norm was lower than $10^{-4}$.

Tables 12.6 and 12.7 show only information corresponding to the tests with the lowest value of the objective function attained by *GENCAN*. The total number of distances between pairs of atoms (*ndist*), the number of distances used to solve the problem (12.1) (*nd*), and the final value of the objective function ($f(x^*)$) are reported in Table 12.6. The performance of routine is indicated in Table 12.7. These results show that *GENCAN* can find configurations of points that fit all distances between atoms using less than 36 % of the known distances.

We illustrate a test with the protein 6PAX where we attempt to recover the true structure considering only distances not greater than 10 Å. Figure 12.5a shows the true structure with 133 alpha carbons and Fig. 12.5b compares the original distances between atoms ($\hat{d}_{ij}$) with the distances in the numerical solution ($d_{ij}$). We can see in the graph that the points are concentrated around the line $y = x$. To create Fig. 12.6a, we solved the Procrustes problem using a singular value decomposition and we obtained a transformation involving a reflection. In the case of Fig. 12.6b, we applied the quaternion approach and, as a result, we obtained a pure rotation matrix. These results are shown in Table 12.8.

Table 12.6 Final values of objective function attained by *GENCAN*

| | | $d_{\text{fix}} = 6$ | | $d_{\text{fix}} = 10$ | | $d_{\text{fix}} = 15$ | |
|---|---|---|---|---|---|---|---|
| Protein | ndist | nd | $f(x^*)$ | nd | $f(x^*)$ | nd | $f(x^*)$ |
| 1AMU | 129,286 | 1,614 | 8.438471E+00 | 4,650 | 2.514611E+02 | 14,193 | 5.554569E−17 |
| 1OOH | 7,875 | 386 | 1.929190E−01 | 1,074 | 3.657396E−17 | 2,824 | 1.198010E−18 |
| 2O12 | 82,621 | 1,221 | 4.402781E+00 | 3,899 | 5.413680E+00 | 11,083 | 2.446779E−13 |
| 3RAT | 7,626 | 393 | 1.189479E+00 | 1,045 | 2.463908E−10 | 2,724 | 2.767508E−18 |
| 6PAX | 8,778 | 386 | 4.858342E−03 | 916 | 2.438740E+02 | 2,292 | 1.311720E−13 |
| | | $d_{\text{fix}} = 10$ | | $d_{\text{fix}} = 9$ | | $d_{\text{fix}} = 8$ | |
| Protein | ndist | nd | $f(x^*)$ | nd | $f(x^*)$ | nd | $f(x^*)$ |
| 1lmO | 6246345 | 1085393 | 1.393982E−17 | 839689 | 6.667112E−15 | 677434 | 4.904778E−13 |

**Table 12.7** Performance of *GENCAN* in the resolution of problems

| | $d_{\text{fix}}=6$ | | | $d_{\text{fix}}=10$ | | | $d_{\text{fix}}=15$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Protein | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ |
| 1AMU | 121 | 428 | 48.88 | 52 | 173 | 2.53 | 64 | 192 | 2.88 |
| 1OOH | 150 | 304 | 8.490 | 34 | 93 | 0.15 | 18 | 35 | 0.08 |
| 2O12 | 328 | 733 | 137.91 | 57 | 176 | 3.06 | 36 | 127 | 1.60 |
| 3RAT | 111 | 244 | 5.19 | 44 | 115 | 0.45 | 28 | 78 | 0.14 |
| 6PAX | 66 | 160 | 3.54 | 142 | 326 | 6.34 | 49 | 84 | 1.80 |
| | $d_{\text{fix}}=10$ | | | $d_{\text{fix}}=9$ | | | $d_{\text{fix}}=8$ | | |
| Protein | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ | iter | evalf | $t(s)$ |
| 1lmO | 63 | 245 | 103.51 | 65 | 235 | 111.52 | 35 | 98 | 66.41 |

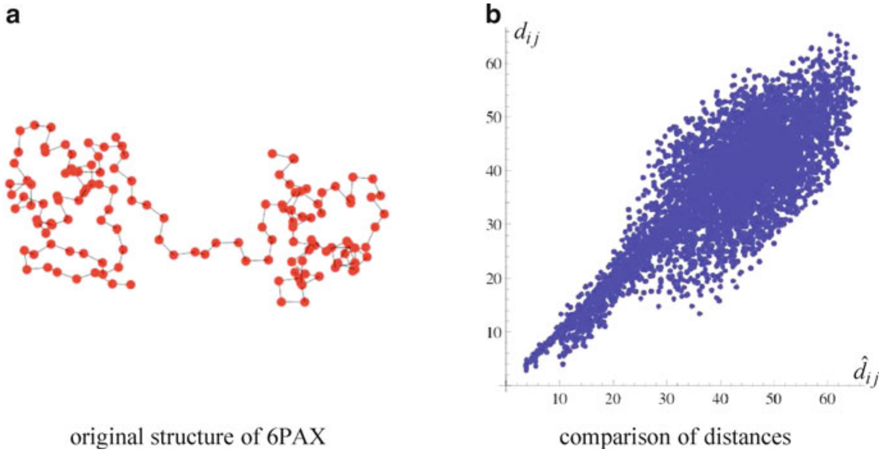


**Fig. 12.5** Experiments with 6PAX protein

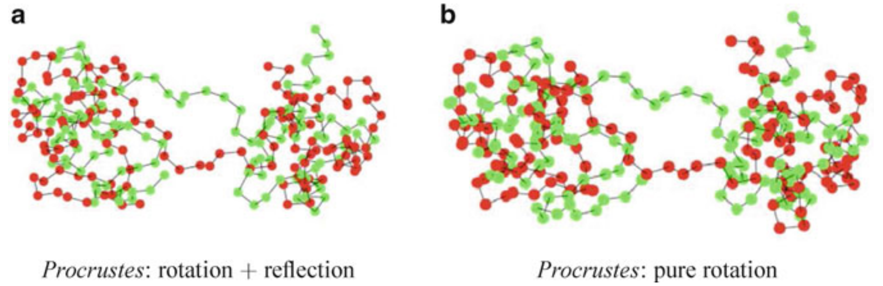

a
b
*Procrustes*: rotation + reflection
*Procrustes*: pure rotation

**Fig. 12.6** Comparison of structures via *procrustes*

According to the experiments, we can conclude that it is possible to use a continuous optimization routine to recover a 3D structure of a protein using only a subset of known distances between pairs of atoms. In particular, if we provide to *GENCAN* a reasonable starting point, the routine solves the problem very quickly.

**Table 12.8** Results of procrustes problem with `6PAX` protein

Procrustes: Golub and Van Loan's strategy

$$Q = \begin{pmatrix} 0.619471 & -0.528366 & -0.580591 \\ -0.756835 & -0.59837 & -0.262972 \\ 0.208462 & -0.602315 & 0.770558 \end{pmatrix}, \qquad g(Q) = 1.17023\text{E}{+}02,$$

Procrustes: Kearsley's strategy

$$Q = \begin{pmatrix} 0.584185 & -0.546667 & -0.599903 \\ 0.787414 & 0.20257 & 0.582189 \\ -0.196741 & -0.812478 & 0.548792 \end{pmatrix}, \qquad g(Q) = 1.30287\text{E}{+}02.$$

**Table 12.9** Comparing *GENCAN* and *MDJEEP*

| | | | | | *GENCAN* | | *MDJEEP* | |
|---|---|---|---|---|---|---|---|---|
| Protein | nat | ndist | nd | $d_{\text{fix}}$ | $t(s)$ | $E_{\text{sol}}$ | $t(s)$ | $E_{\text{sol}}$ |
| `1CRN` | 138 | 9,453 | 1,250 | 6.0 | 1.41 | 9.63E−08 | 0.001 | 9.63E−05 |
| `1PTQ` | 150 | 11,175 | 1,263 | 6.0 | 1.65 | 9.53E−04 | 0.001 | 9.78E−05 |
| `2ERL` | 120 | 7,140 | 1,136 | 6.0 | 0.44 | 4.17E−08 | 0.001 | 8.65E−05 |
| `1PPT` | 108 | 5,778 | 1,039 | 6.5 | 0.74 | 6.15E−04 | 0.001 | 9.51E−05 |
| `1PHT` | 249 | 30,876 | 2,631 | 6.5 | 5.60 | 3.66E−08 | 0.002 | 9.34E−05 |
| `1HOE` | 222 | 24,531 | 2,715 | 7.0 | 0.79 | 1.85E−10 | 0.002 | 8.12E−05 |
| `3RAT` | 372 | 69,006 | 4,567 | 7.0 | 3.37 | 1.53E−09 | 0.004 | 8.82E−05 |
| `1A70` | 291 | 42,195 | 4,472 | 8.0 | 33.48 | 6.37E−10 | 0.003 | 7.59E−05 |

### *12.2.4 Comparing* GENCAN *and* MD-jeep

To finish this work, we compare the performances of *GENCAN* to the ones of a software tool named *MD-jeep*. *MD-jeep* was developed specifically to solve molecular distance geometry problems using combinatorial optimization techniques [13]. This software was written in C by Mucherino et al., and it is freely distributed at

www.antoniomucherino.it/en/mdjeep.php

In order to run the experiments, we used eight instances obtained from protein conformations downloaded from Protein Data Bank. We extracted the coordinates of atoms $N$, $C_\alpha$, and $C$ from each structure. For each protein, only distances not greater than $d_{\text{fix}} = 6$ Åwere considered as input for the routines. To solve the problems with *GENCAN*, we employ a multistart strategy: we perform runs until the routine provides a solution that differs from the original structure by a transformation involving a pure rotation. *GENCAN* stopped in all tests with the gradient norm smaller than $10^{-4}$. The solutions of *MD-jeep* listed in Table 12.9 differ from original structure by a linear transformation involving a rotation matrix. The columns of Table 12.9 have the following meaning: *nat* is the number of atoms $N$, $C_\alpha$, and $C$ present in each protein, *ndist* is the total number of distances between pairs of atoms, *nd* is the number of distances not greater than $d_{\text{fix}} = 6$ Å, and $t(s)$ is the CPU time, in seconds.

After solving the problems with both packages, we analyze the quality of the solutions obtained using the error formula

$$E_{\text{sol}} = \frac{1}{n_d} \sum_{i,j} \frac{|\hat{d}_{ij} - d_{ij}|}{d_{ij}}, \tag{12.3}$$

where $\hat{d}_{ij}$ is the original distance between the atoms $i$, $j$, $d_{ij}$ is the final distance between the points $x^i, x^j \in \mathbb{R}^3$, and $n_d$ is the number of distances used in each test. We employed a Fortran procedure to evaluate the numerical solutions using the formula (12.3). The results are shown in the columns $E_{\text{sol}}$ of Table 12.9. According to Table 12.9, we can see that both routines attain good solutions to the problems. *GENCAN* obtains smaller values to the error (12.3), but *MD-jeep* is much faster.

**Acknowledgements** The authors are thankful to PRONEX-Optimization (PRONEX - CNPq / FAPERJ E-26 / 171.164/2003 - APQ1), FAPESP (Grant 06/53768-0), and CNPq.

## References

1. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Res. **28**, 235–242 (2000)
2. Birgin, E.G., Martínez, J.M.: Large-scale active-set box-constrained optimization method with spectral projected gradients. Comput. Optim. Appl. **23**, 101–125 (2002)
3. Curtis, M.L.: Matrix Groups. Springer, New York (1984)
4. De Leeuw, J.: Differentiability of Kruskal's Stress at a Local Minimum. Psychometrika **49**, 111–113 (1984)
5. Golub, G.H., Van Loan, C.F.: Matrix Computations. The Johns Hopkins University Press (1996)
6. Gower, J.C., Dijksterhuis, G.B.: Procrustes Problems. Oxford University Press (2004)
7. Horn, B.K.P.: Closed-form solution of absolute orientation using unit quaternions. J. Opt. Soc. Am. **4**, 629–642 (1987)
8. Kabsch, W.: A discussion of the solution for the best rotation to relate two sets of vectors. Acta Crystallogr. **A34**, 827–828 (1978)
9. Kearsley, S.K.: On the orthogonal transformation used for structural comparisons Acta Crystallogr. **A45**, 208–210 (1989)
10. Kuipers, J.B.: Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality. Princeton University Press, New Jersey (2002)
11. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
12. Lima, R.S.: Representation of rotations: advantages and disadvantages in theory and practice. Master Thesis, Department of Applied Mathematics, IMECC, UniCamp (2007)
13. Mucherino, A., Liberti, L., Lavor, C.: `MD-jeep`: an implementation of a branch and prune algorithm for distance geometry problems. In: Fukuda, K., et al. (eds.) Proceedings of the Third International Congress on Mathematical Software (ICMS10), Lectures Notes in Computer Science, vol. 6327, pp. 186–197. Kobe, Japan (2010)

# Chapter 13
# DC Programming Approaches for Distance Geometry Problems

**Hoai An Le Thi and Tao Pham Dinh**

**Abstract** In this chapter, a so-called DCA method based on a DC (difference of convex functions) optimization approach for solving large-scale distance geometry problems is developed. Two main problems are considered: the exact and the general distance geometry problems. Different formulations of equivalent DC programs are introduced. Substantial subdifferential calculations permit to compute sequences of iterations in the DCA quite simply and allow exploiting sparsity in the large-scale setting. For improving the computational efficiency of the DCA schemes we investigate several techniques. A two-phase algorithm using shortest paths between all pairs of atoms to generate the complete dissimilarity matrix, a spanning trees procedure, and a smoothing technique are investigated in order to compute a good starting point (SP) for the DCAs. An important issue in the DC optimization approach is well exploited, say the *nice* effect of DC decompositions of the objective functions. For this purpose we propose several equivalent DC formulations based on the stability of Lagrangian duality and the regularization techniques. Finally, many numerical simulations of the molecular optimization problems with up to $12,567$ variables are reported which prove the practical usefulness of the nonstandard nonsmooth reformulations, the globality of found solutions, the robustness, and the efficiency of our algorithms.

H.A. Le Thi (✉)
Laboratory of Theoretical and Applied Computer Science (LITA EA 3097) UFR MIM,
University of Lorraine, Ile du Saulcy, 57045 Metz, France
e-mail: hoai-an.le-thi@univ-lorraine.fr

T. Pham Dinh
Laboratory of Mathematics, National Institute for Applied Sciences, Rouen, BP 08,
Avenue de l'Université, F 76800 Saint-Etienne-du-Rouvray, France
e-mail: pham@insa-rouen.fr

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_13,

## 13.1 Introduction

A distance geometry problem is generally stated as to find the coordinates for a set of points, given the distances between certain pairs of points. The problem has an important application in molecular modeling, where the coordinates of the atoms in a given molecule need to be determined based on a given set of pairwise distances between atoms.

In recent years there has been a very active research in the molecular optimization, especially in the protein folding framework which is one of the most important problems in biophysical chemistry. Molecular optimization problems arise also in the study of clusters (molecular cluster problems) and of large, confined ionic systems in plasma physics [32]. The determination of a molecular conformation can be tackled by either minimizing a potential energy function (if the molecular structure corresponds to the global minimizer of this function) or solving the distance geometry problem [3, 12] (when the molecular conformation is determined by distances between pairs of atoms in the molecule). Both methods are concerned with global optimization problems.

The *exact distance geometry problem* consists in finding positions $x^1,\dots,x^n$ of $n$ points in $\mathbb{R}^p$ such that

$$\|x^i - x^j\| = \delta_{ij}, (i,j) \in \mathscr{S}, \tag{13.1}$$

where $\mathscr{S}$ is a subset of the point pairs, $\delta_{ij}$ with $(i,j) \in \mathscr{S}$ is the given distance between atoms $i$ and $j$, and $\|\cdot\|$ denotes the Euclidean norm. Usually, a small subset of pairwise distances is known, i.e., $\mathscr{S}$ is small, and in practice, lower and upper bounds of the distances are given instead of the exact values. In the molecular conformation, the problem is considered in three-dimensional Euclidean space, say $p = 3$ (each point is an atom). In general, it can be defined in arbitrary dimensions.

It should be noted that, by the error in the theoretical or experimental data, there may not exist a solution to this problem, then an $\varepsilon$-optimal solution of Eq. (13.1), namely a configuration $x^1,\dots,x^n$ satisfying

$$|\ \|x^i - x^j\| - \delta_{ij}\ | \leq \varepsilon, (i,j) \in \mathscr{S}, \tag{13.2}$$

is useful. When only the lower and upper bounds of $\delta_{ij}$ are given, we are faced with the so-called *general distance geometry problem* which consists of finding a set of positions $x^1,\dots,x^n$ in $\mathbb{R}^p$ such that

$$l_{ij} \leq \|x^i - x^j\| \leq u_{ij}, (i,j) \in \mathscr{S}, \tag{13.3}$$

where $l_{ij}$ and $u_{ij}$ are lower and upper bounds of the distance constraints, respectively.

In Euclidean distance geometry problems, we must take into consideration the symmetry of the subset $\mathscr{S}$ (i.e., $(i,j) \in \mathscr{S}$ implies $(j,i) \in \mathscr{S}$). For simplifying the presentation, let $\mathscr{S}^w$ be the set defined by

$$\mathscr{S}^w := \{(i,j) \in \mathscr{S} : i < j\}.$$

In the sequel $\mathscr{M}_{n,p}(\mathrm{I\!R})$ denotes the space of real matrices of order $n \times p$ and for $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$, $X_i$ (resp. $X^i$) is its $i$th row (resp. $i$th column). By identifying a set of positions $x^1, \ldots, x^n$ with the matrix $X$ (i.e., $X_i = x^i$ for $i = 1, \ldots, n$), we can advantageously express the exact and/or general distance geometry problems in the matrix space $\mathscr{M}_{n,p}(\mathrm{I\!R})$:

$$\text{(EDP)} \qquad 0 = \min \left\{ \sigma(X) := \frac{1}{2} \sum_{(i,j) \in \mathscr{S}^w} w_{ij} \theta_{ij}(X) : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) \right\},$$

where $w_{ij} > 0$ for $i \neq j$ and $w_{ii} = 0$ for all $i$. The pairwise potential $\theta_{ij} : \mathscr{M}_{n,p}(\mathrm{I\!R}) \longrightarrow \mathrm{I\!R}$ is defined for problem (13.1) by either

$$\theta_{ij}(X) = \left(\delta_{i,j}^2 - \|X_i^T - X_j^T\|^2\right)^2 \tag{13.4}$$

or

$$\theta_{ij}(X) = \left(\delta_{ij} - \|X_i^T - X_j^T\|\right)^2, \tag{13.5}$$

and for problem (13.3) by

$$\theta_{ij}(X) = \min{}^2 \left\{ \left(\|X_i^T - X_j^T\|^2 - l_{ij}^2\right)/l_{ij}^2, 0 \right\} + \max{}^2 \left\{ \left(\|X_i^T - X_j^T\|^2 - u_{ij}^2\right)/u_{ij}^2, 0 \right\}. \tag{13.6}$$

When all pairwise distances are available and a solution exists, the exact distance geometry problem (13.1) can be solved by a polynomial time algorithm (Blumenthal [2], Crippen and Havel [3], Dong and Wu [9]). However, in practice, one knows only a subset of the distances, and it is well known (Saxe [45]) that $p$-dimensional distance geometry problems are strongly NP-complete with $p = 1$ and strongly NP-hard for all $p > 1$. The visible sources of difficulties of these problems are (1) the question of the existence of a solution, (2) the nonuniqueness of solutions, (3) the presence of a large number of local minimizers, and (4) the large scale of problems that arise in practice.

Several methods have been proposed for solving the distance geometry problems (13.1) and/or (13.3). De Leeuw [6,7] proposed the well-known majorization method for solving the Euclidean metric multidimensional scaling problem (MDS) which includes (EDP) with $\theta_{ij}$ given by Eq. (13.5). Cripen and Havel [3] used the function $\theta_{ij}$ defined in Eq. (13.6) for solving Problem (13.3) by the EMBED algorithm. Their method consists of solving a sequence of exact distance geometry problems where all pairwise distances are included. It relies on the SVD or alternative

Cholesky decomposition with diagonal pivoting. Current implementations of the EMBED algorithm use a local minimizer of the problem (EDP)-Eq. (13.4) as a starting point for a simulated annealing. Glunt, Hayden,and Raydan [11] studied a special gradient method for determining a local minimizer of Problem (13.1) with $\theta_{ij}$ defined in Eq. (13.5). From a graph-theoretic viewpoint Hendrickson [14] developed an algorithm to solve Problem (13.1) where $\theta_{ij}$ is given by Eq. (13.4). His method works well for his test problems where a protein contains at most 124 amino acids (at most 777 atoms). The protein actually has 1,849 atoms, but some simple structure exploitation allowed the author to start the numerical method with only 777 atoms. With a smoothing technique and a continuation approach based on the Gaussian transform of the objective function and on the trust region method, Moré and Wu [31] proposed an algorithm for solving Problem (13.1) with $\theta_{ij}$ defined by Eq. (13.4). By the Gaussian transform, the original function becomes a smoother function with fewer local minimizer. Computational experiments with up to 648 variables ($n = 216$) in [31] proved that the continuation method is more reliable and efficient than the multistart approach, a standard procedure for finding the global minimizer to (EDP). Also by the Gaussian transform, Moré and Wu [33] considered the general distance geometry problem with the function $\theta_{ij}$ defined by Eq. (13.6). Another smoothing technique and continuation approach is developed in [46]. A stochastic/perturbation algorithm was proposed by Zou, Bird and Schnabel [48] for both general and exact distance geometry problems. This is a combination of a stochastic phase that identifies an initial set of local minimizers and a more deterministic phase that moves from a low to an even lower local minimizer. The numerical experiments presented there (with the same data as in Moré and Wu [31] and Hendrickson [14]) showed that this approach is promising. It is worth noting that (EDP) is intimately related to the Euclidean distance matrix completion problem [1, 18]. This problem has been formulated as a semidefinite programming problem in [1, 8, 17]. Other work includes the $\alpha BB$ algorithm by Floudas [10], the branch-and-bound algorithm by Pardalos [16], and the branch-and-prune algorithm by Liberti el al. [28].

In the convex analysis approach to nondifferentiable nonconvex programming, the DC (difference of convex functions) optimization and its solution algorithms (DCA) were developed by Pham Dinh Tao and Le Thi Hoai An ([19, 21, 38–40] and references therein). These tools constitute a natural and logical extension of Pham Dinh Tao's earlier works concerning convex maximization and its subgradient algorithms ([34–37] and references therein). The majorization method proposed by De Leeuw [6, 7] is a suitable adaptation of the above subgradient methods for maximizing a seminorm over the unit ball of another one. Our method in this chapter, based on the DC optimization approach, aims at solving the exact and the general geometry distance problems (13.1) and (13.3) with different standard and nonstandard formulations as smooth/nonsmooth DC programs.

The purpose of this chapter is to demonstrate that the DCA can be suitably adapted to devise efficient algorithms for solving large-scale distance geometry problems. We propose various versions of DCA based on different formulations for these problems. The DCA is a primal-dual subgradient method for solving a general

DC program that consists in the minimization of the difference of convex functions on the whole space. Featured as a descent method without linesearch, it is at present one of a few algorithms in the local approach which has been successfully applied to many large-scale DC optimization problems and proved to be more robust and efficient than related standard methods (see, e.g., [19–26] and the list of references in [4]). One of the key features of the DCA is the effect of DC decomposition. For finding good DC decompositions we propose several equivalent DC formulations based on the stability of Lagrangian duality and the regularization techniques. The norms $l_1$ and $l_2$ as well as the nonstandard $l_1 - l_\infty$ norm have served to model these nonconvex programs. On the other hand, due to its local character, DCA cannot guarantee the globality of computed solutions for general DC programs. However, we observe that with a suitable starting point, it converges quite often to a global one (see, e.g., [19,21,41]). This property motivates us to investigate different techniques for computing a "good" starting point for the DCA: a two phases algorithm using shortest paths between all pairs of atoms to generate the complete dissimilarity matrix, a spanning trees procedure, and a smoothing technique are investigated.

Our interest in the DC programming and DCA has increased recently, motivated by its success to a great deal of various large-scale DC programs [19,20,40–42]. The positive aspects of the DCA that come out of numerical solutions of these problems are:

- It often converges to a global solution.
- The number of concave variables, say the number of variables associated to the concave term of the objective function, does not affect the complexity for the algorithm.
- It can be used for large-scale problems at little cost.
- It is applicable to nonsmoothness.

Our work relies on the DC programming and its main algorithmic tool, the DCA. A short description of the background indispensable for understanding this approach is described in Sect. 13.2. In Sects. 13.3 and 13.4 we show, respectively, how to use the generic DCA scheme to solve the exact distance geometry problem (13.1) and the general distance geometry problem (13.3). Section 13.5 is devoted to the description of several techniques for finding a good starting point of DCA. Finally, numerical simulations reported in Sect. 13.6 demonstrate the practical usefulness of the nonstandard reformulations, the globality of sought solutions, and the efficiency and the reliability of our algorithms.

## 13.2 DC Programming and DCA

In this section we summarize the material needed for an easy understanding of DC programming and DCA which will be used to solve the distance geometry problems. We introduce the notions of convex conjugate function, subdifferential and dual DC program, and present the optimality conditions for the primal and dual programs.

We then describe the DCA and discuss its convergence properties. The presentation follows the seminal works in [40, 41] with a slightly different organization.

We are working with the space $X = \mathbb{R}^n$ which is equipped with the canonical inner product $\langle \cdot, \cdot \rangle$ and the corresponding Euclidean norm $\| \cdot \|$; thus, the dual space $Y$ of $X$ can be identified with $X$ itself. We follow [15, 44] for definitions of usual tools in modern convex analysis where functions could take the infinite values $\pm\infty$. A function $\theta : X \to \mathbb{R} \cup \{\pm\infty\}$ is said to be proper if it takes nowhere the value $-\infty$ and is not identically equal to $+\infty$. The effective domain of $\theta$, denoted by dom $\theta$, is

$$\text{dom}\ \theta = \{x \in X : \theta(x) < +\infty\}.$$

The set of all lower semicontinuous proper convex functions on $X$ is denoted by $\Gamma_0(X)$. Let $\theta \in \Gamma_0(X)$, then the conjugate function of $\theta$, denoted $\theta^*$, is defined by

$$\theta^*(y) = \sup\{\langle x, y\rangle - \theta(x) : \ x \in X\}.$$

We have $\theta^* \in \Gamma_0(Y)$ and $\theta^{**} = \theta$.
For $\theta \in \Gamma_0(X)$ and $x_0 \in$ dom $\theta$, $\partial\theta(x_0)$ denotes the subdifferential of $\theta$ at $x_0$, i.e.,

$$\partial\theta(x_0) := \{y \in Y : \theta(x) \geq \theta(x_0) + \langle x - x_0, y\rangle, \ \forall x \in X\}. \tag{13.7}$$

Each $y \in \partial\theta(x_0)$ is called a subgradient of $\theta$ at $x_0$. The subdifferential $\partial\theta(x_0)$ is a closed convex set in $Y$. It generalizes the derivative in the sense that $\theta$ is differentiable at $x_0$ if and only if $\partial\theta(x_0)$ is a singleton which is exactly $\{\nabla\theta(x_0)\}$. Recall the well-known property related to subdifferential calculus of $\theta \in \Gamma_0(X)$:

$$y \in \partial\theta(x) \Leftrightarrow x \in \partial\theta^*(y) \Leftrightarrow \langle x, y\rangle = \theta(x) + \theta^*(y). \tag{13.8}$$

The domain of $\partial\theta$, denoted dom $\partial\theta$, is defined by: dom $\partial\theta := \{x \in$ dom $\theta$ : $\partial\theta(x) \neq \emptyset\}$. There holds *ri*(dom $\theta$) $\subset$ dom $\partial\theta \subset$ dom $\theta$, where *ri C* is the relative interior of the convex set $C$.

A function $\theta \in \Gamma_0(X)$ is said to be polyhedral convex if

$$\theta(x) = \max\{\langle a_i, x\rangle - \beta_i : i = 1, \ldots, m\} + \chi_C(x) \ \forall x \in X,$$

where $C$ is a nonempty polyhedral convex set in $X$ and $\chi_C$ is the indicator function of $C$, i.e., $\chi_C(x) := 0$ if $x \in C, +\infty$ otherwise.

A general DC program is of the form

$$(P_{dc}) \ \ \alpha = \inf\{f(x) := g(x) - h(x) : \ x \in X\}, \tag{13.9}$$

with $g, h \in \Gamma_0(X)$. Such a function $f$ is called a DC function, and $g - h$, a DC decomposition of $f$, while the convex functions $g$ and $h$ are DC components of $f$. In DC programming [40, 41], the convention

$$(+\infty) - (+\infty) := +\infty \tag{13.10}$$

has been adopted to avoid the ambiguity on the determination of $(+\infty)-(+\infty)$. Such a case does not present any interest and can be discarded. In fact, we are actually concerned with the following problem:

$$\alpha = \inf\{f(x) := g(x) - h(x) \ : \ x \in \mathrm{dom}\, h\},$$

which is equivalent to $(P_{dc})$ under the convention (13.10).

It should be noted that a convex constrained DC program

$$\alpha = \inf\{\varphi(x) - \psi(x) \ : \ x \in C\}$$

is equivalent to the unconstrained DC program by adding the indicator function $\chi_C$ of $C$ to the first DC component $\varphi$ :

$$\alpha = \inf\{g(x) - h(x) \ : \ x \in X\}, \qquad \text{where } g := \varphi + \chi_C \text{ and } h := \psi.$$

Based on the conjugate functions, the dual program of $(P_{dc})$ is defined as

$$(D_{dc}) \quad \alpha_D = \inf\{h^*(y) - g^*(y) : y \in Y\}. \tag{13.11}$$

One can prove that $\alpha = \alpha_D$ (see, e.g., [40]), and there is the perfect symmetry between primal and dual DC programs: the dual to $(D_{dc})$ is exactly $(P_{dc})$.

Note that the finiteness of $\alpha$ merely implies that dom $g \subset$ dom $h$ and dom $h^* \subset$ dom $g^*$. Such inclusions will be assumed throughout this chapter.

If $g$ or $h$ are polyhedral convex functions, then $(P_{dc})$ is called a polyhedral DC program, which plays a main role in nonconvex programming (see [21, 26, 40–42] and references therein), and enjoys interesting properties concerning the local optimality and the convergence of the DCA.

DC programming investigates the structure of the vector space $DC(\mathrm{I\!R}^n) := \Gamma_0(\mathrm{I\!R}^n) - \Gamma_0(\mathrm{I\!R}^n)$, DC duality, and optimality conditions for DC programs. The complexity of DC programs resides, of course, in the lack of practical optimal globality conditions. We developed instead the following necessary local optimality conditions for DC programs in their primal part, by symmetry their dual part is trivial (see [21,26,40–42]):

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset \tag{13.12}$$

(such a point $x^*$ is called *critical point* of $g-h$ or generalized KKT point for $(P_{dc})$), and

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*). \tag{13.13}$$

The condition (13.13) is also sufficient (for local optimality) in many important classes of DC programs (see [21,40,41]).

The transportation of global solutions between $(P_{dc})$ and $(D_{dc})$ is expressed by

$$[\bigcup_{y^*\in\mathscr{D}} \partial g^*(y^*)] \subseteq \mathscr{P}\ ,\ [\bigcup_{x^*\in\mathscr{P}} \partial h(x^*)] \subseteq \mathscr{D}, \tag{13.14}$$

where $\mathscr{P}$ and $\mathscr{D}$ denote the solution sets of $(P_{dc})$ and $(D_{dc})$, respectively. Moreover, equality holds in the first inclusion of Eq. (13.14) if $\mathscr{P} \subseteq$ dom $\partial h$, in particular if $\mathscr{P} \subseteq$ ri(dom $h$). Similar property relative to the second inclusion can be stated by duality. On the other hand, under technical conditions, this transportation holds also for local solutions of $(P_{dc})$ and $(D_{dc})$ (see [26, 40] and references therein).

Based on local optimality conditions and duality in DC programming, the DCA consists in constructing of two sequences $\{x^k\}$ and $\{y^k\}$ of trial solutions of the primal and dual programs, respectively, such that the sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing, and $\{x^k\}$ (resp. $\{y^k\}$) converges to a primal feasible solution $x^*$ (resp. a dual feasible solution $y^*$) satisfying local optimality conditions and

$$x^* \in \partial g^*(y^*), \quad y^* \in \partial h(x^*). \tag{13.15}$$

It implies, according to Eq. (13.8), that $x^*$and $y^*$ are critical points of $g-h$ and $h^*-g^*$, respectively.

The sequences $\{x^k\}$ and $\{y^k\}$ are determined in the way that $x^{k+1}$ (resp. $y^{k+1}$) is a solution to the convex program $(P_k)$ (resp. $(D_{k+1})$) defined by ($x^0 \in$ dom $\partial h$ being a given initial point and $y^0 \in \partial h(x^0)$ being chosen)

$$\begin{aligned} (P_k) \quad & \inf\{g(x) - [h(x^k) + \langle x - x^k, y^k\rangle] : x \in \mathrm{I\!R}^n\},\\ (D_{k+1}) \quad & \inf\{h^*(y) - [g^*(y^k) + \langle y - y^k, x^{k+1}\rangle] : y \in \mathrm{I\!R}^n\}. \end{aligned}$$

The DCA has the quite simple interpretation: at the $k$th iteration, one replaces in the primal DC program $(P_{dc})$ the second component $h$ by its affine minorization $h^{(k)}(x) := h(x^k) + \langle x - x^k, y^k\rangle$ defined by a subgradient $y^k$ of $h$ at $x^k$ to give birth to the primal convex program $(P_k)$, the solution of which is nothing but $\partial g^*(y^k)$. Dually, a solution $x^{k+1}$ of $(P_k)$ is then used to define the dual convex program $(D_{k+1})$ obtained from $(D_{dc})$ by replacing the second DC component $g^*$ with its affine minorization $(g^*)^{(k)}(y) := g^*(y^k) + \langle y - y^k, x^{k+1}\rangle$ defined by the subgradient $x^{k+1}$ of $g^*$ at $y^k$ : the solution set of $(D_{k+1})$is exactly $\partial h(x^{k+1})$. The process is repeated until convergence. DCA performs a double linearization with the help of the subgradients of $h$ and $g^*$ and the DCA then yields the next scheme (starting from given $x^0 \in$ dom $\partial h$):

$$y^k \in \partial h(x^k); \quad x^{k+1} \in \partial g^*(y^k),\ \forall k \geq 0. \tag{13.16}$$

DCA's distinctive feature relies upon the fact that DCA deals with the convex DC components $g$ and $h$ but not with the DC function $f$ itself. Moreover, a DC function $f$ *has infinitely many DC decompositions which have crucial implications for the qualities* (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA. For a given DC program, the choice of *optimal* DC decompositions is still open. Of course, this depends strongly on the very specific structure of the problem being considered. In order to tackle the large-scale setting, one tries in practice to choose $g$ and $h$ such that sequences $\{x^k\}$ and $\{y^k\}$ can be easily calculated, i.e., either they are in an explicit form or their computations are inexpensive.

**DCA's Convergence Properties**

We mention now the main convergence properties of DCA. First let us define the *modulus of strong convexity* of a function $\theta$ on a convex set $C$, denoted by $\rho(\theta,C)$:

$$\rho(\theta,C) = \sup\{\rho \geq 0 : \theta - (\rho/2)\|\cdot\|^2 \text{ is convex on } C\}. \tag{13.17}$$

Clearly, $\theta$ is convex on $C$ if and only if $\rho(\theta,C) \geq 0$. One says that $\theta$ is *strongly convex* on $C$ if $\rho(\theta,C) > 0$.

Let $C$ (resp. $D$) a convex set containing the sequence $\{x^k\}$ (resp. $\{y^k\}$). We have the following: DCA is a descent method without linesearch which enjoys the following properties:

(i) The sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing and

- $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$ iff $y^k \in \partial g(x^k) \cap \partial h(x^k), y^k \in \partial g(x^{k+1}) \cap \partial h(x^{k+1})$ and $[\rho(g,C) + \rho(h,C)]\|x^{k+1} - x^k\| = 0$. Moreover, if $g$ or $h$ are strictly convex on $C$, then $x^k = x^{k+1}$.

  In such a case DCA terminates at the $k$th iteration (finite convergence of DCA).
- $h^*(y^{k+1}) - g^*(y^{k+1}) = h^*(y^k) - g^*(y^k)$ iff $x^{k+1} \in \partial g^*(y^k) \cap \partial h^*(y^k), x^{k+1} \in \partial g^*(y^{k+1}) \cap \partial h^*(y^{k+1})$ and $[\rho(g^*,D) + \rho(h^*,D)]\|y^{k+1} - y^k\| = 0$. Moreover, if $g^*$ or $h^*$ is strictly convex on $D$, then $y^{k+1} = y^k$.

  In such a case DCA terminates at the $k$th iteration (finite convergence of DCA).

(ii) If $\rho(g,C) + \rho(h,C) > 0$ (resp. $\rho(g^*,D) + \rho(h^*,D) > 0$)) then the series $\{\|x^{k+1} - x^k\|^2$ (resp. $\{\|y^{k+1} - y^k\|^2\}$ converges.

(iii) If the optimal value $\alpha$ of problem $(P_{dc})$ is finite and the infinite sequences $\{x^k\}$ and $\{y^k\}$ are bounded then every limit point $x^*$(resp. $y^*$) of the sequence $\{x^k\}$ (resp. $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).

(iv) DCA has a linear convergence for general DC programs.

(v) DCA has a finite convergence for polyhedral DC programs.

For DC programming with subanalytic data, rate convergence of DCA is established in [27].

At last, it is worth pointing out that, with suitable DC decompositions, DCA generates most standard algorithms in convex and nonconvex programming. For a complete study of DC programming and DCA the reader is referred to [21, 26, 40–42] and references therein.

*Remark 13.1.* In general, the qualities (cost, robustness, stability, rate of convergence, and globality of sought solutions) of the DCA depend upon the DC decomposition of the function $f$. Assertion (ii) shows how the strong convexity of DC components in primal and dual problems can influence on the DCA. To make the DC components (of the primal objective function $f = g - h$) strongly convex, we usually apply the following decomposition *(proximal regularization in DC programming)*:

$$f = g - h = \left(g + \frac{\lambda}{2}\|\cdot\|^2\right) - \left(h + \frac{\lambda}{2}\|\cdot\|^2\right). \tag{13.18}$$

In this case the DC components in the dual problem will be differentiable. It is worth mentioning, for instance, that by using conjointly suitable DC decompositions of convex functions and proximal regularization techniques [29, 30, 44] we can obtain the proximal point algorithm and the Goldstein–Levitin–Polyak subgradient method (in convex programming) as special cases of DCA. For a detailed study of regularization techniques in DC programming, see [19, 26, 38, 40]. Since there are as many DCA as there DC decompositions are, it is of particular interest to study various equivalent DC forms for the primal and dual DC programs.

The choice of the DC decomposition of the objective function in a DC program and the initial point for DCA are open questions to be studied. Of course, this depends strongly on the very specific structure of the problem being considered. In practice, for solving a given DC program, we try to choose g and h such that sequences $\{x^k\}$ and $\{y^k\}$ can be easily calculated, i.e., either they are in explicit form or their computations are inexpensive. On the other hand, for a method based on local optimality conditions like DCA, it is crucial to highlight different equivalent reformulations of the DC program which have no the same local optimality because they may serve to restart DCA (escaping local solutions procedure).

The major difficulty in nonconvex programming resides in the fact that there is, in general, no verifiable global optimality conditions. Thus, checking globality of solutions computed by local algorithms is very difficult. It is only possible for the cases where optimal values are known a priori (*e.g., they are zero in exact distance geometry problems*) or by comparison with global optimization techniques which, unfortunately, are too expensive to handle large-scale problems. A pertinent comparison of local algorithms should be based on the following aspects: (1) mathematical foundations of the algorithms; (2) rate of convergence and running time; (3) ability to treat large-scale problems; (4) quality of computed solutions: the lower the corresponding value of the objective is, the better the local algorithm will be; and (5) the degree of dependence on initial points: the larger the set composed of starting points which ensure convergence of the algorithm to a global solution is, the better the algorithm will be.

DCA seems to meet these features since it was successfully applied to a lot of different and various nonconvex optimization problems to which it gave global solutions and proved to be more robust and more efficient than related standard methods, especially in the large-scale setting ([19, 20, 40–42] and references therein).

We shall apply *all these DC enhancement features* to solve distance geometry problems which are formulated as DC programs.

## 13.3 Solving the Exact Distance Geometry Problem by DCA

This section is devoted to the formulation of the exact distance geometry problem (13.1) in terms of DC programs and the computation of the sequences $\{X^k\}$ and $\{Y^k\}$ generated by DCA for solving them. It will be proved that both problems (EDP) with $\theta_{ij}$ defined by Eq. (13.4) or (13.5) are DC programs; moreover, the objective function of the first (the usual formulation of problem (13.1) as a global optimization problem) is infinitely differentiable, while the latter is a nondifferentiable nonconvex optimization problem. Paradoxically, the second formulation is advantageous in using DCA for solving the exact distance geometry problem since the sequences $\{X^k\}$ and $\{Y^k\}$ have explicit forms. On the other hand, the zero gap of the Lagrangian duality relative to a special convex maximization problem allows stating interesting equivalent DC programs of the exact distance geometry problem.

Throughout this chapter we work on an appropriate matrix framework. By identifying an $n \times p$ matrix $X$ with a $p \times n-$ vector, in what follows, we use either $\mathscr{M}_{n,p}(\mathrm{I\!R})$ or $\mathrm{I\!R}^{p\times n}$ for indicating the same notation. We can identify by rows (resp. columns) each matrix $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$ with a row-vector (resp. column-vector) in $(\mathrm{I\!R}^p)^n$ (resp. $(\mathrm{I\!R}^n)^p$) by writing, respectively,

$$X \longleftrightarrow \mathscr{X} = (X_1,\ldots,X_n), X_i^T \in \mathrm{I\!R}^p, \mathscr{X}^T \in (\mathrm{I\!R}^p)^n, \tag{13.19}$$

and

$$X \longleftrightarrow \overline{\mathscr{X}} = (X^1,\ldots,X^p)^T, \quad X^i \in \mathrm{I\!R}^n, \overline{\mathscr{X}} \in (\mathrm{I\!R}^n)^p. \tag{13.20}$$

The inner product in $\mathscr{M}_{n,p}(\mathrm{I\!R})$ is defined as the inner product in $(\mathrm{I\!R}^p)^n$ or $(\mathrm{I\!R}^n)^p$. That is,

$$\begin{aligned}\langle X,Y\rangle_{\mathscr{M}_{n,p}(\mathrm{I\!R})} &= \langle \mathscr{X}^T,\mathscr{Y}^T\rangle_{(\mathrm{I\!R}^p)^n} = \sum_{i=1}^{n}\langle X_i^T,Y_i^T\rangle_{\mathrm{I\!R}^p} = \sum_{i=1}^{n} X_iY_i^T \\ &= \langle\overline{\mathscr{X}},\overline{\mathscr{Y}}\rangle_{(\mathrm{I\!R}^n)^p} = \sum_{k=1}^{p}\langle X^k,Y^k\rangle_{\mathrm{I\!R}^n} = \sum_{k=1}^{p}(X^k)^TY^k = Tr(X^TY).\end{aligned}$$

Here $Tr(X^TY)$ denotes the trace of the matrix $X^TY$. In the sequel, for simplicity, we shall suppress, if no possible ambiguity, the indices for the inner product and denote by $\|\cdot\|$ the corresponding Euclidean norm on $\mathscr{M}_{n,p}(\mathrm{I\!R})$. Evidently, we must choose either representation in a convenient way.

The data of the distance geometry problems can be succinctly represented by a graph $G(N,\mathscr{S})$. The vertices $N = \{1,\ldots,n\}$ correspond to the atoms and an edge $(i,j) \in \mathscr{S}$ connects vertices $i$ and $j$ if the distance $\delta_{ij}$ between the corresponding atoms is known. The weight matrix $W = (w_{ij})$ of (EDP$_1$) is defined by taking $w_{ij} = 0$ when $(i,j) \notin \mathscr{S}$. Throughout this chapter, we assume that $W$ is irreducible, i.e., the graph $G(N,\mathscr{S})$ is connected. This assumption is not restrictive for problem (EDP$_1$) since it can be decomposed into a number of smaller problems otherwise. Then we work under the next assumptions for the two symmetric matrices $\Delta = (\delta_{ij})$ (the distance matrix) and $W = (w_{ij})$:

(a1) For $i \neq j$, $\delta_{ij} > 0$ when $(i,j) \in \mathscr{S}$ (i.e., two different atoms are not in the same position), and $w_{ii} = 0$ for all $i$.
(a2) For $i \neq j$, $w_{ij} = 0$ if and only if $\delta_{ij}$ is unknown, say $(i,j) \notin \mathscr{S}$.
(a3) The weight matrix $W$ is irreducible.

Remark that if we set $\delta_{ij} = 0$ for $(i,j) \notin \mathscr{S}$, then $G(N,\mathscr{S})$ is the graph associated with the distance matrix $\Delta$ too.

The case where $w_{ij} = c$ ($c$ is a given positive number) for all $i \neq j$ is called *the normal case*. Clearly, this case can occur if and only if the distance matrix $\Delta$ is completely defined, say all pairwise distances are known.

### *13.3.1 The $l_1$-Norm Approach for Solving Problem (EDP)*

The first equivalent nonconvex optimization problem of the exact distance geometry problem (13.1) is

$$(\text{EDP}_1)\quad 0 = \min\left\{\sigma(X) := \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left(\|X_i^T - X_j^T\| - \delta_{ij}\right)^2 : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\}.$$

The objective function of (EDP$_1$) can be written as

$$\sigma(X) = \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}d_{ij}^2(X) - \sum_{(i,j)\in\mathscr{S}^w} w_{ij}\delta_{ij}d_{ij}(X) + \frac{1}{2}\eta_\delta^2, \tag{13.21}$$

with $d_{ij}(X) = \|X_i^T - X_j^T\|$ and $\eta_\delta := \left[\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\delta_{ij}^2\right]^{\frac{1}{2}}$.

#### 13.3.1.1 The Primal Formulation of (EDP$_1$)

Under assumption (a2), although $\delta_{ij}$ is unknown for any $(i,j) \notin \mathscr{S}$, in Eq. (13.21) the summation over pairs $(i,j) \in \mathscr{S}$ can be extended to that overall pairs $(i,j)$. This fact must be taken into account later while computing sequences of iterations in DCA. Then (EDP$_1$) is equivalent to the following problem:

$$-\frac{1}{2}\eta_\delta^2 = \min\left\{F_1(X) := \frac{1}{2}\eta^2(X) - \xi(X) : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\}, \tag{13.22}$$

where $\eta$ and $\xi$ are the functions defined on $\mathscr{M}_{n,p}(\mathrm{IR})$ by

$$\eta(X) = \left[\sum_{i<j} w_{ij} d_{ij}^2(X)\right]^{1/2} \quad \text{and} \quad \xi(X) = \sum_{i<j} w_{ij}\delta_{ij} d_{ij}(X). \tag{13.23}$$

It is not difficult to verify that $\eta$ and $\xi$ are two seminorms in $\mathscr{M}_{n,p}(\mathrm{IR})$, and thus Problem (13.22) is a DC program to which the DCA can be applied.

Under assumptions (a2) and (a3), we can restrict the working matrix space to an appropriate set which is, as will be seen later, favorable to our calculations. Indeed, let $\mathscr{A}$ denote the set of matrices in $\mathscr{M}_{n,p}(\mathrm{IR})$ whose rows are all identical, i.e.,

$$\mathscr{A} := \{X \in \mathscr{M}_{n,p}(\mathrm{IR}) : X_1 = \cdots = X_n\},$$

and let $Proj_{\mathscr{A}}$ be the orthogonal projection on $\mathscr{A}$, we have the following result.

**Lemma 13.1.** *(i)* $\mathscr{A} = \{ev^T : \quad v \in \mathrm{IR}^p\}$ *is a p-dimensional subspace of* $\mathscr{M}_{n,p}(\mathrm{IR})$ *whose orthogonal subspace is given by* $\mathscr{A}^\perp = \{Y \in \mathscr{M}_{n,p}(\mathrm{IR}) : \sum_{i=1}^n Y_i = 0\}$.

*(ii)* $\mathscr{A} \subset \xi^{-1}(0); \quad \mathscr{A} \subset \eta^{-1}(0)$.

*(iii)* $Proj_{\mathscr{A}} = (1/n)ee^T; \quad Proj_{\mathscr{A}^\perp} = I - (1/n)ee^T$ *(e is the vector of ones in* $\mathrm{IR}^n$*).*

*(iv) If the weight matrix W is irreducible (resp. W is irreducible and* $w_{ij}\delta_{ij} > 0$ *whenever* $w_{ij} > 0$*), then* $\mathscr{A} = \eta^{-1}(0)$ *(resp.* $\mathscr{A} = \xi^{-1}(0)$*). If* $\mathscr{A} = \eta^{-1}(0) = \xi^{-1}(0)$*, then solving the problem (13.22) leads to solving*

$$-\frac{\eta_\delta^2}{2} = \min\left\{\frac{1}{2}\eta^2(X) - \xi(X) : \quad X \in \mathscr{A}^\perp\right\}, \tag{13.24}$$

*and* $X^*$ *is an optimal solution of Problem (13.24) if and only if* $X^* + Z$ *is an optimal solution of Problem (13.22) for all* $Z \in \mathscr{A}$.

*Proof.* (i) and (ii) are straightforward from the definition of $\mathscr{A}$. The proof of (iii) is easy. To prove (iv) let $X \in \mathscr{M}_{n,p}(\mathrm{IR})$ such that $\eta(X) = 0$ (or $\xi(X) = 0$) and $(i,j) \in \{1,\dots,n\}^2$ with $i \neq j$. Since the matrix $W$ is irreducible, there is a finite sequence $\{i_1,\dots,i_r\} \subset \{1,\dots,n\}$ verifying $w_{ii_1} > 0, w_{i_k i_{k+1}} > 0$ for $k = 1,\dots,r-1$, and $w_{i_r j} > 0$. It follows that $X_i = X_{i_1} = \cdots = X_{i_r} = X_j$ and then $\eta^{-1}(0) = \mathscr{A} = \xi^{-1}(0)$. The proof is thus completed. □

*Remark 13.2.* As a consequence of Lemma 13.1, the restrictions of the seminorms $\eta$ and $\xi$ on the subspace $\mathscr{A}^\perp$ are actually norms under the assumptions (a1), (a2), and (a3). It follows that their polars $\eta^0$ and $\xi^0$ defined by [44]

$$\eta^0(Y) = \sup\{\langle X,Y\rangle : \eta(X) \le 1\}, \forall Y \in \mathscr{M}_{n,p}(\mathrm{IR}),$$

$$\xi^0(Y) = \sup\{\langle X,Y\rangle : \eta(X) \le 1\}, \forall Y \in \mathscr{M}_{n,p}(\mathrm{IR})$$

satisfy the following properties:

(i) $\eta^0(Y) = \xi^0(Y) = +\infty$ if $Y \notin \mathscr{A}^\perp$.
(ii) $\eta^0(Y) = \sup\{\langle X, Y\rangle : X \in \mathscr{A}^\perp, \eta(X) \leq 1\}, \forall Y \in \mathscr{A}^\perp$.
$\xi^0(Y) = \sup\{\langle X, Y\rangle : X \in \mathscr{A}^\perp, \eta(X) \leq 1\}, \forall Y \in \mathscr{A}^\perp$.

We shall now compute subdifferentials of the functions $\xi$, $((1/2)\eta^2)^*$. These calculations will fortunately permit to state new matrix expressions of these functions and thus to provide the simplest computations of the sequences $\{X^{(k)}\}$ and $\{Y^{(k)}\}$ generated by DCA applied to problem (13.22). They also point out interesting relations between the trust region subproblem and problem (13.22).

### Computing $\partial\xi$

By definition $\xi(X) = \sum_{i<j} w_{ij}\delta_{ij}d_{ij}(X)$. Thus, $\partial\xi(X) = \sum_{i<j} w_{ij}\delta_{ij}\partial d_{ij}(X)$. Further, since $d_{ij}$ can be expressed as (using the row-representation of $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$)

$$d_{ij} = \|\cdot\| \circ L_{ij} : (\mathrm{I\!R}^p)^n \longrightarrow \mathrm{I\!R}^p \longrightarrow \mathrm{I\!R},$$
$$X \longmapsto L_{ij}(X) = X_i^T - X_j^T \longmapsto \|X_i^T - X_j^T\|,$$

it follows that [44] $\partial d_{ij}(X) = L_{ij}^T \partial\left(\|\cdot\|\right)(L_{ij}(X))$. Hence

$$Y(i,j) \in \partial d_{ij}(X) \Leftrightarrow Y(i,j) = L_{ij}^T y, \quad y \in \partial\left(\|\cdot\|\right)(X_i^T - X_j^T)$$

which implies

$$Y(i,j)_k = 0 \text{ if } k \notin \{i, j\} \text{ and } Y(i,j)_i^T = -Y(i,j)_j^T \in \partial\left(\|\cdot\|\right)(X_i^T - X_j^T). \quad (13.25)$$

Thus, $\xi$ is *not differentiable* on the closed set $\{X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) : X_i = X_j \text{ for } (i,j) \in \mathscr{S}^w\}$ but on its complement in $\mathscr{M}_{n,p}(\mathrm{I\!R})$, i.e., the open set $\Omega$ defined by

$$\Omega = \{X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) : \|X_i^T - X_j^T\| > 0, \forall (i,j) \in \mathscr{S}^w\}. \quad (13.26)$$

It is clear that $\Omega + \mathscr{A} = \Omega$. Now for $(i,j) \in \mathscr{S}^w$ let us choose the particular subgradient $Y(i,j) \in \partial d_{ij}(X)$ defined by

$$Y(i,j)_i = -Y(i,j)_j = \frac{X_i - X_j}{\|X_i^T - X_j^T\|} \text{if } X_i \neq X_j, 0 \text{ if } X_i = X_j. \quad (13.27)$$

In this case, the resulting subgradient $Y \in \partial\xi(X)$ is explicitly given by

$$\begin{aligned} Y_k &= \sum_{i<j} w_{ij}\delta_{ij}Y(i,j)_k = \sum_{i<k} w_{ik}\delta_{ik}Y(i,k)_k + \sum_{j>k} w_{kj}\delta_{kj}Y(k,j)_k \\ &= \left[\sum_{i=1}^n w_{ki}\delta_{ki}s_{ki}(X)\right] X_k - \sum_{i=1}^n w_{ki}\delta_{ki}s_{ki}(X)X_i, \end{aligned}$$

where $s_{ij}(X) = 1/(\|X_i^T - X_j^T\|)$ if $X_i \neq X_j$, 0 otherwise.

Let $B(X) = (b_{ij}(X))$ be the $n \times n$ matrix defined by

$$b_{ij}(X) = -w_{ij}\delta_{ij}s_{ij}(X) \text{ if } i \neq j, -\sum_{k=1\, k\neq i}^{n} b_{ik}(X) \text{ if } i = j. \tag{13.28}$$

It follows that

$$Y = B(X)X,\ B(X+U) = B(X),\ \forall X \in \mathscr{M}_{n,p}(\mathrm{I\!R}), \forall U \in \mathscr{A}. \tag{13.29}$$

In the sequel for $i \neq j$, $M_{ij}$ denotes the $n \times n$ matrix given by

$$M_{ij} = e_i e_i^T + e_j e_j^T - (e_i e_j^T + e_j e_i^T) \tag{13.30}$$

where $\{e_i : i = 1,..,n\}$ forms the canonical basis of $\mathrm{I\!R}$. It is clear that the particular subgradient $Y(i,j) \in \partial d_{ij}(X)$ relates to $M_{ij}$ by

$$Y(i,j) = s_{ij}(X)M_{ij}X. \tag{13.31}$$

We will denote by $\mathscr{N}(M)$ and Im $M$ the null space and the range of the matrix $M$, respectively.

**Proposition 13.1.** *Let $B(X)$ be the matrix defined by Eq. (13.28). Then*

*(i)* $\mathscr{N}(B(X)) \supset \mathscr{A}$, *Im* $(B(X)) \subset \mathscr{A}^{\perp}$ *for all* $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$. *Moreover, the preceding inclusions become inequalities under the assumptions (a1), (a2), and (a3).*
*(ii)* $B(X)$ *depends only on* $X_i - X_j$ *for* $(i,j) \in \mathscr{S}$, $i < j$ *and* $B : \mathscr{M}_{n,p}(\mathrm{I\!R}) \longmapsto \Sigma_n^+$ *(the set of $n \times n$ symmetric positive semidefinite matrices) is continuous on* $\Omega$ *and* $B(X)X \in \partial\xi(X)$ *for all* $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$.
*(iii) The seminorm* $\xi$ *is differentiable (and so continuously differentiable) on* $\Omega$, *and* $\xi(X) = \langle X, B(X)X\rangle$ *for all* $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$.
*(iv)* $\langle X, B(Y)Y\rangle \leq \langle X, B(X)X\rangle$ *for all* $X, Y \in \mathscr{M}_{n,p}(\mathrm{I\!R})$.

*Proof.* (i) follows immediately from Lemma 13.1 and the fact that $\mathscr{A} = \{ev^T : v \in \mathrm{I\!R}^p\}$ and $B(X)e = 0$ for all $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$.
(ii) $B(X)$ is symmetric, diagonally dominant, and its diagonal entries are nonnegative. Thus it is positive semidefinite [47]. The continuity of the mapping $B$ on $\Omega$ directly follows from Eq. (13.28).
(iii) The differentiability of the seminorm $\xi$ is straightforward from Eq. (13.27) since the subdifferential $\partial\xi(X)$ is reduced to the singleton $\{B(X)X\}$ for $X \in \Omega$. The last inequality is in fact the generalized Euler relation for convex nondifferentiable functions which are positively homogeneous of degree 1 [44].
(iv) By the definition of the subdifferential, it follows from the assertion (ii) that

$$\xi(X) = \langle X, B(X)X\rangle \geq \xi(Y) + \langle X - Y, B(Y)Y\rangle, \quad \forall X, Y \in \mathscr{M}_{n,p}(\mathrm{I\!R}),$$

so the proof is completed since $\xi(Y) = \langle Y, B(Y)Y\rangle$. □

*Remark 13.3.* We have [44]

$$\partial\xi(X) = \{Y \in \mathscr{A}^{\perp} : \xi^0(Y) \leq 1,\ \langle X, Y\rangle = \xi(X)\},\ \forall X \in \mathscr{M}_{n,p}(\mathrm{I\!R}),$$

and the range of the subdifferential $\partial\xi$ then is bounded (Remark 13.2)

$$range\ \ \partial\xi = \{Y \in \mathscr{A}^{\perp} : \xi^0(Y) \leq 1\}.$$

### Computing $\partial\left((\mathbf{1/2})\eta^{\mathbf{2}}\right)^*$

First we state some fundamental properties of the function $(1/2)\eta^2$ which will be needed for the calculation of $\partial\left((1/2)\eta^2\right)^*$. From the definition of $\eta$, say

$$\eta^2(X) = \sum_{i<j} w_{ij}\|X_i^T - X_j^T\|^2 = \sum_{i<j} w_{ij} d_{ij}^2(X), \tag{13.32}$$

we have $\partial\left(\frac{1}{2}\eta^2\right)(X) = \sum_{i<j} w_{ij} d_{ij}(X)\partial d_{ij}(X)$. Thus

$$Y \in \partial\left(\frac{1}{2}\eta^2\right)(X) \Leftrightarrow Y = \sum_{i<j} w_{ij} d_{ij}(X) Y(i,j)$$

with $Y(i,j)$ being defined by Eq. (13.25). It follows that $\eta^2$ is differentiable on $\mathscr{M}_{n,p}(\mathrm{I\!R})$ and $Y = \nabla(\frac{1}{2}\eta^2)(X)$ is defined as

$$Y_k = \sum_{i<k} w_{ki}(X_k - X_i) + \sum_{j>k} w_{kj}(X_k - X_j) = \left(\sum_{i=1}^{n} w_{ki}\right) X_k - \sum_{i=1}^{n} w_{ki} X_i.$$

Hence $Y = VX$ where $V = (v_{ij})$ given by

$$v_{ij} = -w_{ij} \text{ if } i \neq j, \sum_{k=1}^{n} w_{ik} \text{ if } i = j. \tag{13.33}$$

Similarly to Proposition 13.1 for the function $\xi$, one has the following results.

**Proposition 13.2.** *Let $V$ be the matrix defined by Eq. (13.33). Then*

*(i)* *$V$ is positive semidefinite, $\nabla(\frac{1}{2})\eta^2(X) = VX$ and $\eta^2(X) = \langle X, VX\rangle$.*

*(ii)* *If the weight matrix $W$ is irreducible assumption (a3), then $\mathscr{A} = \eta^{-1}(0) = \{X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) : VX = 0\} = \mathscr{N}(V)$, rank $V = n-1$ and $\mathscr{A}^{\perp} = \{Y = VX : X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\} = Im\mathrm{V}$.*

*(iii)* *$(\frac{1}{2}\eta^2)^*(Y) = \frac{1}{2}\langle Y, V^+Y\rangle$ if $Y \in \mathscr{A}^{\perp}, +\infty$ otherwise. In other words*

$$\left(\frac{1}{2}\eta^2\right)^*(Y) = \frac{1}{2}\langle V^+Y, Y\rangle + \chi_{\mathscr{A}^{\perp}}(Y) \text{ for } Y \in \mathscr{M}_{n,p}(\mathrm{I\!R}).$$

*(iv)* *dom $(\frac{1}{2}\eta^2)^* =$ dom $\partial(\frac{1}{2}\eta^2)^*$ and $\partial(\frac{1}{2}\eta^2)^*(Y) = V^+Y + \mathscr{A}$ for $Y \in \mathscr{A}^{\perp}$.*

*Proof.* (i) The positive semidefiniteness of $V$ follows from [47] as in Proposition 13.1. Since $\nabla(\frac{1}{2}\eta^2)(X) = VX$, the generalized Euler relation [44] yields $\eta^2(X) = \langle X, VX\rangle$.

(ii) It remains to prove that $\operatorname{rank} V = n-1$, since the other assertions follow from the fact that $V$ is symmetric positive semidefinite. First, we see that $\operatorname{rank} V \leq n-1$ because $Ve = 0$. Suppose now $\operatorname{rank} V < n-1$. Then there exists $v \notin \mathbb{R}e$ such that $Vv = 0$. Let $X = v(e^p)^T$ ($e^p$ is the vector of ones in $\mathbb{R}^p$). Clearly, $VX = 0$, and therefore $X \in \mathscr{A}$. By the definition of $\mathscr{A}$, all rows of $X$ are identical, which implies that $v \in \mathbb{R}e$. This contradiction proves that $\operatorname{rank} V = n-1$.

(iii) By definition, $(\frac{1}{2}\eta^2)^*(Y) := \sup\{\langle X, Y\rangle - (\frac{1}{2}\eta^2)(X) : X \in \mathscr{M}_{n,p}(\mathbb{R})\}$. It follows that $(\frac{1}{2}\eta^2)^*(Y) = +\infty$ if $Y \notin \mathscr{A}^\perp$. For $Y \in \mathscr{A}^\perp$, $X$ solves the above problem if and only if $VX = Y$, i.e., $X \in V^+Y + \mathscr{A}$, where $V^+$ denotes the pseudo-inverse of $V$. Hence $(\frac{1}{2}\eta^2)^*(Y) = \frac{1}{2}\langle V^+Y, Y\rangle$ if $Y \in \mathscr{A}^\perp$, and so

$$\left(\frac{1}{2}\eta^2\right)^*(Y) = \frac{1}{2}\langle V^+Y, Y\rangle + \chi_{\mathscr{A}^\perp}(Y) \text{ for } Y \in \mathscr{M}_{n,p}(\mathbb{R}).$$

Since $V^+$ is symmetric positive semidefinite, we have

$$\partial\left(\frac{1}{2}\eta^2\right)^*(Y) = V^+Y + \mathscr{A} \text{ for } Y \in \mathscr{A}^\perp.$$

The proof then is completed. □

Hence, determining the gradient of $(\frac{1}{2}\eta^2)^*(Y)$ with $Y \in \mathscr{A}^\perp$ amounts to compute the pseudo-inverse of $V$. The next result permits to calculate $V^+$.

**Proposition 13.3.** *If the weight matrix W is irreducible assumption (a3), then we have:*

*(i)* $\mathit{Im}\, V^+ = \mathit{Im}\, V = \mathscr{A}^\perp$ *and* $\langle V^+Y, Y\rangle > 0$ *for* $Y \in \mathscr{A}^\perp \setminus \{0\}$.

*(ii)* $V^+Y = \left(V + \frac{1}{n}ee^T\right)^{-1} Y - \frac{1}{n}ee^T Y \forall Y \in \mathscr{M}_{n,p}(\mathbb{R})$. *That implies, for* $Y \in \mathscr{A}^\perp$,

$$X = V^+Y = \left(V + \frac{1}{n}ee^T\right)^{-1} Y, \imath.e., (V + \frac{1}{n}ee^T)X = Y. \tag{13.34}$$

*Hence, in the normal case where* $V = ncI - cee^T$ *(i.e.,* $w_{ij} = c$ *for all* $i \neq j$*), the solution to Eq. (13.34) is* $X = Y/(nc)$. *In other words*

$$V^+Y = \frac{Y}{nc}, \text{ for } Y \in \mathscr{A}^\perp. \tag{13.35}$$

*Proof.* Assertion (i) is a well-known property for pseudo-inverses of symmetric positive semidefinite matrices (see also Eq. (13.34)) while the assertion (ii) is easy to prove and is omitted here.

Before going further, it is worth noting the following crucial consequences on both theoretical and algorithmic viewpoints of DCA for solving problem (13.22).

□

*Remark 13.4.* (i) It follows from the very definition of the seminorms $\eta$ and $\xi$ and the Cauchy–Schwarz inequality $\xi(X) \leq \eta_\delta \eta(X) \quad \forall X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$. Hence

$$\frac{1}{\eta_\delta}\eta^0 = (\eta_\delta \eta)^0 \leq \xi^0.$$

(ii) We have $VV^+Y = Y = V^+VY$ for $Y \in \operatorname{Im} V = \operatorname{Im} V^+ = \mathscr{A}^\perp$. Hence

$$\eta(V^+Y) = \langle Y, V^+Y\rangle^{1/2} = \eta^0(Y) \leq \eta_\delta \;\text{ for }\; Y \in \mathscr{A}^\perp. \tag{13.36}$$

(iii) Under the assumptions (a1), (a2), and (a3), X$^*$ is a solution to problem (EDP$_1$) if and only if X$^* \in \Omega$ and

$$B(X^*) = V \tag{13.37}$$

according to Eqs. (13.28), (13.33). Moreover, $\rho(\frac{1}{2}\eta^2, \mathscr{A}^\perp)$ and $\rho((\frac{1}{2}\eta^2)^*, \mathscr{A}^\perp)$ are positive.

From the above displayed calculations, we can now give the description of the DCA applied to Problem (13.22) (or, equivalently Problem (13.24).

### The Description of DCA for Solving Problem (13.24)

We present below the DCA applied to problem (13.24) in the general case and the normal case which are respectively denoted by **DCA1** and **DCA1bis**. The latter will be used to compute an initial point for the former.

**DCA1** *(DCA applied to Problem (13.24)) Generate two sequences* $\{X^{(k)}\}$ *and* $\{Y^{(k)}\}$ *in* $\mathscr{A}^\perp$ *as follows*:

Let $\tau_1 > 0$, $\tau_2 > 0$ and $0 \neq X^{(0)} \in \mathscr{A}^\perp$ be given.

For $k = 0, 1, \ldots$ until

either $\|X^{(k+1)} - X^{(k)}\| \leq \tau_1 \|X^{(k+1)}\|$ or $|F_1(X^{(k)}) - F_1(X^{(k+1)})| \leq \tau_2(|F_1(X^{(k+1)}| + 1)$

take

$$Y^{(k)} = B(X^{(k)})X^{(k)},$$

and solve the following nonsingular system to obtain $X^{(k+1)}$

$$\left(V + \frac{1}{n}ee^T\right)X = Y^{(k)}. \tag{13.38}$$

**DCA1bis** *(DCA applied to Problem (13.24) in the normal case, say* $w_{ij} = c \neq 0$ $\forall i, j$*).*

Replace Eq. (13.38) in **DCA1** by

$$X^{(k+1)} = \frac{1}{nc} Y^{(k)}. \tag{13.39}$$

The main results on DCA's convergence for general DC programs (see Sect. 13.2) can be refined as follows.

**Proposition 13.4.** *The sequences $\{X^{(k)}\}$ and $\{Y^{(k)}\}$ generated by DCA1 satisfy the following properties:*

*(i)* $\eta(X^{(k+1)}) = \eta(V^+Y^{(k)}) = \langle Y^{(k)}, V^+Y^{(k)}\rangle^{1/2} = \eta^0(Y^{(k)}) \le \eta_\delta \ \forall k$*, i.e., they are bounded.*

*(ii)*

$$\frac{1}{2}\eta^2(X^{(k+1)}) - \xi(X^{(k+1)}) \le -\frac{1}{2}\langle Y^{(k)}, V^+Y^{(k)}\rangle \le \frac{1}{2}\eta^2(X^{(k)}) - \xi(X^{(k)}) - \delta_k \ \forall k$$

*where* $\delta_k := \max\{\frac{1}{2}\rho(\frac{1}{2}\eta^2, \mathscr{A}^\perp)\|X^{(k+1)} - X^{(k)}\|^2, \frac{1}{2}\rho(\frac{1}{2}(\eta^0)^2, \mathscr{A}^\perp)\|Y^{(k)} - Y^{(k-1)}\|^2\}.$

*(iii) The sequences* $\{\eta(X^{(k)})\}, \{\xi(X^{(k)})\}$ *and* $\{\eta^0(Y^{(k)})\}$ *are increasing*

$$\eta^2(X^{(k)}) \le \xi(X^{(k)}) \le \frac{1}{2}[\eta^2(X^{(k)}) + \eta^2(X^{(k+1)})] - \delta_k \le \eta_\delta - \delta_k \ \forall k$$

$$\xi(X^{(k+1)}) \ge \xi(X^{(k)}) + \frac{1}{2}[\eta^2(X^{(k+1)}) - \eta^2(X^{(k)})] + \delta_k \ \forall k.$$

*(iv) (Finite convergence of DCA1)* $\frac{1}{2}\eta^2(X^{(k+1)}) - \xi(X^{(k+1)}) = \frac{1}{2}\eta^2(X^{(k)}) - \xi(X^{(k)})$ *if and only if* $X^{(k+1)} = X^{(k)}$ *(or equivalently* $Y^{(k)} = Y^{(k-1)}$*). In such a case DCA1 stops at the kth iteration with*

$$Y^{(k)} = B(X^{(k)})X^{(k)} = VX^{(k)}, \ \xi(X^{(k)}) = \eta^2(X^{(k)}) = (\eta^0)^2(Y^{(k-1)}),$$

*and* $X^{(k)}$ *solves problem (13.24) if and only if* $\xi(X^{(k)}) = \eta_\delta^2$ *(i.e.,* $\eta(X^{(k)}) = \eta_\delta$*).*

*(v) (Infinite convergence of DCA1) If the sequences* $\{X^{(k)}\}$ *and* $\{Y^{(k)}\}$ *are infinite, then the two series* $\{\|X^{(k+1)} - X^{(k)}\|^2\}$, $\{\|Y^{(k+1)} - Y^{(k)}\|^2\}$ *converge, and we have for every limit point* $(X^*, Y^*)$ *of* $\{X^{(k)}, Y^{(k)}\}$

$$X^* = V^+Y^*, \ Y^* = VX^* \in \partial\xi(X^*), \ \xi(X^*) = \eta^2(X^*) = (\eta^0)^2(Y^*).$$

*In such a case* $X^*$ *solves problem (13.24) if and only if* $\xi(X^*) = \eta_\delta^2$ *(i.e.,* $\eta(X^*) = \eta_\delta$*).*

*Proof.* (i) follows from Remark 13.6. The remaining assertions are consequences of the main results on DCA (see Sect. 13.2) after simple calculations (related to the conjugate function, the polar, and the subdifferential of the two seminorms $\eta$, $\xi$), the fact that the modulus of strong convexity $\rho(\frac{1}{2}\eta^2, \mathscr{A}^\perp)$ and $\rho(\frac{1}{2}(\eta^0)^2, \mathscr{A}^\perp)$ are positive, and Proposition 13.5. □

#### 13.3.1.2 The Proximal Regularized DC Program of Problem (13.22): DCA1r

As indicated in Remark 13.1, it is worth introducing the proximal regularized DC program of (EDP$_1$): ( $\rho$ being a nonnegative number, called regularization parameter)

$$\min\left\{F_1(X) := \left[\frac{\rho}{2}\|X\|^2 + \frac{1}{2}\eta^2(X)\right] - \left[\frac{\rho}{2}\|X\|^2 + \xi(X)\right] : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\} \tag{13.40}$$

where the DC decomposition of $F(X)$ is

$$G(X) := \frac{\rho}{2}\|X\|^2 + \frac{1}{2}\eta^2(X),\ H(X) := \frac{\rho}{2}\|X\|^2 + \xi(X). \tag{13.41}$$

The original DC program (EDP$_1$) is a special case of Problem (13.40) with $\rho = 0$.

The DCA applied to Problem (13.40) differs from the DCA applied to (EDP$_1$) by the following two facts: the symmetric positive semidefinite matrices $B(X)$ and $V$ are replaced by $\rho I + B(X)$ and $\rho I + V$, respectively. More precisely, the regularized version of **DCA1** can be described as follows:

**DCA1r** (*the DCA applied to the proximal regularized DC program (13.40)*)

Let $\tau_1 > 0$, $\tau_2 > 0$, and $0 \neq X^{(0)} \in \mathscr{A}^\perp$ be given.

For $k = 0, 1, \ldots$ until

either $\|X^{(k+1)} - X^{(k)}\| \leq \tau_1 \|X^{(k+1)}\|$ or $|F_1(X^{(k)}) - F_1(X^{(k+1)})| \leq \tau_2(|F_1(X^{(k+1)}| + 1)$

take $Y^{(k)} = B(X^{(k)})X^{(k)} + \rho X^{(k)}$, and solve the next equation to obtain $X^{(k+1)}$:

$$(V + \rho I)X = Y^{(k)}. \tag{13.42}$$

*Remark 13.5.* The other visible advantage of **DCA1r** concerns the computation of the pseudo-inverse $V^+$ of $V$: for computing $V^+$, we have to apply the Cholesky factorization to the matrix $V + \frac{1}{n}ee^T$ which destroys the sparsity structure of $V$ while the sparse Cholesky factorization can be advantageously applied to the symmetric positive matrix $\rho I + V$ which preserves the sparsity structure of $V$. In our experiments DCA1r seems to be more robust and efficient than DCA1 (see Sect. 13.6).

#### 13.3.1.3 The Dual Formulation of (EDP$_1$) via Lagrangian Duality

As indicated above, an important issue in the DCA is a *good* DC decomposition of the problem being considered. For this purpose, by using the Lagrangian duality with zero gap relative to the problem of maximization of a finite gauge $\psi$ over the unit ball defined by a finite gauge $\phi$ such that $\phi^{-1}(0)$ is a subspace contained in $\psi^{-1}(0)$ [19, 39], we will state an equivalent problem of (EDP$_1$) which is a DC program too. Let us first recall the following result [19,39]:

**Lemma 13.2.** *The convex maximization program*

$$\omega := \max\{\xi(X) : \quad X \in U(\eta)\} \tag{13.43}$$

*with $U(\eta) := \{X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) : \eta(X) \leq 1\}$ formulated as a DC program*

$$-\omega := \min\{\chi_{\mathscr{U}(\eta)}(X) - \xi(X) : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\} \tag{13.44}$$

*is equivalent to the DC program*

$$-\frac{\omega^2}{2} = \min\left\{F_1(X) := \frac{1}{2}\eta^2(X) - \xi(X) : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\} \tag{13.45}$$

*in the sense that*

*(i) The solutions to problem (13.44) are of the form $X^*/\eta(X^*)$ with $X^*$ being a solution to problem (13.45).*
*(ii) The solutions to problem (13.45) are of the form $\xi(X^*)X^*$ with $X^*$ being a solution to problem (13.44).*

For our distance geometry problem, $\omega = \eta_\delta$, and the next useful result is a consequence of Lemma 13.2

**Proposition 13.5.** *The convex maximization program*

$$\eta_\delta^2 = \max\{\xi(X) : \quad \eta(X) \leq \eta_\delta\} \tag{13.46}$$

*formulated as a DC program (with $C := \{X \in \mathscr{M}_{n,p}(\mathrm{I\!R}) : \frac{1}{2}\eta^2(X) \leq \frac{1}{2}\eta_\delta^2\}$)*

$$-\eta_\delta^2 = \min\{\chi_C(X) - \xi(X) : \quad X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\} \tag{13.47}$$

*is equivalent to the DC program (13.22) in the sense that they have the same solution set. Moreover $X^*$ solves these problems if and only if $\xi(X^*) = \eta^2(X^*) = \eta_\delta^2$.*

Similarly to lemma 13.1, if $\mathscr{A} = \eta^{-1}(0) = \xi^{-1}(0)$, then solving the problem (13.47) leads to solve the following problem:

$$-\eta_\delta^2 = \min\{\chi_{\mathscr{C}}(X) - \xi(X) : \quad X \in \mathscr{A}^\perp\}, \tag{13.48}$$

and $X^*$ is an optimal solution of Eq. (13.48) if and only if $X^* + Z$ is an optimal solution of Problem (13.47) for all $Z \in \mathscr{A}$.

Applying DCA on problem (13.47) amounts to computing $\partial\xi$ and $\partial\chi_C^*$. According to Proposition 13.1 $\xi$ is differentiable and $\nabla\xi(X) = B(X)X$.

### Computing $\partial\chi_C^*$

Since $C = \{X \in \mathscr{M}_{n,p}(\mathbb{R}) : \eta(X) \leq \eta_\delta\}$, according to [44], $\chi_C^*$ is $\eta_\delta$ times the polar $\eta^0$ of the gauge (seminorm) $\eta$:

$$\eta^0(Y) := \sup\{\langle X,Y\rangle : \eta(X) \leq 1\} = \sup\left\{\langle X,Y\rangle : \frac{1}{2}\eta^2(X) \leq \frac{1}{2}\right\}.$$

We have $\eta^0(Y) = +\infty$ if $Y \notin \mathscr{A}^\perp$. Let now $Y \in \mathscr{A}^\perp$. It is clear that $X$ is an optimal solution to the above problem if and only if there is a positive number $\lambda$ such that (i) $\langle X, VX\rangle \leq 1$, (ii) $Y = \lambda VX$, (iii) $\lambda(\langle X, VX\rangle - 1) = 0$. First assume that $Y \neq 0$. Then $\lambda$ must be positive and (ii) implies that $X \in \frac{1}{\lambda}V^+Y + \mathscr{A}$. The value of $\lambda$ is given according to (iii) $\lambda = \langle Y, V^+Y\rangle^{\frac{1}{2}}$. Hence $\eta^0(Y) = \langle Y, V^+Y\rangle^{\frac{1}{2}}$.

This formulation holds also for $Y = 0$ because $\eta^0(0) = 0$. Finally we get

$$\eta^0(Y) = \eta_\delta\langle Y, V^+Y\rangle^{\frac{1}{2}} + \chi_{\mathscr{A}^\perp}(Y) \;\; \forall Y \in \mathscr{M}_{n,p}(\mathbb{R}). \tag{13.49}$$

It follows that

**Proposition 13.6.** *(i) The support function $\chi_{U(\eta)}^*$ of $U(\eta)$ is the polar $\eta^0$ of $\eta$, and we have $\chi_{U(\eta)}^*(Y) = \eta^0(Y) = \langle Y, V^+Y\rangle^{\frac{1}{2}} + \chi_{\mathscr{A}^\perp}(Y)$ $\forall Y \in \mathscr{M}_{n,p}(\mathbb{R})$ and $(\frac{1}{2}\eta^2)^* = \frac{1}{2}(\eta^0)^2$. Hence*

$$\partial\chi_{U(\eta)}^*(Y) = \begin{cases} \emptyset & \text{if} \quad Y \notin \mathscr{A}^\perp, \\ V^+Y/\langle Y, V^+Y\rangle^{1/2} + \mathscr{A} & \text{if} \quad Y \in \mathscr{A}^\perp \setminus \{0\}, \\ U(\eta) & \text{if} \quad Y = 0. \end{cases} \tag{13.50}$$

*In the normal case, we have $V^+Y = Y/nc$ for $Y \in \mathscr{A}^\perp$; therefore*

$$\partial\chi_{U(\eta)}^*(Y) = \begin{cases} \emptyset & \text{if} \quad Y \notin \mathscr{A}^\perp, \\ Y/(\sqrt{nc}\|Y\|) + \mathscr{A} & \text{if} \quad Y \in \mathscr{A}^\perp \setminus \{0\}, \\ U(\eta) & \text{if} \quad Y = 0. \end{cases} \tag{13.51}$$

*(ii) For $C := \eta_\delta U(\eta)$, we have $\chi_C^* = \eta_\delta\chi_{U(\eta)}^*$.*

Before going further, it is worth noting the following crucial consequences on both theoretical and algorithmic viewpoints of DCA for solving problems (13.22) and (13.47).

*Remark 13.6.* (i) It follows from the very definition of the seminorms $\eta$ and $\xi$ and the Cauchy–Schwarz inequality $\xi(X) \leq \eta_\delta \eta(X) \quad \forall X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$. Hence

$$\frac{1}{\eta_\delta}\eta^0 = (\eta_\delta \eta)^0 \leq \xi^0.$$

(ii) We have $VV^+Y = Y = V^+VY$ for $Y \in \operatorname{Im} V = \operatorname{Im} V^+ = \mathscr{A}^\perp$. Hence

$$\eta(V^+Y) = \langle Y, V^+Y\rangle^{1/2} = \eta^0(Y) \leq \eta_\delta \;\text{ for }\; Y \in \mathscr{A}^\perp. \tag{13.52}$$

(iii) Under the assumptions (a1), (a2), and (a3), X* is a solution to problem (EDP$_1$) if and only if X* $\in \Omega$ and

$$B(X^*) = V \tag{13.53}$$

according to Eqs. (13.28), (13.33). Moreover, $\rho(\frac{1}{2}\eta^2, \mathscr{A}^\perp) > 0$ and $\rho((\frac{1}{2}\eta^2)^*, \mathscr{A}^\perp) > 0$.

#### 13.3.1.4 The Description of DCA for Solving Problem (13.48)

**DCA2** *(DCA applied to Problem (13.48)) Generate two sequences $\{X^{(k)}\}$ and $\{Y^{(k)}\}$ in $\mathscr{A}^\perp$ as follows*:

Let $\tau_1 > 0$, $\tau_2 > 0$ , and $0 \neq X^{(0)} \in \mathscr{A}^\perp \cap C$ be given.

For $k = 0, 1, \ldots$ until

either $\|X^{(k+1)} - X^{(k)}\| \leq \tau_1 \|X^{(k+1)}\|$ or $|\xi(X^{(k)}) - \xi(X^{(k+1)})| \leq \tau_2(\xi(X^{(k+1)} + 1)$

take

$$Y^{(k)} = B(X^{(k)})X^{(k)}, \tag{13.54}$$

$$X^{(k+1)} = \frac{\eta_\delta V^+ Y^{(k)}}{\langle Y^{(k)}, V^+Y^{(k)}\rangle^{1/2}} = \frac{\eta_\delta V^+ Y^{(k)}}{\eta(V^+Y^{(k)})}. \tag{13.55}$$

**DCA2bis** *(DCA applied to Problem (13.48) in the normal case).*

Replace Eq. (13.55) in DCA2 by

$$X^{(k+1)} = \frac{Y^{(k)}}{\sqrt{nc}\|Y^{(k)}\|}. \tag{13.56}$$

Like Proposition 13.4, we have the following convergence result for DCA2.

**Proposition 13.7.** *The sequences $\{X^{(k)}\}$ and $\{Y^{(k)}\}$ generated by DCA2 satisfy the following properties:*

*(i) $\eta(X^{(k)}) = \eta_\delta$, for all k, and $\eta^0(Y^{(k)}) \leq \eta_\delta$, i.e., they are bounded.*

(ii) *The sequences* $\{\xi(X^{(k)})\}$ *and* $\{\eta^0(Y^{(k)})\}$ *are increasing:*

$$\xi(X^{(k)}) \leq \eta_\delta \eta^0(Y^{(k)}) \leq \xi(X^{(k+1)}) \;\; \forall k.$$

(iii) *(Finite convergence of DCA2)* $\xi(X^{(k+1)}) = \xi(X^{(k)})$ *if and only if* $X^{(k+1)} = X^{(k)}$ *(or equivalently* $Y^{(k+1)} = Y^{(k)}$*). In such a case DCA2 stops at the kth iteration with*

$$X^{(k)} = \frac{\eta_\delta V^+ Y^{(k)}}{\eta(V^+ Y^{(k)})}, Y^{(k)} = \frac{\xi(X^{(k)})}{\eta_\delta^2} V X^{(k)},$$

*and* $X^{(k)}$ *solves problem (13.48) if and only if* $\xi(X^{(k)}) = \eta_\delta^2$ *(i.e.,* $\eta^0(Y^{(k)}) = \eta_\delta$*).*

(iv) *(Infinite convergence of DCA2) If the sequences* $\{X^{(k)}\}$ *and* $\{Y^{(k)}\}$ *are infinite, then for every limit point* $(X^*, Y^*)$ *of* $\{X^{(k)}, Y^{(k)}\}$*, it holds*

$$Y^* = \frac{\xi(X^*)}{\eta_\delta^2} V X^* \in \partial \xi(X^*), \quad \eta(X^*) = \eta_\delta \eta^0(Y^*) \leq \eta_\delta \xi(X^*) = \eta_\delta \eta^0(Y^*).$$

*Moreover such an* $X^*$ *solves problem (13.48) if and only if* $\xi(X^*) = \eta_\delta^2$ *(i.e.,* $\eta^0(Y^*) = \eta_\delta$*).*

*Proof.* Assertions (i) and (ii) follow from the main results on DCA for general DC programs (Sect. 13.2) and Proposition 13.5. The remaining assertions can be proved in the same way. Let us demonstrate (iii). According to the results collected in Sect. 13.2, $\xi(X^{(k+1)}) = \xi(X^{(k)})$ implies $Y^{(k)} \in \partial \chi_C\,(X^{(k)})$, i.e., $Y^{(k)} = \lambda_k V X^{(k)}$. But $Y^{(k)} \in \partial \xi(X^{(k)})$, so

$$\xi(X^{(k)}) = \langle Y^{(k)}, X^{(k)} \rangle = \; \lambda_k \langle X^{(k)}, V X^{(k)} \rangle = \lambda_k \eta^2(X^{(k)}) = \lambda_k \eta_\delta^2.$$

It follows that

$$Y^{(k)} = \frac{\xi(X^{(k)})}{\eta_\delta^2} V X^{(k)} \text{ and } X^{(k)} = \frac{\eta_\delta V^+ Y^{(k)}}{\eta(V^+ Y^{(k)})} = X^{(k+1)}.$$

The converse is obvious and the proof is completed. □

*Remark 13.7.* (i) Computing $V^+ Y^{(k)}$ in DCA1 and/or DCA2 amounts to solving

$$\left(V + \frac{1}{n} e e^T\right) X = Y^{(k)} \tag{13.57}$$

for which the Cholesky factorization seems to be one of the efficient methods.

(ii) The calculation of $X^{(k+1)}$ in DCA1bis and DCA2bis requires only matrix-vector products.

(iii) In DCA1bis we have $\rho(\frac{1}{2}\eta^2, \mathscr{A}^\perp) = nc$ and $\rho((\frac{1}{2}\eta^2)^*, \mathscr{A}^\perp) = \frac{1}{nc}$.

#### 13.3.1.5 The Proximal Regularized DC Program of Problem (13.47): DCA2r

As for problem (13.22), we introduce the proximal regularization technique into problem (13.47) in order to obtain the robustness and the stability in numerical computations. Here we will not use the Hilbertian kernel $\frac{\rho}{2}\|.\|^2$ but the quadratic function $\frac{\rho}{2}\eta^2$ (which is positive definite on $\mathscr{A}^\perp$) because we have explicit calculations for the latter. The regularized DC program of problem (13.47) is thus its equivalent DC program:

$$\min\left\{\left[\frac{\rho}{2}\eta^2(X)+\chi_{\eta_\delta U(\eta)}(X)\right]-\left[\xi(X)+\frac{\rho}{2}\eta^2(X)\right]:X\in\mathscr{M}_{n,p}(\mathrm{I\!R})\right\}. \quad (13.58)$$

The DCA applied to problem (13.58) computes $Y^{(k)}=B(X^{(k)})X^{(k)}+\rho VX^{(k)}$ and $X^{(k+1)}\in\mathscr{A}^\perp$ which is the unique solution of the convex program

$$\min\left\{\frac{\rho}{2}\eta^2(X)-\langle X,Y^{(k)}\rangle:X\in\mathscr{A}^\perp,\eta(X)\le\eta_\delta\right\}. \quad (13.59)$$

**Lemma 13.3.** *Let $\overline{Y}\in\mathscr{A}^\perp$ be fixed. The unique solution $\overline{X}$ to the convex program*

$$\min\left\{\frac{\rho}{2}\eta^2(X)-\langle X,\overline{Y}\rangle:X\in\mathscr{A}^\perp,\eta(X)\le\eta_\delta\right\} \quad (13.60)$$

*is given by*

$$\overline{X}:=\frac{1}{\rho}V^+Y \quad if \quad \eta(V^+\overline{Y})\le\rho\eta_\delta,\ \frac{\eta_\delta}{\eta(V^+\overline{Y})}V^+\overline{Y}\ \textit{otherwise.} \quad (13.61)$$

*Moreover if $\overline{Y}=B(X)X+\rho VX$ with $X\in\mathscr{A}^\perp,\rho^2\eta^2(X)+2\rho\xi(X)+\eta^2(V^+B(X)X)\ge\rho^2\eta_\delta^2$, then the unique solution $\overline{X}$ to problem (13.60) is simply $\overline{X}=\frac{\eta_\delta}{\eta(V^+\overline{Y})}V^+\overline{Y}$.*

*Proof.* Since the quadratic function $\eta^2$ is positive definite on $\mathscr{A}^\perp$, problem (13.60) has a unique solution $\overline{X}\in\mathscr{A}^\perp$ defined by ($\overline{\lambda}$ being a nonnegative number)

$$\eta(\overline{X})\le\eta_\delta,\ \rho V\overline{X}-\overline{Y}=-\overline{\lambda}V\overline{X},\ \overline{\lambda}(\eta(\overline{X})-\eta_\delta)=0.$$

It follows that $\overline{X}=\frac{1}{\lambda+\rho}V^+\overline{Y}$ and $\eta^2(\overline{X})=\frac{1}{(\lambda+\rho)^2}\ \eta^2(V^+\overline{Y})$. According to the first and third conditions we get

$$\overline{\lambda}=0\ \ if\ \ \eta(V^+\overline{Y})\le\rho\eta_\delta,\ \frac{\eta(V^+\overline{Y})}{\eta_\delta}-\rho\ \text{ otherwise.}$$

The formulation (13.61) then is immediate. It remains to prove that if $\overline{Y}=B(X)X+\rho VX$ with $X\in\mathscr{A}^\perp$ given as above, then $\eta(V^+\overline{Y})\ge\rho\eta_\delta$. Indeed we have:

$$\begin{aligned}\eta^2(V^+\overline{Y}) &= \langle VV^+\overline{Y},V^+\overline{Y}\rangle=\langle\overline{Y},V^+\overline{Y}\rangle=\langle B(X)X+\rho VX,V^+B(X)X+\rho X\rangle\\ &=\rho^2\langle VX,X\rangle+2\rho\langle X,B(X)X\rangle+\langle B(X)X,V^+B(X)X\rangle\\ &=\rho^2\eta^2(X)+2\rho\xi(X)+\eta^2(V^+B(X)X)\ge\rho^2\eta_\delta^2.\end{aligned}$$

□

The DCA for solving the proximal regularized DC program (13.58) then is given by

```
**DCA2r** (*the DCA applied to the proximal regularized DC program (13.58)*)
Let $\tau_1 > 0$, $\tau_2 > 0$ and $X^{(0)} \in \mathscr{A}^{\perp}$, $\eta(X^{(0)}) = \eta_\delta$ be given.
For $k = 0, 1, \ldots$ until
```

either $\|X^{(k+1)} - X^{(k)}\| \leq \tau_1 \|X^{(k+1)}\|$ or $|\xi(X^{(k)}) - \xi(X^{(k+1)})| \leq \tau_2(\xi(X^{(k+1)}) + 1)$

take $Y^{(k)} = B(X^{(k)})X^{(k)} + \rho V X^{(k)}$, and solve

$$(V + \frac{1}{n}ee^T)X = Y^{(k)} \tag{13.62}$$

to obtain $V^+Y^{(k)}$ and set $X^{(k+1)} = \frac{\eta_\delta}{\eta(V^+Y^{(k)})} V^+Y^{(k)}$.

*Remark 13.8.* Regarding Proposition 13.7, the above regularization implies the following property for DCA2: since the modulus of strong convexity $\rho(\frac{1}{2}\eta^2, \mathscr{A}^{\perp})$ and $\rho(\frac{1}{2}(\eta^0)^2, \mathscr{A}^{\perp})$ are positive, the two series $\{\|X^{(k+1)} - X^{(k)}\|^2\}$, $\{\|Y^{(k+1)} - Y^{(k)}\|^2\}$ converge.

### 13.3.2 The Nonstandard $l_\infty$ and Combined $l_1 - l_\infty$ Approaches

The preceding two DC programs (13.22) and (13.47) for (EDP$_1$) involve the $l_1$-norm in the definition of their objective functions. At least from the computational point of view, as will be shown in the numerical simulations (Sect. 13.6), it is important to use the $l_\infty$-norm to formulate the following nonstandard DC programs for (EDP$_1$). The first reformulation is

$$0 = \min\left\{\Phi(X) := \max_{(i,j)\in\mathscr{S}^w}\left\{\Phi_{ij}(X) := \frac{1}{2}w_{ij}[d_{ij}(X) - \delta_{ij}]^2\right\} : X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\}.$$

In fact we will tackle the equivalent constrained problem (Lemma 13.1 and Proposition 13.3)

$$\begin{aligned} 0 &= \min\{\Phi(X) : X \in \mathscr{C}\}, \\ \mathscr{C} &:= \{X \in \mathscr{A}^{\perp} : \sum_{i<j} \|X_i^T - X_j^T\|^2 = n\|X\|^2 \leq r^2\} \text{ and } r^2 := \sum_{i<j} \widetilde{\delta}_{ij}^2, \end{aligned} \tag{13.63}$$

where $\tilde{\delta}_{ij}$ denotes the length of the shortest paths between nodes $i$ and $j$ (Sect. 13.5). Note that $\mathscr{C}$ is a compact convex set. Let us now prove that problem (13.63) is actually a DC program. It is clear that

$$\Phi_{ij}(X) = \frac{1}{2}w_{ij}d_{ij}^2(X) + \frac{1}{2}w_{ij}\delta_{ij}^2 - w_{ij}\delta_{ij}d_{ij}(X)$$

is a DC function on $\mathscr{M}_{n,p}(\mathbb{R})$. Hence its finite pointwise supremum $\Phi$ is DC too with the following DC decomposition [40]:

$$\Phi(X) := \zeta(X) - \xi(X),$$
$$\text{where } \zeta(X) := \max_{(i,j)\in\mathscr{S}^w}\{\zeta_{ij}(X) := \frac{1}{2}w_{ij}d_{ij}^2(X) + \frac{1}{2}w_{ij}\delta_{ij}^2 + \sum_{\substack{(k,l)\in\mathscr{S}^w\\(k,l)\neq(i,j)}} w_{kl}\delta_{kl}d_{kl}(X)\}. \tag{13.64}$$

Hence problem (13.63) can be recasted into the following DC program:

$$0 = \min\{\zeta(X) - \xi(X) : X \in \mathscr{C}\}. \tag{13.65}$$

In the combined $l_1 - l_\infty$ approach, the distance geometry problem is equivalently stated as the DC program

$$\begin{aligned}-\frac{\rho}{2}\eta_\delta^2 &= \min\left\{F_2(X) := [\zeta(X) - \xi(X)] + \rho\left[\frac{1}{2}\eta^2(X) - \xi(X)\right] : X \in \mathscr{C}\right\}\\ &= \min\left\{F_2(X) = \left[\zeta(X) + \frac{\rho}{2}\eta^2(X)\right] - (1+\rho)\xi(X) : X \in \mathscr{C}\right\}. \end{aligned}\tag{13.66}$$

The positive constant $\rho$ is to be chosen according to problems under consideration. It is clear that problems (13.65) and (13.66) are actually nonsmooth DC programs, i.e., they cannot be transformed into equivalently smooth nonconvex programs. The practical usefulness of these reformulations resides in the fact that DCA applied to problem (13.66) may better approximate global solutions than DCA applied to the standard problems (13.22), (13.47).

*Remark 13.9.* Problems (13.22), (13.47), (13.65), and (13.66) have the same (global) solution set. But the local optimality condition (13.15) used for constructing DCA is not the same for the first two problems and the last two. It is crucial for DCA since we could restart DCA applied to problem (13.66) from an initial point computed by DCA applied to problem (13.22). By this way local solutions computed by DCA could be improved (see computational results in Sect. 13.6).

To perform the DCA applied to the $l_1 - l_\infty$ DC program (13.66), we have only to calculate the subdifferential of the convex functions $\zeta$ and $(\zeta + \frac{\rho}{2}\eta^2 + \chi_{\mathscr{C}})^*$.

Let $\mathscr{S}_\zeta(X) := \{(i,j) \in \mathscr{S}^w : \zeta_{ij}(X) = \zeta(X)\}$. Clearly that it is simpler to compute $\mathscr{S}_\zeta(X)$ with the following formulation:

$$\mathscr{S}_\zeta(X) = \{(i,j) \in \mathscr{S}^w : \Phi_{ij}(X) = \Phi(X)\}. \tag{13.67}$$

Using usual rules for subdifferential calculus we have (*co* stands for the convex hull)

$$\partial\zeta(X) = co\{\cup\partial\zeta_{ij}(X) : (i,j) \in \mathscr{S}_\zeta(X)\}.$$

According to the above computation of $\partial\xi$ and Remark 13.3:

(i) Range $\partial\zeta \subset \mathscr{A}^{\perp}$.
(ii) We can choose the particular subgradient of $\zeta$:

$$B(X)X + w_{ij}[1-\delta_{ij}s_{ij}(X)]M_{ij}X = B_{ij}(X)X \in \partial\zeta(X) \text{ for } (i,j) \in \mathscr{S}_{\zeta}(X). \tag{13.68}$$

*Remark 13.10.* It follows from the definition of the matrix $M_{ij}$ in Eq. (13.30) that (i) the symmetric matrices $B(X)$ and $B_{ij}(X)$, serving to calculate subgradients of the convex functions $\xi$ and $\zeta$, respectively, ($B(X)X \in \partial\xi(X)$ and $B_{ij}(X)X \in \partial\zeta(X)$ for $(i,j) \in \mathscr{S}_{\zeta}(X)$), differ from each other at four entries:

$$[B_{ij}(X)]_{ii} = [B(X)]_{ii} + w_{ij}[1-\delta_{ij}s_{ij}(X)],\ [B_{ij}(X)]_{jj} = [B(X)]_{jj} + w_{ij}[1-\delta_{ij}s_{ij}(X)],$$
$$[B_{ij}(X)]_{ij} = [B_{ij}(X)]_{ji} = [B(X)]_{ij} - w_{ij}[1-\delta_{ij}s_{ij}(X)].$$

(ii) $X^*$ is an optimal solution to $(EDP_1)$ if and only if $B(X^*) = B_{ij}(X^*)$ for all $(i,j) \in \mathscr{S}$.

### Computing $\partial(\zeta + \frac{\rho}{2}\eta^2 + \chi_{\mathscr{C}})^*$

Since the convex function $\eta^2$ is strongly convex on $\mathscr{A}^{\perp}$ (Remark 13.6), the function $(\zeta + \frac{\rho}{2}\eta^2 + \chi_{\mathscr{C}})^*$ is differentiable on $\mathscr{M}_{n,p}(I\!R)$. But unlike the preceding convex functions, it seems that the gradient $\nabla(\zeta + \frac{\rho}{2}\eta^2 + \chi_{\mathscr{C}})^*(Y)$, which is the unique solution of the convex program

$$\min\left\{\zeta(X) + \frac{\rho}{2}\eta^2(X) - \langle X, Y\rangle : X \in \mathscr{A}^{\perp},\ n\|X\|^2 \le r^2\right\}, \tag{13.69}$$

cannot be explicitly calculated. On the other hand, since the projection on the convex set $\mathscr{C} := \{X \in \mathscr{A}^{\perp} : \sum_{i<j}\|X_i^T - X_j^T\|^2 = n\|X\|^2 \le r^2\}$ is explicit, for solving problem (13.69), we suggest the use of the subgradient projection method [5,43] that we succinctly describe below:

### Subgradient Projection Algorithm for Solving Problem (13.69): SGPA

From a given initial point $Z^0 \in \mathscr{C}$, SGPA generates a sequence $\{Z^k\}$ in $\mathscr{C}$ as follows: Assume that $Z^k$ has already been calculated. Calculate the particular subgradient $G^k$ of the objective function at $Z^k$ : $G^k := \rho V Z^k + B_{ij}(Z^k)Z^k - Y$ with $(i,j) \in \mathscr{S}_{\zeta}(Z^k)$. If $G^k = 0$, then $Z^k$ is an optimal solution to problem (13.69). Otherwise calculate the next iteration $Z^{k+1} := Proj_{\mathscr{C}}(Z^k - \lambda_k \frac{G^k}{\|G^k\|})$, where the sequence of positive numbers$\{\lambda_k\}$ is chosen such that $\lambda_k \to 0$ as $k \to +\infty$ and $\sum_{k=1}^{+\infty}\lambda_k = +\infty$. As said above the projection $Proj_{\mathscr{C}}$ is explicit, and $Z^k$, $G^k$ are in $\mathscr{A}^{\perp}$. It has been shown [5,43] that the sequence $\{Z^k\}$ converges to the unique optimal solution to the convex program (13.69).

### 13.3.3 DCA for Solving the $l_1 - l_\infty$ DC Program (13.66): DCA3

**DCA3** *(DCA applied to Problem (13.66)) Generate two sequences* $\{X^{(k)}\} \subset \mathscr{C}$ *and* $\{Y^{(k)}\} \subset \mathscr{A}^\perp$ *as follows*:

Let $\tau_1 > 0, \tau_2 > 0$, and $0 \neq X^{(0)} \in \mathscr{C}$ be given.

For $k = 0, 1, \ldots$ until

$$\text{either } \|X^{(k+1)} - X^{(k)}\| \leq \tau_1 \|X^{(k+1)}\| \text{ or } |F_2(X^{(k)}) - F_2(X^{(k+1)})| \leq \tau_2(|F_2(X^{(k+1)})| + 1)$$

take

$$Y^{(k)} = (1+\rho)B(X^{(k)})X^{(k)} \in (1+\rho)\partial\xi(X^{(k)})$$

and compute the sequence $\{X^{(k,l)} : l \geq 0\}$ generated by SGPA for solving the convex program (starting with $(X^{(k,0)} := X^{(k)})$

$$\min\{\zeta(X) + \frac{\rho}{2}\eta^2(X) - \langle X, Y^{(k)}\rangle : X \in \mathscr{C}\} \tag{13.70}$$

to obtain $X^{(k+1)}$ as the unique optimal solution to Eq. (13.70): $X^{(k+1)} := \lim_{l \to +\infty} X^{(k,l)}$.

*Remark 13.11.* Like DCA1, the two sequences $\{X^{(k)}\}$ and $\{Y^{(k)}\}$ generated by DCA3 are bounded, and the general convergence result for DCA is strengthened by the strong convexity of $\zeta + \frac{\rho}{2}\eta^2$ on $\mathscr{C}$ .

## 13.4 The General Distance Geometry Problem

This section is devoted to the new formulations of the general distance geometry problem and solution methods based on DCA. The standard optimization problem (EDP)-Eq. (13.6) is a DC program, but the objective DC function is too complex and inconvenient for DCA.

### 13.4.1 A Nonsmooth DC Formulation

To simplify matters, the objective function in [22] to the general distance geometry problem (13.3) has been chosen in an *elegant* way (a nonsmooth nonconvex function)

$$(\text{GDP}_1) \quad 0 = \min\left\{\frac{1}{4}\sum_{(i,j)\in\mathscr{S}} w_{ij}(\|x^i - x^j\| - t_{ij})^2 : x^1, \ldots, x^n \in \mathbb{R}^3, l_{ij} \leqslant t_{ij} \leqslant u_{ij}, (i,j) \in \mathscr{S}\right\}$$

where $w_{ij} = w_{ji} > 0$ for $i \neq j, (i,j) \in \mathscr{S}$. It is a *nonsmooth nonconvex linearly constrained optimization problem.* Different formulations of Eq. (13.3) as DC programs are possible; however, the formulation (GDP$_1$) seems to be advantageous to DCA.

Firstly, we need to work only once with both *dense* and *sparse* sets of constraints. In contrast, the existing methods for the general distance geometry problem work many times with full and sparse sets of constraints (see, e.g., the *embed* algorithm) or a sparse set of constraints [33, 48]. Secondly, we can exploit sparsity of the given bound matrices. This is important because only a small subset of constraints is known in practice. Thirdly, although the addition of variable $t_{ij}$ increases the dimension of the problem, it does not really cause any trouble since the problem is solved separately in variables $x$ and $t$.

The algorithms are quite simple and easy to implement. They only require matrix-vector products and one Cholesky factorization.

Note that this formulation corresponds to both exact and general distance geometry problems, because in the exact distance geometry problem one has $l_{ij} = u_{ij} = \delta_{ij}$, then we take $t_{ij} = \delta_{ij}$, and (GDP$_1$) becomes (EDP$_1$).

We first prove that problem (GDP$_1$) is a DC program and point out simple convex functions (on the convex constraint set) whose difference is the objective function of this problem. Since

$$-2t_{ij}\|x^i - x^j\| = -(\|x^i - x^j\| + t_{ij})^2 + \|x^i - x^j\|^2 + {t_{ij}}^2,$$

the objective function of (GDP$_1$) can be expressed as

$$\frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\|x^i - x^j\|^2 + \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\, t_{ij}^2 - \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\|x^i - x^j\| + t_{ij})^2. \tag{13.71}$$

Remark that by setting $w_{ij} = 0$ for $(i,j) \notin \mathscr{S}$ the constraint $(i,j) \in \mathscr{S}$ can be omitted in (GDP$_1$). Since the functions

$$\frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\|x^i - x^j\|^2 + \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\, t_{ij}^2 \quad \text{and} \quad \frac{1}{4}\left\{\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\|x^i - x^j\| + t_{ij})^2\right\}$$

are convex with respect to the variables $((x^1,\ldots,x^n),T)$ with $x^1,\ldots,x^n \in \mathbb{R}^3$ and $T = (t_{ij})$ on the convex constraint set, it is clear that Eq. (13.71) is a DC function. As above, the matrix spaces are useful for various calculations of subgradients in DCA.

Let $\varphi_{ij} : \mathscr{M}_{n,n}(\mathbb{R}) \longmapsto \mathbb{R}$ be the pairwise function defined by $\varphi_{ij}(T) = t_{ij}$, and let $\eta$ be the function defined by Eq. (13.32).

Problem (GDP$_1$) can be now written in the matrix form

$$0=\inf\{\mathscr{F}_1(X,T):=L(X,T)-K(X,T):(X,T)\in\Omega:=\mathscr{M}_{n,p}(\mathrm{I\!R})\times\mathscr{T}\},$$
$$\text{with } L(X,T):=\frac{1}{2}\eta^2(X)+\mu(T),\ \mu(T):=\frac{1}{2}\sum_{i<j}w_{ij}\ \varphi_{ij}^2(T),$$
$$K(X,T):=\frac{1}{4}\sum_{i<j}w_{ij}[d_{ij}(X)+\varphi_{ij}(T)]^2,$$
$$\mathscr{T}:=\{T\in\mathscr{M}_{n,n}(\mathrm{I\!R}):l_{ij}\leqslant t_{ij}\leqslant u_{ij},(i,j)\in\mathscr{S}^w\}. \tag{13.72}$$

Clearly, the function $L$ is finite and convex on $\mathscr{M}_{n,p}(\mathrm{I\!R})\times\mathscr{M}_{n,n}(\mathrm{I\!R})$. Since for every $(i,j)\in\mathscr{S}$, the function: $(X,T)\to d_{ij}(X)+\varphi_{ij}(T)$ is finite and convex on $\mathscr{M}_{n,p}(\mathrm{I\!R})\times\mathscr{M}_{n,n}(\mathrm{I\!R})$ and nonnegative on $\Omega$, the function $K$ then is convex on $\Omega$ too. Let $\chi_\Omega$ be the indicator function of $\Omega$, then problem (GDP$_1$) can be expressed in the standard form of DC programs:

$$0=\inf\left\{\mathscr{F}_1(X,T):=\ \mathscr{G}(X,T)-\ K(X,T):(X,T)\in\mathscr{M}_{n,p}(\mathrm{I\!R})\times\mathscr{M}_{n,n}(\mathrm{I\!R})\right\}, \tag{13.73}$$

where $\mathscr{G}(X,T):=L(X,T)+\chi_\Omega(X,T)$ is the separable function in its variables $X$ and $T$. Before going further let us make precise the obvious relation between problems (13.3) and (13.73).

**Proposition 13.8.** *(i) If a set of positions $(x^1,\dots,x^n)$ is a solution to problem (13.3), then the couple of matrices $(X,T)$, with $X=(x^1,\dots,x^n)^T$ and $t_{ij}=\|x^i-x^j\|$ for $(i,j)\in\mathscr{S}$, is a solution to Eq. (13.73).*

*(ii) If a couple of matrices $(X,T)$ is a solution to Problem (13.73), then the set of positions $(x^1,\dots,x^n)=X^T$ is a solution to Problem (13.3) and $t_{ij}=\|X_i^T-X_j^T\|$ for $(i,j)\in\mathscr{S}$.*

Similarly to the exact distance geometry problem, under assumption (a2) and (a3), we can restrict the working space to $\mathscr{A}^\perp$.

**Lemma 13.4.** *If $W$ is irreducible, then problem (13.73) is equivalent to*

$$0=\inf\left\{F(X,T):=\mathscr{G}(X,T)-K(X,T):(X,T)\in\mathscr{A}^\perp\times\mathscr{M}_{n,n}(\mathrm{I\!R})\right\}. \tag{13.74}$$

## 13.4.2 Solving (GDP$_1$) by DCA

Performing this scheme thus is reduced to calculating subdifferentials of $K$ and $\mathscr{G}^*$:

$$(Y^{(k)},Z^{(k)})\in\partial K(X^{(k)},T^{(k)}),(X^{(k+1)},T^{(k+1)})\in\partial\mathscr{G}^*(Y^{(k)},Z^{(k)}). \tag{13.75}$$

According to Lemma 13.4 the sequence $\{X^{(k)}\}$ can be taken in the subspace $\mathscr{A}^{\perp}$. Likewise, we shall show that (see Proposition 13.2 above) the sequence $\{\mathrm{Y}^{(k)}\}$ is contained in $\mathscr{A}^{\perp}$ too.

### Computing $\partial K$

Remark that problem (13.73) only involves the restriction of $K$ to $\Omega$ where this function is convex. Actually we shall compute the conditional subdifferential of $K$ with respect to the convex set $\Omega$ ([5]) that we again denote by $\partial K$ for simplicity, i.e., $(Y^0, W^0)$ is a conditional subgradient of $K$ at $(X^0, T^0) \in \Omega$ with respect to $\Omega$ if

$$K(X,T) \geq K(X^0,T^0) + \langle (X,T) - (X^0,T^0), (Y^0\ ,W^0)\rangle \ \text{ for } (X,T) \in \Omega.$$

By the very definition, it is easy to state the following inclusion for $(X,T) \in \Omega$:

$$\partial K(X,T) \supset \frac{1}{2}\sum_{i<j} w_{ij}[d_{ij}(X) + \varphi_{ij}(T)][\partial d_{ij}(X) \times \{0\} + \{0\} \times \partial \varphi_{ij}(T)].$$

Since $\varphi_{ij}(T) = t_{ij} = \langle T, E_{ij}\rangle_{\mathscr{M}_{n,n}(\mathrm{I\!R})}$, with $E_{ij} = e_i e_j^T$ ($e_i \in \mathrm{I\!R}^{\mathrm{n}}$ is the unit vector with value one in the $i$th component and zero otherwise), $\varphi_{ij}$ is differentiable on $\mathscr{M}_{n,n}(\mathrm{I\!R})$, and $\nabla\varphi_{ij}(T) = E_{ij}$. Hence $(Y,Z) \in \partial K(X,T)$ can be determined as

$$(Y,Z) = \frac{1}{2}\sum_{i<j} w_{ij}(d_{ij}(X) + \varphi_{ij}(T))(Y(i,j), E_{ij})\ ,$$

with $Y(i,j) \in \partial d_{ij}(X), i.e.,$

$$Y = \frac{1}{2}\sum_{i<j} w_{ij} d_{ij}(X) Y(i,j) + \frac{1}{2}\sum_{i<j} w_{ij}\varphi_{ij}(T) Y(i,j), Z = \frac{1}{2}\sum_{i<j} w_{ij}(d_{ij}(X) + \varphi_{ij}(T)) E_{ij}.$$

Thus $Z = (z_{ij})$ is defined by $z_{ij} = \frac{1}{2} w_{ij}(d_{ij}(X) + t_{ij})$ if $(i,j) \in \mathscr{S}$, 0 if $(i,j) \notin \mathscr{S}$.

The computation $R := \sum_{i<j} w_{ij} d_{ij}(X) Y(i,j)$ has been done in Sect. 13.3.1.1:

$$R = \sum_{i<j} w_{ij} d_{ij}(X) Y(i,j) = VX, \tag{13.76}$$

where $V = (v_{ij})$ is the $n \times n$ matrix given by Eq. (13.33).

By the same way, we get

$$\sum_{i<j} w_{ij}\varphi_{ij}(T) Y(i,j) = C(X,T)X, \tag{13.77}$$

where $C(X,T) = (c_{ij}(X,T))$ is the $n \times n$ matrix valued function given by

$$c_{ij}(X,T) = -w_{ij} t_{ij} s_{ij}(X) \text{ if } i \neq j, -\sum_{k=1, k\neq i}^{n} c_{ik} \text{ if } i = j, \tag{13.78}$$

Finally, we have

$$Y = \frac{1}{2}(V + C(X,T))X. \tag{13.79}$$

We are now proving that the sequence $\{Y^{(k)}\}$ defined by Eq. (13.75) is contained in $\mathscr{A}^{\perp}$. According to Eq. (13.79) and the fact that the range $V \subset \mathscr{A}^{\perp}$ (Proposition 13.2) it suffices to prove that the range $C(X,T) \subset \mathscr{A}^{\perp}$, for a given couple $(X,T)$.

**Proposition 13.9.** *Let $T = (t_{ij})$ be a given matrix in the set $\mathscr{C}$ given by Eq. (13.72). Then the function $\xi_T$ defined on $\mathscr{M}_{n,p}(\mathrm{I\!R})$ by*

$$\xi_T(X) := \sum_{i<j} w_{ij} t_{ij} d_{ij}(X) \tag{13.80}$$

*is a seminorm such that $C(X,T)X \in \partial\xi_T(X)$ whose kernel contains the subspace $\mathscr{A}$ and*

$$\xi_T(X) = \langle X, C(X,T)X\rangle, \forall X \in \mathscr{M}_{n,p}(\mathrm{I\!R}). \tag{13.81}$$

*Furthermore, for every $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$, the symmetric matrix $C(X,T)$ is positive semidefinite and range $C(X,T) \subset \mathscr{A}^{\perp}$.*

*Proof.* It is clear that the function $\xi_T$ is a seminorm on $\mathscr{M}_{n,p}(\mathrm{I\!R})$ whose kernel contains the subspace $\mathscr{A}$. Like the matrix $V$, the positive semidefiniteness of the matrix $C(X,T)$, for every $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$, comes from [47]. The preceding computations leading to formulations (13.77) and (13.78) show that for a given matrix $T = (t_{ij}) \in \mathscr{C}$, $C(X,T)X \in \partial\xi_T(X)$ for every $X \in \mathscr{M}_{n,p}(\mathrm{I\!R})$. The extended Euler's relation (13.80) for positively homogeneous convex functions is then immediate. Finally, since $C(X,T)e = 0$, according to Lemma 13.1, $C(X,T) \subset \mathscr{A}^{\perp}$. □

*Remark 13.12.* Under the assumptions (a1),(a2), and (a3), one can prove, as in Lemma 13.1, that $\mathscr{A}^{\perp} =$ range $C(X,T)$.

### Computing $\partial\mathscr{G}^*$

The calculation of $\partial\mathscr{G}^*(Y^{(k)}, Z^{(k)})$ consists of solving the next problem:

$$\min\{\mathscr{G}(X,T) - \langle (Y^{(k)}, Z^{(k)}), (X,T)\rangle : (X,T) \in \mathscr{M}_{n,p}(\mathrm{I\!R}) \times \mathscr{M}_{n,n}(\mathrm{I\!R})\}.$$

Since the functions $\mathscr{G}$ are separable in its variables, the last problem can be decomposed into the following two problems:

$$\min\left\{\frac{1}{2}\eta^2(X) - \langle Y^{(k)}, X\rangle : X \in \mathscr{M}_{n,p}(\mathrm{I\!R})\right\}, \tag{13.82}$$

$$\min\left\{\mu(T) - \langle Z^{(k)}, T\rangle : T \in \mathscr{T}\right\}. \tag{13.83}$$

According to Proposition 13.2, solving Problem (13.82) is reduced to computing the solution of the nonsingular linear system (13.84).

$$\left(V+\frac{1}{n}ee^T\right)X = Y. \tag{13.84}$$

Expressing the objective function of Problem (13.83) in the form

$$\mu(T) - \langle Z^{(k)}\ ,T\rangle = \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}t_{ij}^2 - \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(d_{ij}(X^{(k)})+t_{ij}{}^{(k)})t_{ij},$$

and using the following result (whose proof is easy), we can compute explicitly an optimal solution to Problem (13.83).

**Lemma 13.5.** *Let $M$ be a subset of $\{1,\ldots,m\}$. Let $a_i, b_i, d_i$, and $f_i$ be real numbers such that $a_i \le b_i$ and $d_i > 0$ for $i \in M$. Then the solution set of the following convex program in $\mathbb{R}^m$*

$$\min\left\{\frac{1}{2}\sum_{i\in M} d_i u_i^2 - \sum_{i\in M} f_i u_i : a_i \le u_i \le b_i,\ i\in M\right\}$$

*is $\{u : u_i = \frac{1}{d_i}f_i$ if $d_i a_i \le f_i \le d_i b_i$, $a_i$ if $f_i \le d_i a_i$, and $b_i$ if $f_i \ge d_i b_i\}$.*

From Lemma 13.5 it follows that $T^* = (t^*_{ij})$ is a solution to problem (13.83) if and only if $t^*_{ij}$ is arbitrary for $(i,j)\notin\mathscr{S}$ and

$$t^*_{ij} = \begin{cases} \frac{1}{2}(d_{ij}(X^{(k)})+t_{ij}^{(k)}) & \text{if} \quad l_{ij} \leqslant \frac{1}{2}(d_{ij}(X^{(k)})+t_{ij}^{(k)}) \leqslant u_{ij}, (i,j)\in\mathscr{S}, \\ l_{ij} & \text{if} \quad \frac{1}{2}(d_{ij}(X^{(k)})+t_{ij}^{(k)}) < l_{ij}, (i,j)\in\mathscr{S}, \\ u_{ij} & \text{if} \quad \frac{1}{2}(d_{ij}(X^{(k)})+t_{ij}^{(k)}) > u_{ij}, (i,j)\in\mathscr{S}. \end{cases} \tag{13.85}$$

Only the components $t_{ij}^{(k+1)}$ with $(i,j)\in\mathscr{S}$ actually intervene in problem (13.73), so we set $T^{(k+1)} = (t_{ij}^{(k+1)})$ as follows:

$$t_{ij}^{(k+1)} = 0 \text{ for } (i,j)\notin\mathscr{S} \text{ and } t_{ij}^{(k+1)} = t^*_{ij} \ \ for\ (i,j)\in\mathscr{S}. \tag{13.86}$$

From the above displayed calculations, we can now provide the description of the DCA applied to Problem (13.74).

### Description of the DCA for Solving Problem (13.74)

**GDCA1** *(DCA applied to Problem (13.74)).*
*The sequence $\{(X^{(k)},T^{(k)})\}$ with $X^{(k)}\in\mathscr{A}^\perp$ is generated as follows:*
Let $\tau > 0$ and $X^{(0)}\in\mathscr{A}^\perp\setminus\{0\}, T^{(0)}\in\mathscr{T}$ be given.

For $k = 0, 1, \dots$ until

$$(1-\tau)l_{ij} \leq \|X_i^{(k)} - X_j^{(k)}\| \leq (1+\tau)u_{ij}, \text{ for all } (i,j) \in \mathscr{S} \tag{13.87}$$

determine $T^{(k+1)}$ following Eq. (13.85), and solve the following nonsingular linear system to obtain $X^{(k+1)}$

$$\left(V + \frac{1}{n}ee^T\right)X = \frac{1}{2}(V + C(X^{(k)}, T^{(k)}))X^{(k)}. \tag{13.88}$$

*Remark 13.13.* (i) According to Proposition 13.8, if (13.87) occurs, then $F(X^{(k)}, \overline{T} = (\bar{t}_{ij})) = 0$, with $\bar{t}_{ij} = \|X_i^{(k)T} - X_j^{(k)T}\|$.

(ii) To solve the linear system with positive definite constant matrix (13.88), one can use only one Cholesky factorization. So algorithm **GDCA1** requires only matrix-vector products and one Cholesky factorization.

### 13.4.3 A Smooth DC Formulation

We now consider a new DC formulation whose objective function is differentiable:

$$(\mathrm{GDP}_2)\ 0 = \inf\left\{\frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\|x^i - x^j\|^2 - t_{ij}^2)^2 : x^1, \dots, x^n \in \mathbb{R}^3,\ l_{ij} \leqslant t_{ij} \leqslant u_{ij}, (i,j) \in \mathscr{S}^w\right\}.$$

Besides the same advantages indicated above for (GDP$_1$), the immediate advantages of this formulation are the following: the objective function is smooth, its Gaussian transform can be computed easily, and from it a *nice* DC decomposition is directly derived.

If the matrix spaces are useful for various calculations of subgradients in DCA for the nonsmooth formulation (GDP$_1$), the use of vector spaces is more convenient for DCA applied on (GDP$_2$). Since only the pairs $(i,j) \in \mathscr{S}^w$ are involved in the constraints, we denote by $ind(i,j)$ the indices of the pairs $(i,j)$ in the set $\mathscr{S}^w$ and consider only the elements $t_{ij}$ with $(i,j) \in \mathscr{S}^w$. Let $T$ be the column-vector in $\mathbb{R}^s$ ($s = |\mathscr{S}^w|$, the cardinality of the set $\mathscr{S}^w$) defined by these elements: $T = \left(T_1, \dots, T_{ind(i,j)}, \dots, T_s\right)^T \in \mathbb{R}^s$ with $T_{ind(i,j)} = t_{ij}$ for $(i,j) \in \mathscr{S}^w$. Let also $X$ be the column-vector in $\mathbb{R}^{p.n}$ that contains $n$ elements $x^j \in \mathbb{R}^p$, $j = 1, \dots, n$, say $X = \left(x^1, \dots x^n\right)^T$. Denote by $c(i,j)$ the indices in $X$ of $i$th component of the vector $x^j$ in $\mathbb{R}^p$. Then $c(i,j) = 3(j-1) + i$ for $i = 1, \dots 3$, $j = 1, \dots, n$, and $X_{c(i,j)} = (x^j)_i$. Define now the pairwise functions $\theta_{ij} : \mathbb{R}^{pn} \longmapsto \mathbb{R}$ and $\pi_{ij} : \mathbb{R}^s \longmapsto \mathbb{R}$ by $\theta_{ij}(X) := \|x^i - x^j\|^2, \pi_{ij}(T) := T_{ind(i,j)}^2 = t_{ij}^2$. (GDP$_2$) can be rewritten in the next form which is also denoted (GDP$_2$)

$$(\mathrm{GDP}_2)\ \ 0 = \inf\left\{\mathscr{F}_2(X,T) := \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\theta_{ij}(X) - \pi_{ij}(T))^2 : (X,T) \in \mathbb{R}^{p.n} \times \mathscr{T}\right\}.$$

The results in Proposition 13.8 are also valid to (GDP$_2$).

We first prove that problem (GDP$_2$) is a DC program. Since

$$-2\pi_{ij}(T)\theta_{ij}(X) = -(\theta_{ij}(X)+\pi_{ij}(T))^2 + \theta_{ij}^2(X)+\pi_{ij}^2(T),$$

the objective function of (GDP$_2$) can be expressed as

$$\mathscr{F}_2(X,T) = \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\theta_{ij}^2(X) + \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\pi_{ij}^2(T) - \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\theta_{ij}(X)+\pi_{ij}(T))^2. \tag{13.89}$$

The function $\mathscr{L}$ defined by

$$\mathscr{L}(X,T) := \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\theta_{ij}^2(X) + \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\,\pi_{ij}^2(T) \tag{13.90}$$

is finite and convex on $\mathrm{I\!R}^{p.n}\times\mathrm{I\!R}^s$. On the other hand, since for every $(i,j)\in\mathscr{S}^w$, the function: $(X,T)\to\theta_{ij}(X)+\pi_{ij}(T)$ is finite, convex, and nonnegative, the function $\mathscr{K}$ given by

$$\mathscr{K}(X,T) := \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}(\theta_{ij}(X)+\pi_{ij}(T))^2 \tag{13.91}$$

is convex, too. So $\mathscr{F}_2 = \mathscr{L} - \mathscr{K}$ is a DC function.

Then (GDP$_2$) is a DC program of the standard form

$$\inf\{\mathscr{P}(X,T)-\mathscr{K}(X,T) : (X,T)\in\mathrm{I\!R}^{p.n}\times\mathrm{I\!R}^s\} \tag{13.92}$$

$$\text{with}\quad \mathscr{P}(X,T) = \mathscr{L}(X,T) + \chi_{\mathrm{I\!R}^{p.n}\times\mathscr{T}}(X,T). \tag{13.93}$$

## 13.4.4 Solving (GDP$_2$) by DCA

For constructing a DCA scheme applied to (GDP$_2$) we have to compute the subdifferentials of the functions $\mathscr{K}$ and $\mathscr{P}^*$.

### Computing $\partial\mathscr{K}$

By the very definition, the function $\mathscr{K}$ is convex on $\mathrm{I\!R}^{p.n}\times\mathrm{I\!R}^s$ and then

$$\partial\mathscr{K}(X,T) = \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\theta_{ij}(X)+\pi_{ij}(T)][\partial\theta_{ij}(X)\times\{0\}+\{0\}\times\partial\pi_{ij}(T)].$$

So computing $\partial\mathscr{K}$ amounts to compute $\partial\theta_{ij}$ and $\partial\pi_{ij}$.

Clearly that $\pi_{ij}$ is differentiable:

$$\pi_{ij}(T) = T^2_{ind(i,j)} = \langle T, E_{ind(i,j)}\rangle^2_{\mathrm{IR}^s},$$

with $E_{ind(i,j)} = (0,\dots,1_{ind(i,j)},\dots 0)^T \in \mathrm{IR}^s$, ($E_k \in \mathrm{IR}^s$ is the unit vector with value one in the $k$th component and zero otherwise). Hence $\nabla\pi_{ij}(T) = 2T_{ind(i,j)}E_{ind(i,j)}$.

Similarly to the computation of $\partial\theta_{ij}$ in Sect. 13.3 we have

$$\partial\theta_{ij}(X) = Q_{ij}\partial(\|.\|^2)(Q_{ij}^T X) = 2Q_{ij}(x^i - x^j),$$

where $Q_{ij}$ is the $(p.n \times p)-$ matrix defined by

$$Q_{ij} = (e_{c(1,i)} - e_{c(1,j)}, \dots, e_{c(p,i)} - e_{c(p,j)}). \tag{13.94}$$

So $\theta_{ij}$ is also differentiable, and $\nabla\theta_{ij}(X) = 2Q_{ij}(x^i - x^j)$. Since both $\pi_{ij}$ and $\theta_{ij}$ are differentiable, $\mathscr{K}$ is differentiable too,

$$\nabla\mathscr{K}(X,T) = \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\theta_{ij}(X) + \pi_{ij}(T)][\nabla\theta_{ij}(X)\times\{0\} + \{0\}\times\nabla\pi_{ij}(T)] \tag{13.95}$$

$$= \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\theta_{ij}(X) + \pi_{ij}(T)][2Q_{ij}(x^i - x^j)\times\{0\} + \{0\}\times 2T_{ind(i,j)}E_{ind(i,j)}]$$

So $\nabla\mathscr{K}(X,T) = (Y,W)$ with

$$Y = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\theta_{ij}(X) + \pi_{ij}(T)]Q_{ij}(x^i - x^j), \tag{13.96}$$

$$W = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\theta_{ij}(X) + \pi_{ij}(T)]T_{ind(i,j)}E_{ind(i,j)}. \tag{13.97}$$

$$\text{or again } W = (W_{ind(i,j)})_{(i,j)\in\mathscr{S}^w}, \quad W_{ind(i,j)} = w_{ij}[\theta_{ij}(X) + \pi_{ij}(T)]T_{ind(i,j)}.$$

### Computing $\partial\mathscr{P}^*$

The calculation of $\partial\mathscr{P}^*(Y^k, W^k)$ consists of solving the next problem

$$\min\left\{\mathscr{P}(X,T) - \langle (X,T),(Y^{(k)},Z^{(k)})\rangle : (X,T)\in\mathrm{IR}^{p.n}\times\mathrm{IR}^s\right\}.$$

Observing that $\mathscr{P}$ is separable function:

$$\mathscr{P}(X,T) = \mathfrak{f}(X) + \varPi(T) + \chi_{\mathscr{T}}(T)$$

$$\text{with} \quad \mathfrak{f}(X) := \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\theta^2_{ij}(X), \varPi(T) := \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\ \pi^2_{ij}(T),$$

we can compute separately $X^{(k+1)}$ and $T^{(k+1)}$ by solving the following two problems:

$$\min\left\{\mathfrak{f}(X)-\langle Y^{(k)},X\rangle : X\in\mathrm{I\!R}^{p.n}\right\}, \tag{13.98}$$

$$\min\left\{\Pi(T)-\langle Z^{(k)},T\rangle : T\in\mathscr{T}\ \right\}. \tag{13.99}$$

Similarly to the solution of problem (13.83), a solution $T^{*}=(T^{*}_{ind(i,j)})_{(i,j)\in\mathscr{S}^w}$ of problem (13.99) can be explicitly determined as

$$T^{*}_{ind(i,j)}=\begin{cases}\sqrt[3]{\dfrac{Z^{(k)}_{ind(i,j)}}{2w_{ij}}} & \text{if } l_{ij}\le\sqrt[3]{\dfrac{Z^{(k)}_{ind(i,j)}}{2w_{ij}}}\le u_{ij},\\ l_{ij} & \text{if } \sqrt[3]{\dfrac{Z^{(k)}_{ind(i,j)}}{2w_{ij}}}<l_{ij},\\ u_{ij} & \text{if } \sqrt[3]{\dfrac{Z^{(k)}_{ind(i,j)}}{2w_{ij}}}>u_{ij}.\end{cases} \tag{13.100}$$

For solving the convex program (13.98), several methods in convex programming are available. The universality of the DCA suggests us to use it more one time in this work to solve problem (13.98). With a suitable choice of DC decomposition for the objective function of Problem (13.98) the corresponding DCA is very simple and not expensive. Indeed, the objective function of Problem (13.98) can be written in the form

$$\begin{aligned}\mathfrak{f}(X)-\langle Y^{(k)},X\rangle &:= \Lambda(X)-\Psi(X),\\ \Lambda(X) &:= \frac{1}{2}\rho\|X\|^2-\langle Y^{(k)},X\rangle,\\ \Psi(X) &= \frac{1}{2}\rho\|X\|^2-\mathfrak{f}(X),\end{aligned}$$

where $\rho>0$ is chosen such that the function $\Psi$ is convex. Moreover, the solution set of Problem (13.98) can be bounded in the set $\bar{\Omega}$ defined by

$$\bar{\Omega}:=\left\{X\in\mathscr{A}^{\perp}:\sum_{i<j}\|x^i-x^j\|^2\le\sum_{i<j}u^2_{ij}\right\},$$

where the upper bounds $u_{ij}$ of distances $\|x^i-x^j\|^2$ for $(i,j)\notin\mathscr{S}^w$ are computed by using the relationships $u_{ij}=\min(u_{ij},u_{ik}+u_{kj})$. Since $\sum_{i<j}\|x^i-x^j\|^2=n\|X\|^2$ for all $X=(x^1,x^2,\ldots x^n)^T\in\mathscr{A}^{\perp}$, $\bar{\Omega}=\{X\in\mathscr{A}^{\perp}:\|X\|\le r:=\sqrt{\frac{1}{n}\sum_{i<j}u^2_{ij}}\}$. Then we can reformulate problem (13.98) in the form

$$\min\left\{\mathfrak{f}(X)-\langle Y^{(k)},X\rangle : X\in\bar{\Omega}\right\}. \tag{13.101}$$

Hence the sequence $\{X^{(k)}\}$ defined from Problem (13.101) is well defined and bounded.

We can now rewrite Problem (13.98) as DC program

$$\min\{\chi_{\bar{\Omega}}(X)+\Lambda(X)-\Psi(X) : X \in \mathrm{IR}^{p.n}\}. \tag{13.102}$$

By the very definition of $\mathfrak{f}$, this function is differentiable, and

$$\nabla\mathfrak{f}(X) = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}\theta_{ij}(X)\nabla\theta_{ij}(X) = \sum_{(i,j)\in\mathscr{S}^w} 2w_{ij}\theta_{ij}(X)Q_{ij}Q_{ij}^T X.$$

So $\Psi$ is differentiable too, and $\nabla\Psi(X) = \rho X - \nabla\mathfrak{f}(X)$. Then DCA applied to Problem (13.102) consists of generating the sequence $\{\mathrm{X}^{(l)}\}$ such that $\mathrm{X}^{(l+1)}$ solves

$$\min\left\{\frac{1}{2}\rho\|X\|^2 - \langle Y^{(k)}, X\rangle - \langle \rho X^{(l)} - \vartheta^{(l)}), X\rangle : X \in \bar{\Omega}\right\}$$

where $\vartheta^{(l)} = \nabla\mathfrak{f}(X^{(l)})$. In other words, $\mathrm{X}^{(l+1)}$ verifies

$$X^{(l+1)} = Proj_{\bar{\Omega}}(X^{(l)} - \frac{1}{\rho}(\nabla\mathfrak{f}(X^{(l)} - Y^{(k)})).$$

```
Algorithm Calx (DCA applied to Problem (13.98))
    Choose $\mathrm{X}^{(0)} \in \mathrm{IR}^{p.n}$, set $l := 0$.
    Repeat $X^{(l+1)} = Proj_{\bar{\Omega}}(X^{(l)} - \frac{1}{\rho}(\nabla\mathfrak{f}(X^{(l)} - Y^{(k)}))$
    until    $\|X^{(l+1)} - X^{(l)}\| \le \varepsilon$.
```

Finally DCA applied to Problem (13.92) can be described as follows.

**Algorithm GDCA2** *(DCA applied to (GDP$_2$) with DC decomposition (13.92).*
Let $\tau > 0$, $\varepsilon > 0$, and $X^{(0)} \in \mathrm{IR}^{p.n}$ ,$T^{(0)} \in \mathscr{T}$ be given, $k = 0$.
Step k1: Set $Y^{(k)} = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\left\|x^{i^{(k)}} - x^{j^{(k)}}\right\|^2 + T^{2(k)}_{ind(i,j)})]Q_{ij}(x^{i^{(k)}} - x^{j^{(k)}})$.
Step k2: Set $T^{(k+1)} = (T_{ind(i,j)})^{(k+1)}_{(i,j)\in\mathscr{S}^w}$ as follows:

$$T^{(k+1)}_{ind(i,j)} = \begin{cases} \gamma := \sqrt[3]{\frac{1}{2}\left(\left\|x^{i^{(k)}} - x^{j^{(k)}}\right\|^2 + T^{2(k)}_{ind(i,j)}\right) T^{(k)}_{ind(i,j)}} & \text{if } \; l_{ij} \le \gamma \le u_{ij}, \\ l_{ij} & \text{if } \quad \gamma < l_{ij}, \\ u_{ij} & \text{if } \quad \gamma > u_{ij}. \end{cases} \tag{13.103}$$

Step k3: Apply algorithm **calx** to Problem (13.98) to obtain $\mathrm{X}^{(k+1)}$, more precisely

```
    Set $l := 0$, $\mathrm{X}^{(k,l)} := \mathrm{X}^{(k)}$
    Repeat $X^{(k,l+1)} = Proj_{\bar{\Omega}}(X^{(k,l)} - \frac{1}{\rho}(\nabla\mathfrak{f}(X^{(k,l)} - Y^{(k)}))$
        until    $\|X^{(k,l+1)} - X^{(k,l)}\| \le \varepsilon$.
        Set      $X^{(k+1)} := X^{(k,l+1)}$
```

Step k4: **If**

$$(1-\tau)l_{ij} \le \left\|x^{i^{(k+1)}} - x^{j^{(k+1)}}\right\| \le (1+\tau)u_{ij}, \text{ for all } (i,j) \in \mathscr{S}^w \qquad (13.104)$$

**then** STOP, $(\mathrm{X}^{(k+1)}, T^{(k+1)})$ is an optimal solution to **(GDP$_2$)**, the set of positions $(x^{1^{(k+1)}}, \ldots, x^{n^{(k+1)}})$ is a solution to Problem (13.3))

**else** set $k = k+1$ and go to step k1.

*Remark 13.14.* (i) Algorithm GDCA2 is not expensive. By the very special structure of the matrix $Q_{ij}$, the computationals run in fact on vectors, and are all explicit.

(ii) Ones can also stop algorithm GDCA2 when either $\mathscr{F}_2(X^{(k)}, T^{(k)}) - \mathscr{F}_2(X^{(k+1)}, T^{(k+1)}) \le \varepsilon$ or $\|(X^{(k+1)}, T^{(k+1)}) - (X^{(k)}, T^{(k)})\| \le \varepsilon$. In such a case if Eq. (13.87) is not hold, then $(x^{1^{(k+1)}}, \ldots, x^{n^{(k+1)}})$ is not an optimal solution to Problem (13.3).

## 13.5 Finding a Good Starting Point for DCA

We present in this section some efficient techniques for finding a good starting point of DCA.

### 13.5.1 *An Optimal Solution of the Complete Distance Matrix: The Two Phase Algorithm*

We first *complete* the matrix of distances by using the *shortest path* between all pairs of atoms and then apply the DCA to the new problem where all pairwise "distances" (rather dissimilarities) are known. The idea of this technique comes from two facts:

- When all pairwise distances are known, the DCA applied to (EDP)-Eq. (13.5) is very simple. Although the DCA is not a polynomial time algorithm, it works very well in practice, because it has an explicit form and requires only matrix-vector products (see algorithm **DCA1bis**).
- In the general case where only a small subset of distances (or bounded distances in the general distance geometry problem) is known one can approximate a solution of the distance geometry problem by using a dense set of constraints which is extended from the given distances and then work with this set.

An approximate distance matrix can be determined by several ways as in the *embed* algorithm [3]. Indeed, by taking $u_{ij} = l_{ij} = \delta_{ij}$ for $(i,j) \in \mathscr{S}$, and using the relationships $u_{ij} = \min(u_{ij}, u_{ik} + u_{kj}), \quad l_{ij} = \max(l_{ij}, l_{ik} - u_{kj}, l_{jk} - u_{ki}\}$ one obtains a full set of bounds $[l_{ij}, u_{ij}]$, and then one can take $\tilde{\delta}_{ij} \in [l_{ij}, u_{ij}]$. In our algorithm, we attempt to use a simpler procedure for computing the approximate matrix $\tilde{\Delta} = (\tilde{\delta}_{ij})$: the length of the shortest paths (within the connected graph $G(N, \mathscr{S})$) between atom $i$ and atom $j$. Its direct calculation does not require computing both the bounds $l_{ij}$ and $u_{ij}$ and so is less expensive. From our experiments we observe that this choice of $\tilde{\Delta}$ for DCA is the most efficient, in comparing with the choice $\tilde{\delta}_{ij} = 0.5(u_{ij} + l_{ij})$ for $(i, j) \notin \mathscr{S}$ and the conditional choice $\tilde{\delta}_{ij} = 0.5u_{ij} + l_{ij}$ for $(i, j) \notin \mathscr{S}$ if $l_{ij} \leq 0.5u_{ij}$.

A variant of the DCA, called the two-phase algorithm, is composed of two phases. In Phase 1 we *complete* the matrix of distances by using the *shortest path* between all pairs of atoms and then apply the DCA to the new problem where all pairwise "distances" (rather dissimilarities) are known. In Phase 2 we solve the original problem by applying the DCA from the point obtained by Phase 1.

This two-phase algorithm has some advantages. First, we work with both (*dense* and *sparse*) sets of constraints. The use of a complete matrix which is an *approximate* distance matrix aims at finding a good initial point for the DCA applied to the original problem: such a SP is computed by DCA applied to the resulting problem (13.1) with the complete dissimilarity matrix. By contrast, the existing methods work only on either a full set of constraints (see, e.g., [3]) or a sparse set of constraints [33, 48].

**The Two Phase Algorithm**

*Phase 1. Find an initial point for Phase 2.*

Step 1. *Determine an approximate distance matrix* $\tilde{\Delta} = (\tilde{\delta}_{ij})$.

**For** $i = 1, \ldots, n, j = i + 1, \ldots, n$, compute $\tilde{\delta}_{ij}$, the length of the shortest path between $i$ and $j$, within the connected graph $G(N, \mathscr{S})$.

Step 1. *Solve the problem*

$$\min\left\{\frac{1}{2}\sum_{i<j} c(\tilde{\delta}_{ij} - \|X_i^T - X_j^T\|)^2 : X \in \mathscr{M}_{n,p}(\mathrm{IR})\right\} \tag{13.105}$$

by applying either **DCA1bis** to problem (13.22) or **DCA2bis** to problem (13.47), where $w_{ij}$ and $\delta_{ij}$ are replaced by $c$ and $\tilde{\delta}_{ij}$, respectively, to obtain a point denoted by $\tilde{X}$.

*Phase 2. Solve the original problem (EDP*$_1$*)* (resp. (GDP$_1$) or (GDP$_2$)) by applying either **DCA1, DCA1r** to problem (13.22) or **DCA2, DCA2r** to problem (13.47) (resp. either **GDCA1** to problem (GDP$_1$) or **GDCA2** to problem (GDP$_2$)) from the point $\tilde{X}$.

### *13.5.2 A Spaning Tree Procedure*

As an alternative of Phase 1, we also propose the SP procedure which is an adaptation of the *inexpensive approach using spanning trees* (algorithm Struct of Moré–Wu [33]) to compute acceptable starting points. In our experiments the SP procedure is more (resp. less) efficient than Phase 1 when the number of distance constraints is small (resp. large).

For the general (resp. exact) distance geometry problem we take $T^0$ by $T^0_{ind(i,j)} = \frac{u_{ij}+l_{ij}}{2}$ (resp. $T^0_{ind(i,j)} = \delta_{ij}$) for all $(i,j) \in \mathscr{S}^w$ and then choose $X^0$ so that $\left\| x^{i(0)} - x^{j(0)} \right\| = T^0_{ind(i,j)}$ for at least $n-1$ pairs $(i,j) \in \mathscr{S}^w$:

**Procedure SP**: Let $(i_0, j_0) \in \mathscr{S}^w$ such that $T^0_{ind(i_0,j_0)} = \max \left\{ T^0_{ind(i,j)} : (i,j) \in \mathscr{S}^w \right\}$. Let $x^{i_0} = (0,0,0)^T$ and randomly generate $x^{j_0}$ such that $\left\| x^{i_0} - x^{j_0} \right\| = T^0_{ind(i,j)}$. Set $\mathscr{M} := \{i_0, j_0\}, k := j_0$.

```
do while $|\mathscr{M}| < n$
    Choose $(k, j_k) \in \mathscr{S}$ such that $T^0_{ind(k,j_k)} = \max_j \left\{ T^0_{ind(k,j)} : (k,j) \in \mathscr{S}^w \right\}$. Randomly generate $x^{j_k}$ such that $\left\| x^k - x^{j_k} \right\| = T^0_{ind(k,j_k)}$.
    Set $\mathscr{M} := \mathscr{M} \cup \{j_k\}, k := j_k$
end do
```

This procedure is an amelioration of algorithm Struct given in [33]: it provides a point satisfying the *largest* distance constraint in $\mathscr{S}^w$ and the *largest* constraint $(k, j_k)$ among the pairs $(k,j) \in \mathscr{S}^w$ for a given $k$, while the point generated by algorithm Struct satisfies some $n-1$ distance constraints. In our computational experiments this procedure is better than algorithm Struct for finding a SP to the DCA.

### 13.5.3 A Continuation Approach via Gaussian Transformation

A disadvantage of the two-phase algorithm is that, although the strategy of Phase 1 is quite suitable for DCA to reach global solutions to the distance geometry problem, it is quite expensive (the running time of Phase 1 is equal to that of Phase 2). To get around this drawback, more precisely, to find a good SP of DCA without using Phase 1, we develop a combined DCA-smoothing technique.This technique can be used for the smooth DC formulation (GDP$_2$) (recall that this formulation is also valid to the exact distance geometry problem by taking $\delta_{ij} = l_{ij} = u_{ij}$). Given a sequence of smoothing parameters $\lambda_0 > \lambda_1 > \ldots > \lambda_{step} = 0$, our continuation algorithm uses the DCA to compute a minimizer $(X^{q+1}, T^{q+1})$ of $\langle \mathscr{F}_2 \rangle_{\lambda_q}$ (the Gaussian transformation of $\mathscr{F}_2$) with the previous one $(X^q, T^q)$ as the starting point. The algorithm generates then a sequence $\{X^q, T^q\}$, and $(X^{step+1}, T^{step+1})$ is a candidate for a global minimizer of (GDP$_2$).

The idea of the use of the continuation approach via the Gaussian transform for distance geometry problems is not new: Moré and Wu [31] proposed an algorithm for solving the exact distance geometry problem (13.4) by the Gaussian smoothing technique via the trust region method. Computational experiments with up to 648 variables ($n = 216$) in [31] proved that the continuation method is more reliable and efficient than the multistart approach, a standard procedure for finding the global minimizer to Eq. (13.4). However, we observe that the trust region method applied to the sequence of subproblems in the continuation approach is expensive and thus may not be efficient in the large-scale setting. For the general distance geometry problem (13.3), using the formulation (EDP)-Eq. (13.6), Moré and Wu [33] introduced the dgsol algorithm based on the Gauss–Hermite transform and the variable-metric limited-memory code MINPACK-2. Computational experiments on protein fragments with 100 and 200 atoms from the PDB data bank showed that the dgsol code is also more efficient than the multistart algorithm.

Our continuation approach for the distance geometry problems is completely different from that of Moré–Wu's work: with the formulation (GDP$_2$) we get exactly and explicitly the Gaussian transformed function while with the formulation (EDP)-Eq. (13.6), Moré and Wu [33] were able only to get an approximation to the Gaussian transformed function, say the Gauss–Hermite approximation which is quite complicated and inconvenient to use DCA. On the other hand, our optimization method for the transformed problem is based on D.C. programming and DCA while Moré and Wu used the trust region method in [31] (for the exact distance geometry problem) and the limited-memory code in [33].

Our approach has several advantages. First, by using the continuation approach which traces the minimizers of the smooth function back to the original function, the Phase 1 in [22, 23] is no longer needed. Second, since the DCA applied to Gaussian transformed problems works only on vector products and does not require the Cholesky factorization, it is efficient in large-scale problems and faster than the trust region approach. Third, we can exploit sparsity of the given distance matrix. This is important because only a small subset of constraints is known in practice.

The Gaussian transform of a function $f : \mathbb{R}^n \to \mathbb{R}$, denoted by $\langle f \rangle_\lambda$, is defined as

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

or again, by the change of variable $y \mapsto x + \lambda u$ :

$$\langle f \rangle_\lambda (x) = \frac{1}{\pi^{n/2}} \int_{\mathbb{R}^n} f(x + \lambda u) \exp\left(- \| u \|^2\right) du.$$

In this section we compute the Gaussian transform of $\mathscr{F}_2(X,T)$, the objective function of (GDP$_2$), by using the basic results on the computational Gaussian transform of functions given in [31]. Let $h_{ij} : \mathbb{R}^p \times \mathbb{R} \mapsto \mathbb{R}$ be the function defined by

$$h_{ij}(x,t) := \left( \|x\|^2 - \frac{1}{2}t^2 \right)^2 . \tag{13.106}$$

Then we can express $\mathscr{F}_2(X,T)$ in the form

$$\mathscr{F}_2(X,T) := \frac{1}{4} \sum_{(i,j)\in\mathscr{S}^w} w_{ij} h_{ij}(x^i - x^j, \sqrt{2}t_{ij}), = \frac{1}{4} \sum_{(i,j)\in\mathscr{S}^w} w_{ij} h_{ij}(\mathscr{P}_{ij}^T(X,T)), \tag{13.107}$$

where $\mathscr{P}_{ij}$ is the $(pn+d)\times(p+1)$-matrix with $\mathscr{P}_{ij}^T = \begin{pmatrix} Q_{ij} & 0_{d\times 1} \\ 0_{1\times np} & q_{ij} \end{pmatrix}$. Here $Q_{ij}$ is the $(pn \times p)-$matrix defined in Eq. (13.94), $\mathrm{q}_{ij}$ is the row-vector in $\mathbb{R}^s$ such that $[q_{ij}]_{ind(i,j)} = \sqrt{2}, [q_{ij}]_k = 0, \forall k \neq ind(i,j)$, and $e_k \in \mathbb{R}^{p.n}$ is the unit vector with one in the $k$th component and zero otherwise.

Let $F_{ij} : \mathbb{R}^{pn} \times \mathbb{R}^s \mapsto \mathbb{R}$ be such that $F_{ij}(X,T) = h_{ij}(\mathscr{P}_{ij}^T(X,T))$. Then we have

$$\langle \mathscr{F}_2 \rangle_\lambda (X,T) = \frac{1}{4} \sum_{(i,j)\in \mathscr{S}^w} w_{ij} \langle F_{ij} \rangle_\lambda (X,T) = \frac{1}{4} \sum_{(i,j)\in \mathscr{S}^w} w_{ij} \langle h_{ij} \rangle_{\sqrt{2}\lambda} (x^i - x^j, \sqrt{2} t_{ij}), \tag{13.108}$$

where the first equality is obtained from the linear property of the operator $\langle F \rangle_\lambda$ (see [31, p. 819]) and the second by applying Theorem 4.1 of [31].

So computing the Gaussian transform of $\mathscr{F}_2$ amounts to computing that of $h_{ij}$ which is quite simple. Since $h(x,t) = \|x\|^4 - \|x\|^2 t^2 + \frac{1}{4} t^4$, the linear property of the operator $\langle h \rangle_\lambda$ implies that

$$\langle h \rangle_\lambda (x,t) = \langle h_1 \rangle_\lambda (x) + \langle h_2 \rangle_\lambda (x,t) + \langle h_3 \rangle_\lambda (t), \tag{13.109}$$

where $h_1(x) := \|x\|^4, h_2(x,t) := -\|x\|^2 t^2, h_3(t) := \frac{1}{4} t^4$. An application of Theorem 3.4 of [31] gives

$$\langle h_1 \rangle_\lambda (x) \; = \; \|x\|^4 + (2+p)\lambda^2 \|x\|^2 + \frac{1}{4} p(p+2)\lambda^4, \tag{13.110}$$

$$\langle h_3 \rangle_\lambda (t) \; = t^4 + 3\lambda^2 t^2 + \frac{3}{4}\lambda^4. \tag{13.111}$$

On the other hand, noting that $h_2(x,t) := -\|x\|^2 t^2 = -\sum_{i=1}^p x_i^2 t^2$ and using the formulation of the Gaussian transform of a decomposable function given in [31, p. 820], we get

$$\langle h_2 \rangle_\lambda (x,t) = -\sum_{i=1}^p \left( x_i^2 + \frac{1}{2}\lambda^2 \right) \left( t^2 + \frac{1}{2}\lambda^2 \right) = - \left( t^2 + \frac{1}{2}\lambda^2 \right) \left( \|x\|^2 + \frac{p}{2}\lambda^2 \right). \tag{13.112}$$

So

$$\langle h \rangle_\lambda (x,t) = h(x,t) + \frac{1}{2}(3+2p)\lambda^2 \|x\|^2 + \frac{1}{4}( \; 3-2p)\lambda^2 t^2 + \frac{1}{16}(3+4p+4p^2)\lambda^4. \tag{13.113}$$

Finally, from Eqs. (13.108) and (13.113), we get the expression of the Gaussian transform of $F$:

**Proposition 13.10.** *If $\mathscr{F}_2 : \mathrm{I\!R}^{pn} \times \mathrm{I\!R}^{nn} \mapsto \mathrm{I\!R}$ is defined by Eqs. (13.107) and (13.106), then*

$$\begin{aligned}&\langle \mathscr{F}_2(X,T)\rangle_\lambda(X,T)\\ &\quad = F(X,T) + \left(\frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left[(3+2p)\lambda^2\|x^i-x^j\|^2+(3-2p)\lambda^2 t_{ij}^2\right]\right)+\gamma,\end{aligned} \tag{13.114}$$

*where $\gamma = \frac{1}{16}(3+4p+4p^2)\lambda^4\sum_{(i,j)\in\mathscr{S}^w} w_{ij}$.*

The main subproblem in our continuation approach to solve (GDP$_2$) is the following:

$$(GDP_\lambda) \qquad \min\{\langle\mathscr{F}_2\rangle_\lambda(X,T) : (X,T)\in \mathrm{I\!R}^{p.n}\times T\}.$$

### DCA for Solving Problem (GDP$_\lambda$)

In a similar way to (GDP$_2$) we get the following DC decomposition of the objective function of $(GDP_\lambda)$ without the constant $\overline{\gamma}$ which is also denoted by $\langle\mathscr{F}_2\rangle_\lambda$:

$$\langle\mathscr{F}_2\rangle_\lambda = \mathscr{L}_\lambda(X,T) - \mathscr{K}_\lambda(X,T),$$

$$\mathscr{L}_\lambda(X,T) := \Theta_\lambda(X) + \Pi(T),\ \Theta_\lambda(X) := \frac{1}{2}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left(\theta_{ij}^2(X)+\frac{9}{4}\lambda^2\theta_{ij}(X)\right),$$

$$\mathscr{K}_\lambda(X,T) := \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left[\,\theta_{ij}(X)+\pi_{ij}(T)]^2+3\lambda^2\pi_{ij}(T)\right]. \tag{13.115}$$

Hence problem (GDP$_\lambda$) can be expressed in the standard form of DC programs

$$\min\{F_\lambda(X,T) := \mathscr{P}_\lambda(X,T) - \mathscr{K}_\lambda(X,T) : (X,T)\in\mathrm{I\!R}^{p.n}\times\mathrm{I\!R}^s\}, \tag{13.116}$$

where $\mathscr{P}_\lambda$ is defined on $\mathrm{I\!R}^{p.n}\times\mathrm{I\!R}^s$ by $\mathscr{P}_\lambda(X,T) = \mathscr{L}_\lambda(X,T) + \chi_{onC}(T)$.
The computation of $\nabla\mathscr{K}_\lambda$ is immediate from $\nabla\mathscr{K}$ : $\nabla\mathscr{K}_\lambda(X,T) = (Y, Z_\lambda)$ with Y defined in Eq. (13.96) and

$$Z_\lambda = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left[(\theta_{ij}(X)+\pi_{ij}(T)\right]T_{ind(i,j)}E_{ind(i,j)} + \frac{3}{2}\lambda^2\sum_{(i,j)\in\mathscr{S}^w} w_{ij}T_{ind(i,j)}E_{ind(i,j)}.$$

The computing of $\partial\mathscr{P}_\lambda^*$ is similar to the one of $\partial\mathscr{P}^*$ where $\Theta(X)$ and $W$ are replaced, respectively, by $\Theta_\lambda$ and $W_\lambda$. Finally DCA applied to (GDP)$_\lambda$ can be describes as follows.

**Algorithm GDCA$\lambda$** *(DCA applied to (GDP)$_\lambda$)*

Let $\tau, \varepsilon_1, \varepsilon_2 > 0$, and $X^{(0)} \in \mathscr{A}^\perp, T^{(0)} \in T$ be given, $k = 0$.

Step k1: Set

$$Y^{(k)} = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}\left[\left\|x^{i^{(k)}} - x^{j^{(k)}}\right\|^2 + T^{2(k)}_{ind(i,j)})\right]Q_{ij}(x^{i^{(k)}} - x^{j^{(k)}}). \tag{13.117}$$

Step k2: Set $T^{(k+1)} = (T_{ind(i,j)})^{(k+1)}_{(i,j)\in\mathscr{S}^w}$ as follows:

$$T^{(k+1)}_{ind(i,j)} = \begin{cases} \overline{\mu} := \sqrt[3]{\frac{1}{2}\left(\left\|x^{i^{(k)}} - x^{j^{(k)}}\right\|^2 + T^{2(k)}_{ind(i,j)} + 3\lambda^2\right) T^{(k)}_{ind(i,j)}} & \text{if } \ l_{ij} \le \overline{\mu} \le u_{ij}, \\ l_{ij} & \text{if } \quad \overline{\mu} < l_{ij}, \\ u_{ij} & \text{if } \quad \overline{\mu} > u_{ij}, \end{cases} \tag{13.118}$$

Step k3: Set $l := 0$, $X^{(k,l)} := X^{(k)}$

**Repeat**

$$X^{(k,l+1)} = \begin{cases} \varkappa := X^{(k,l)} - \frac{1}{\rho}\left(\nabla(X^{(k,l)} - Y^{(k)}\right) & \text{if } \|\xi\| \le r \\ (r/\|\varkappa\|)\varkappa & \text{if } \|\xi\| > r \end{cases} \tag{13.119}$$

**until** $\|X^{(k,l+1)} - X^{(k,l)}\| \le \varepsilon_1$.

Set X$^{(k+1)}$ :=X$^{(k,l+1)}$

If either $|\langle\mathscr{F}_2\rangle_\lambda(X^{(k)}, T^{(k)}) - \langle\mathscr{F}_2\rangle_\lambda(X^{(k+1)}, T^{(k+1)})| \le \varepsilon_2\langle\mathscr{F}_2\rangle_\lambda(X^{(k+1)}, T^{(k+1)})$ or

$$(1-\tau)l_{ij} \le \left\|x^{i^{(k+1)}} - x^{j^{(k+1)}}\right\| \le u_{ij}(1+\tau), \ \text{ for all } (i,j) \in \mathscr{S}^w \tag{13.120}$$

then STOP, (X$^{(k+1)}, T^{(k+1)}$) is an solution to (GDP)$_\lambda$
else set $k = k + 1$ and go to step k1.
The continuation algorithm for solving (GDP)$_2$ is summarized as follows.

**The Main Algorithm CGDCA**

Choose $T^0$ by $T^0_{ind(i,j)} = \frac{u_{ij}+l_{ij}}{2}$ for $(i,j)\in \mathscr{S}^w$ and determine $X^0$ by Procedure starting point. Set $q := 0$.

**Do while** $(q \le step)$

Compute $(X^{q+1}, T^{q+1})$, a solution to problem (GDGP$_{\lambda_q}$) by applying algorithm GDCA$\lambda$ (and/or algorithm GDCA2 when $\lambda_q = 0$) from the starting point $(X^q, T^q)$.

Increase $q$ by 1.

**end do**

*Remark 13.15.* To solve the exact geometry problem (13.1) the variable T is not needed. So in algorithm GDCA$\lambda$, the step k2 is eliminated, and the vector $Y^{(k)}$ at the step k1 is defined by $Y^{(k)} = \sum_{(i,j)\in\mathscr{S}^w} w_{ij}[\|x^{i^{(k)}} - x^{j^{(k)}}\|^2 + \delta^2_{ij}]Q_{ij}(x^{i^{(k)}} - x^{j^{(k)}})$.

## 13.6 Computational Experiments

Our algorithms are coded in FORTRAN 77 with double precision and run on an SGI Origin 2000 multiprocessor with IRIX system.

### *13.6.1 Computational Experiments of the Exact Distance Geometry Problem (EDP)*

We have tested our code on three sets of data: the first one is the artificial data from Moré–Wu [31], the second is derived from proteins in the PDB data bank, and the third is generated by Hendrickson [13, 14].

The purpose of the experiments is threefold. The first is to show that the DCA can efficiently solve large-scale distance geometry problems ($\text{EDP}_1$). We consider molecules containing at most $4,096$ atoms ($12,288$ variables) in the artificial data and at most $4,189$ atoms in the PDB data.

The second is to study the effect of starting points for the DCA applied to the main problem ($\text{EDP}_1$). We compare the efficiency of the two-phase algorithm EDCA and algorithm SDCA, i.e., DCA applied to ($\text{EDP}_1$) with and/or without Phase 1.

The third goal is to exploit the effect of the DC decomposition on the solution of ($\text{EDP}_1$) by the DCA via the regularization technique and the Lagrangian duality.

For these purposes, we have tested the following variants of our methods:

- EDCA1: the two-phase algorithm which uses **DCA1bis** and **DCA1** in Phase 1 and Phase 2, respectively
- EDCA2: the two-phase algorithm which uses **DCA2bis** and **DCA2** in Phase 1 and Phase 2, respectively
- SDCA: **DCA1** with procedure SP for computing a starting point
- RSDCA: DCA1r with procedure SP for computing a starting point (the regularized version of SDCA)
- REDCA1: the two-phase algorithm which uses **DCA1bis** and **DCA1r** in Phase 1 and Phase 2, respectively (the regularized version of EDCA1)
- EDCA1-3: a variant of EDCA1 which uses a combination of DCA1 and DCA3 in Phase 2: in EDCA1, we perform only a number of iterations of DCA1 to get a sufficiently good guess $X^{(k)}$ (we stop DCA1 with a quite large tolerance $\tau_2$) and then apply DCA3 to terminate Phase 2

We consider $w_{ij} = 1$ for all $i \neq j$ in Phase 1, and for $(i,j) \in \mathscr{S}$, $i \neq j$ in Phase 2.

For starting **DCA1bis**, we use Procedure **SP** to compute $X^{(0)}$ and then set $X^{(0)} := Proj_{\mathscr{A}^\perp}(X^{(0)})$. The initial point of **DCA2bis** is then set to $\frac{\tilde{\eta}_{\tilde{\delta}} X^{(0)}}{(\sqrt{n}\|X^{(0)}\|)}$. We take $\tau_1 = 10^{-8}$ and $\tau_2 = 10^{-9}$ in all algorithms (except for **DCA1** in the combined EDCA1-3 where we choose $\tau_2 = 10^{-3}$) .

For solving the linear system (13.57) (resp. Eq. (13.42)) in Phase 2 we first decompose the matrix $V+\frac{1}{n}ee^T = R^T R$ (resp. $V+\rho I = R^T R$) by the Cholesky factorization, and then at each iteration we solve two systems $R^T U = Y^{(k)}$ and $RX = U$.

In the tables presented below we indicate the following values:

- Data: the number of given distances, i.e., the cardinality of $\mathscr{S}^w$
- T0: CPU time of Procedure SP and the completion of the matrix $\tilde{\Delta}$ in the two-phase algorithm EDCA, and/or CPU time of Procedure SP in algorithm SDCA
- It1 and time1: the number of iterations and CPU time, of **DCA1bis** and/or **DCA2bis**, respectively
- It2 and time2: the number of iteration and CPU time, of **DCA1** and/or **DCA2**, respectively
- Ttotal: the total CPU time of the algorithm
- Aver: the average relative error defined by $\frac{1}{|\mathscr{S}|}\sum_{(i,j)\in\mathscr{S}} \frac{|\delta_{ij}-\|X_i^{*T}-X_j^{*T}\||}{\delta_{ij}}$
- Maxer: the maximum relative error defined by $\max\left\{|\delta_{ij}-\|X_i^{*T}-X_j^{*T}\||/\delta_{ij} : (i,j)\in\mathscr{S}\right\}$

Note that all CPU times are computed in seconds.

#### 13.6.1.1 The Data

**(i) The Artificial Data:** We consider two models of problems given in Moré–Wu [31] where the molecule has $n = s^3$ atoms located in the three-dimensional lattice $\{(i_1,i_2,i_3) : 0\le i_1 < s, 0\le i_2 < s, 0\le i_3 < s\}$ for some integer $s\ge 1$.

In the first model problem the ordering for the atoms is specified by letting $i$ be the atom at the position $(i_1,i_2,i_3)$: $i = 1+i_1+si_2+s^2 i_3$, and distance data are generated for all pairs of atoms in

$$\mathscr{S} = \{(i,j) : |i-j|\le r\}, \tag{13.121}$$

where $r$ is an integer between 1 and $n$.

In the second model problem the set $\mathscr{S}$ is specified by ( $X_i^T = (i_1,i_2,i_3)$)

$$\mathscr{S} = \{(i,j) : \|X_i^T - X_j^T\| \le \sqrt{r}\}. \tag{13.122}$$

As indicated in Moré–Wu [31], one difference between both definitions of $\mathscr{S}$ is that Eq. (13.122) includes *all nearby atoms*, while Eq. (13.121) includes *some of nearby atoms and some relatively far away atoms*. Then these model problems may capture various features in distance data from applications.

**(ii) The PDB Data:** We consider 16 problems whose data are derived from 16 structures of proteins given in PDB data bank. Table 13.1 gives the summarized information about these structures (in this table, “Exp.” is the abbreviation of “exploitation” and “MAS” is the one of “minimized average structure”).

**Table 13.1** Summarized information about test problems from PDB data bank

| ID code | Exp. method | Classification | Atoms (n) | Residues |
|---|---|---|---|---|
| 1A1D | NMR (MAS) | Nucleotidyltransferase | 146 | 146 |
| 304D | X-ray diffraction | Deoxyribonucleic acid | 237 | 52 |
| 8DRH | NMR (MAS) | Deoxyribonucleic acid/ribonucleic acid | 329 | 16 |
| 1AMD | NMR (MAS) | Deoxyribonucleic acid | 380 | 12 |
| 2MSJ | X-ray diffraction | Antifreeze protein | 480 | 66 |
| 124D | NMR | Deoxyribonucleic/rinonucleic acid | 509 | 16 |
| 141D | NMR | Deoxyribonucleic acid | 527 | 26 |
| 132D | NMR | Deoxyribonucleic acid | 750 | 24 |
| 1A84 | NMR | Deoxyribonucleic acid | 758 | 24 |
| 104D | NMR | DNA/RNA chimeric hybrid duplex | 766 | 24 |
| 103D | NMR (MAS) | Deoxyribonucleic acid | 772 | 24 |
| 2EQL | X-ray diffraction | Hydrolase (O-glycosyl) | 1,023 | 129 |
| 1QS5 | X-ray diffraction | Hydrolase | 1,429 | 162 |
| 1QSB | X-ray diffraction | Hydrolase | 1,431 | 162 |
| 1ITH | X-ray diffraction | Oxygen transport | 2,366 | 282 |
| 2CLJ | Theoretical model | Hydrolase | 4,189 | 543 |

For each structure we generate a set of distances by using all distances between the atoms in the same residue as well as those in the neighboring residues. More precisely, if $\mathscr{R}_k$ is the $k^{th}$ residue, then $\mathscr{S} = \{(i,j) : x_i \in \mathscr{R}_k,\ x_j \in \mathscr{R}_k \cup \mathscr{R}_{k+1}\}$ specifies the set of distances. The given distances in the *exact distance geometry problem* are defined by $\delta_{ij} = \|x^i - x^j\|,\ \forall (i,j) \in \mathscr{S}$.

**(iii) Hendrickson's Benchmark Problems:** This set of data is composed of significantly more difficult test problems. We consider the twelve problems generated by Hendrickson [13, 14] from the bovine pancreatic ribonuclease by using fragments consisting of the first 20, 40, 60, 80, and 100 amino acids as well as the full protein (124 amino acids), with two sets of distance constraints for each size corresponding to the largest unique subgraphs and the reduced graphs. These problems have from 63 up to 777 atoms. The protein actually has 1,849 atoms, but some simple structure exploitation allowed the author to start the numerical method with only 777 atoms.

#### 13.6.1.2 Experimental Results

*(i) The Performance of the Two-Phase Algorithm REDCA1*

In this experiment we have tested algorithm REDCA1 (the regularized version of the two-phase algorithm EDCA1) on the first two sets of data (the second model of the artificial data and the PDB data). To observe the behavior of our method when the number of given distance varies, we consider three different values of $r$ in the artificial data: $r = 1$, $r = 2$, and $r = s^2$ (Table 13.2). The results on the PDB data are reported in Table 13.3. The regularization parameter $\rho$ is set to $\rho = 0.01$.

**Table 13.2** The performance of REDCA1 for the second model problem, $r = 1$, $r = 2$, and $r = s^2$

| n | $r$ | Data | T0 | Iter1 | Time1 | Iter2 | Time2 | Ttotal | Aver | Maxer |
|---|---|---|---|---|---|---|---|---|---|---|
| 64 | 1 | 144 | 0.00 | 55 | 0.07 | 23 | 0.02 | 0.09 | 1.24E−8 | 8.66E−5 |
|  | 2 | 360 | 0.01 | 209 | 0.26 | 20 | 0.02 | 0.29 | 2.23E−8 | 9.90E−5 |
|  | $s^2$ | 1,880 | 0.00 | 68 | 0.09 | 21 | 0.05 | 0.14 | 2.24E-8 | 8.84E−5 |
| 125 | 1 | 300 | 0.03 | 74 | 0.37 | 59 | 0.13 | 0.52 | 8.12E−8 | 9.44E−5 |
|  | 2 | 780 | 0.05 | 79 | 0.38 | 24 | 0.07 | 0.50 | 2.21E−7 | 9.14E−5 |
|  | $s^2$ | 7,192 | 0.03 | 71 | 0.34 | 26 | 0.24 | 0.61 | 2.13E−8 | 8.57E−6 |
| 216 | 1 | 540 | 0.17 | 75 | 1.07 | 80 | 0.47 | 1.71 | 3.02E−8 | 9.60E−5 |
|  | 2 | 1,440 | 0.22 | 77 | 1.10 | 25 | 0.19 | 1.52 | 1.22E−8 | 9.70E−5 |
|  | $s^2$ | 21,672 | 0.15 | 64 | 0.92 | 27 | 0.77 | 1.84 | 1.02E−8 | 9.01E−5 |
| 343 | 1 | 882 | 0.66 | 69 | 2.61 | 109 | 1.56 | 4.83 | 2.36E−8 | 9.96E−5 |
|  | 2 | 2,394 | 0.80 | 71 | 2.69 | 32 | 0.60 | 4.09 | 2.23E−8 | 9.11E−5 |
|  | $s^2$ | 53,799 | 0.64 | 90 | 3.38 | 27 | 2.00 | 6.02 | 1.02E−8 | 8.95E−5 |
| 512 | 1 | 1,344 | 2.75 | 65 | 6.63 | 145 | 6.43 | 15.81 | 2.12E−8 | 9.98E−5 |
|  | 2 | 3,696 | 3.10 | 68 | 6.94 | 32 | 1.79 | 11.84 | 1.23E−8 | 9.97E−5 |
|  | $s^2$ | 119,692 | 2.14 | 81 | 8.10 | 33 | 6.22 | 16.46 | 4.52E−8 | 9.07E−5 |
| 729 | 1 | 1,944 | 11.40 | 78 | 20.26 | 184 | 28.10 | 59.76 | 2.23E−8 | 9.90E−5 |
|  | 2 | 5,400 | 12.22 | 73 | 18.96 | 35 | 6.35 | 37.54 | 1.45E−8 | 9.02E−5 |
|  | $s^2$ | 243,858 | 6.59 | 88 | 22.38 | 33 | 15.69 | 44.66 | 1.23E−7 | 9.14E−5 |
| 1,000 | 1 | 2,700 | 35.33 | 72 | 40.55 | 230 | 89.05 | 164.93 | 3.24E−7 | 9.90E−5 |
|  | 2 | 7,560 | 37.00 | 79 | 44.53 | 37 | 16.83 | 98.36 | 1.23E−8 | 9.19E−5 |
|  | $s^2$ | 456,872 | 17.55 | 89 | 48.38 | 1,261 | 1,197.99 | 1,263.92 | 2.35E−6 | 2.21E−4 |
| 1,331 | 1 | 3,630 | 111.08 | 78 | 95.62 | 272 | 405.56 | 611.91 | 1.23E−7 | 8.92E−5 |
|  | 2 | 10,230 | 112.95 | 81 | 99.14 | 40 | 61.48 | 271.52 | 1.24E−7 | 9.98E−5 |
|  | $s^2$ | 809,763 | 58.52 | 83 | 115.89 | 28 | 72.19 | 246.60 | 1.28E−6 | 8.90E−4 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1,728 | 1 | 4,752 | 368.25 | 75 | 215.89 | 326 | 860.11 | 1,444.25 | 1.89E−7 | 7.78E−5 |
| | 2 | 13,464 | 349.02 | 76 | 209.58 | 50 | 135.26 | 693.85 | 2.45E−7 | 9.23E−5 |
| | $s^2$ | – | 133.80 | 96 | 228.90 | 29 | 130.76 | 493.46 | 1.25E−6 | 8.52E−4 |
| 2,197 | 1 | 6,084 | 719.79 | 78 | 351.22 | 406 | 1,642.10 | 2,713.11 | 3.22E−7 | 1.05E−5 |
| | 2 | 17,316 | 726.37 | 77 | 348.01 | 44 | 208.87 | 1,283.25 | 1.08E−7 | 6.72E−5 |
| | $s^2$ | 2,014,666 | 315.59 | 85 | 383.87 | 60 | 480.28 | 1,179.74 | 1.0E−6 | 1.0E−3 |
| 2,744 | 1 | 7,644 | 1,620.07 | 88 | 741.54 | 237 | 1,979.76 | 4,341.37 | 8.52E−5 | 1.00E−3 |
| | 2 | 21,840 | 1,629.80 | 77 | 648.85 | 25 | 280.51 | 2,559.16 | 1.22E−6 | 1.00E−3 |
| | $s^2$ | 3,436,528 | 595.18 | 86 | 729.34 | 75 | 1,215.57 | 2,600.75 | 1.24E−6 | 1.00E−3 |
| 3,375 | 1 | 9,450 | 3,552.69 | 92 | 1,274.72 | 270 | 3,738.10 | 8,565.50 | 5.23E−5 | 1.00E−3 |
| | 2 | 27,090 | 3,570.90 | 85 | 1,175.36 | 28 | 529.58 | 5,275.86 | 2.34E−5 | 1.00E−3 |
| | $s^2$ | 5,196,129 | 1,201.28 | 111 | 1,526.32 | 5 | 266.82 | 2,994.42 | 1.0E−6 | 4.92E−2 |
| 4,096 | $s^2$ | 7,640,952 | 5,434.57 | 83 | 6,668.17 | 3 | 1,282.16 | 13,384.90 | 1.00E−4 | 4.85E−2 |

**Table 13.3** The performance of algorithm REDCA1 in the PDB data

| ID code | n | Data | T0 | Iter1 | Time1 | Iter2 | Time2 | Ttotal | Aver | Maxer |
|---|---|---|---|---|---|---|---|---|---|---|
| 1A1D | 146 | 145 | 0.0 | 0 | 0.00 | 0 | 0 | 0.00 | 0.00E+0 | 0.00E+0 |
| 304D | 237 | 2,319 | 0.38 | 251 | 4.85 | 175 | 1.52 | 6.76 | 7.00E−4 | 1.00E−3 |
| 8DRH | 329 | 3,549 | 0.91 | 481 | 17.36 | 660 | 10.65 | 28.92 | 1.80E−4 | 1.00E−2 |
| 1AMD | 380 | 6,186 | 1.91 | 912 | 40.82 | 844 | 19.31 | 65.51 | 1.00E−6 | 1.00E−4 |
| 2MSJ | 480 | 715 | 2.04 | 110 | 9.81 | 918 | 22.96 | 34.81 | 1.50E−4 | 1.00E−2 |
| 124D | 509 | 8,307 | 4.16 | 401 | 36.06 | 492 | 18.03 | 58.26 | 2.00E−6 | 1.00E−3 |
| 141D | 527 | 5,615 | 3.44 | 544 | 52.01 | 498 | 26.95 | 88.53 | 1.00E−5 | 1.00E−3 |
| 132D | 750 | 12,094 | 14.21 | 262 | 65.94 | 182 | 29.81 | 106.84 | 1.00E−4 | 1.00E−2 |
| 1A84 | 758 | 12,345 | 19.55 | 452 | 122.00 | 886 | 118.51 | 260.07 | 1.02E−6 | 1.00E−3 |
| 104D | 766 | 12,609 | 15.56 | 524 | 135.50 | 13,281 | 1,704.65 | 1,855.71 | 1.00E−5 | 1.50E−3 |
| 103D | 772 | 12,777 | 18.08 | 1,179 | 343.39 | 414 | 58.12 | 419.59 | 2.36E−5 | 2.02E−3 |
| 2EQL | 1,023 | 4,888 | 54.60 | 169 | 133.42 | 1,665 | 939.40 | 1,127.41 | 3.00E−4 | 3.05E−2 |
| 1QS5 | 1,429 | 6,355 | 154.88 | 72 | 104.66 | 3,572 | 4,439.67 | 4,698.99 | 1.10E−3 | 5.02E−2 |
| 1QSB | 1,431 | 6,344 | 165.67 | 112 | 166.93 | 2,555 | 3,215.16 | 3,547.75 | 1.20E−03 | 7.53E−2 |
| 1ITH | 2,366 | 6,239 | 957.38 | 87 | 458.62 | 1,441 | 6,944.09 | 8,360.08 | 1.00E−4 | 8.58E−2 |
| 2CLJ | 4,189 | 19,833 | 8,238.57 | 119 | 3,056.88 | 1,230 | 31,949.27 | 43,244.71 | 1.00E−3 | 1.00E−1 |

*(ii) Comparison of SDCA and RSDCA*

In the second experiment we study the efficiency of DCA applied to (EDP$_1$) without Phase 1. In Table 13.4 we report the experimental results of algorithms SDCA and RSDCA (the regularized version of SDCA) on the PDB data. Here we are interested in the effect of the regularization technique for DCA. The regularization parameter is taken as $\rho = 0.01$.

*(iii) Comparison of Two Variants EDCA1 and EDCA2*

In this experiment we consider two versions of EDCA which correspond to different DC decompositions to solve the first model problem where the parameter $r$ in Eq. (13.121) is set to $r = s^2$. This data set is also considered in [48]. We note that when $r = s^2$ the set defined by Eq. (13.121) is included in the set defined by Eq. (13.122). In Table 13.5 we present the performance of two algorithms EDCA1 and EDCA2.

*(iv) The Performance of EDCA1 and EDCA1-3 for Hendrickson's Problems*

In this experiment we are interested in the efficiency of EDCA1 and the combined EDCA1-3 to the last set of data, say Hendrickson's benchmark problems. The results are summarized in Table 13.6.

#### 13.6.1.3 Comments

Our main concerns in this chapter are both the ability to treat large-scale problems and the cost of algorithms. The numerical results show that our algorithms are quite efficient to all sets of data. Our experiments suggest the following comments:

*(i) About the Two-Phase Algorithm EDCA and Its Variants*

The most important fact is that *in all experiments algorithm EDCA gives an $\varepsilon$-global solution of (EDP$_1$).* Moreover, since the basic DCA is efficient, EDCA can solve large-scale problems in a short time when $n \leq 1{,}000$ (3,000 variables) and in a reasonable time when $1{,}331 \leq n \leq 4{,}189$ (to 12,567 variables).

We observe from Tables 13.3 and 13.5 that the rate of convergence of the DCA in Phase 1 does not depend much on the distance data (i.e., the number and the length of given distances between *nearby* or *far away* atoms). In other words, the DCA is quite stable in the normal case (in the artificial data).

On the contrary, the DCA in Phase 2 (**DCA1** and **DCA2**) is quite sensitive to the data. In the first model, (where given distances are determined between both nearby and far away atoms), the number of iterations of DCA1 and/or DCA2 is much greater than that in the second model. A simple explanation is that for the given distances *between relatively far away atoms* the approximate distance matrix $\tilde{\Delta}$ does not seem to be "good" and the resulting initial point $\tilde{X}$ (given by Phase 1) is not relatively *near* a solution of (EDP)$_1$. Then **DCA1** and **DCA2** need more

**Table 13.4** The performance of algorithms RSDCA and SDCA in the PDB data

| | | RSDCA | | | | SDCA | | | |
|---|---|---|---|---|---|---|---|---|---|
| ID code | n | Iter2 | Ttotal | Aver | Maxer | Iter2 | Ttotal | Aver | Maxer |
| 1A1D | 146 | 0 | 0.00 | 0.00E+0 | 0.00E+0 | 0 | 0.00 | 0.00E+0 | 0.00E+0 |
| 304D | 237 | 701 | 6.30 | 1.22E−7 | 1.00E−4 | 575 | 4.38 | 1.00E−4 | 4.87E−2 |
| 8DRH | 329 | 560 | 9.08 | 2.10E−4 | 2.00E−1 | 604 | 9.26 | 2.10E−3 | 2.00E−1 |
| 1AMD | 380 | 631 | 16.06 | 1.02E−8 | 9.89E−5 | 226 | 6.07 | 2.0E−4 | 4.77E−2 |
| 2MSJ | 480 | 666 | 16.88 | 1.24E−7 | 8.55E−5 | 454 | 11.92 | 2.0E−4 | 4.55E−2 |
| 124D | 509 | 715 | 32.53 | 1.24E−7 | 5.75E−4 | 259 | 11.44 | 3.0E−4 | 4.93E−2 |
| 141D | 527 | 637 | 26.61 | 3.72E−3 | 2.28E−1 | 688 | 26.61 | 4.0E−3 | 2.28E−1 |
| 132D | 750 | 337 | 46.31 | 2.50E−3 | 4.53E−1 | 388 | 65.23 | 3.2E−3 | 5.0E−1 |
| 1A84 | 758 | 4,438 | 565.91 | 0.00E+0 | 1.70E−3 | 780 | 133.24 | 8.0E−4 | 2.92E−1 |
| 104D | 766 | 875 | 151.46 | 2.46E−06 | 1.38E−04 | 335 | 61.18 | 2.0E−4 | 4.69E−2 |
| 103D | 772 | 743 | 116.18 | 2.22E−3 | 4.73E−1 | 717 | 126.85 | 2.2E−3 | 4.73E−1 |
| 2EQL | 1,023 | 1,219 | 669.25 | 5.76E−4 | 1.30E−1 | 922 | 373.65 | 1.0E−3 | 2.49E−1 |
| 1QS5 | 1,429 | 1,109 | 1,402.66 | 8.00E−4 | 1.50E−1 | 1,041 | 1,488.72 | 7.0E−4 | 7.05E−2 |
| 1QSB | 1,431 | 1,282 | 1,445.91 | 7.56E−4 | 1.44E−1 | 1,282 | 1,545.91 | 8.0E−4 | 1.44E−1 |
| 1ITH | 2,366 | 2,101 | 9,799.02 | 4.82E−4 | 8.21E−2 | 2,201 | 10,299.02 | 1.0E−4 | 8.52E−2 |
| 2CLJ | 4,189 | 1,002 | 27,001.29 | 6.72E−4 | 2.03E−1 | 1,052 | 28,361.29 | 7.0E−4 | 2.06E−1 |

**Table 13.5** The performance of EDCA1 and EDCA2 for the first model problem, $r = s^2$

| n | Data | Algorithm | T0 | Iter1 | Time1 | Iter2 | Time2 | Ttotal | Aver | Maxer |
|---|---|---|---|---|---|---|---|---|---|---|
| 64 | 888 | EDCA1 | 0.00 | 70 | 0.09 | 70 | 0.09 | 28 | 1.00E−3 | 4.84E−2 |
| | | EDCA2 | 0.00 | 66 | 0.01 | 28 | 0.04 | 0.05 | 1.00E−3 | 4.85E−2 |
| 125 | 2,800 | EDCA1 | 0.03 | 117 | 0.58 | 52 | 0.24 | 0.85 | 6.00E−4 | 4.89E−2 |
| | | EDCA2 | 0.03 | 112 | 0.52 | 52 | 0.24 | 0.79 | 6.00E−4 | 4.87E−2 |
| 216 | 7,110 | EDCA1 | 0.23 | 90 | 1.51 | 87 | 1.75 | 3.50 | 4.00E−4 | 4.94E−2 |
| | | EDCA2 | 0.23 | 84 | 1.32 | 87 | 1.75 | 3.26 | 4.00E−4 | 4.96E−2 |
| 343 | 15,582 | EDCA1 | 0.54 | 86 | 3.66 | 137 | 6.56 | 10.77 | 3.00E−4 | 4.97E−2 |
| | | EDCA2 | 0.54 | 83 | 3.25 | 136 | 6.55 | 10.34 | 3.00E−4 | 4.97E−2 |
| 512 | 30,688 | EDCA1 | 3.20 | 81 | 7.68 | 205 | 20.68 | 31.55 | 0.0002 | 4.98E−2 |
| | | EDCA2 | 3.20 | 78 | 7.18 | 206 | 20.55 | 30.93 | 0.0002 | 4.97E−2 |
| 729 | 55,728 | EDCA1 | 15.48 | 92 | 28.61 | 288 | 77.18 | 121.33 | 1.00E−4 | 4.98E−2 |
| | | EDCA2 | 15.48 | 90 | 28.24 | 288 | 77.19 | 120.91 | 1.00E−4 | 4.98E−2 |
| 1,000 | 94,950 | EDCA1 | 76.20 | 95 | 61.05 | 369 | 271.33 | 347.53 | 1.00E−4 | 4.99E−2 |
| | | EDCA2 | 76.20 | 92 | 59.01 | 369 | 271.33 | 345.49 | 1.00E−4 | 4.99E−2 |
| 1,331 | 153,670 | EDCA1 | 178.85 | 85 | 103.68 | 471 | 537.68 | 820.21 | 1.00E−4 | 4.99E−2 |
| | | EDCA2 | 178.85 | 83 | 100.64 | 470 | 537.62 | 817.11 | 1.00E−4 | 4.99E−2 |
| 1,728 | 238,392 | EDCA1 | 404.15 | 87 | 285.34 | 581 | 1,930.06 | 2,619.55 | 1.00E−4 | 4.99E−2 |
| | | EDCA2 | 404.15 | 84 | 277.92 | 432 | 1,929.01 | 2,611.14 | 1.00E−4 | 4.99E−2 |
| 2,197 | 356,928 | EDCA1 | 1,073.46 | 103 | 563.41 | 702 | 4,009.28 | 5,646.15 | 1.00E−4 | 4.99E−2 |
| | | EDCA2 | 1,073.46 | 99 | 542.84 | 703 | 4,012.22 | 5,628.52 | 1.00E−4 | 4.99E−2 |
| 2,744 | 518,518 | EDCA1 | 2,745.87 | 130 | 1,132.52 | 848 | 7,593.98 | 11,472.37 | 1.00E−4 | 4.99E−2 |
| | | EDCA2 | 2,745.87 | 124 | 1,073.21 | 850 | 7,601.22 | 11,420.30 | 1.00E−4 | 4.99E−2 |

**Table 13.6** The performance of EDCA1 and EDCA1-3 for Hendrickson's problems

| | | EDCA1 | | | EDCA1-3 | | |
|---|---|---|---|---|---|---|---|
| n | Data | Ttotal | Aver | Maxer | Ttotal | Aver | Maxer |
| 63 | 236 | 12.23 | 9.36E−5 | 7.85E−4 | 5.74 | 1.81E−4 | 1.58E−3 |
| 102 | 336 | 17.62 | 9.46E−5 | 1.12E−3 | 11.48 | 1.34E−4 | 1.35E−3 |
| 174 | 786 | 28.19 | 5.76E−4 | 2.49E−2 | 31.62 | 5.59E−4 | 2.49E−2 |
| 236 | 957 | 100.85 | 2.73E−5 | 6.25E−4 | 52.88 | 8.56E−5 | 9.05E−4 |
| 287 | 1,319 | 74.59 | 3.40E−3 | 1.43E−1 | 310.24 | 2.43E−4 | 3.60E−2 |
| 362 | 1,526 | 316.24 | 3.24E−4 | 2.48E−2 | 111.38 | 3.0E−4 | 2.38E−2 |
| 377 | 1,719 | 325.68 | 5.75E−4 | 4.32E−2 | 165.25 | 7.0E−4 | 2.92E−2 |
| 472 | 2,169 | 359.95 | 3.63E−3 | 1.66E−1 | 258.22 | 3.67E−4 | 5.44E−2 |
| 480 | 2,006 | 747.49 | 6.50E−4 | 4.97E−2 | 287.14 | 7.02E−4 | 3.89E−2 |
| 599 | 2,532 | 331.56 | 3.24E−3 | 1.77E−1 | 1,746.20 | 2.32E−3 | 7.20E−2 |
| 695 | 3,283 | 1,067.15 | 1.35E−2 | 2.49E−1 | 9,899.00 | 1.50E−3 | 8.50E−2 |
| 777 | 3,504 | 619.57 | 6.48E−4 | 2.50E−2 | 620.80 | 6.48E−4 | 2.50E−2 |

iterations to yield a solution. In general, the cost of algorithm EDCA in the first model problem is more important than in the second one (see Tables 13.3 and 13.5).

Consider now the influence of the number of given distances, *data*, in the first experiment. We observe that, with $n \geq 1,000$, when the number of given distances is small ($r = 1$ and $r = 2$) the cost for determining $\tilde{\Delta}$ is very important in the two-phase algorithm (21 to 68 % of the total cost). However, when the number of given distances is large ($r = 2$ and $r = s^2$ in the artificial data) this step is necessary, because Phase 1 is important for EDCA to obtain a global solution of (EDP$_1$) in such a case. The results given in Table 13.2 show that when $n \geq 512$ the more the number of given distances increases, the more t0 decreases, and thus the faster EDCA is.

On the other hand, although the sequences $\{X^{(k)}\}$ in **DCA1** and **DCA2** are not in an explicit form, one iteration of these algorithms (which comprises computing the matrix $B(X^{(k)})$, the product $B(X^{(k)})X^{(k)}$, and the solution of two triangular systems) is not more expensive than that of **DCA1bis** and **DCA2bis** which need only matrix-vector products. This shows that **DCA1** and **DCA2** exploit well the sparsity of $\mathscr{S}$ (in the determination of matrix $B(X^{(k)})$ and the product $B(X^{(k)})X^{(k)}$).

### *(ii) About the Algorithm SDCA and Its Regularized Version*

The two algorithms SDCA and RSDCA are very efficient when the number of constraints is not large. In the artificial data with $r = 1$ they provided an optimal solution for all test problems with the maximal error *maxer* $\leq 0.009$ (for not increasing the length of the paper we do not present here these numerical results). In any case we see that the objective function decreases very fast during some first iterations of DCA1. In the PDB data SDCA successfully solved 11 of 16 problems with *aver* $\leq 0.001$ and *aver* $\leq 0.003$ in all test problems. Hence, Phase 1 in the two-phase algorithm can be replaced efficiently by procedure SP when a small subset of distances is known. Among SDCA and RSDCA, the regularized version RSDCA gives better results in most cases.

### *(iii) The Effect of Phase 1 in the Two-Phase Algorithm EDCA*

From experimental results we see that Phase 1 is *important* for EDCA when the number of constraints is large. In other words, for our DC approach, the technique using the shortest path in Phase 1 of EDCA seems to be more advantageous than procedure SP when the number of distance constraints is large. Nevertheless when a small subset of constraints is known, Phase 1 does not seem to be efficient because it is expensive to complete the "distance" matrix, and the resulting complete dissimilarity matrix may not be a good approximation to the complete exact distance matrix.

### *(iv) The Effect of the Regularization Technique*

As indicated in Sect. 13.3.1.2, the regularization technique has a visible advantage. In all test problems (most of them have not been presented here), with an appropriate choice of the regularization parameter $\rho$, DCA1r is better than DCA1 (in our experiments the best choice of $\rho$ is $\rho \in [0.01, 0.001]$).

### *(v) About Two Variants EDCA1 and EDCA2: The Effect of DC Decomposition*

The sequence $\{\|X^{(k+1)} - X^{(k)}\|\}$ in **DCA2** (resp. **DCA2bis**) decreases faster than in **DCA1** (resp. **DCA1bis**). In all problems, the number of iterations of **DCA2bis** is smaller than that of **DCA1bis**. In several test problems, EDCA2 is less expensive than EDCA1.

### *(vi) More About the Results on PDB Data and on Hendrickson's Problems*

Not surprisingly, the problems derived from PDB data are more difficult to solve than the artificial problems. For these real-life problems Phase 2 needs much more iterations than for artificial problems, while Phase 1 has the same behavior. On the other hand, in contrast to the artificial data, the smaller the number of distance constraints is, the more efficient EDCA1 is. We note that the average error of the obtained solution is very small in both algorithms REDCA1 and RSDCA.

The twelve problems generated by Hendrickson [14] are the most difficult: the cost of the algorithm is higher than that for the two first sets of test problems. However, our algorithm is still efficient to these problems: the maximal error on distance constraints of the solution given by EDCA1-3 is below, respectively, 0.025, 0.055, and 0.085 in 6, 4, and 2 problems. We observe that the solutions obtained by DCA3 are better than those provided by **DCA1**, but **DCA3** is more expensive than **DCA1.** Then it is interesting to use the combined algorithm EDCA1-3. This set of test problems has been considered in [48] with exact distances for the first seven problems and with inexact distances, say the general problem (13.2) with $0.01 \leq \varepsilon \leq 0.04$, for the rest five problems ($n \geq 472$). Here we consider the exact distances for all test problems, and these results indicate that our approach has the potential to locate exact (or nearly exact) solutions.

## *13.6.2 Computational Experiments for the General Distance Geometry Problems*

We test the efficiency of the three proposed algorithms for the general distance geometry problem on the artificial data and the PDB data considered in the previous section:

- The two-phase algorithm which uses **DCA1bis** and **GDCA1**in Phase 1 and Phase 2, respectively, also denoted **GDCA1**
- The two-phase algorithm which uses **DCA1bis** and **GDCA2** in Phase 1 and Phase 2, respectively, also denoted **GDCA2**
- The continuation algorithm **CGDCA**

We considered $w_{ij} = 1$ for all $i \neq j$ in (GDP$_1$) and (GDP$_2$).

### 13.6.2.1 Experimentation on the Artificial Data

We have tested our algorithm on the second model problems from Moré–Wu [31] described above. We study the efficiency of **GDCA1** on various bounds $l_{ij}$ and $u_{ij}$. For this reason, we used the same procedure from [33] to generate the given bounds:

$$l_{ij} = (1-\varepsilon)\|X_i^T - X_j^T\|, \;\; u_{ij} = (1+\varepsilon))\|X_i^T - X_j^T\|) \tag{13.123}$$

for some $\varepsilon \in (0,1)$. We are then able to examine the behavior of the algorithm as $\varepsilon$ varies over $(0,1)$.

The computational experiment allows studying the effect of the number of given bounds on the performance of our algorithm. We have tested the algorithm on different values of $r$ that vary the cardinality of $\mathscr{S}$.

For solving the linear system (13.88) in Phase 2 we first decomposed the matrix $(V + \frac{1}{n}ee^T) = R^T R$ by the Cholesky factorization, and then at each iteration we solved two triangular linear systems $R^T U = (B + C(X^{(k)}, T^{(k)}))X^{(k)}$ and $RX = U$.

For stopping **GDCA1** and **GDCA2** we used the same tolerance considered in [33] when $n < 3,375$, i.e., $\tau = 0.01$. When $n = 3,375$ we took $\tau = 0.02$.

Similarly to the previous tables, “it1” and “time1” means respectively, the number of iterations and CPU time in seconds of Phase 1 while “it2” and “time2” denotes respectively, the number of iterations and CPU time in seconds of Phase 2. Also, “ttotal” is the total CPU time of the main algorithm **GDCA1**, and “data” denotes the number of given distances, i.e., the cardinality of $\mathscr{S}^w$.

In Table 13.7 we present the performance of algorithm **GDCA1** with $\varepsilon = 0.04$, and $\varepsilon = 0.08$, in the cases $r = 1$, $r = 2$, and $r = s$.

**Table 13.7** The performance of algorithm **GDCA1** with $\varepsilon = 0.04$ and $\varepsilon = 0.08$

| | | Data | $\varepsilon = 0.04$ | | | | | $\varepsilon = 0.08$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| n | | $r$ | Iter1 | Time1 | Iter2 | Time2 | Ttotal | Iter1 | Time1 | Iter2 | Time2 | Ttotal |
| 27 | 1 | 54 | 54 | 0.025 | 5 | 0.001 | 0.026 | 56 | 0.022 | 3 | 0.000 | 0.023 |
| | 2 | 126 | 307 | 0.120 | 10 | 0.003 | 0.124 | 223 | 0.087 | 9 | 0.003 | 0.091 |
| | s | 347 | 75 | 0.029 | 0 | 0.000 | 0.030 | 70 | 0.027 | 0 | 0.000 | 0.027 |
| 64 | 1 | 144 | 60 | 0.021 | 7 | 0.013 | 0.146 | 64 | 0.138 | 5 | 0.005 | 0.143 |
| | 2 | 360 | 125 | 0.280 | 10 | 0.012 | 0.293 | 125 | 0.287 | 10 | 0.112 | 0.299 |
| | s | 1,880 | 89 | 0.198 | 97 | 0.250 | 0.448 | 103 | 0.228 | 33 | 0.086 | 0.314 |
| 125 | 1 | 300 | 91 | 0.823 | 14 | 0.042 | 0.866 | 92 | 0.837 | 10 | 0.032 | 0.869 |
| | 2 | 780 | 123 | 1.541 | 7 | 0.029 | 1.123 | 115 | 1.027 | 7 | 0.029 | 1.056 |
| | s | 7,192 | 102 | 0.872 | 170 | 1.707 | 2.580 | 103 | 0.881 | 189 | 1.889 | 2.770 |
| 216 | 1 | 540 | 67 | 2.047 | 19 | 0.167 | 2.214 | 65 | 2.005 | 14 | 0.132 | 2.137 |
| | 2 | 1,440 | 78 | 2.342 | 7 | 0.087 | 2.429 | 70 | 2.126 | 7 | 0.087 | 2.213 |
| | s | 21,672 | 84 | 2.250 | 243 | 7.763 | 10.012 | 80 | 2.145 | 449 | 14.306 | 16.451 |
| 343 | 1 | 882 | 98 | 8.424 | 19 | 0.590 | 9.014 | 96 | 8.381 | 12 | 0.403 | 8.784 |
| | 2 | 2,394 | 83 | 7.334 | 9 | 0.343 | 7.677 | 80 | 7.130 | 9 | 0.344 | 7.474 |
| | s | 53,799 | 108 | 7.981 | 406 | 37.127 | 45.108 | 90 | 6.728 | 547 | 49.791 | 56.519 |
| 512 | 1 | 1,344 | 77 | 20.858 | 18 | 1.696 | 22.545 | 69 | 19.175 | 10 | 1.100 | 20.275 |
| | 2 | 3,696 | 77 | 20.863 | 11 | 1.213 | 22.076 | 76 | 20.669 | 11 | 1.230 | 21.899 |
| | s | 119,692 | 92 | 18.184 | 550 | 133.238 | 151.413 | 109 | 21.338 | 779 | 188.287 | 209.625 |

(continued)

**Table 13.7** (continued)

|  |  | Data | $\varepsilon = 0.04$ |  |  |  |  | $\varepsilon = 0.08$ |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| n |  | *r* | Iter1 | Time1 | Iter2 | Time2 | Ttotal | Iter1 | Time1 | Iter2 | Time2 | Ttotal |
| 729 | 1 | 1,944 | 77 | 56.189 | 23 | 6.180 | 62.469 | 73 | 54.728 | 14 | 4.155 | 58.783 |
|  | 2 | 5,400 | 88 | 61.609 | 13 | 4.020 | 65.628 | 83 | 59.105 | 13 | 4.011 | 63.116 |
|  | s | 243,858 | 97 | 46.688 | 777 | 462.851 | 509.539 | 108 | 51.527 | 845 | 499.930 | 551.457 |
| 1,000 | 1 | 2,700 | 70 | 127.147 | 33 | 19.939 | 147.086 | 73 | 130.980 | 19 | 13.163 | 144.143 |
|  | 2 | 7,560 | 87 | 143.921 | 14 | 10.559 | 154.481 | 86 | 142.22 | 14 | 9.906 | 152.126 |
|  | s | 456,872 | 90 | 92.120 | 750 | 928.538 | 1,021.464 | 91 | 93.474 | 1,079 | 1,330.681 | 1,424.154 |
| 1,331 | 1 | 3,630 | 77 | 373.150 | 41 | 70.598 | 443.749 | 73 | 362.41 | 19 | 36.097 | 398.507 |
|  | 2 | 10,230 | 83 | 383.735 | 15 | 29.840 | 413.57507 | 95 | 412.564 | 15 | 29.868 | 442.431 |
|  | s | 809,763 | 109 | 238.950 | 846 | 2,227.612 | 2,466.562 | 114 | 248.459 | 1,460 | 3,839.354 | 4,087.813 |
| 1,728 | 1 | 4,752 | 73 | 724.046 | 51 | 149.920 | 873.966 | 69 | 709.026 | 15 | 54.005 | 763.031 |
|  | 2 | 13,464 | 81 | 754.888 | 17 | 59.954 | 814.843 | 100 | 826.779 | 16 | 57.052 | 883.831 |
| 2,197 | 1 | 6,084 | 79 | 1,695.226 | 62 | 363.544 | 2,058.771 | 77 | 1,638.067 | 16 | 117.323 | 1,755.390 |
|  | 2 | 17,316 | 100 | 1,814.960 | 17 | 122.642 | 1,937.603 | 96 | 1,764.107 | 17 | 122.618 | 1,886.725 |
| 2,744 | 1 | 7,644 | 83 | 3,550.647 | 74 | 890.792 | 4,441.439 | 86 | 3,585.365 | 19 | 292.622 | 3,877.987 |
|  | 2 | 21,840 | 99 | 3,698.951 | 18 | 274.561 | 3,973.511 | 96 | 3,662.845 | 20 | 302.656 | 3,965.501 |
| 3,375 | 1 | 9,450 | 68 | 6,745.238 | 16 | 666.760 | 7,412.002 | 71 | 6,800.952 | 13 | 515.420 | 7,316.372 |
|  | 2 | 27,090 | 83 | 7,065.840 | 22 | 674.270 | 7,740.112 | 87 | 7,145.402 | 11 | 338.600 | 7,484.002 |

#### 13.6.2.2 Experimentation on the PDB Data

We considered ten problems whose data are derived from ten structures of proteins given in PDB data bank (See Table 13.1). The set $\mathscr{S}$ is generated by the same way in the first experiment. To generate the given bounds in the general distance geometry problem we use Eq. (13.123) with $\varepsilon = 0.01$.

In **GDCA2** and **GDCA**$_\lambda$ we took $\varepsilon = \varepsilon_1 = 10^{-7}$, $\varepsilon_2 = 10^{-8}$. If Eq. (13.120) is satisfied when stopping algorithm, we say that an $\tau$-global minimizer of the geometry distance problem (13.3) is obtained. The parameters $\lambda_0$ and *step* in the continuation algorithm are chosen as in [33], say $\lambda_0$ is the median of all $\lambda_{ij} = \left(\frac{l_{ij}}{\sqrt{5}u_{ij}} + \sqrt{2}(1 - \frac{l_{ij}}{u_{ij}}\right) u_{ij}$ with $(i, j) \in \mathscr{S}$ and $step = \lceil 20\lambda_0 \rceil$.

In Table 13.8 we present the summary performance of the tree algorithms **GDCA1**, **GDCA2**, and **CGDCA**. Here $F^* := F(X^*, T^*) = \frac{1}{4}\sum_{(i,j)\in\mathscr{S}^w} w_{ij}$ $(\|x^{*i} - x^{*j}\|^2 - t_{ij}^{*2})^2$ and "maxer" denotes

$$\max\left\{\max\left\{\left(\left\|x^{*^i} - x^{*^j}\right\| - u_{ij}\right)/u_{ij}, \left(l_{ij} - \left\|x^{*^i} - x^{*^j}\right\|\right)/l_{ij}, 0\right\} : (i, j) \in \mathscr{S}^w\right\}.$$

In the next experiment we are interested in the influence of the length of bounds on the behavior of **CGDCA** on four test problems. We generate the data in the same way as in the first experiment with different values of $\varepsilon : 0.0, 0.001, 0.01, 0.02, 0.04$, and 0.08. The results are reported in Table 13.9. Remember that when $\varepsilon = 0.0$ we are faced with the *exact distance geometry problem* (13.1).

#### 13.6.2.3 Comments

*(i) About the Two-Phase Algorithm GDCA1*

The most important fact is that in all experiments algorithm **GDCA1** gives a global solution to (GDP$_1$). Moreover, since the basic DCA is efficient, GDCA1 can solve large-scale problems in a reasonable time (the maximum running time is about 2 hr for a problem with 10,125 variables).

About the influence of the length of bounds (when $\varepsilon$ varies), from Table 13.7 we constate that:

- In the cases $r = 1$ and/or $r = 2$, the more $\varepsilon$ increases (i.e., the more the length of bounds increases), the faster GDCA is (except for the cases where $n = 1{,}331$, $n = 1{,}728$ with $r = 2$ for which GDCA is the most expensive when $\varepsilon = 0.08$).
- In the case $r = s$, **GDCA1** is the most expensive when $\varepsilon = 0.08$. The ratio of CPU time between these cases goes up to 2.

We observe also that the rate of convergence of the DCA in Phase 1 does not seem to depend on data. In contrast, the DCA in Phase 2 (**GDCA1**) is quite sensitive

**Table 13.8** Summary performance of **GDCA1**, **GDCA2**, and **CGDCA**

| | | CGDCA | | | GDCA1 | | | GDCA2 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ID code | numva | Ttotal | $F^*$ | Maxer | Ttotal | $F^*$ | Maxer | Ttotal | $F^*$ | Maxer |
| 304D | 3,030 | 146.06 | 0.23E+00 | 2.02E−03 | 284.37 | 3.63E−03 | 9.99E−03 | 224.37 | 3.83E−03 | 9.99E−03 |
| 8DRH | 4,536 | 35.03 | 1.32E−03 | 9.98E−05 | 198.80 | 8.66E−02 | 6.68E−02 | 64.95 | 1.36E−02 | 9.99E−03 |
| 2MSJ | 2,155 | 32.69 | 1.86E−01 | 3.53E−02 | 360.39 | 4.65E−04 | 9.95E−03 | 53.16 | 1.58E−05 | 3.22E−03 |
| 1AMD | 7,326 | 102.85 | 9.50E−04 | 9.99E−03 | 192.56 | 9.70E−04 | 9.99E−03 | 142.12 | 1.93E−02 | 9.99E−03 |
| 2EQL | 7,957 | 183.27 | 3.69E−03 | 9.99E−03 | 4,318.9 | 4.00E−03 | 9.99E−03 | 242.12 | 1.95E−02 | 9.99E−03 |
| 132D | 14,344 | 320.04 | 4.50E−03 | 9.99E−03 | 1,794.52 | 3.79E−03 | 9.98E−02 | 630.19 | 1.48E−03 | 1.36E−02 |
| 104D | 314,907 | 71.84 | 2.54E−04 | 1.30E−02 | 6,198.2 | 1.32E−02 | 1.00E−02 | 486.56 | 7.49E−03 | 9.99E−03 |
| 1A84 | 758 | 484.84 | 5.12E−04 | 9.99E−03 | 1,182.48 | 1.83E−03 | 1.99E−02 | 782.48 | 1.83E−03 | 9.99E−03 |
| 103D | 15,093 | 411.48 | 2.68E−03 | 9.99E−03 | 1,107.0 | 1.69E+00 | 2.70E−01 | 722.48 | 1.92E−03 | 9.99E−03 |
| 6GAT | 28,288 | 1,229.52 | 2.00E−02 | 9.99E−03 | 2,602.50 | 1.05E−01 | 4.99E−02 | 3,569.90 | 1.47E−02 | 9.99E−03 |
| 7HSC | 29,962 | 1,129.08 | 3.66E−03 | 9.99E−03 | 2,512.70 | 3.86E−03 | 9.99E−03 | 912.83 | 4.22E−03 | 9.99E−03 |
| 2CLJ | 32,400 | 919.04 | 1.37E+00 | 1.12E−01 | 3,524.22 | 7.66E−02 | 6.78E−02 | 1,554.00 | 1.75E−01 | 1.016E−01 |

**Table 13.9** The performance of **CGDCA** as the length of bounds varies

| ID code | 1AMD | | 124D | | 104D | | 2EQL | |
|---|---|---|---|---|---|---|---|---|
| $\varepsilon$ | Ttotal | Maxer | Ttotal | Maxer | Ttotal | Maxer | Ttotal | Maxer |
| 0.08 | 102.85 | 9.99E−03 | 387.06 | 9.99E−03 | 371.74 | 1.30E−02 | 183.27 | 9.99E−03 |
| 0.04 | 71.03 | 5.84E−02 | 388.08 | 1.02E−02 | 265.28 | 9.99E−03 | 339.41 | 9.75E−02 |
| 0.02 | 49.18 | 7.15E−02 | 372.16 | 1.03E−02 | 241.63 | 9.99E−03 | 204.86 | 3.39E−01 |
| 0.01 | 79.20 | 1.04E−02 | 272.04 | 9.99E−03 | 219.26 | 9.99E−03 | 228.89 | 6.47E−02 |
| 0.001 | 47.68 | 9.99E−03 | 153.20 | 9.99E−03 | 151.35 | 9.99E−03 | 237.47 | 9.99E−03 |
| 0.0 | 48.06 | 6.42E−03 | 129.85 | 8.30E−03 | 134.30 | 9.99E−03 | 263.07 | 9.99E−03 |

to data. On the other hand, although the sequences $\{X^{(k)}\}$ in **GDCA1** are not in explicit form, the cost of one iteration of this algorithm (that contains the cost of the computation of matrices $C(X^{(k)},T^{(k)})$, $T^{(k+1)}$ and the cost of the solution of two triangular linear systems) is not more expensive than the cost of one iteration of **DCA1bis** which requires only matrix-vector products. This shows that **GDCA1** exploits well sparsity of $\mathscr{S}$ (in the determination of matrices $C(X^{(k)},T^{(k)})$, $T^{(k+1)}$, and the product $C(X^{(k)},T^{(k)})X^{(k)}$).

Moreover, we observe from Table 13.7 that when the number of given bounds is small ($r=1$ and $r=2$), Phase 1 is much more expensive than Phase 2 (the cost of Phase 1 occupies up to 96.4 % of the total cost when $n \geq 1{,}728$). However, this phase is important for **GDCA1** to obtain a global solution of (GDP$_1$) in these cases. On the contrary, when the number of given bounds is large $(r=s)$, the cost of Phase 2 is the most important. But the last property is no more true in Experiment 2 where Phase 2 is inexpensive.

### *(ii) Comparison Between the Three Algorithms GDCA1, GDCA2, and CGDCA*

According to the results in Table 13.8, **CGDCA** is faster and more reliable than the two-phase algorithm **GDCA2**. In fact, the main difference between **CGDCA** and **GDCA2** is that the first phase of **GDCA2** is replaced by the smoothing technique in **CGDCA**; we then need not complete an approximate distance matrix (this is expensive in case of large dimension and the number of constraints is small) and work only with given distances (sparse distance matrices). On the other hand, among the two versions of the two-phase algorithm, **GDCA2** is faster than **GDCA1.**

### *(iii) Comparison Between the Performance of CGDCA for the Exact and the General Distance Geometry Problems*

Surprisingly enough, the results in Table 13.9 show that in general the reliability and the convergence rate of **CGDCA** increase when $\varepsilon$ decreases (the length of bounds decreases). Moreover, the algorithm is very efficient to the exact distance geometry problem (13.1). In other words, for our algorithm, the exact problem is easier than the general problem.

## 13.7 Conclusion

We have presented several approaches based on DC programming and DCA for solving large-scale molecular optimization problems from distance matrices. The main points in our approaches are:

1. Formulation and reformulation of the exact distance geometry problem (13.1) in various forms using the $l_1$ norm, the combined $l_1 - l_\infty$ norm, the Lagrangian duality, and the regularization techniques in DC programming
2. Reformulation in an elegant way the general distance geometry problem as bound constrained nonsmooth/smooth optimization problems
3. DC optimization algorithms with suitable DC decompositions to the resulting problems
4. Strategies of choosing a good starting point of DCA: the two-phase algorithm, the Smoothing technique via the Gaussian transform of the objective function, and the spaning tree procedure
5. Exploiting the nice effect of DC decompositions for DCA

Computational experiments show that our method is successful in locating the large configurations satisfying given distance constraints: the DCA globally solved distance geometry problems with up to 4,189 atoms (32,400 variables).

Several interesting issues arise from the present work. The first deals with the *reformulation* of the distance geometry problems. As indicated above, our new formulation has several advantages not only to DCA but also to existing methods for bound constrained smooth problems. Nevertheless, from the numerical experiments, we observe that the maximal error of distance constraints occurs on the same pair of atoms when the current point is *near* a solution. So it is interesting to consider an objective function dealing simultaneously with the sum of all errors and the maximal error of distance constraints. In other words, we can investigate a predictor-corrector algorithm to exploit the efficiency of DCA.

The second is the amelioration of our code by using a parallel solver.

The third concerns the data. In this chapter we generated the distance data from the complete protein given in PDB data bank which are real models (see Table 13.1). We follow the way of Moré–Wu [33] for constructing the data because our idea to use the continuation approach is suggested by their work. We wish to expand our testing to distance data generated by more realistic ways and to distance data derived from NMR experiments. These issues are currently in progress.

## References

1. Alfakih, A.Y., Khandani, A., Wolkowicz, H.: An interior-point method for the Euclidean distance matrix completion problem. Research Report CORR 97-9, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada
2. Blumenthal, L.M.: Theory and Applications of Distance Geometry. Oxford University Press (1953)

3. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
4. Le Thi H.A.: DC programming and DCA, available on the website http://lita.sciences.univ-metz.fr/~lethi/DCA.html
5. Demyanov, V.F., Vasilev, L.V.: Nondifferentiable optimization. Optimization Software, Inc. Publications Division, New York (1985)
6. De Leeuw, J.: Applications of convex analysis to multidimensional scaling. In: Barra, J.R., et al. (eds.) Recent Developments in Statistics, pp. 133–145. North-Holland Publishing Company (1977)
7. De Leeuw, J.: Convergence of the majorization method for multidimensional scaling. Journal of Classification **5**, 163–180 (1988)
8. Ding, Y., Krislock, N., Qian, J., Wolkowicz, H.: Sensor network localization, Euclidean distance matrix completions, and graph realization. Optim. Eng. **11**(1), 45–66 (2010)
9. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. J. Global Optim. **22**(1), 365–375 (2002)
10. Floudas, C., Adjiman, C.S., Dallwig, S., Neumaier, A.: A global optimization method, $\alpha$BB, for general twice differentiable constrained NLPs – I: theoretical advances. Comput. Chem. Eng. **22**, 11–37 (1998)
11. Glunt, W., Hayden, T.L., Raydan, M.: Molecular conformation from distance matrices. J. Comput. Chem. **14**, 114–120 (1993)
12. Havel, T.F.: An evaluation of computational strategies for use in the determination of protein structure from distance geometry constraints obtained by nuclear magnetic resonance. Progr. Biophys. Mol. Biol. **56**, 43–78 (1991)
13. Hendrickson, B.A.: The molecule problem: determining conformation from pairwise distances. Ph.D. thesis, Cornell University, Ithaca, New York (1991)
14. Hendrickson, B.A.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
15. Hirriart Urruty, J.B., Lemarechal, C.: Convex Analysis and Minimization Algorithms. Springer, Berlin (1993)
16. Huang, H.X., Liang, Z.A., Pardalos, P.M.: Some properties for the Euclidean distance matrix and positive semi-Definite matrix completion problems. Department of Industrial and Systems Engineering, University Florida (2001)
17. Krislock, N., Wolkowicz, H.: Euclidean distance matrices and applications. In: Anjos, M.F., Lasserre, J.B. (eds.) Handbook on Semidefinite, Conic and Polynomial Optimization, pp. 879–914 (2012)
18. Laurent, M.: Cuts, matrix completions and a graph rigidity. Math. Program. **79**(1-3), 255–283 (1997)
19. Le Thi, H.A.: Contribution à l'optimisation non convexe et l'optimisation globale: Théorie, Algorithmes et Applications. Habilitation à Diriger des Recherches, Université de Rouen, Juin (1997)
20. Le Thi, H.A., Le Hoai, M., Nguyen, V.V., Pham Dinh, T.: A DC Programming approach for Feature Selection in Support Vector Machines learning. Journal of Advances in Data Analysis and Classification **2**(3), 259–278 (2008)
21. Le Thi, H.A., Pham Dinh, T.: Solving a class of linearly constrained indefinite quadratic problems by d.c. algorithms. J. Global Optim. **11**, 253–285 (1997)
22. Le Thi, H.A., Pham Dinh, T.: D.c. programming approach for large scale molecular optimization via the general distance geometry problem. In: Floudas, C.A., Pardalos, P.M. (eds.) Optimization in Computational Chemistry and Molecular Biology: Local and Global Approaches, pp. 301–339. Kluwer Academic Publishers (2000)
23. Le Thi, H.A., Pham Dinh, T.: Large scale molecular optimization from distance matrices by a d.c. optimization approach. SIAM J. Optim. **14**(1), 77–114 (2003)
24. Le Thi, H.A., Pham Dinh, T.: A new algorithm for solving large scale molecular distance geometry problems. special issue of Applied Optimization, HighPerformance Algorithms and Software for Nonlinear Optimization, pp. 279–296. Kluwer Academic Publishers (2003)

25. Le Thi, H.A.: Solving large scale molecular distance geometry problems by a smoothing technique via the gaussian transform and d.c. programming. J. Global Optim. **27**(4), 375–397 (2003)
26. Le Thi, H.A., Pham Dinh, T.: The DC programming and DCA revisited with DC models of real world nonconvex optimization problems. Ann. Oper. Res. **133**, 23–46 (2005)
27. Le Thi, H.A., Pham Dinh, T., Huynh, V.N.: Convergence analysis of DC algorithm for DC programming with subanalytic data. Research Report, National Institute for Applied Sciences (2009)
28. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**(1), 1–17 (2008)
29. Mahey, P., Pham Dinh, T.: Partial regularization of the sum of two maximal monotone operators. Math. Model. Numer. Anal. ($M^2AN$) **27**, 375–395 (1993)
30. Mahey, P., Pham Dinh, T.: Proximal decomposition of the graph of maximal monotone operator. SIAM J. Optim. **5**, 454-468 (1995)
31. Moré, J.J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **8**, 814–836 (1997)
32. Moré, J.J., Wu, Z.: Issues in large-scale molecular optimization. preprint MCS-P539-1095, Argonne National Laboratory, Argonne, Illinois 60439, March 1996
33. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. preprint MCS-P628-1296, Argonne National Laboratory, Argonne, Illinois 60439, December 1996
34. Pham Dinh, T.: Contribution à la théorie de normes et ses applications à l'analyse numérique. Thèse de Doctorat d'Etat Es Science, Université Joseph Fourier-Grenoble (1981)
35. Pham Dinh, T.: Convergence of subgradient method for computing the bound norm of matrices. Lin. Algebra. Appl. **62**, 163–182 (1984)
36. Pham Dinh, T.: Algorithmes de calcul d'une forme quadratique sur la boule unité de la norme maximum. Numer. Math. **45**, 377–440 (1985)
37. Pham Dinh, T.: Algorithms for solving a class of non convex optimization problems. Methods of subgradients. Mathematics for Optimization, Elsevier Science Publishers B.V., North-Holland (1986)
38. Pham Dinh, T.: Duality in d.c. (difference of convex functions) optimization. Subgradient methods. Trends in Mathematical Optimization, International Series of Numer Math., vol. 84, pp. 277–293. Birkhäuser (1988)
39. Pham Dinh, T., Le Thi, H.A.: Stabilité de la dualité lagrangienne en optimisation d.c. (différence de deux fonctions convexes). C.R. Acad. Paris, t.318, Série I, pp. 379–384 (1994)
40. Pham Dinh, T., Le Thi, H.A.: Convex analysis approach to d.c. programming: Theory, Algorithms and Applications (dedicated to Professor Hoang Tuy on the occasion of his 70th birthday). Acta Mathematica Vietnamica **22**, 289–355 (1997)
41. Pham Dinh, T., Le Thi, H.A.: D.c. optimization algorithms for solving the trust region subproblem. SIAM J. Optim. **8**, 476–505 (1998)
42. Pham Dinh, T., Nguyen, C.N., Le Thi, H.A.: An efficient combined DCA and B&B using DC/SDP relaxation for globally solving binary quadratic programs. J. Global Optim. **48**(4), 595–632 (2010)
43. Polyak, B.: Introduction to optimization. Optimization Software, Inc. Publications Division, New York (1987)
44. Rockafellar, R.T.: Convex Analysis. Princeton University, Princeton (1970)
45. Saxe, J.B.: Embeddability of weighted Graphs in *k*-space is strongly NP-hard. In: Proceedings of the 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
46. Souza, M., Xavier, A.E., Lavor, C., Maculan, N.: Hyperbolic smoothing and penalty techniques applied to molecular structure determination. Oper. Res. Lett. **39**(6), 461–465 (2011)
47. Varga, R.: Matrix Iterative Analysis. Prentice Hall (1962)
48. Zou, Z., Richard, H.B., Schnabel, R.B.: A stochastic/pertubation global optimization algorithm for distance geometry problems. J. Global Optim. **11**, 91–105 (1997)

# Chapter 14
# Stochastic Proximity Embedding: A Simple, Fast and Scalable Algorithm for Solving the Distance Geometry Problem

**Dimitris K. Agrafiotis, Deepak Bandyopadhyay, and Eric Yang**

**Abstract** Stochastic proximity embedding (SPE) is a simple, fast, and scalable algorithm for generating low-dimensional Euclidean coordinates for a set of data points so that they satisfy a prescribed set of geometric constraints. Like other related methods, SPE starts with a random initial configuration and iteratively refines it by updating the positions of the data points so as to minimize the violation of the input constraints. However, instead of minimizing all violations at once using a standard gradient minimization technique, SPE stochastically optimizes one constraint at a time, in a manner reminiscent of back-propagation in artificial neural networks. Here, we review the underlying theory that gives rise to the SPE formulation and show how it can be successfully applied to a wide range of problems in data analysis, with particular emphasis on computational chemistry and biology.

**Keywords** Dimensionality reduction • Nonlinear mapping • Multidimensional scaling • Conformational analysis • Stochastic proximity embedding • SPE • Self-organizing superimposition • Self-organizing superposition • SOS • Conformational boosting • Distance geometry

D.K. Agrafiotis (✉)
Janssen Research & Development, Johnson & Johnson, Welsh & McKean Roads,
Spring House, PA 19477, USA
e-mail: dagrafio@its.jnj.com

D. Bandyopadhyay
GlaxoSmithKline, 1250 S. Collegeville Road, Mail Code UP12-210, Collegeville,
PA 19426, USA
e-mail: deepak.2.bandyopadhyay@gsk.com

E. Yang
Janssen Research & Development, Johnson & Johnson, Welsh & McKean Roads,
Spring House, PA 19477, USA
e-mail: eyang3@its.jnj.com

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_14,

## 14.1 Introduction

As large, high-dimensional data sets are becoming ubiquitous, the need for scalable methods to help elucidate their intrinsic structure is becoming more and more pressing. Machine learning approaches such as clustering, automated model building, and mathematical programming have been used for this task for many years with considerable success [43]. However, for many applications, the ability to employ human intuition through visual means can greatly enhance the exploration of the data space and offer insights that may not be possible from purely numerical methods. In order to do so, one must overcome the fundamental limitation of humans in visualizing multiple dimensions simultaneously. Dimensionality reduction techniques such as principal component analysis (PCA), singular value decomposition (SVD), or multidimensional scaling (MDS) [20, 28, 39] have been very effective in this regard, but their poor scaling with respect to the number of data point n—typically $O(n^2)$ or worse—makes them unsuitable for data sets beyond a few thousand items.

The fundamental problem with all the traditional dimensionality reduction algorithms is the need to compare each element of the data set to every other element, either through the Jacobian calculation as in MDS or a bidimensional matrix in PCA/SVD. This leads to poor scaling in both time and memory as the data sets become larger and larger. Stochastic proximity embedding (SPE) was formulated as a computationally efficient alternative to MDS. Unlike MDS, it is able to calculate an approximation of the direction of steepest descent through an iterative process using a small subset of points in $O(n)$ time, and still produce embeddings of equal quality. This makes the algorithm very fast and suitable for data sets beyond the reach of established methods. In addition, the method is trivial to implement and is relatively insensitive to missing data. More importantly, SPE can be generalized to embed data objects whose proximities are known only approximately or within certain ranges, which makes it an ideal alternative to distance geometry [22] for generating molecular conformations and addressing an array of related problems in structural chemistry and biology.

In the following sections, we summarize the basic elements and some of the key applications of the SPE algorithm and its variants. We hope to convince the reader that this is a useful and widely applicable data analysis technique, whose full potential remains relatively untapped.

## 14.2 Dimensionality Reduction

In its prototypical form, SPE generates coordinates for a set of points given only the distances between them. More specifically, given a set of $n$ objects, a symmetric matrix of proximities between these objects $r_{ij}$, and a set of images on a $m$-dimensional display map $\{x_i, i = 1, 2, \ldots, n; x_i \in \mathbb{R}^m\}$, SPE attempts to place $x_i$ into

**Algorithm 7**

```
Initialize the coordinates $x_i$ and select an initial learning rate $\lambda$
for ($C$ cycles) do
    for ($S$ steps) do
        select randomly a pair of points $i$ and $j$
        compute their relative distance $d_{ij} = ||x_i - x_j||$
        if ($d_{ij} \neq r_{ij}$) then
            $x_i = x_i + \lambda \frac{1}{2} \frac{r_{ij} - d_{ij}}{d_{ij} + \varepsilon} (x_i - x_j)$
            $x_j = x_j + \lambda \frac{1}{2} \frac{r_{ij} - d_{ij}}{d_{ij} + \varepsilon} (x_j - x_i)$
        end if
    end for
    Decrease the learning rate $\lambda$ by a prescribed decrement $\delta\lambda$
end for
```

the map in such a way that their Euclidean distances $d_{ij} = ||x_i - x_j||$ approximate as closely as possible the corresponding values $r_{ij}$ [5]. The method starts with random initial coordinates and iteratively refines them by repeatedly selecting two points at random and adjusting their positions so that their distance on the map $d_{ij}$ matches more closely their corresponding proximity $r_{ij}$. Details are given in Algorithm 7. This algorithm is controlled by four parameters: the number of cycles $C$, the number of steps $S$, the learning rate $\lambda$, and the term $\varepsilon$. $\lambda$ controls the magnitude of the correction and decreases over time from an initial value close to 1 to a final value close to 0. This forces the update rule to take more or less the full Newton–Raphson step at the initial cycles and successively reduces the magnitude of the updates as the embedding becomes more refined to prevent the algorithm from oscillating, particularly when the proximities $r_{ij}$ are not perfectly satisfiable (e.g., when attempting to embed the vertices of a tetrahedron in two dimensions so that their 3D distances are preserved—a clearly impossible task). The epsilon term, empirically chosen as $\varepsilon = 10^{-6}$, is added to $d_{ij}$ to prevent division by zero if points $i$ and $j$ happen to coincide. We have empirically determined that the algorithm scales linearly with the number of data points and that 10,000 total pairwise refinements per data point ($C \times S = n \times 10,000$) are sufficient to achieve a practically perfect embedding. The quality of the embedding is insensitive to the exact values of $C$ and $S$ as long as the total number of refinements, $C \times S$, is the same (in practice, we set $C \ll S$). That quality is measured by an error function known as stress, which is minimized during the course of the refinement:

$$S = \frac{\sum_{i<j} \frac{(d_{ij} - r_{ij})^2}{r_{ij}}}{\sum_{i<j} r_{ij}}.$$

The SPE algorithm has two primary uses: (1) reduce the dimensionality of high-dimensional data spaces and (2) produce coordinate vectors from data supplied directly in the form of proximities so that they can be analyzed with conventional statistical methods. SPE has been applied to visualize large chemical libraries to enable diversity profiling and compound selection [1, 3, 13], protein families to reveal sequence-function relationships [2, 6], QSAR models to identify molecular features that explain biological activity [26], cell images in high-content screens (HCS) to identify compounds that induce a particular biological effect [40], and nodes in wireless sensor networks to compute their optimal localization [17]. Each of these applications involved different object types (small organic molecules, protein sequences, QSAR models, HCS well abstractions, sensor nodes) and different proximity definitions (Euclidean distances in high-dimensional molecular descriptor spaces, evolutionary distances in sequence space, feature loadings in model space, differences in the distribution of phenotypic parameters), but in all cases the method produced meaningful maps that offered useful insights into the structure of the underlying data. We have also shown that the underlying nonlinear transforms produced by these embeddings can be "learned" by artificial neural networks, which can then be used to instantaneously embed additional objects beyond the original set with minimal distortion [11, 12, 14, 37].

## 14.3 Nonlinear Manifold Learning

The algorithm described above produces informative low-dimensional maps but does not guarantee that the resulting dimensions will have any physical significance. High-dimensional data often include extraneous and/or highly correlated dimensions that can cause the space to appear of higher dimensionality than it really is. Indeed, the *intrinsic* dimensions that truly determine the behavior of a system may be fewer than and different from the *apparent* dimensions that we measure through experimentation. Unfortunately, the relationship between those variables is often nonlinear, which greatly complicates the analysis. If we assume that the data of interest lies on an embedded nonlinear manifold within the higher-dimensional space, the task is to extract the true intrinsic dimensions from the apparent dimensions that serve as their surrogates.

The dimensionality and nonlinear geometry of a manifold are invariably embodied in the similarities between the data points. Consequently, many nonlinear manifold learning approaches try to embed the data points in a low-dimensional space so as to best preserve these similarities, a problem for which SPE is ideally suited. However, the question is what similarities to preserve. Conventional similarity measures such as the Euclidean distance tend to underestimate the proximity of points on a nonlinear manifold and can lead to erroneous embeddings. Consider, for instance, the "Swiss roll" example illustrated in Fig. 14.1. It is evident that the Euclidean distance does not always accurately represent the similarity between two points. To properly reconstruct this manifold, one needs to preserve the geodesic

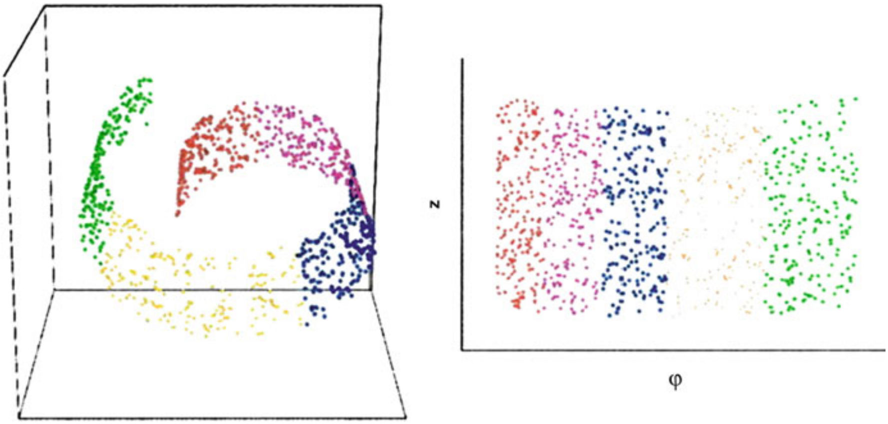


**Fig. 14.1** Swiss roll. (Left) Original 3D coordinates. (Right) Isometric SPE (ISPE) embedding in 2D. Standard Euclidean distances do not necessarily capture the similarity between the points, and in some cases, some of the red points would be identified as being more similar to the green points than to the magenta points. However, by preserving the geodesic distances of the points along the surface of the manifold, the intrinsic two-dimensional nature of the data set is revealed. Reproduced with permission from the *Proceedings of the National Academy of Sciences*, USA (DOI: http://dx.doi.org/10.1073/pnas.242424399)

distances between the data points, i.e., the lengths of the shortest paths between two points along the surface of the manifold itself [38, 41]. The problem is that these distances are not known a priori and must be inferred from the data sample.

While the Euclidean distance is not a good global approximation of the geodesic distance, it is reasonably accurate when two points are relatively close to each other. Moreover, the Euclidean distance (or any other proximity measure that is a true metric) is always smaller than the geodesic distance. Thus, when two points are close to each other, the input proximity provides a good approximation to their geodesic distance; when they are further away, the input proximity provides a lower bound to the geodesic distance.

The isometric variant of SPE (ISPE) [4, 15, 16] capitalizes on this observation by forcing the distances of nearby points on the low-dimensional map to match their corresponding input proximities and never allowing the distances between distant points—i.e., points whose input proximities are larger than a prescribed cutoff—to drop below those proximities. The error function in ISPE is given by

$$S = \frac{\sum\limits_{i<j} \frac{f(d_{ij}, r_{ij})}{r_{ij}}}{\sum\limits_{i<j} r_{ij}},$$

where

$$f(d_{ij}, r_{ij}) = \begin{cases} (d_{ij} - r_{ij}) & \text{if}\, r_{ij} < r_c \,\text{or}\, d_{ij} < r_{ij} \\ 0 & \text{otherwise.} \end{cases}$$

ISPE proceeds exactly the same way as regular SPE with the only difference being that the coordinate update is applied only when $r_{ij} \leq r_c$ or $d_{ij} < r_{ij}$; otherwise, the coordinates are left unchanged.

The utility of ISPE is illustrated in the mapping of 1,000 random conformations of methylpropylether ($C_1C_2C_3O_4C_5$), using RMSD as a measure of similarity between two conformations [4, 15, 16]. Since RMSD is calculated using all the atomic coordinates as input, it would appear that the space is 15-dimensional (five atoms times three spatial coordinates). In the SPE formalism, we can infer the intrinsic dimensionality of the conformational manifold by embedding the RMSD distance matrix into spaces of increasing dimensionality and identifying the minimum number of dimensions required to achieve a perfect embedding (i.e., where the distances of conformations on the map are the same as the respective RMSDs and the stress is zero). As shown in Fig. 14.2b, in the case of regular SPE, the residual stress cannot be entirely eliminated in less than 15 dimensions (only five shown here), but with its isometric variant a perfect embedding can be achieved in two dimensions. Figure 14.2a shows that these two dimensions correspond to the dihedral angles of the two central rotatable bonds, $\phi(C_2C_3O_4C_5)$ and $\phi(C_1C_2C_3O_4)$, which we know from the molecule's covalent structure to be its only true degrees of conformational freedom.

The method can also be used to visualize the general structure and clustering of a data set, even when its intrinsic dimensionality is higher than 2 or 3. Figure 14.3 shows a two-dimensional embedding of a combinatorial library obtained by ISPE. Even though the intrinsic dimensionality of the molecular diversity space is much higher than 2, the resulting two-dimensional map reveals chemically meaningful clusters.

## 14.4 Conformational Sampling

The SPE algorithm lends itself naturally to one of the most central problems in computational and structural chemistry, namely the generation of a molecule's three-dimensional structure(s) directly from its connection table (the topology of the molecule described by a set of atoms, bonds, and stereochemical flags). Since most flexible molecules can assume multiple 3D conformations that coexist in equilibrium, the problem can be formulated either as *conformation generation* which aims at producing a single representative structure (typically the one that corresponds to the global energy minimum) or conformational sampling which seeks to identify all energetically accessible conformations.

Conformation generation was first formulated as an embedding problem by Crippen and Havel, who derived a set of geometric constraints from a molecule's

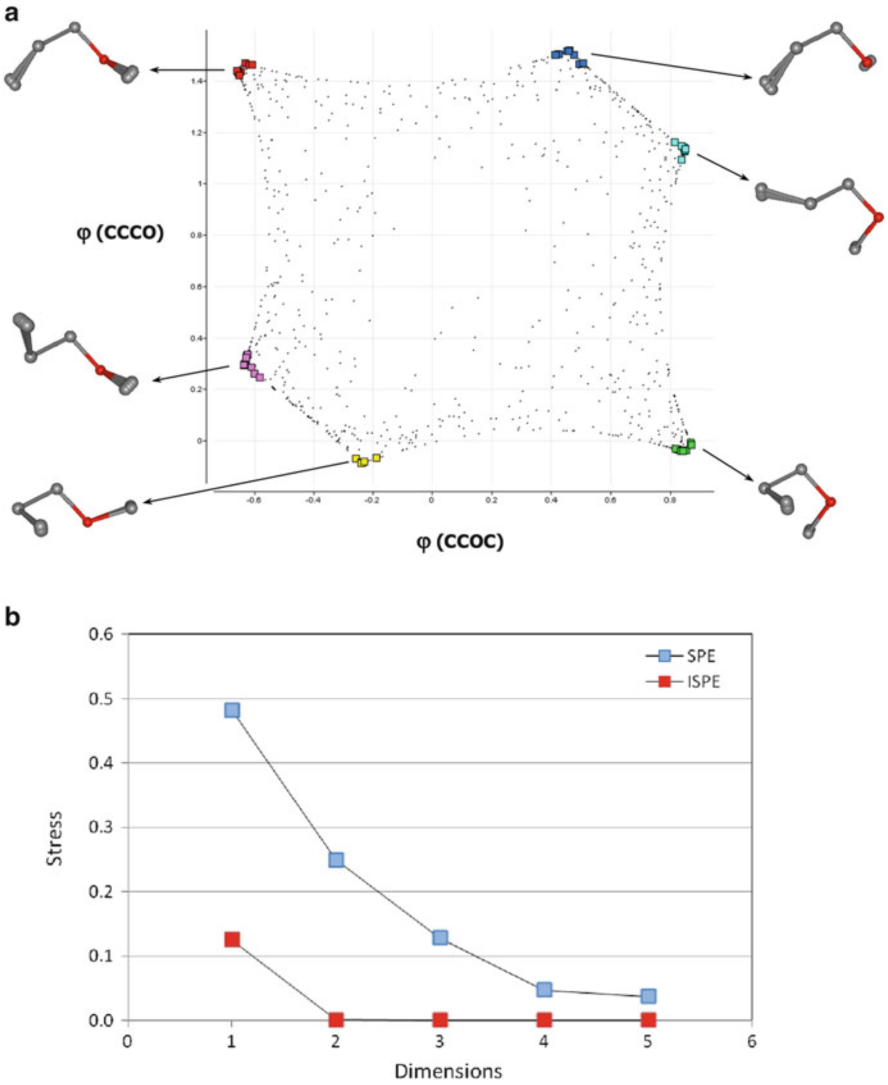


**Fig. 14.2** Embedding of 1,000 conformations of methylpropylether, $C_1C_2C_3O_4C_5$, generated by a distance geometry algorithm and compared by RMSD. (**a**) Two-dimensional embedding obtained by ISPE. Representative conformations are shown next to highlighted points in different parts of the map. The horizontal and vertical directions represent the torsional angles $\phi C_2C_3O_4C_5$ and $\phi C_1C_2C_3O_4$, respectively. The unoccupied upper-left and bottom-right corners represent conformations that are inaccessible because of the steric hindrance between the two terminal carbon atoms $C_1$ and $C_5$. (**b**) Final stress obtained by ISPE (mean and standard deviation over 30 independent runs) and regular SPE (SPE) as a function of embedding dimensionality. Reproduced with permission from the *Proceedings of the National Academy of Sciences*, USA ([15], DOI: http://dx.doi.org/10.1073/pnas.242424399)

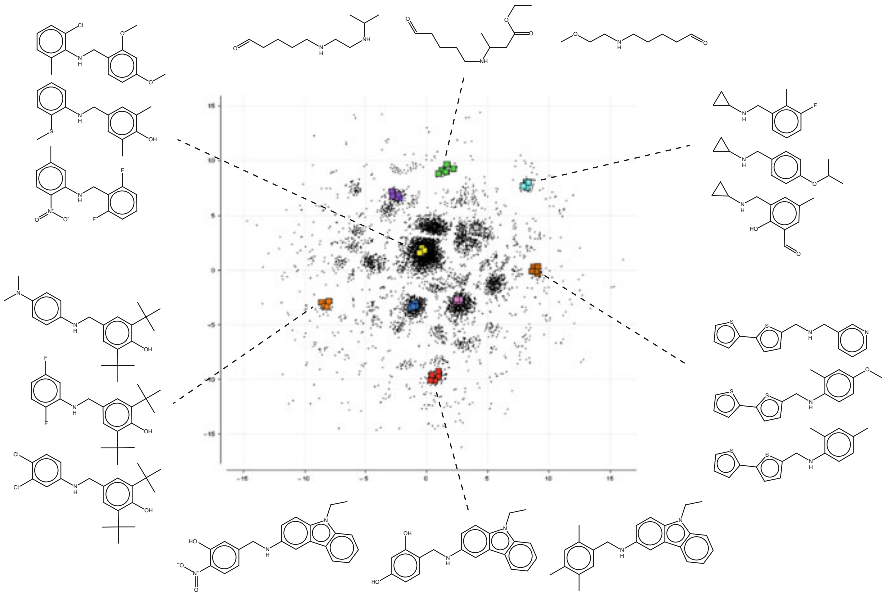


**Fig. 14.3** Two-dimensional embedding of a combinatorial library obtained by ISPE and a neighborhood radius of 0.4. Even though the intrinsic dimensionality of the molecular diversity space is much higher than 2, the resulting two-dimensional map reveals chemically meaningful clusters. Reproduced with permission from the *Journal of Chemical Information and Computer Science* ([16], DOI: http://dx.doi.org/10.1021/ci025631m)

covalent structure and then used distance geometry to embed the atoms in $\mathbb{R}^3$ so as to minimize the violation of these constraints [23]. Two types of constraints can be defined: (1) distance constraints that require the distance between two atoms $i$ and $j$, $d_{ij}$, to fall within a certain range $l_{ij} \leq d_{ij} \leq u_{ij}$ and (2) volume constraints that require the signed volume $V_{ijkl}$ formed by four atoms $i$, $j$, $k$, $l$ to fall within a certain range $V^l_{ijkl} \leq V_{ijkl} \leq V^u_{ijkl}$. Distance constraints are derived from standard bond lengths and angles and are used to enforce the proper covalent geometry, while volume constraints are used to enforce the correct chirality of stereocenters and the planarity of conjugated systems. Collectively, these constraints define all possible three-dimensional geometries attainable by a given molecule. Distance geometry has been applied to conformational analysis [30] as well as a number of related problems such as protein structure prediction [25], ligand docking [34], and NMR structure determination [24, 29, 35] and has been shown to identify conformations missed by alternative systematic search methods [33].

Once the constraints are defined, the embedding essentially comprises two minimization problems, the minimization of distance constraint violations and the minimization of volume constraint violations, a problem that is ideally suited for SPE. We formulate this problem as the minimization of an objective function comprised of two parts:

$$S = S_d + S_v = \sum_{i<j} f(d_{ij}, u_{ij}, l_{ij}) + \alpha \sum_k h(V_k, V_k^l, V_k^u). \tag{14.1}$$

$S_d$ captures the violation of distance constraints which are defined as

$$f(d_{ij}, u_{ij}, l_{ij}) = \begin{cases} (d_{ij} - u_{ij})^2 & \text{if}\, d_{ij} > u_{ij}, \\ (d_{ij} - l_{ij})^2 & \text{if}\, d_{ij} < l_{ij}, \\ 0 & \text{otherwise.} \end{cases}$$

This equation resembles the distance refinements used in nonlinear manifold learning, except that instead of setting an upper bound to determine whether an update will occur, in conformation generation the objective function is non-zero only when the distance between two atoms falls outside the lower and upper bounds.

$S_v$ captures the violation of volume constraints which are defined as

$$h(V_k, V_k^l, V_k^u) = \begin{cases} (V_k - V_k^l)^2 & \text{if}\, V_k < V_k^l, \\ (V_k - V_k^u)^2 & \text{if}\, V_k > V_k^u, \\ 0 & \text{otherwise,} \end{cases}$$

where $V_k$, $V_k^u$, and $V_k^l$ represent the current value and the upper and lower bounds of the $k^{th}$ volume constraint, respectively.

Minimizing the error function $S$ with respect to the atomic coordinates generates conformations that satisfy the distance and volume constraints. In SPE, this is done by first assigning random initial coordinates to the atoms and then refining their positions in an iterative fashion by selecting one random constraint at a time (either a distance or a volume constraint, each with different probability) and adjusting the positions of the constituent atoms so as to satisfy that constraint [8,44]. The parameter $\alpha$ in Eq. (14.1) determines the relative frequency by which volume and distance constraints are sampled and therefore their relative weight in the error function. By convention, we set $\alpha = 0.1$, which means that distance constraints are sampled ten times more frequently than volume constraints. Algorithm 8 provides more details about this algorithm. An illustration of the algorithm is in Fig. 14.4.

By running this algorithm multiple times, each time starting from a different random initial configuration and random number seed, multiple conformations that satisfy all the constraints can be produced. However, because SPE does not explicitly take into account the potential energy of the molecule, it may be necessary to refine the resulting geometries with standard energy minimization techniques to remove minor imperfections and drive them towards their nearest local minima. While it may seem conceptually simpler to utilize a single optimization method, SPE is much more efficient at generating good starting geometries and eliminating unrealistic conformations more rapidly than traditional methods. We must also point out that for non-trivial molecules it is unlikely that SPE will generate the global minimum with a single embedding since that is just as viable as any other conformation that satisfies all the geometric constraints. Thus, identifying the

**Algorithm 8**

```
Initialize geometric constraints. Define lower $\{l_{ij}\}$ and upper $\{u_{ij}\}$ distance bounds for every pair of atoms $i$ and $j$. Define upper $\{V^u_{ijkl}\}$ and lower $\{V^l_{ijkl}\}$ volume bounds for every set of four atoms $i$, $j$, $k$, $l$ that are attached to tetrahedral atoms with explicit chirality or required to have a planar configuration.
Assign random coordinates $x_i$ to every atom in the molecule, select an initial distance learning rate $\lambda_d$, an initial volume learning rate $\lambda_v$, and a relative frequency for enforcing a distance or a volume constraint $v$.
for ($C$ cycles) do
  for ($S$ steps) do
    if (probability $v$) then
      Randomly select a pair of atoms, $i$ and $j$, and compute their distance $d_{ij} = ||x_i - x_j||$.
      if ($l_{ij} \leq d_{ij} \leq u_{ij}$ is not satisfied) then
        update the coordinates $x_i$ and $x_j$ by
          $x_i = x_i + \lambda_d \frac{1}{2} \frac{t_{ij} - d_{ij}}{d_{ij} - \varepsilon}(x_i - x_j),$
          $x_j = x_j + \lambda_d \frac{1}{2} \frac{t_{ij} - d_{ij}}{d_{ij} - \varepsilon}(x_j - x_i).$
        where $t_{ij}$ is the nearest bound to $d_{ij}$ (i.e. $t_{ij} = l_{ij}$ if $d_{ij} < l_{ij}$, or $t_{ij} = u_{ij}$ if $d_{ij} > u_{ij}$), and $\varepsilon$ is a small number used to avoid divisions by zero.
      end if
    else
      Randomly select a volume constraint $k$, and the four atoms involved, $p$, $q$, $s$, $t$. Compute the signed volume $V_{pqst}$ formed by the 4 atoms.
      if ($V^l_k < V_{pqst} < V^u_k$ is not satisfied) then
        compute the gradient of the signed volume with respect to the atomic positions, $\mathbf{g}_\mu = \Delta_\mu V_{pqst}$, where $\mu = p,q,s,t$, and update the atomic coordinates by
          $x_\mu = x_\mu + \lambda_v (V^0_k - V_{pqst}) \frac{\mathbf{g}_\mu}{\sum_{\beta=p,q,s,t} |\mathbf{g}_\mu|^2},$
        where $V^0_k$ is the nearest bound to $V_{pqst}$ (i.e. $V^0_k = V^l_k$ if $V_{pqst} < V^l_k$, or $V^0_k = V^u_k$ if $V_{pqst} > V^u_k$).
      end if
    end if
  end for
  Decrease the learning rate $\lambda_d$ and $\lambda_v$ by prescribed decrements $\delta\lambda_d$ and $\delta\lambda_v$.
end for
```

global minimum would necessitate extensive conformational sampling achieved by performing multiple independent embeddings and minimizing the resulting structures.

Extensive testing has shown that the conformations generated via SPE tend to be relatively compact. While for some applications this may be irrelevant or even desirable (e.g., when one seeks a single representative conformation or the global minimum in the gas phase), it can be limiting in other contexts, for example, in identifying the bioactive conformations of biological ligands, many of which are known to bind to their biomolecular hosts in relatively extended forms. To minimize the

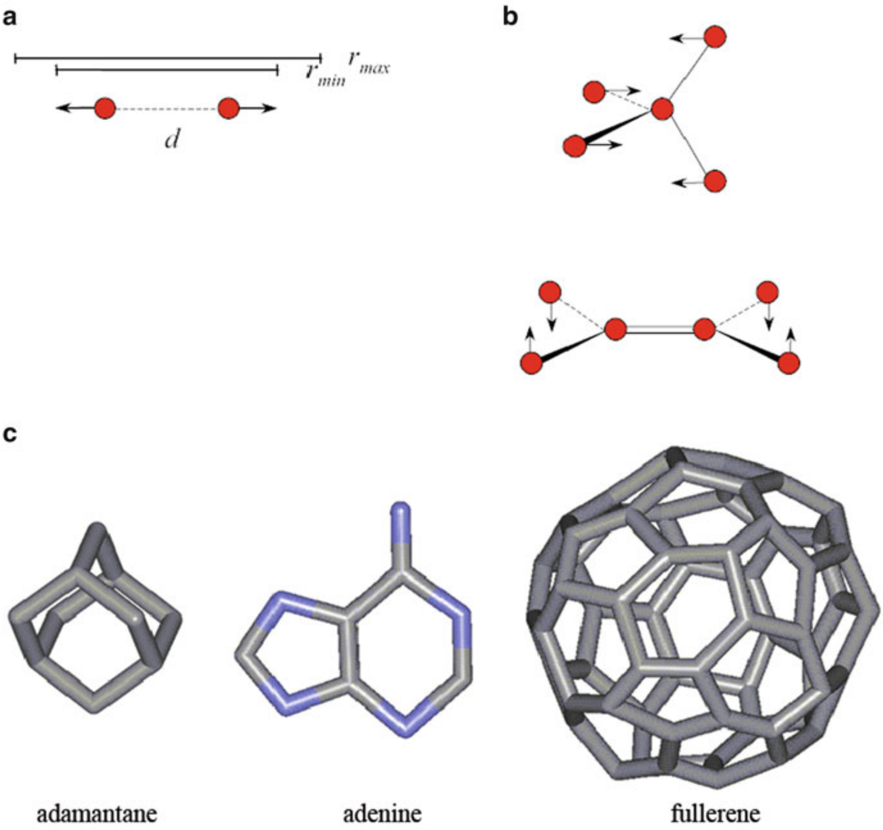


**Fig. 14.4** Refinement steps used by the SPE conformational sampling algorithm. (**a**) Distance constraints are satisfied through pairwise atomic adjustments that bring the atoms towards their target range (specified by the lower and upper bounds). (**b**) Volume constraints, used to enforce the correct chirality of stereocenters and planarity of $\pi$ systems, are satisfied by adjusting the positions of the four atoms attached to the corresponding chiral center or double bond. (**c**) Representative conformations obtained for adamantane, adenine, and fullerene. These geometries are within 0.04, 0.08, and 0.24 Å RMSD from their corresponding nearest energy minima. Caged molecules like adamantane and fullerene are beyond the reach of most conformational search algorithms currently in use

bias for compact geometries, a boosting heuristic has been devised that can be used in conjunction with SPE to generate increasingly extended conformations iteratively [9, 27]. The process starts by generating an initial conformation $c_1$ using standard SPE. The lower bounds of all atom pairs $\{l_{ij}\}$ are then replaced by the corresponding actual interatomic distances $\{d_{ij}\}$ in conformation $c_1$ and used along with the unchanged upper bounds $\{u_{ij}\}$ and volume constraints $\{V^u_{ijkl}\}$ and $\{V^l_{ijkl}\}$ to perform a second embedding to generate another conformation, $c_2$. This process (which is illustrated in Fig. 14.5) is repeated for a prescribed number of iterations and rerun

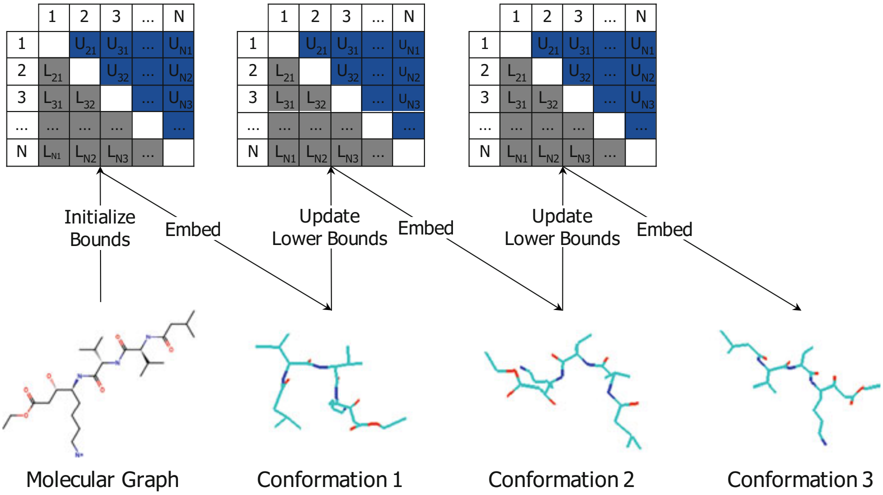


**Fig. 14.5** Illustration of the conformational boosting algorithm. The method involves a sequence of embeddings, where each successive embedding uses the actual interatomic distances in the preceding conformation as lower (or upper) bounds. The method generates increasingly extended (or compact) conformations, while remaining stochastic in nature. The updated bounds are by definition satisfiable because there is at least one conformation that satisfies them—the one generated in the preceding step

with different random number seeds to produce multiple conformations (note that increasingly compact conformations can be generated analogously, by replacing upper rather than lower bounds). It has been shown that the conformations generated via this method cover a wider range of geometries than utilizing standard SPE.

An alternative application of conformational sampling is the generation of elegant two-dimensional drawings of chemical structures. Instead of embedding the atoms in three dimensions using 3D geometric constraints, we can embed them in two dimensions using distance constraints specifically designed to ensure an optimal 2D layout of the molecular graph. These include forcing all bonds to have the same length and requiring all atoms to be in an ideal 2D geometry (e.g., an atom with three attachments should have a perfect trigonal geometry with attached atoms 120° from each other, etc.). Once the embedding has completed, wedges are applied to indicate the pre-designated stereochemistry of chiral centers. The algorithm works very well for bridged, caged, or otherwise sterically hindered molecules which are very difficult to handle with empirical drawing rules. This method is used to render chemical structure drawings in third dimension explorer (3DX), the application front-end to Janssen R&D's discovery informatics platform known as ABCD [7,21].

## 14.5 Self-organizing Superimposition

Though SPE has proven successful in generating valid conformations rapidly, the method can be further improved. Self-organizing superimposition (SOS) is a variant of SPE devised in an effort to improve the quality of the resulting geometries and speed up the embedding [46]. SOS leverages the fact that many fragments of a given molecule are inherently rigid. For instance, the atoms of a phenyl group can be prearranged in a regular hexagon and refined in unison, instead of relying on continuous pairwise atomic adjustments to enforce both local and global geometry.

SOS works by interleaving the standard pairwise distance refinements of SPE with template fitting operations aimed at fixing the geometry of a rigid fragment in a single step. First, SOS breaks down the molecule into a list of rigid fragments and assigns a precomputed geometry to each one. This geometry can either be retrieved from a database of reference templates or computed on the fly using SPE. The algorithm then starts sampling pairwise distance constraints and making coordinate adjustments as in standard SPE. However, after adjusting the coordinates of an atom to satisfy the selected distance constraint, the SOS algorithm checks whether that atom is part of a rigid fragment. If that is the case, SOS moves all the atoms in the fragment towards their ideal geometry by superimposing the reference template with the current configuration of the corresponding atoms to minimum RMSD and then replacing the atomic coordinates with those in the superimposed template (see Fig. 14.6).

The benefit of this method is that it can rapidly optimize the geometry of a rigid group of atoms in a single iteration rather than the several hundred iterations that standard SPE would require. In essence, the algorithm resolves steric clashes by updating interatomic distances and then reconstructs the local geometry of rigid fragments if the update step has altered their structure. Since chirality and planarity are already represented in the template conformations, SOS no longer needs the volume constraints used in SPE.

Conformations generated by SOS require fewer refinements than those from SPE, and are also of much higher quality, since atoms within rigid fragments are forced to conform to their ideal local geometry. Furthermore, SOS can be combined with conformational boosting to increase its ability to sample conformation space and was found to be particularly effective for macrocycles, a class of molecules that are particularly challenging for other methods.

## 14.6 Validation of Conformational Sampling

A thorough comparison with other conformational sampling algorithms including Catalyst, Macromodel, Omega, MOE, CAESAR, and Rubicon revealed that SPE was significantly faster and yielded a much wider range of low-energy conformations compared to these other methods, which showed a bias towards more extended or more compact geometries [10].

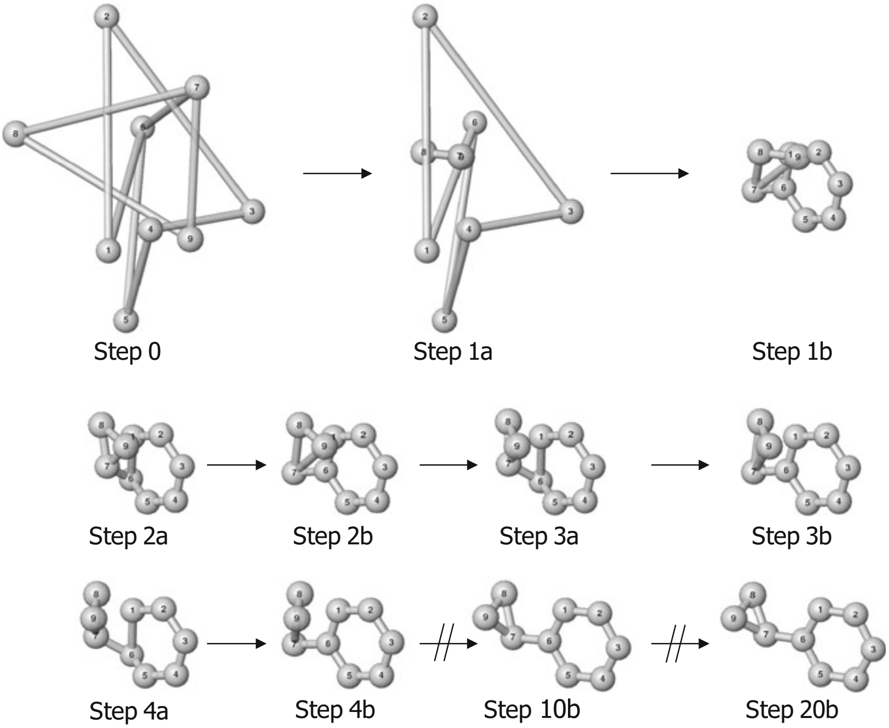


**Fig. 14.6** Illustration of the SOS algorithm for conformational sampling. The method involves alternating template superpositions and pairwise distance adjustments to generate the final conformation. Template superposition allows the atomic coordinates of rigid fragments to be adjusted at once without violating their internal geometry. Distance adjustments are used for pairs between two different fragments and are used to remove steric classes. Stereochemistry and planarity are naturally preserved. Reproduced with permission from the *Journal of Computational Chemistry* (DOI: http://dx.doi.org/10.1002/jcc.20622)

This comparison was subsequently extended to macrocyclic molecules [19] such as cycloglycines, cyclodextrins, and cyclic bioactive peptides, which are notoriously difficult to handle by other methods because of the ring closure problem. This study confirmed the superiority of SPE and SOS by every relevant metric, including speed, diversity of conformations and pharmacophores, energy range, and ability to identify the global minimum.

Because one of the main uses of conformational sampling is to generate biologically relevant conformations, we also compared SPE and SOS to other methods with regard to their ability to identify bioactive conformations [42]. This was assessed by allowing each method to generate the same number of conformations and comparing how many of them were within 0.5, 1, 1.5, or 2 Å RMSD from a known co-crystal structure. It was found that SPE and SOS were able to identify a comparable number of bioactive conformations after only 500 trials that other methods required 10,000 or more trials to find, suggesting a higher intrinsic propensity to sample biologically relevant conformational space. Finally, it is important to note that SPE and SOS are able to accomplish this goal while simultaneously decreasing the amount of memory and CPU time required to generate each conformation.

## 14.7 Pharmacophore Modeling

Pharmacophore modeling is the process of identifying the spatial arrangement of key molecular features that are necessary for recognition of a ligand by a biological macromolecule. The pharmacophore is usually derived from a set of known actives and is subsequently used to search large chemical databases for novel ligands possessing that pharmacophore and thus likely to bind to the same macromolecule. The first step is being able to identify the common pharmacophore given a set of active compounds. This is related to the previously described process of conformational analysis, insomuch as the goal is to identify conformations that may be biologically active. However, since each ligand may assume multiple conformations, the challenge is to determine which combinations of these conformations can be aligned sufficiently well so that their key pharmacophoric features are presented in a similar relative orientation in 3D space.

To reduce the combinatorial complexity of this problem, one approach is to perform the conformational sampling and the alignment in a single step. The so-called ensemble SPE [18] maps pharmacophoric features onto each molecule, abstracts them by the centroids of the atoms comprising each feature, and sets up additional distance constraints between these centroids to force matching features to overlap. Distances to external points, such as protein atoms, can be also used to signify H-bond or other interactions (if upper bound is specified) and excluded volumes around the protein atoms (if lower bound is specified).

The ensemble SPE algorithm differs from that for conformational analysis in a few crucial steps. The ensemble is stored as a single molecule with many components, and distance, volume, and external constraints are only enforced between atoms in the same component. The input molecules are used to derive pairwise pharmacophore distance constraints and satisfied when matching atoms or groups from both molecules in the pair are superimposed (lower/upper bounds 0 and 0.05 Å). Rings and hydrophobic regions are represented by centroids; aromatic rings also include bidirectional normal vectors that must be superimposed to within 18° by a pairwise pharmacophore angle constraint. This constraint is satisfied in a superimposed pair of rings by rotating atoms of the second ring about an axis through their centroid perpendicular to both ring normals. The distance of each rotating atom to this axis provides an arc length that can substitute for distance in SPE's distance error function.

After SPE convergence of the ensemble and energy minimization of the individual molecules, a valid pharmacophore hypothesis produces a well-aligned set of molecules in low-energy conformations, from which 3D pharmacophore point coordinates, distances, and tolerances can be extracted.

## 14.8 Loop Modeling

Protein loops are the flexible short segments connecting two rigid secondary structural units in proteins. Loops tend to be located at the solvent-exposed surface of globular proteins and play an important role in ligand binding, allosteric regulation, and other protein functions. Because they are conformationally flexible, they are often associated with regions of low electron density and are difficult to resolve through X-ray crystallography. In fact, their movement is often an important and necessary component of their function. Thus, to understand their function, we need to know the conformations they can attain.

Just like macrocycles, protein loops are particularly challenging because one needs to seamlessly bridge the gap between the anchor points without introducing any steric clashes. Most loop modeling algorithms generate the loop structure and resolve steric clashes in two distinct steps. The SPE/SOS formalism makes it easy to incorporate the additional anchor constraints and allows low-energy loop conformations to emerge naturally during the search [32].

To generate the loop conformations, the SOS algorithm is modified so that the terminal atoms are constrained to predefined fixed positions. After this initial placement, randomly selected rigid fragments in the protein loop are updated via least-squares fitting, and steric clashes are resolved through pairwise refinements between atoms in different fragments (see Fig. 14.7). The embedding is run until the maximum atomic displacement during template superimposition and the end-point violations are below prescribed thresholds. Since template superimpositions are the rate-limiting step, a novel method to compute the optimal rotational matrix was devised that sped up calculations by 100 % [31].

When compared to other algorithms such as CCD and CSJD on a data set with three sets of ten loops each with 4, 8, and 12 residues, SOS consistently produced good backbone conformations with a mean best RMSD of 0.20, 1.19, and 2.29 Åand runtime of 5, 13, and 19 ms. It is also straightforward to incorporate additional geometric constraints derived from experimental techniques such as 2D NMR and fluorescent resonance energy transfer or from known interactions such as hydrogen bonds or saltbridges. Our work has shown that it is easy to model multiple interlocking loops and solve multiple loop problems, two problems that pose a tremendous challenge to conventional loop construction algorithms.

## 14.9 Graphics Processing Unit Acceleration

While the SPE algorithm has proven to be efficient and rapid, there is still a desire to operate on larger data sets and in near real time. Graphics processing units (GPUs) represent a low-cost highly parallel processor containing anywhere from ten to several hundred independent compute units, each with a small amount of high-speed local memory. On algorithms that can be readily parallelized, GPUs have

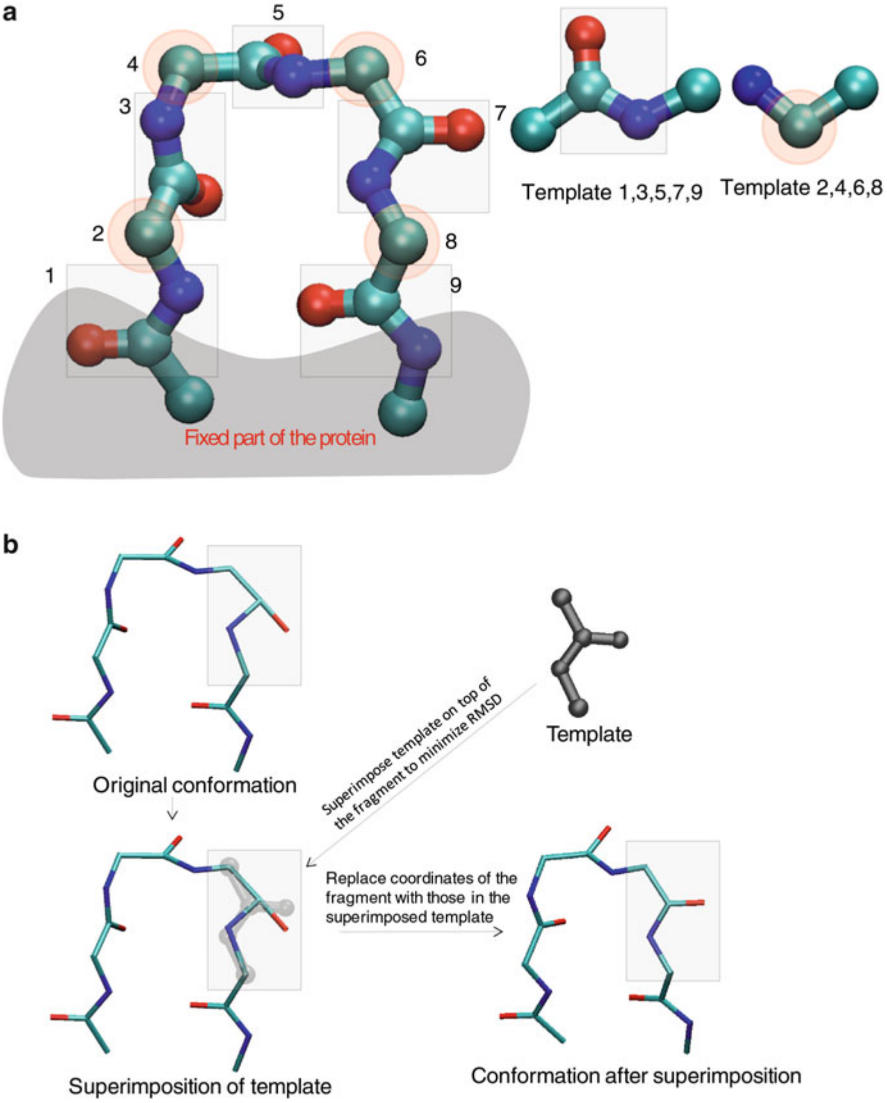

a
1
2
3
4
5
6
7
8
9
Template 1,3,5,7,9
Template 2,4,6,8
Fixed part of the protein
b
Original conformation
Template
Superimpose template on top of the fragment to minimize RMSD
Superimposition of template
Replace coordinates of the fragment with those in the superimposed template
Conformation after superimposition

**Fig. 14.7** SOS algorithm for generating conformations of protein loops. (**a**) Identification of rigid fragments. These are the subgraphs that remain after all rotatable bonds (acyclic non-terminal single bonds, which are not part of a small ring or a delocalized system) are removed. The atoms directly attached to a rigid fragment are also included in the template. (**b**) Template superposition step. The five atoms of the amide bond are adjusted at once by fitting the idealized template geometry onto the current fragment. Reproduced with permission from *PLoS Computational Biology* (DOI: http://dx.doi.org/10.1371/journal.pcbi.1000478)

shown speedups of 100-fold versus conventional CPUs. On algorithms that are not embarrassingly parallel, speedups of 10–20-fold are still easily within reach. SPE is an attractive candidate for multi-threading as well as porting to a GPU architecture because its inner loop, if executed in parallel, could yield substantial increases in performance.

To obtain maximum performance from the GPU implementation, a slight modification to SPE is warranted. The pivoted variant of SPE [36] selects a random pivot in the outermost loop and a random set of points for refinement in the innermost loop. The coordinates of this inner set of points can be updated independently, which makes it more suitable for parallelization. Doing so eliminates potential race conditions, i.e., incorrect updates as multiple threads attempt to update the same point simultaneously.

This GPU implementation of SPE can result in speedups of anywhere from 20× to 30× compared to a single-threaded CPU implementation, which is in line with other algorithms that have been ported to GPUs [45]. Compared to a multi-threaded implementation running on eight CPU cores, the speedup was reduced to tenfold, still a considerable improvement. However, it should be noted that in the example cases, the GPU was underutilized since our data sets of about one million points were still not large enough to fully occupy all the computational units in the GPU. Given the overhead of synchronization, we found that SPE performed best when utilizing around 20 % of the available computational units. Thus, data sets consisting of five million points could be tackled without requiring an increase in the run time with the GPU fully utilized, resulting in greater than tenfold speedups versus multi-threaded implementations.

GPUs provide several advantages over CPUs for SPE embedding. First, the GPU contains many more execution units than a typical CPU. Also, since the execution of SPE consists of one simple update rule, in many cases even a single GPU execution unit performs faster than a CPU. This is primarily due to the relatively large high-speed memory buffer present on the GPU but not on the CPU. Finally, we have shown that the thread synchronization construct on the GPU is much more efficient than the barrier synchronization construct on the CPU. This means that for the SPE-based algorithms presented previously, the benefits of switching over to GPUs can be quite substantial.

## 14.10 Conclusions

Stochastic proximity embedding is ideally suited to any class of problems that involve finding optimal solutions that satisfy a set of geometric constraints and which may not be easy to tackle by deterministic optimization techniques. While we have presented examples focused primarily on dimensionality reduction and structural chemistry and biology, it should be possible to apply SPE to broader classes of optimization problems. This is because the underlying mathematics of SPE is based upon a well-characterized optimization technique. Furthermore, because of its conceptual simplicity, SPE can be easily ported to a number of parallel architectures, allowing us to tackle large problems in near real time.

## References

1. Agrafiotis, D.K.: Stochastic algorithms for maximizing molecular diversity. J. Chem. Inform. Comput. Sci. **37**(5), 841–851 (1997)
2. Agrafiotis, D.K.: A new method for analyzing protein sequence relationships based on Sammon maps. Protein Sci. **6**(2), 287–293 (1997)
3. Agrafiotis, D.K.: Diversity of chemical libraries. In: Allinger, N.L., Clark, T., Gasteiger, J., Kollman, P.A., Schaefer III, H.F., Schreiner, P.R. (eds.) The Encyclopedia of Computational Chemistry, vol. 1, pp. 742–761. Wiley, Chichester (1998)
4. Agrafiotis, D.K.: Exploring the nonlinear geometry of sequence homology. Protein Sci. **12**, 1604–1612 (2003)
5. Agrafiotis, D.K.: Stochastic proximity embedding. J. Comput. Chem. **24**, 1215–1221 (2003)
6. Agrafiotis, D.K.: Exploring the nonlinear geometry of sequence homology. Protein Sci. **12**, 1604–1612 (2003)
7. Agrafiotis, D.K., Alex, S., Dai, H., Derkinderen, A., Farnum, M., Gates, P., Izrailev, S., Jaeger, E.P., Konstant, P., Leung, A., Lobanov, V.S., Marichal, P., Martin, D., Rassokhin, D.N., Shemanarev, M., Skalkin, A., Stong, J., Tabruyn, T., Vermeiren, M., Wan, J., Xu, X.Y., Yao, X.: Advanced Biological and Chemical Discovery (ABCD): centralizing discovery knowledge in an inherently decentralized world. J. Chem. Inform. Model. **47**(6), 1999–2014 (2007)
8. Agrafiotis, D.K., Bandyopadhyay, D., Carta, G., Knox, A.J.S., Lloyd, D.G.: On the effects of permuted input on conformational sampling of druglike molecules: an evaluation of stochastic proximity embedding (SPE). Chem. Biol. Drug. Des. **70**(2), 123–133 (2007)
9. Agrafiotis, D.K., Gibbs, A., Zhu, F., Izrailev, S., Martin, E.: Conformational boosting. Aust. J. Chem. **59**, 874–878 (2006)
10. Agrafiotis, D.K., Gibbs, A., Zhu, F., Izrailev, S., Martin, E.: Conformational sampling of bioactive molecules: a comparative study. J. Chem. Inform. Model. **47**, 1067–1086 (2007)
11. Agrafiotis, D.K., Lobanov, V.S.: Nonlinear mapping networks. J. Chem. Inform. Comput. Sci. **40**, 1356–1362 (2000)
12. Agrafiotis, D.K., Lobanov, V.S.: Multidimensional scaling of combinatorial libraries without explicit enumeration. J. Comput. Chem. **22**(14), 1712–1722 (2001)
13. Agrafiotis, D.K., Lobanov, V.S., Salemme, F.R.: Combinatorial informatics in the post-genomics era. Nat. Rev. Drug. Discov. **1**, 337–346 (2002)
14. Agrafiotis, D.K., Rassokhin, D.N., Lobanov, V.S.: Multidimensional scaling and visualization of large molecular similarity tables. J. Comput. Chem. **22**(5), 488–500 (2001)
15. Agrafiotis, D.K., Xu, H.: A self-organizing principle for learning nonlinear manifolds. Proc. Natl. Acad. Sci. USA **99**, 15869–15872 (2002)
16. Agrafiotis, D.K., Xu, H.: A geodesic framework for analyzing molecular similarities. J. Chem. Inform. Comput. Sci. **43**, 475–484 (2003)
17. Allor, G., Jacob, L.: Distributed wireless sensor network localization using stochastic proximity embedding. Comput. Comm. **33**, 745–755 (2010)
18. Bandyopadhyay, D., Agrafiotis, D.K.: A self-organizing algorithm for molecular alignment and pharmacophore development. J. Comput. Chem. **29**, 965–982 (2009)
19. Bonnet, P., Agrafiotis, D.K., Zhu, F., Martin, E.J.: Conformational analysis of macrocycles: finding what common search methods miss. J. Chem. Inform. Model. **49**, 2242–2259 (2009)
20. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications. Springer, New York (1997)

21. Cepeda, M.S., Lobanov, V.S., Farnum, M., Weinstein, R., Gates, P., Agrafiotis, D.K., Stang, P., Berlin, J.A.: Broadening access to electronic health care databases. Nat. Rev. Drug. Discov. **9**, 84 (2010)
22. Crippen, G.M.: Rapid calculation of coordinates from distance matrices. J. Comput. Phys. **26**, 449–452 (1978)
23. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
24. Havel, T.F., Wüthrich, K.: An evaluation of the combined use of nuclear magnetic resonance and distance geometry for the determination of protein conformations in solution. J. Mol. Biol. **182**, 281–294 (1985)
25. Huang, E.S., Samudrala, R., Ponder, J.W.: Distance geometry generates native-like folds for small helical proteins using the consensus distances of predicted protein structures. Protein Sci. **7**, 1998–2003 (1998)
26. Izrailev, S., Agrafiotis, D.K.: A method for quantifying and visualizing the diversity of QSAR models. J. Mol. Graph. Model. **22**, 275–284 (2004)
27. Izrailev, S., Zhu, F., Agrafiotis, D.K.: A distance geometry heuristic for expanding the range of geometries sampled during conformational search. J. Comput. Chem. **27**(16), 1962–1969 (2006)
28. Kruskal, J.B.: Non-metric multidimensional scaling: a numerical method. Phychometrika **29**, 115–129 (1964)
29. Kuszewski, J., Nilges, M., Brünger, A.T.J.: Sampling and efficiency of metric matrix distance geometry: A novel partial metrization algorithm. J. Biomol. NMR. **2**, 33–56 (1992)
30. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2011)
31. Liu, P., Agrafiotis, D.K., Theobald, D.L.: Fast determination of the optimal rotation matrix for weighted superpositions. J. Comput. Chem. **31**, 1561–1563 (2010)
32. Liu, P., Zhu, F., Rassokhin, D.N., Agrafiotis, D.K.: A self-organizing algorithm for modeling protein loops. PLoS Comput. Biol. **5**(8), e1000478 (2009)
33. Martin, E.J., Hoeffel, T.J.: Oriented Substituent Pharmacophore PRopErtY Space (OSPPREYS): A substituent-based calculation that describes combinatorial library products better than the corresponding product-based calculation. J. Mol. Graph. Model. **18**, 383–403 (2000)
34. Meng, E.C., Gschwend, D.A., Blaney, J.M., Kuntz, I.D.: Orientational sampling and rigid-body minimization in molecular docking. Proteins: Structure, Function, and Bioinformatics **17**, 266–278 (1993)
35. Mumenthaler, C., Braun, W.: Automated assignment of simulated and experimental NOESY spectra of proteins by feedback filtering and self-correcting distance geometry. J. Mol. Biol. **254**, 465–480 (1995)
36. Rassokhin, D.N., Agrafiotis, D.K.: A modified update rule for stochastic proximity embedding. J. Mol. Graph. Model. **22**, 133–140 (2003)
37. Rassokhin, D.N., Lobanov, V.S., Agrafiotis, D.K.: Nonlinear mapping of massive data sets by fuzzy clustering and neural networks. J. Comput. Chem. **22**(4), 373–386 (2011)
38. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**, 2323–2326 (2000)
39. Sammon, J.W.: A nonlinear mapping for data structure analysis. IEEE Trans. Comput. **C18**, 401–409 (1969)
40. Smellie, A., Wilson, C.J., Ng, S.C.: Visualization and interpretation of high content screening data. J. Chem. Inform. Model. **46**, 201–207 (2006)
41. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2323 (2000)
42. Tresadern, G., Agrafiotis, D.K.: Conformational sampling with stochastic proximity embedding (SPE) and self-organizing superimposition (SOS): Establishing reasonable parameters for their practical use. J. Chem. Inform. Model. **49**, 2786–2800 (2009)

43. Witten, I.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann (2010)
44. Xu, H., Izrailev, S., Agrafiotis, D.K.: Conformational sampling by self-organization. J. Chem. Inform. Comput. Sci. **43**, 1186–1191 (2003)
45. Yang, E., Liu, P., Rassokhin, D., Agrafiotis, D.K.: Stochastic proximity embedding on graphics processing units: Taking multidimensional scaling to a new scale. J. Chem. Inform. Model. **51**(11), 2852–2859 (2011)
46. Zhu, F., Agrafiotis, D.K.: A self-organizing superposition (SOS) algorithm for conformational sampling. J. Comput. Chem. **28**, 1234–1239 (2007)

# Part III
# Applications to Protein Conformations

# Chapter 15
# Distance Geometry for Realistic Molecular Conformations

**Gordon M. Crippen**

**Abstract** In order to better understand the many different distance geometry numerical algorithms, it is necessary to relate them to real-world problems in computational chemistry. Here we consider small molecule applications, determination of protein conformation from nuclear magnetic resonance experiments (NMR), protein homology modeling, and more abstract applications to conformational classification and protein sequence comparison. Underlying methods involve conformations in more than three dimensions, low resolution treatments of large problems, and special abstract algebras for dealing with geometry.

## 15.1 Introduction

For computer modeling of molecules using empirical energy functions, a molecule is customarily represented by a point for each atom, located in $\Re^3$ and corresponding to the position of the atomic nucleus. There is very little interest in one or two dimensional molecules. Covalent bonds between pairs of atoms can be thought of as undirected edges in a graph whose vertices correspond to the atoms. Empirical energy functions generally consist of a sum of terms, almost all of which involve particular pairs of atoms and are functions of the Euclidean distance between them. The potentially most important terms are strong positive (unfavorable) steric repulsions between all nonbonded pairs of atoms when they are very close in space. Small deviations from optimal bond lengths and bond angles are strongly penalized in order to model the high frequency, low amplitude vibrational modes in real molecules. More complex terms involve certain ordered quartets of atoms closely linked by bonds to require them to be coplanar or to have a specified chirality,

G.M. Crippen (✉)
University of Michigan, Ann Arbor, MI 48109, USA
e-mail: gcrippen@umich.edu

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_15,

corresponding to requiring the signed volume of the spanned tetrahedron to be zero, positive, or negative. Weaker terms represent mild attractions or repulsions of nonbonded pairs of atoms at longer distances.

Conformational analysis amounts to determining positions of the atoms such that steric clashes and distorted bond lengths, bond angles, and chiralities are avoided, while otherwise having generally favorable weaker pairwise interactions. For $N$ atoms there are $3N$ Cartesian coordinates, but since the overall translation and rigid rotation of the atoms is of no interest, there are $3N-6$ degrees of freedom. Finding low energy conformations amounts to a global optimization problem in terms of the atomic coordinates, where there are many local minima for even small molecules having only 20 atoms, much less for macromolecules having over 1,000 atoms. One standard way to reduce the number of variables is the rigid valence approximation where bond lengths, bond angles, and planar or chiral groups of atoms are held fixed at their ideal values. This leaves as variables only the torsion angles about the relatively few rotatable bonds. Unfortunately, the long range interactions between pairs of atoms separated by multiple rotatable bonds are very complicated functions of these torsion angles. Alternatively, the $N(N-1)/2$ interatomic distances are directly linked to most of the potential energy terms, but the distances are interrelated by the requirement that the conformation exist in three dimensions.

Distance geometry methods for conformational analysis ultimately produce Cartesian coordinates for the atoms, but at some stage of the calculation, they deal with distances as variables. There have been a number of reviews of distance geometry methods [15, 33, 45–47, 49, 54, 64, 85], and here I emphasize more their relevance to real scientific problems rather than details of the algorithms.

## 15.2 Problem Definition

It helps to keep in mind what a real molecule does at reasonable temperatures and either in the gas phase or in solution with a reasonable solvent. The atoms of the molecule are constantly moving as it collides with neighboring molecules, such as solvent, causing the molecule to explore a variety of conformations. At something like room temperature, the molecule very rarely crosses high energy barriers in conformation space, and the motions are confined to small variations in bond lengths and bond angles, but wide changes in torsion angles about rotatable bonds. Planar groups of atoms deviate only slightly from coplanarity, and chiral centers maintain their proper chirality. For the usual small molecule having fewer than 100 atoms and having several rotatable bonds, there is generally no single conformation that accounts for most of the time spent for one molecule or for most of the molecules at one instant in a macroscopic sample. However, if the small molecules are packed together in a regular crystal, they may all assume a single conformation that is rather precisely specified, and it can be determined by x-ray crystallography.

The case for macromolecules (having thousands of atoms) can be different. Most macromolecules explore wide regions of conformation space over reasonable time spans at reasonable temperatures, mostly by large variations in torsion angles for their many rotatable bonds. However, some proteins and nucleic acids spend most of their time in rather compact conformations having favorable interactions between atoms separated by many bonds (so-called “long range” interactions) but close in space. Thus they acquire a somewhat well-defined conformation due to internal interactions rather than interactions between small molecules packed together in a crystal. For a protein consisting of a linear polypeptide chain built out of 50–500 amino acid residues, there are two rotatable bonds per residue in the main chain, plus numerous others in the individual amino acid sidechains. In the so-called folded state, the relative positions of a large fraction of the atoms may be rather well defined, but other parts of the polypeptide chain may continually flex in large amplitude motions.

### *15.2.1 Small Molecules*

Given the covalent structure for a small molecule in terms of atoms, their types, bonds between pairs of atoms, and their types, a common task in molecular modeling is to produce at least one three-dimensional conformation of the molecule. For very small molecules, one can choose initial atomic coordinates at random and then adjust them by local optimization of a suitable energy function. For larger molecules and whole databases of them, it is much more efficient to cleverly link together rigid parts from a comprehensive library of standard ring systems, etc. [31, 62, 79]. This is a generalization of fixed bond lengths and bond angles to take into account a wide variety of molecular substructures. In any case, the objective is to quickly produce reasonable starting conformations that can be subsequently varied, say, by systematic exploration of torsion angles.

Flexible ring structures can be remarkably challenging. The classical case is cyclohexane, which consists of only 18 atoms, six of which are bonded together in a six-membered ring by rotatable bonds. Requiring precisely fixed bond lengths and angles produces an allowed conformation space consisting of two isolated points and one one-dimensional loop [14]. For conformations on the loop, the six torsion angles for the ring bonds are reduced to one “pseudorotation” degree of freedom. To reach the isolated points, bond angles must be deformed, and indeed this is what the real molecule does occasionally at room temperature.

### *15.2.2 Macromolecules*

There has been a great deal of interest in the conformations of those proteins that spontaneously adopt a compact, fairly well-defined conformation in solution under biologically relevant conditions. Of the more than 75,000 structures archived in the Protein Data Bank in 2011, almost all are proteins and only 5,000 involve nucleic acids. Only about 15 % of the structures have been determined by nuclear magnetic resonance (NMR), and the rest resulted from x-ray crystallography. One of the great challenges of protein crystallography is to find conditions for each protein where well-ordered crystals will form. Once packed into a crystal, the protein molecules are more restricted in their motions, due to the close proximity of other protein molecules. There may be some variation in conformation due to motions of individual molecules over the time span of experimental data collection or due to differences in conformation between the many molecules comprising the crystal. In addition, the standard interpretation of the experimental diffraction results is to calculate the single set of atomic coordinates that best fits the diffraction data. If some part of the molecule is substantially disordered, no coordinates will be given for that part, but otherwise the coordinates are something like mean positions, subject to a priori constraints on bond lengths, bond angles, etc.

Structure determination by NMR is carried out in more biologically relevant conditions, namely in solution. Once the peaks in the different NMR spectra have been assigned to particular atoms, further experiments can detect nuclear Overhauser effects (NOEs) between particular pairs of atoms. It is also possible to use NOEs to aid the assignment step [73]. An NOE is taken as evidence that the two atoms involved are *almost always* near each other in space even though they might be separated by many rotatable bonds in the covalent structure of the molecule [33]. Depending on experimental conditions, the interatomic distance may have an upper bound of 5–10 Å (compared to typical bond lengths of about 1.5 Å) (Zuiderweg, E.R.P., personal communication, 2011). An upper bound of 8 Å is relatively high [59]. Failure to observe an NOE does not imply that the two atoms are necessarily always further apart. Sometimes the upper bound on distances from all NOEs may not be consistent with a single structure, and fitting to multiple structures is required [9,22,57,76,92]. There is, however, the danger of overfitting when using many structures [8]. For smaller proteins having few NOEs, multiple substantially different conformations may be consistent with the data [75].

An additional NMR source of conformational information is from residual dipolar coupling (RDC) measurements, which are related to the angle between a particular bond vector in the protein and the external magnetic field vector. Although the conformation of a protein cannot be unambiguously determined from a full set of RDCs plus a priori bond lengths, etc., it is possible to get good conformations from a combination of RDCs plus substantially fewer NOEs than would otherwise be required if only NOEs are used [12, 13, 83, 96].

The customary general work flow is to start with geometric constraints such as NOEs, produce a sampling of initial coordinates by distance geometry or other methods, and then refine these by constrained molecular dynamics [33]. The motivation behind this last step is to produce conformations that not only satisfy all the constraints, but are also of higher quality in that they have low values of the empirical energy function. This raises the issue of what created the final range of conformations, the geometric constraints or the energy function? It is always necessary to add to the energy function penalty terms to enforce the constraints, because the energy function by itself would not favor the experimentally determined constraints. One might ask therefore what does the energy function contribute to the quality of the result?

## 15.3 Algorithms

### *15.3.1 Distance Geometry Mathematics*

The mathematical basis of distance geometry focused on the matrix of squared distances for a set of points [7] and led to the EMBED algorithm for calculating atomic coordinates plus many other issues of interest in applications to molecular conformations [20]. The squared distances have maintained fundamental importance in that any Euclidean invariant (angles, dihedral angles, areas, etc.) can be expressed as a polynomial in the squared interpoint distances [23, 45]. Geometric algebra refers to a special algebra involving scalars, distances, angles, oriented areas, etc., such that geometric problems can be solved in terms of the algebra [28]. In a similar spirit, another such algebra features operations between circles and vectors [81]. Another generalization of distance geometry deals not only with the squared distance between points but also signed distances between a point and a hyperplane, and the dot product between hyperplane normals [102].

The full set of bond lengths, bond angles, and torsion angles constitute a redundant coordinate system [78]. The customary way to reduce the number of variables in conformational problems is to fix the bond lengths and bond angles, leaving the torsion angles as variables. Other sets of nonredundant curvilinear coordinates can be used [25]. In linearized embedding, the molecule is broken down into groups of atoms that are rigid due to the fixed valence geometry assumption. The locations of atoms within each rigid group are expressed as linear combinations of the unit vectors that serve as the local coordinate axes for the rigid group. Relations between rigid groups are expressed in terms of scalar products of the unit vectors, and long-range distance constraints can also be expressed this way, for example, closure of a flexible ring. This leads naturally to embedding in terms of the metric matrix, which is the matrix of inner products between all the unit vectors of all the rigid groups. Conformational sampling tends to be broader and more uniform than with standard distance geometry embedding [14, 17].

Another way to reduce the number of variables is to go from the squared distances between points to the sums of squared distances between subsets of points. This is particularly convenient for a linear polymer, such as a protein, where the subsets can be a few sequentially adjacent amino acid residues. That way, the diagonal elements of the distance matrix reflect how compact the local parts of the chain are, and the off-diagonal elements indicate interactions between different parts of the chain. The requirements on the distance matrix for embedding in three dimensions remain the same as for single-point distances. In this cluster distance geometry approach [16, 18, 19], it is easy to examine the entire conformation space of proteins at low resolution, calculate the canonical partition function, and devise empirical energy functions that favor the native fold for a training set of proteins and a large test set of unrelated proteins.

### *15.3.2 Exact Constraints*

Suppose we are given desired values for all interpoint distances. In general, this will not correspond to a realizable configuration in three dimensions, so the question is how to adjust the proposed distance matrix until it does, after which calculating coordinates is easy. The multiple alternating projections (MAP) method finds a unique least-squares approximation to the given matrix that corresponds to a realizable configuration in $N-1$ dimensions for $N$ points [35], but there are in general multiple corresponding configurations in three dimensions [36]. Subsequent improvements yielded a practical algorithm for finding three-dimensional conformations of small proteins [34, 37, 38, 97].

Alternatively, suppose we are given not many more than $3N-6$ exact distances having at least no incompatibility with a three-dimensional configuration, such as violations of the triangle inequality, or five points having the same fixed distance among them. It is possible to calculate the missing distances from the given ones [3, 82]. However, a generally appealing way to proceed is to first determine a subset of the points that are constrained to be rigid and then successively add points having sufficiently many fixed distances to already positioned points to be positioned in turn. Even the initial rigid group can be tricky because there are some sets of exact distances such that the points are locally rigid but have more than one such configuration in space [50]. The general build-up approach from sparse exact distances has been developed [26, 27], and improvements have been made to deal with error propagation [11, 86, 87, 100]. As mentioned above, the case of exact distance constraints does not generally arise in molecular conformation applications, except for generating representative conformations of small molecules.

Taking some distances to be effectively fixed leads to an analysis of all allowed conformations in terms of rigid subsets of points/atoms. This is a useful way to describe the conformations of a folded protein [57] where some parts of the chain are tightly packed together, and other parts can flex. This sort of analysis can be extended to subsets of atoms that are not totally rigid [55, 56]. Distance geometry embedding can be specially adapted to generate conformations where one substructure is held rigid, and the rest of the molecule is more flexible, subject to the usual distance constraints [70].

Instead of building up points with determined coordinates, one can start with a few points and their interpoint distances and then additional successive distance values are chosen to satisfy any distance constraints while remaining consistent with a three-dimensional conformation. This explicit distance geometry approach [42] and similar approaches [60] can even deal with macromolecules.

Another way to deal with a sparse set of exact distance constraints is to use global optimization techniques [99] where the penalty function is initially smoothed to make finding an approximate solution easy and then gradually return to the rougher, more precise penalty function [40, 63, 67–69]. Results vary, depending on the smoothing technique and the subsequent unsmoothing procedure. Similarly, an energy function plus constraint penalty terms can be smoothed in order to make easier the search for low energy conformations satisfying the geometric constraints [53].

### 15.3.3 Qualitative Descriptions

In most studies of conformations, the quantitative values of interatomic distances are important. However, one can reduce the distance matrices of a set of alternative conformations to matrices of the *ordinal* distances, resulting in a smaller set of equivalence classes of conformations [29]. In a similar spirit, one can calculate from the three-dimensional atomic coordinates the chirotope, which is the qualitative chirality ($+1$ or $-1$) for each ordered quartet of atoms. Then two different conformations can be compared by calculating the Hamming distance between their respective chirotopes [61].

### 15.3.4 More than Three Dimensions

One way to make easier a search for nearly global minima of a function is to smooth the function. However, another way is to increase the number of variables. For example, inverting a chiral center in three dimensions requires passing through high energy intermediates with distorted bond angles, which is why real molecules at room temperature seldom do this. In four dimensions, such an inversion is merely a rigid rotation requiring no such distortions. In order to satisfy a set of distance constraints, it is much easier to compute high-dimensional coordinates than three-dimensional coordinates. Then there are good ways to compute lower-dimensional coordinates from these that approximately satisfy the constraints [24]. In energy embedding, the reduction of dimension also takes into account a given energy function, so as to eventually find low energy three-dimensional coordinates. In rotational energy embedding, the dimension reduction involves trying 0 and $180^\circ$ alternate values of high-dimensional torsion angles [21]. Since the number of local minima for a given potential function generally decreases substantially with increasing dimension, an alternative approach is to alternate between relaxing the conformation into four dimensions while minimizing the energy, then flatten the conformation back into three dimensions [88].

Molecular dynamics is often used to derive relatively low energy conformations from initial coordinates that obey the given distance constraints. A more effective way is to run the molecular dynamics in four dimensions and then gradually return to three [6, 66, 89, 94].

Dimension reduction can arise in a quite different sense. Suppose we have a large number of three-dimensional conformations that hopefully constitute a fairly dense and exhaustive sampling of the abstract conformation space. Then with some measure of conformational dissimilarity that can be used as a distance in conformation space, the goal is to characterize the set of conformations as points in some lower-dimensional manifold. This is something like a distance geometry embedding procedure that emphasizes the near neighbor points to get approximate geodesic distances between all pairs of points [1, 84, 91, 101].

### *15.3.5 Sequence Alignment*

A protein consists of an unbranched chain of amino acid residues having almost always 20 standard types. Two proteins may have similar amino acid sequences in the sense that the sequences are identical except for a few insertions and deletions in the sequence of one relative to the sequence of the other. Alternatively, given the atomic coordinates, one can make a gapped alignment of the two distance matrices so that interresidue distances match, regardless of matching residue types [2, 51]. Generally, substantial sequence similarity between two proteins is reflected in conformational similarity in the aligned segments but conformational dissimilarity in the insertions and deletions. Thus, one can calculate conformations for a new protein sequence, given the known conformation of a protein having strong sequence similarity [90]. If there are several proteins of known conformation that all have weak sequence similarity to the new protein, then consensus distances among aligned residues can be used to generate a good conformation [52].

Generally, residues in a folded protein that are in close contact with each other tend to be hydrophobic, while hydrophilic residues tend to interact more with the aqueous solvent than with other residues. Reducing the interresidue distance matrix for a known protein conformation to a qualitative contact matrix having 1 entry for contact vs. 0 for lack of contact, the eigenvector of the contact matrix corresponding to the maximal eigenvalue summarizes the conformation. The eigenvector components correlate strongly with the hydrophobicity profile of the sequence [4]. Moreover, one can recover the original conformation from the contact map by a simulated annealing procedure [95].

Given a measure of sequence similarity after an alignment procedure, one can use the dissimilarities as distances between points that are the sequences in an abstract sequence space. From this, one can generate three-dimensional displays of a set of sequences and their similarities [32].

### *15.3.6 Protein Structure from NMR Data*

Given the experimentally determined NOEs for a protein in solution, the usual way to calculate a sample of satisfactory conformations is by distance geometry embedding, proceeding from allowed distance matrices to corresponding metric matrices, to three-dimensional coordinates [20]. There are alternative ways to ensure that the proposed matrix of distances is consistent with a three-dimensional conformation [30, 39]. Many variations on the standard distance geometry embedding procedure have been devised, and they vary with regard to the breadth and uniformity of the distribution of conformations they sample, as well as general tendencies toward compact or expanded conformations [44, 48, 71, 72, 74, 77, 80, 93, 98].

An alternative approach is to simply use the fixed valence geometry and optimize satisfaction of the NOE distance constraints by varying the dihedral angles of the relatively few rotatable bonds [5, 41, 58, 65]. As in global optimization methods, the penalty function enforcing the constraints needs to be varied in the process [10]. Monte Carlo methods, such as multicanonical ensembles, give broader sampling than molecular dynamics [43].

## 15.4 Conclusions

The mathematical concepts of distance geometry have given rise to many computational algorithms for exploring the conformational possibilities for small and large molecules. Which approaches are best depends on an understanding of the sources of experimental data for the problem at hand. Efficient algorithms are of course preferable, but given that the size of molecules of general interest is strictly limited, one need merely ensure that the algorithm performs well enough for the largest cases. The real challenges are more conceptual issues about how to describe the allowed conformation space of molecules.

## References

1. Agrafiotis, D.K.: Stochastic proximity embedding. J. Comput. Chem. **24**(10), 1215–1221 (2003)
2. Akutsu, T.: Protein structure alignment using dynamic programming and iterative improvement. IEICE Trans. Inform. Syst. **E79-D**, 1–8 (1996)
3. Alfakih, A.Y., Khandani, A., Wolkowicz, H.: Solving Euclidean distance matrix completion problems via semidefinite programming. Comput. Optim. Appl. **12**, 13–30 (1999)
4. Bastolla, U., Porto, M., Roman, H.E., Vendruscolo, M.: Principal eigenvector of contact matrices and hydrophobicity profiles in proteins. Proteins **58**, 22–30 (2005)
5. Berndt, K.D., Guentert, P., Wuethrich, K.: Conformational sampling by NMR solution structures calculated with the program DIANA evaluated by comparison with long-time molecular dynamics calculations in explicit water. Proteins **24**, 304–313 (1996)

6. Beutler, T.C., van Gunsteren, W.F.: Molecular dynamics free energy calculation in four dimensions. J. Chem. Phys. **101**, 1417–1422 (1994)
7. Blumenthal, L.M.: Theory and Applications of Distance Geometry. Chelsea Publishing Company, New York (1970)
8. Bonvin, A.M.J.J., Brünger, A.T.: Conformational variability of solution NMR structures. J. Mol. Biol. **250**, 80–93 (1995)
9. Bonvin, A.M.J.J., Rullmann, J.A.C., Lamerichs, R.M.J.N., Boelens, R., Kaptein, R.: "Ensemble" iterative relaxation matrix approach: A new NMR refinement protocol applied to the solution structure of crambin. Proteins Struct. Funct. Genet. **15**, 385–400 (1993)
10. Braun, W.: Distance geometry in distance and torsion angle space: Flexibility, convergence and sampling properties. In: Renugoplakrishnan, V. (ed.) Proteins, pp. 116–22. ESCOM, Leiden, Netherlands (1991)
11. Carvalho, R., Lavor, C., Protti, F.: Extending the geometric build-up algorithm for the molecular distance geometry problem. Inform. Process. Lett. **108**(4), 234–237 (2008)
12. Clore, G.M., Gronenborn, A.M., Bax, A.: A robust method for determining the magnitude of the fully asymmetric alignment tensor of oriented macromolecules in the absence of structural information. J. Mag. Reson. **133**, 216–221 (1998)
13. Clore, G.M., Gronenborn, A.M., Tjandra, N.: Direct structure refinement against residual dipolar couplings in the presence of rhombicity of unknown magnitude. J. Mag. Reson. **131**, 159–162 (1998)
14. Crippen, G.M., Smellie, A.S., Richardson, W.W.: Conformational sampling by a general linearized embedding algorithm. J. Comput. Chem. **13**(10), 1262–1274 (1992)
15. Crippen, G.M.: Chemical distance geometry: Current realization and future projection. J. Math. Chem. **6**, 307–324 (1991)
16. Crippen, G.M.: Cluster distance geometry of polypeptide chains. J. Comput. Chem. **25**, 1305–1312 (2004)
17. Crippen, G.M.: Exploring the conformation space of cycloalkanes by linearized embedding. J. Comput. Chem. **13**(3), 351–361 (1992)
18. Crippen, G.M.: Recognizing protein folds by cluster distance geometry. Proteins **60**, 82–89 (2005)
19. Crippen, G.M.: Statistical mechanics of protein folding by cluster distance geometry. Biopolymers **75**(3), 278–289 (2004)
20. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
21. Crippen, G.M., Havel, T.F.: Global energy minimization by rotational energy embedding. J. Chem. Inform. Comput. Sci. **30**(3), 222–7 (1990)
22. Cuniasse, P., Raynal, I., Yiotakis, A., Dive, V.: Accounting for conformational variability in NMR structure of cyclopeptides: Ensemble averaging of interproton distance and coupling constant restraints. J. Am. Chem. Soc. **119**, 5239–5248 (1997)
23. Dalbec, J.P.: Straightening Euclidean invariants. Ann. Math. Artif. Intell. **13**, 97–108 (1995)
24. Dasgupta, S., Gupta, A.: An elementary proof of the Johnson-Lindenstrauss lemma, Technical Report TR-99-006, International Computer Science Institute (ICSI), Berkeley, California (1999)
25. De Vico, L., Olivucci, M., Lindh, R.: New general tools for constrained geometry optimizations. J. Chem. Theor. Comput. **1**, 1029–1037 (2005)
26. Dong, Q., Wu, Z.: A Linear-time algorithm for solving the molecular distance geometry problem with exact interatomic distances. J. Global Optim. **22**, 365–375 (2002)
27. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global Optim. **26**(3), 321–333 (2003)
28. Dress, A.W.M., Havel, T.F.: Distance geometry and geometric algebra. Found. Phys. **23**, 1357–1374 (1993)
29. Easthope, P.L.: Classifying the conformations of a chemical system using matrices of integers. J. Math. Chem. **13**, 73–94 (1993)

30. Emiris, I.Z., Nikitopoulos, T.G.: Molecular conformation search by distance matrix perturbations. J. Math. Chem. **37**(3), 233–253 (2005)
31. Feuston, B.P., Miller, M.D., Culberson, J.C., Nachbar, R.B., Kearsley, S.K.: Comparison of knowledge-based and distance geometry approaches for generation of molecular conformations. J. Chem. Inform. Comput. Sci. **41**(3), 754–763 (2001)
32. Forster, M.J., Heath, A.B., Afzal, M.A.: Application of distance geometry to 3D visualization of sequence relationships. Bioinformatics **15**(1), 89–90 (1999)
33. Gippert, G.P., Yip, P.F., Wright, P.E., Case, D.A.: Computational Methods for Determining Protein Structures from NMR Data. Biochem. Pharmacol. **40**, 15–22 (1990)
34. Glunt, W., Hayden, T.: Improved convergence and speed for the distance geometry program APA to determine protein structure. Comput. Chem. **25**, 223–230 (2001)
35. Glunt, W., Hayden, T.L., Hong, S., Wells, J.: An Alternating Projection Algorithm for Computing the Nearest Euclidean Distance Matrix. SIAM J. Matrix Anal. Appl. **11**, 589–600 (1990)
36. Glunt, W., Hayden, T.L., Liu, W.M.: The Embedding Problem for Predistance Matrices. Bull. Math. Biol. **53**, 769–796 (1991)
37. Glunt, W., Hayden, T.L., Raydan, M.: Molecular conformations from distance matrixes. J. Comput. Chem. **14**(1), 114–120 (1993)
38. Glunt, W., Hayden, T.L., Raydan, M.: Preconditioners for distance matrix algorithms. J. Comput. Chem. **15**, 227–232 (1994)
39. Grooms, I.G., Lewis, R.M., Trosset, M.W.: Molecular embedding via a second order dissimilarity parameterized approach. SIAM J. Sci. Comput. **31**, 2733–2756 (2009)
40. Grosso, A., Locatelli, M., Schoen, F.: Solving molecular distance geometry problems by global optimization algorithms. Comput. Optim. Appl. **43**(1), 23–37 (2009)
41. Guentert, P., Mumenthaler, C., Wuethrich, K.: Torsion angle dynamics for NMR structure calculations with the new program DYANA. J. Mol. Biol. **273**, 283–298 (1997)
42. Hadwiger, M.A., Fox, G.E.: Explicit distance geometry: Identification of all the degrees of freedom in a large RNA molecule. J. Biomol. Struct.Dynam. **8**(4), 759–779 (1991)
43. Hansmann, U.H.E., Okamoto, Y.: New Monte Carlo algorithms for protein folding. Curr. Opin. Struct. Biol. **9**, 177–183 (1999)
44. Havel, T.F.: An evaluation of computational strategies for use in the determination of protein structure from distance constraints obtained by nuclear magnetic resonance. Progress Biophys. Mol. Biol. **56**, 43–78 (1991)
45. Havel, T.F.: Distance geometry: Theory, algorithms, and chemical applications, In: von Rague, P., Schreiner, P.R., Allinger, N.L., Clark, T., Gasteiger, J., Kollman, P.A., Schaefer, H.F. (eds.) Encyclopedia of Computational Chemistry, pp. 723–742. Wiley, New York (1998)
46. Havel, T.F.: Distance Geometry. In: Grant, D.M., Harris, R.K. (eds.) Invited contribution to the Encyclopedia of NMR, pp. 1701–1710. Wiley, New York (1996)
47. Havel, T.F.: Metric matrix embedding in protein structure calculations, NMR spectra analysis, and relaxation theory. Mag. Reson. Chem. **41**, 537–550 (2003)
48. Havel, T.F.: The sampling properties of some distance geometry algorithms applied to unconstrained polypeptide chains: a study of 1830 independently computed conformations. Biopolymers **29**(12–13), 1565–1585 (1990)
49. Havel, T.F., Hyberts, S., Najfeld, I.: Recent advances in molecular distance geometry. Lect. Notes Comput. Sci. **1278**, 62–71 (1997)
50. Hendrickson, B.A.: The molecule problem: Determining conformation from pairwise distances, Technical Report 90-1159, Department of Computer Science, Cornell University, Ithaca, New York (1990)
51. Holm, L., Sander, C.: Protein structure comparison by alignment of distance matrices. J. Mol. Biol. **233**, 123–138 (1993)
52. Huang, E.S., Samudrala, R., Ponder, J.W.: Distance geometry generates native-like folds for small helical proteins using the consensus distances of predicted protein structures. Protein Sci. **7**, 1998–2003 (1998)

53. Huber, T., Torda, A.E., van Gunsteren, W.F.: Structure optimization combining soft-core interaction functions, the diffusion equation method, and molecular dynamics. J. Phys. Chem. **A101**, 5926–5930 (1997)
54. Ikura, T., Go, N.: Determination of three-dimensional structures of proteins in solution by NMR experiment and distance geometry calculation. Kobunshi **39**(3), 210–213 (1990)
55. Jacobs, D.J., Rader, A.J., Kuhn, L.A., Thorpe, M.F.: Protein flexibility predictions using graph theory. Protein. Struct. Funct. Genet. **44**, 150–165 (2001)
56. Jacobs, D.J., Livesay, D.R., Hules, J., Tasayco, M.L.: Elucidating quantitative stability/flexibility relationships within thioredoxin and its fragments using a distance constraint model. J. Mol. Biol. **358**, 882–904 (2006)
57. Kemmink, J., Scheek, R.M.: Dynamic modelling of a helical peptide in solution using NMR data: Multiple conformations and multi-spin effects. J. Biomol. NMR **5**, 33–40 (1995)
58. Koehl, P., Lefevre, J.F., Jardetzky, O.: Computing the geometry of a molecule in dihedral angle space using NMR derived constraints. A new algorithm based on optimal filtering. J. Mol. Biol. **223**(1), 299–315 (1992)
59. Koharudin, L.M.I., Bonvin, A.M.J.J., Kaptein, R., Boelens, R.: Use of very long-distance NOEs in a fully deuterated protein: An approach for rapid protein fold determination. J. Mag. Reson. **163**, 228–225 (2003)
60. Kuszewski, J., Nilges, M., Brunger, A.T.: Sampling and efficiency of metric matrix distance geometry: A novel partial metrization algorithm. J. Biomol. NMR **2**, 33–56 (1992)
61. Laboulais, C., Ouali, M., Bret, M.L., Gabarro-Arpa, J.: Hamming distance geometry of a protein conformational space: Application to the clustering of a 4-ns molecular dynamics trajectory of the HIV-1 integrase catalytic core. Protein. Struct. Funct. Genet. **47**, 169–179 (2002)
62. Leach, A.R., Smellie, A.S.: A combined model-building and distance geometry approach to automated conformational analysis and search. J. Chem. Inform. Comput. Sci. **32**, 379–385 (1992)
63. Liberti, L., Lavor, C., Maculan, N., Marinelli, F.: Double variable neighbourhood search with smoothing for the molecular distance geometry problem. J. Global Optim. **43**, 207–218 (2009)
64. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: From continuous to discrete. Int. Trans. Oper. Res. **18**(1), 33–51 (2011)
65. Mertz, J.E., Guentert, P., Wuethrich, K., Braun, W.: Complete relaxation matrix refinement of NMR structures of proteins using analytically calculated dihedral angle derivatives of NOE intensities. J. Biomol. NMR **1**(3), 257–269 (1991)
66. Morikawa, S., Ogata, K., Sekikawa, A., Sarai, A., Ishii, S., Nishimura, Y., Nakamura, H.: Determination of the NMR solution structure of a specific DNA complex of the Myb DNA-binding domain. J. Biomol. NMR **6**, 294–305 (1995)
67. Moré, J.J., Wu, Z.: $\varepsilon$-optimal solutions to distance geometry problems via global continuation. In: Pardalos, P.M., Shalloway, D., Xue, G. (eds.) Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding, pp. 151–168. American Mathematical Society, Providence (1996)
68. Moré, J.J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**, 814–836 (1997)
69. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. J. Global Optim. **15**, 219–234 (1999)
70. Najfeld, I., Havel, T.F.: Embedding with a rigid substructure. J. Math. Chem. **21**, 223–260 (1997)
71. Nakai, T., Kidera, A., Nakamura, H.: Intrinsic nature of the three-dimensional structure of proteins as determined by distance geometry with good sampling properties. J. Biomol. NMR **3**, 19–40 (1993)
72. Nilges, M., Kuszewski, J., Brunger, A.T.: Sampling properties of simulated annealing and distance geometry. NATO ASI Series, Series A, **225**, 451–455 (1991)
73. Oshiro, C.M., Kuntz, I.D.: Application of distance geometry to the proton assignment problem. Biopolymers **33**(1), 107–115 (1993)

74. Oshiro, C.M., Thomason, J., Kuntz, I.D.: Effects of limited input distance constraints upon the distance geometry algorithm. Biopolymers **31**(9), 1049–1064 (1991)
75. Pastore, A., Atkinson, R.A., Saudek, V., Williams, R.J.P.: Topological mirror images in protein structure computation: An underestimated problem. Protein. Struct. Funct. Genet. **10**(1), 22–32 (1991)
76. Pearlman, D.A.: How is an NMR structure best defined? An analysis of molecular dynamics distance-based approaches. J. Biomol. NMR **4**, 1–16 (1994)
77. Peishoff, C.E., Dixon, J.S.: Improvements to the distance geometry algorithm for conformational sampling of cyclic structures. J. Comput. Chem. **13**(5), 565–569 (1992)
78. Peng, C., Ayala, P.Y., Schlegel, H.B., Frisch, M.J.: Using redundant internal coordinates to optimize equilibrium geometries and transition states. J. Comput. Chem. **17**, 49–56 (1996)
79. Perlman, R.S.: 3D molecular structures: Generation and use in 3D searching. In: Kubinyi, H. (ed.) 3D-QSAR in Drug Design: Theory, Methods and Applications. ESCOM Science Publishers, Leiden, The Netherlands (1993)
80. Petersen, K., Taylor, W.R.: Modelling zinc-binding proteins with GADGET: Genetic algorithm and distance geometry for exploring topology. J. Mol. Biol. **325**(5), 1039–1059 (2003)
81. Pfeifer, R.E., van Hook, C.: Circles, vectors, and linear algebra. Math. Mag. **66**, 86 (1993)
82. Porta, J.M., Ros, L., Tomas, F., Corcho, F., Canto, J., Perez, J.J.: Complete maps of molecular-loop conformation spaces. J. Comput. Chem. **29**, 144–155 (2008)
83. Prestegard, J.H., Al-Hashimi, H.M., Tolman, J.R.: NMR structures of biomolecules using field oriented media and residual dipolar couplings. Q. Rev. Biophys. **33**, 371–424 (2000)
84. Rowels, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**, 2323–2326 (2000)
85. Sherman, S.A., Johnson, M.E.: Derivation of locally accurate spatial protein structure from NMR data. Progress Biophys. Mol. Biol. **59**, 285–339 (1993)
86. Singer, A.: A remark on global positioning from local distances. Proc. Nat. Acad. Sci. **105**, 9507–9511 (2008)
87. Sit, A., Wu, Z., Yuan, Y.: A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation. Bull. Math. Biol. **71**(8), 1914–1933 (2009)
88. Snow, M.E., Crippen, G.M.: Dimensional oscillation. A fast variation of energy embedding gives good results with the AMBER potential energy function. Int. J. Peptide Protein Res. **38**(2), 161–168 (1991)
89. Spellmeyer, D.C., Wong, A.K., Bower, M.J., Blaney, J.M.: Conformational analysis using distance geometry methods. J. Mol. Graph. Model. **15**, 18–36 (1997)
90. Srinivasan, S., March, C.J., Sudarasanam, S.: An automated method for modeling proteins on known templates using distance geometry. Protein Sci. **2**, 227–289 (1993)
91. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2322 (2000)
92. Torda, A.E., Scheek, R.M., van Gunsteren, W.F.: Time-averaged nuclear overhauser effect distance restraints applied to tendamistat. J. Mol. Biol. **214**, 223–235 (1990)
93. van Kampen, A.H.C., Buydens, L.M.C., Lucasias, C.B., Blommers, M.J.J.: Optimisation of metric matrix embedding by genetic algorithms. J. Biomol. NMR **7**, 214–224 (1996)
94. van Schaik, R.C., Berendsen, H.J.C., Torda, A.E., van Gunsteren, W.F.: A structure refinement method based on molecular dynamics in four spatial dimensions. J. Mol. Biol. **234**, 751–762 (1993)
95. Vendruscolo, M., Kussell, E., Domany, E.: Recovery of protein structure from contact maps. Folding Des. **2**, 295–306 (1997)
96. Wang, L., Mettu, R.R., Donald, B.R.: A polynomial-time algorithm for de novo protein backbone structure determination from nuclear magnetic resonance data. J. Comput. Biol. **13**(7), 1267–1288 (2006)
97. Wells, C., Glunt, W., Hayden, T.L.: Searching conformational space with the spectral distance geometry algorithm. J. Mol. Struct. (Theochem) **308**, 263–271 (1994)

98. Wertz, D.A., Shi, C.X., Venanzi, C.A.: A comparison of distance geometry and molecular dynamics simulation techniques for conformational analysis of $\beta$-cyclodextrin. J. Comput. Chem. **13**(1), 41–56 (1992)
99. Williams, G.A., Dugan, J.M., Altman, R.B.: Constrained global optimization for estimating molecular structure from atomic distances. J. Comput. Biol. **8**(5), 523–547 (2001)
100. Wu, D., Wu, Z.: An updated geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data. J. Global Optim. **37**, 661–673 (2007)
101. Xu, H., Izrailev, S., Agrafiotis, D.K.: Conformational sampling by self-organization. J. Chem. Inform. Comput. Sci. **43**(4), 1186–1191 (2003)
102. Yang, L.: Solving spatial constraints with global distance coordinate system. Int. J. Comput. Geom. Appl. **16**, 533–547 (2006)

# Chapter 16
# Distance Geometry in Structural Biology: New Perspectives

**Thérèse E. Malliavin, Antonio Mucherino, and Michael Nilges**

**Abstract** Proteins are polypeptides of amino acids involved in most of the biological processes. In the last 50 years, the study of their structures at the molecular level revolutioned the vision of biology. The three-dimensional structure of these molecules helps in the identification of their biological function. In this chapter, we focus our attention on methods for structure determination based on distance information obtained by nuclear magnetic resonance (NMR) experiments. We give a few details about this experimental technique and we discuss the quality and the reliability of the information it is able to provide. The problem of finding protein structures from NMR information is known in the literature as the molecular distance geometry problem (MDGP). We review some of the historical and most used methods for solving MDGPs with NMR data. Finally, we give a brief overview of a new promising approach to the MDGP, which is based on a discrete formulation of the problem, and we discuss the perspectives this method could open in structural biology.

## 16.1 Introduction

The vision of biology has been fundamentally modified during the second part of the twentieth century by the analysis of the cell function at the molecular

T.E. Malliavin (✉)
Institut Pasteur, Paris, France
e-mail: therese.malliavin@pasteur.fr

A. Mucherino
IRISA, University of Rennes 1, avenue de General Leclerc, Rennes 35042, France
e-mail: antonio.mucherino@irisa.fr

M. Nilges
Institut Pasteur, Paris, France
e-mail: michael.nilges@pasteur.fr

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_16,

level. This brought about a molecular description of the interactions between the molecular agents (biomolecules) which perform important biological processes. Just to mention some examples, molecular motors are the essential agents of movement in living organisms, transcription factors regulate the genetic expressions, enzymes are able to catalyze chemical reactions, and ion channels help establishing and controlling the voltage gradient across the cell membrane. Moreover, transport proteins perform the function of moving other materials inside an organism.

The description of biomolecules at molecular level has been possible for 50 years, due to the development of methods to study the molecular structure of biomolecules. Indeed, these structures are essential in order to understand the function they are able to perform. The slightest modifications in this structure can drastically change the corresponding biomolecular function, as it is encountered, for example, for neurodegenerative diseases [21].

Biomolecular structures can be studied at different levels. A protein is a polypeptide of *amino acids*, named *protein residues* when they are inserted into the polypeptide chain. Polypeptide synthesis is performed through the controlled formation of a peptide bond between two amino acids, where each amino acid pair loses a water molecule. The protein main chain is usually referred to as *backbone*, whereas the atoms that are specific for each residue form the so-called *side chains*.

Proteins display a hierarchical level of organization. Their *primary structures* consist of the sequence of amino acids composing the molecule. Amino acids bond to each other to form a chain, which, under chemical and physical forces, gives rise to three-dimensional structures that are specific for a given primary structure. The *secondary structures* of proteins represent local arrangements of residues: in $\alpha$-helices, the backbone is arranged as a helix, whereas in $\beta$-sheets, the structure is formed by strands of residues over a common plane. The *tertiary structure* is the global arrangement of the amino acids of monomeric proteins, for which there is a unique sequence of amino acids. For more complex proteins, the global three-dimensional structure is given by their *quaternary structure*, build up from the tertiary structures of the various chains of amino acids which can compose the molecule.

In the following, the organization in the three-dimensional space of the atoms of a molecule will be referred to as *conformation* of the molecule. This conformation, together with its chemical architecture, will be referred to as *structure* of the molecule. In the literature, papers may refer to conformations or structures, but in the problem presented here, the actual unknown is the conformation, because the protein structure can be deduced from its conformation, plus its chemical composition.

The focus of this chapter is on methods and algorithms for the identification of the three-dimensional conformations of proteins. The rest of this chapter is organized as follows. Our discussion begins in Sect. 16.2 with the different possible representations for protein conformations that can be considered when solving problems related to such molecules. This representation is strongly related to the complexity of considered solution methods. In Sect. 16.3, we introduce structural biology and discuss its importance for understanding biological processes. Then, we will focus our attention on nuclear magnetic resonance (NMR) experiments and on methods for finding protein conformations from NMR data. In Sect. 16.4, we will give an overview of NMR experiments, and we will discuss about possible sources of errors that may affect the distance information that it is able to provide.

In Sect. 16.5, we will introduce the molecular distance geometry problem (MDGP) and we will study its complexity under different hypotheses. In Sect. 16.6, we will present some basic techniques for refining the distance information given by NMR. In Sect. 16.7, we will briefly present the first method that was used for solving MDGPs with NMR data, and we will mention to some of the issues that caused its replacement with global optimization techniques. A discussion on global optimization for the MDGP is given in Sect. 16.8. The most currently used technique for solving MDGPs by optimization is based on the meta-heuristic simulated annealing (SA): most protein structures that are currently available on the protein data bank (PDB) and that have been analyzed by NMR experiments were obtained by some SA-based global optimization computational tools. We will briefly present the basic idea behind this approach, as well as a new deterministic approach that is based on a discretization of the problem. Finally, in Sect. 16.9, we will give some directions for future research.

## 16.2 Protein Representation

We begin our discussion on methods and algorithms for protein structure determination from NMR data with a short overview of suitable representations for protein conformations. We refer to [40], where a similar discussion was presented in a different context.

An atom can be represented by the coordinates of its mass center in three-dimensional space. Therefore, if a molecule is simply seen as a set of atoms, a possible representation is given by a set of coordinates in the space. This representation is known as *full-atom* representation of a molecule, which involves $3n$ real variables, three for each of the $n$ atoms forming the molecule. Other more efficient representations can be however employed for molecular conformations.

The task of finding an efficient representation is evidently easier when information is available on the chemical composition of the molecule. As already remarked, proteins are chains of amino acids, and the subgroup of atoms that is common to each residue forms the so-called protein backbone. Among the atoms contained in this backbone, more importance is given to the carbon atom usually labeled with the symbol $C_\alpha$. In some works in the literature (see for example [24, 37]), this $C_\alpha$ atom is used for representing an entire residue. In this case, therefore, the protein conformation is represented through the spatial coordinates of $n_{aa}$ atoms, where $n_{aa}$ is the number of residues forming the protein. Considering that each residue can contain 10–20 atoms, it is clear how simplified this representation is. The sequence of $C_\alpha$ atoms is also called *trace* of the protein. We remark that this representation cannot be employed for discriminating among the 20 different amino acids that can make up the protein.

More accurate representations of protein backbones can be obtained if more atoms are considered. If, together with the carbon $C_\alpha$, two other atoms, which are bonded to this $C_\alpha$, are also considered (another carbon $C'$ and a nitrogen N), then the whole protein backbone can be reconstructed. In other words, the coordinates of three atoms per amino acid are sufficient for representing a whole protein conformation without side chains. Therefore, a protein backbone can be represented precisely by a sequence of $3n_{aa}$ atomic coordinates.

This representation is however not much used, because there is another representation for the protein backbones which is much more efficient. Four consecutive atoms in the sequence of atoms N, $C_\alpha$, and C representing a protein backbone form a *torsion angle*, i.e., the angle between the plane formed by the first triplet of atoms and the plane formed by the second triplet of atoms in this quadruplet. Torsion angles can be computed from available atomic coordinates, and, since some distances are angles between bonded atoms are known, the procedure can also be inverted. The representation of a protein which is based on torsion angles is more efficient, because the protein backbone is described by fewer variables.

In the applications, the representation based on torsion angles is further simplified. The sequence of atoms on the protein backbone is a continuous repetition of the atoms N, $C_\alpha$, and C. Each quadruplet defining a torsion angle contains two atoms of the same kind that belong to two bonded amino acids. Then, the torsion angles can be divided into three groups, depending on the kind of atom that appears twice. Torsion angles of the same group are usually denoted by the same symbol: the most used symbols are $\phi$, $\psi$, and $\omega$. The torsion angle $\omega$ is rather constant and its value is generally very close to $180^\circ$: because of the resonance stabilization of the amide (peptide) bond and because of the carbonyl double bond, the four involved atoms, $C_\alpha$, $C'$ (belonging to the first amino acid), and N, $C_\alpha$ (belonging to the second one in the sequence), are constrained to be on the same plane. Therefore, a protein backbone can be represented by a set of $2n_{aa} - 2$ variables, one for each torsion angle $\phi$ and $\psi$ that can be defined. There is a variant of the SA-based algorithm described in Sect. 16.8.1 that is based on this protein representation.

When the whole protein conformation needs to be represented, other torsion angles (usually denoted $\chi$) are defined for the description of the amino acid side chains, where each of such subsequences of angle $\chi$ is specific to the different chemical properties of the amino acids.

Section 16.8.2 is devoted to a novel approach to distance geometry where the problem is discretized. In this case, the variables employed for the protein representation do not need to vary in a continuous space, but they can take a finite number of values. In the simplified case in which there is no uncertainty in the input data [42], a protein can be represented by a vector of binary variables. Otherwise, a vector of integer values can be used for the representation. Naturally, these discrete representations rely on a priori known information on proteins (as in the case of the torsion angle representation). Additional details about these discrete representations are given in Sect. 16.8.2.

For evident reasons, there is no best representation. A representation needs to be selected on the basis of the properties of the problem to be studied. In distance geometry for structural biology, the torsion angle representation is the most currently used one. In different situations, however, other representations could be more appropriate.

## 16.3 Importance of Structural Biology to Understand Biological Processes

The conformation and the structure of biomolecules are very important to understand their function and to analyze possible interactions of the molecule with other molecules, helping in this way the development of new drugs [52]. Because of the essential role of the molecular structure of molecules, a scientific field, called *structural biology*, whose main aim is to identify and study biological structures, has experienced an enormous development.

Structural biology originated from the application of powerful physical techniques to biological objects. It has also largely benefited from the development of molecular biology biochemistry and cellular biology techniques. The widespread application of structural biology methods has produced a quite astonishing molecular description of life. Due to this great impact, structures of biological molecules are deposited in public web databases. The most important database is named PROTEIN DATA BANK (PDB: http://www.rcsb.org) [2], which reached the number of 80,000 deposited molecules at the beginning of 2012.

Biomolecular structures can be investigated at several levels: single biomolecules, biomolecular complexes and assemblies, and cellular organs. Different methods can be applied for studying these biomolecular structures. In general, the information provided by such methods concerns the molecular electronic density and the spatial proximity information. The molecular electronic density describes the position of electronic clouds in molecules. Information on spatial proximity corresponds to the measurement of distances (or angles) between atoms or regions of the molecule.

NMR is one of the major techniques used for studying biomolecular structures. NMR is able to give very sensitive information on distances or angles between atoms. NMR is also able to provide information on the internal dynamics of the molecule. There are also other experimental techniques that can give measurements concerning distances between atoms: an example is given by fluorescence techniques (FRET, cellular imaging) and by hybrid methods, such as mass spectrometry coupled with cross-linking. Due to the very high sensitivity of fluorescence and to the lack of limitation on the size of the analyzed objects, these techniques are likely to continue their development in coming years.

Historically, distance-based methods started to develop when the methods based on electronic density were already well-established. The main objective of such methods was to identify a three-dimensional conformation for a molecule from the experimental data obtained while applying the mentioned experimental methods (for a detailed definition of this problem, see Sect. 16.5). The development of these distance-based methods represented a new challenge in biology and required an intensive intellectual investment.

Other problems in structural biology can also benefit from distance information. One important example is the docking problem, where the conformation of two (or more) molecules, during their interaction, is searched. In general, it is supposed that the conformation of the first molecule $M_1$, as well as the conformation of the second molecule $M_2$, is known. The interest is in discovering the way $M_1$ and $M_2$ arrange their conformations in space during the interaction. NMR and other techniques, such as FRET, can provide distance information between pairs of atoms $(u,v)$ such that $u$ belongs to $M_1$ and $v$ belongs to $M_2$. Since the conformations of $M_1$ and $M_2$, separately, are supposed to be known, other distances can be derived, and, together with the measured distances, can be exploited for analyzing the interaction between the two molecules [22]. In docking, generally, $M_1$ represents a protein, whereas $M_2$ is generally referred to as the ligand and can be another type of molecule.

Homology modeling or, more accurately, comparative modeling [49] attempts the construction of protein conformations by using their chemical composition, which can be easily derived from the sequence (or the sequences) of amino acids forming the molecule, and the similarity of the sequence with other proteins of known conformation. The geometric information required for the structure construction is obtained by comparing the sequence of the protein under study to the sequences of proteins with known structures. The idea is to associate to the atoms of the protein under study some geometric constraints so that they can resemble the local conformation of a protein having a similar sequence. Then, a distance-based method can be used for predicting the conformation of the protein under study.

## 16.4 Geometric Parameters Measured by Nuclear Magnetic Resonance in Biomolecules

NMR studies the behavior of the magnetic moments of spin nuclei and is based on the observation of the so-called NMR resonance. A resonance frequency characterizes the return of each perturbed magnetic moment to equilibrium. In proteins, the nuclei $^{1}$H, $^{13}$C, and $^{15}$N can be observed. The protein sample is submitted to an intense external magnetic field, inducing the alignment of the magnetic moment of the observed nuclei. The perturbation of any aligned magnetic moment is transmitted through dipolar interactions of the moments to the magnetic moments of neighboring nuclei. The transmission of the perturbation is called nuclear Overhauser effect (NOE) and is roughly proportional to $d^{-6}$, where $d$ is the distance between two protons belonging to the two atoms $u$ and $v$. The NOE between $u$ and $v$ is located in the spectrum at coordinates $\delta_u$ and $\delta_v$ representing the *chemical shifts* of $u$ and $v$. These chemical shifts are deduced from the corresponding resonance frequencies.

Errors in the measurement of the distances by NMR can have a number of reasons:

- The sample molecule can undergo dynamics or conformational exchange so that the conversion of the measured signal into a distance becomes difficult.
- The signal recorded during NOE measurement may be distorted by experimental noise or by processing artifacts.
- The NOE measurement related to two atoms $u$ and $v$ is also influenced by other neighboring atoms through a process called *spin diffusion*.

The most common procedure to minimize the effects of spin diffusion and internal mobility is to qualitatively classify the NOE intensities by converting them into distance intervals [25]. A strong NOE is typically assigned to an interproton distance below 2.7 Å, a medium NOE to a distance below 3.3 Å, and a weak NOE to a distance below 5.0 Å [5]. These values define therefore the upper bounds on the possible distances associated to the pair of atoms. The corresponding lower bounds are defined by the sum of the Van der Waals radii of the involved atoms. Thus, from an NOE measurement, a suitable interval can be defined, where the actual distance $d$ between the two atoms $u$ and $v$ is (most likely) contained.

It is important to remark that not all obtained interval distances can be unambiguously assigned to a pair of atoms $(u,v)$. More than one pair of atoms can have the same chemical shifts $\delta_u$ and $\delta_v$ so that one single NOE measurement (one real value) can refer to several distances. For a set of undistinguished pairs $(u,v)$, the relationship between NOE measurement and the distances $d(u,v)$ is approximately

$$\bar{d} = \left[ \sum_{(u,v)} [d(u,v)]^{-6} \right]^{-1/6} .$$

In this case, the assignment of a distance to a pair $(u,v)$ is generally done in an iterative way. Often, only unambiguous NOEs are used at first in order to identify a possible conformation for the protein under study. Then, additional NOEs can be assigned on the basis of some preliminary found three-dimensional conformations [17]. The ambiguous NOEs can be also automatically managed [45, 47] during the structure calculation.

Forms of ambiguity are common in methylene and propyl groups of valines and leucines. In this situation, the distance constraints are often directed to a pseudoatom [56]. A pseudoatom can be placed halfway between the two atoms of a methylene group: the distances concerning the real atoms are successively increased with respect to the ones obtained for the pseudoatom. Pseudoatoms can also be used to describe unresolved NOEs involving protons that are equivalent due to motion, such as protons in methyl groups or aromatic rings. The necessary correction for the

NOE-derived distances can be deduced from theoretical considerations [26], as well as from the size of the rotating group, for example, the upper bound for an NOE involving a methyl group is often increased by 1 Å [56].

The chemical shift strongly depends on the type of the nucleus ($^{1}$H, $^{13}$C, or $^{15}$N) and on the chemical environment. The latter observation led to an empirical relationship in order to correlate the chemical shifts of $C_\alpha$ and $H_\alpha$ atoms to the secondary structures to which the corresponding amino acid belongs. This correlation is commonly named chemical shift index (CSI) [54]. TALOS+ [51] is a software tool based on a neural network that is able to predict the secondary structures of subsequences of amino acids from chemical shifts obtained by NMR experiments. In practice, this software tool is able to provide some constraints on the $\phi$ and $\psi$ torsion angles which are generally employed for the protein backbone representation (see Sect. 16.2).

NMR usually provides only short-range distances. If two atoms are more than 5–6 Å apart, then there is no NOE signal that can be measured for estimating their relative distance. Furthermore, only intervals between pairs of atoms which are visible on the NMR spectra can be estimated. Only distances between pairs of hydrogen atoms are useful for structure determination of biological macromolecules.

This makes it necessary to complement the NOE information by additional information derived from the local geometry of the molecule. If the chemical structure of the molecule is known, distances between bonded atoms or angles among triplets of bonded atoms can be computed. In general, chemical bonds allow some small variations on these relative distances, but it is very common to fix such distances for reducing the degrees of freedom of the whole molecular structure. The exact values for these distances can be obtained, for example, by X-ray crystallography measurements of small molecules or single amino acids [12]. The use of such distances makes it possible to consider the torsion angle representation of proteins discussed in Sect. 16.2. Although this is the mostly common approach, there is a quite original approach where the atoms are isolated [14]; here, NMR distances and bond distances play the same role.

## 16.5 The Molecular Distance Geometry Problem

The information provided by NMR essentially consists in a list of distances between some pairs of atoms of the considered molecule. NMR is also able to provide additional information, such as lower and upper bounds on the backbone torsion angles. However, this additional information can be converted in distance-based constraints, so that we can consider, in general, that the available information about the molecule only consists of distances.

The distance geometry problem (DGP) is therefore the problem of identifying the three-dimensional conformation of a molecule by exploiting a set of available distances between some pairs of its atoms [9, 18]. Formally, we can represent an instance of the DGP as a weighted undirected graph $G = (V, E, d)$ having the following properties. The vertex set $V = \{1, 2, \ldots, n\}$ contains vertices $v$ representing atoms (i.e., the protons of the atoms) which compose the molecule, in a certain predefined ordering. In the following, the cardinality of $V$, i.e., the number of atoms/vertices in the graph, will be referred to as $n$ or $|V|$. The edge set $E$ contains all pairs of vertices $(u, v)$ for which the distance between the atoms corresponding to $u$ and $v$ is known; the weight $d(u, v)$ associated to the edge $(u, v)$ provides the numerical value of the distance. It can be an exact value (i.e., one single real numerical value), or, more often, an interval. Finally, we suppose that a total order relation is associated to the vertices of $G$, which may not correspond to the natural atomic ordering in some molecules such as proteins. When molecules are concerned, the DGP is usually referred to as molecular DGP (MDGP). With a little abuse of notation, we will refer to each $v \in V$ as "vertex" of $G$, as well as "atom" of a molecule.

At the beginning of this discussion, we will suppose that all distances in $G$ are precise. In this case, the MDGP can be seen as the problem of finding a conformation $x = (x_1, x_2, \ldots, x_n)$ such that all constraints

$$||x_u - x_v|| = d(u, v) \qquad \forall (u, v) \in E \tag{16.1}$$

are satisfied. In the formula, $|| \cdot ||$ represents the computed distance between two atomic coordinates belonging to the conformation $x$, whereas $d(u, v)$ represents the known distance between the two atoms (the weight associated to the edge). The MDGP is a constraint satisfaction problem.

Let us suppose that the distance between all pairs of atoms $u$ and $v$ is known. In such a case, the number of equations (16.1) is $n(n-1)/2$ (naturally, two edges $(u, v)$ and $(v, u)$ correspond to the same distance). In order to fix the conformation in the three-dimensional space (for avoiding to consider solutions that can be obtained by translating and/or rotating other solutions), the first three atoms of the molecule can be fixed in space. At this point, there are other $3n - 9$ atomic coordinates to identify in order to find a conformation $x$ which satisfies all constraints (16.1). Note that the number of coordinates is one order smaller than the number of distances. Therefore, in the simple case in which all interatomic distances are available, the distance information is redundant. In other words, only a subset of distances is actually necessary for finding a solution $x$ to the MDGP.

Let us suppose that we need to find the position in space for the atom $v \in V$, and that all the other atoms that precede $v$ in the ordering associated to $V$ have already been positioned. If all distances are available, in particular the distance between $v - 1$ and $v$ is available. Geometrically, the constraint (16.1) associated to this distance defines a sphere which is centered in the position of $v - 1$ and has radius $d(v - 1, v)$. Hence, the possible positions for $v$ belong to this sphere. Since similar spheres can be defined for all the other atoms $u$ such that $u < v$, the coordinates for $v$ can be identified by intersecting all these spheres. As it is well known, in the hypothesis that all distances (radius) are precisely known, the intersection between two spheres gives one circle, the intersection among three spheres gives two points, and the intersection among four spheres gives one point only. As a consequence, any additional sphere to be intersected with the others would not produce any additional information. It is important to remark that there exist particular cases where the sphere intersections can provide different results (e.g., the intersection of three spheres with aligned centers gives a circle and not two points), but the possibility for this to happen has probability 0 in a mathematical sense.

Let $G$ be a graph representing an instance of the MDGP. If we suppose that the first four vertices are placed in fixed positions and that, for each other vertex $v > 4$, there are four adjacent vertices, i.e., four edges $(u,v)$ with $u < v$, then the MDGP can be solved in linear time and there is only one possible solution. In this hypothesis, indeed, for each $v > 4$, there are at least four spheres that can be intersected for the identification of the coordinates of $v$. Since the intersection always produces one point only, the unique solution to this MDGP can be found in linear time. We remark that graphs $G$ satisfying this property for the edges in $E$ are called *trilateration graphs*, and it has been formally proved that MDGPs related to trilateration graphs can be solved in polynomial time [13]. There is, in fact, a solution method for the MDGP with exact distances that is based on this hypothesis [10, 55].

In general, however, one is far from this ideal situation. As discussed in Sect. 16.4, the quantity of distances estimated by NMR is limited to short-range distances, and they mainly concern pairs of hydrogen atoms, while a protein is composed by hydrogens but also carbons, nitrogens, oxygens , and some sulfur. Moreover, NMR distances are estimated and not measured precisely. In general, therefore, the MDGP is an NP-hard problem [50]. We will discuss in the next session some suitable techniques for refining NMR distances and for generating additional distances from the ones obtained by NMR.

## 16.6 Refinement of NMR Distances

As previously discussed, NMR experiments are able to provide a list of distances for a subset of atom pairs from a given molecule. The majority of such distances are imprecise, i.e., they are represented by suitable intervals where the actual distance is supposed to be contained. Moreover, the distances are generally not available for all pairs of atoms, but rather only a small subset of distances can be estimated by NMR.

Before any method for the solution of the MDGP can be applied, it is very important to verify the quality of the distances that were obtained by NMR. An effective and simple test for verifying whether the distances are compatible to one another is the one employing the well-known *triangle inequality*. Suppose there are three vertices $u$, $v$, and $w$ such that the three edges $(u,v)$, $(v,w)$, and $(u,w)$ are present in the edge set $E$. The three vertices form the triangle $\widehat{uvw}$, and the triangle inequality

$$d(u,w) \leq d(u,v) + d(v,w) \tag{16.2}$$

ensures that one side of the triangle ($(u,w)$ in this case) is not larger than the sum of the other two. If the triangle inequality is not satisfied, then some of the involved distances need to be corrected. If there are distances not satisfying the triangle inequalities, then the solution to the MDGP cannot be a Euclidean object, which contradicts the definition of molecular conformation. This compatibility test can be easily generalized to interval distances: in this case, the portion of interval associated to the distance $d(u,w)$, where there are distances not satisfying the inequality (16.2), can be discarded.

Let us suppose now that we have the three vertices $u$, $v$, and $w$ and that the edge $(u,w)$ is not available. In this case, we can estimate the distance associated to this pair of vertices $u$ and $w$ by exploiting the triangle inequality. Because of Eq. (16.2), the distance $d(u,w)$ has as upper bound the sum of the two distances associated to $(u,v)$ and $(v,w)$. If the upper bound is considered for the distance, we have a degenerate triangle $\widehat{uvw}$ (the angle in $v$ is equal to $180°$). Smaller values for the distance produce different triangles with different values for the angles in $v$. However, there is another limit on the values for the distances, which is the one corresponding to an angle in $v$ equal to $0°$. Therefore, the lower bound for the distance is

$$d(u,w) \geq |d(u,v) - d(v,w)|. \tag{16.3}$$

This lower bound can be increased in case it is smaller than the sum of the Van der Waals (VdW) radii of the two involved atoms $u$ and $v$.

Based on these simple rules, there is a procedure for reducing the interval lengths obtained by NMR and for redistributing the distance information along the atoms of the considered molecule. It is generally called *bound smoothing* procedure. In the very first works on this topic [3, 9], the term distance geometry described this procedure, checking the consistency of a set of distance intervals. Only later it became the preprocessing step for the following structure generation step. Bound smoothing allows to reduce the distance intervals before attempting the solution to the problem, and, at the same time, it allows to distribute the NMR information, originally mainly concerning hydrogen atoms, to other atoms of the molecule.

Since there are $n^3$ possible triangles in a molecule formed by $n$ atoms, the bound smoothing procedure can be quite expensive. However, the NMR information is rather sparse, and therefore not all triangles actually have to be checked. The search for the possible triangles can be optimized by considering that, for each given edge $(u,w)$, only pairs of edges $(u,v)$ and $(v,w)$, for some $v \in V$, are of interest. We remark that, in graph theory, these triangles are named *cliques* and that the enumeration of all cliques of a graph $G$ with a predefined size $K$ (in our case, $K = 3$) can be performed in polynomial time.

Apart from the triangle inequalities, there are other higher-order inequalities that can be verified in order to have the compatibility among the distances in $G$. Tetrangle, pentangle, and hexangle inequalities involve the distances between four, five, and six atoms, respectively. These inequalities can, in theory, be used just like the triangle inequalities in the bound smoothing procedure. They are actually able to better refine the NMR distances, by reducing the difference between the lower and the upper bound in the intervals which represent the distances. Whereas the triangle inequality is valid for all dimensions of space, some of these additional inequalities are more specific to three-dimensional space [9]. However, the computational cost increases for the verification of these higher-order inequalities. For this reason, only the triangle and the tetrangular inequalities have been employed in the past [11]. Nowadays, thanks to the increasing computer power, higher-order inequalities might also be considered.

## 16.7 The Metric Matrix Distance Geometry

The metric matrix distance geometry (MMDG) has been the first employed method for NMR structure determination [4, 7–9, 18, 19]. In the following, we will give a few details about this method and discuss the reason why it was discarded in recent years. The interested reader can refer to [1] for additional details about the MMDG. The basic idea is to exploit the properties of a matrix of interatomic distances, to which we refer as *metric matrix*, and to perform the following four main steps:

1. The available NMR distances are checked for consistency and refined by applying a *bound smoothing* procedure, as discussed in Sect. 16.6.
2. For any NMR distance which is represented by an interval, one sample distance is chosen (this process is called *metrization process* when this choice is made consistently).
3. The metric matrix is derived from the distance matrix; the eigenvectors and eigenvalues of the metric matrix are computed: this allows to generate the coordinates of the atoms forming the molecule (*embedding phase*).
4. Possible errors in the obtained conformation are corrected by applying optimization techniques (*optimization phase*).

As discussed in Sect. 16.6, bound smoothing is an important preprocessing step for the solution of an MDGP containing NMR data. However, the resulting set of distances is such that many distances are still represented by intervals (whose length can be up to 3 Å or more). This is the reason why the MMGP has an additional preprocessing (Step 2), where sample distances are chosen from the intervals.

In the simplest implementation, each distance is randomly chosen in each single interval, independently from the choices made for other intervals. However, there is an important issue regarding this simple process. After step 1, all interval distances are compatible to each other: all possible triangles (cliques) satisfy the triangle inequalities. Once three exact distances have been chosen in step 2, however, the corresponding triangle inequality will not be satisfied anymore. To overcome this difficulty, one can use metrization, a process where the bound smoothing is repeated after each distance choice.

Another important aspect of the metrization is given by the ordering in which the sample distances are chosen. If the natural order for the atoms is chosen (we consider the distances related to the first atom, and then we proceed with the ones related to the successive atoms, until the end), then there is the risk of introducing artifacts. Empirically, it was shown that the better results are obtained when the sequence of intervals is randomly chosen [27].

Once a set of exact distances is computed from the available intervals, the metric matrix $\mathscr{G}$ can be defined (see [1] for additional details). Then, the eigenvalues and the eigenvectors of $\mathscr{G}$ are computed. The eigenvectors provide the coordinates of the atoms forming the molecule, i.e., they provide the solution to the MDGP.

Unfortunately, this step of the MMDG does not always provide an acceptable result. The exact distances taken from the intervals during the metrization step, indeed, may not be consistent with a three-dimensional Euclidean object (such as a protein conformation). In this case, the number of eigenvalues which is greater than 0 is $k > 3$ so that our conformation does not belong to the three-dimensional space.

The exact distances taken from the intervals during the metrization step very likely are not consistent with a three-dimensional Euclidean object (such as a protein conformation). In this case, the number of nonzero eigenvalues is $k > 3$ so that our conformation does not belong to the three-dimensional space. One usually truncates therefore the eigenvalue series after the third. This is the optimal projection of the higher-dimensional object into three-dimensional space. In some cases eigenvectors related to more than three strictly positive eigenvalues need to be considered [53].

The last step of the MMGP method consists in *optimizing* the conformation obtained in the embedding step. The generic approach is to define a penalty function which gives penalties to violations for the available constraints, as well as for some local conformations that are not typical in proteins. The chosen penalty function can be minimized, for example, by using a conjugate gradient minimization method [44], which is a local search optimization method.

The MMGP was initially employed in structural biology for solving MDGPs with NMR data. It has largely been replaced by these methods, because of convergence issues and since optimization algorithms are more flexible. More recent approaches to the MDGP are based on suitable reformulations of the MDGP as a global optimization problem.

## 16.8 Methods Based on Global Optimization

Global optimization aims at finding the global minimum (or the set of global minima) of a certain mathematical function (called *objective function*) under the hypothesis that some constraints are satisfied. Many real-life applications lead to the formulation of a global optimization problem [38]. Depending on the properties of the objective function and constraints, suitable methods can be employed for the solution of the optimization problem.

The MDGP can be reformulated as an unconstrained global optimization problem. The satisfaction of the constraints based on the distances can be measured by computing the difference between the left and the right side in the constraints (16.1). In order to verify the overall satisfaction of the available constraints, a penalty function can be defined, whose general term is related to the generic constraint. Different penalty functions can be defined for the MDGP, and the most used one is the largest distance error (LDE):

$$\mathrm{LDE}(x) = \frac{1}{|E|} \sum_{(u,v)} \frac{|\,||x_u - x_v|| - d(u,v)\,|}{d(u,v)}. \tag{16.4}$$

Finding the global minimum of this penalty function allows to obtain solutions to the MDGP. If all distances are compatible to each other and there are no errors, the LDE value in conformations $x$ which are solution for the MDGP is supposed to be zero.

Penalty functions that can be defined for the MDGP generally contain several local minima. This makes the task of finding the global minimum (or the global minima) of the penalty function very difficult. Many methods may get trapped at local minima, and there might not be ways to verify whether the found minimum is global or not. There is a wide literature on global optimization, the interested reader is referred, for example, to [20].

In the following two sections, we discuss some methods for the MDGP which are based on a global optimization reformulation of the problem. We point out that this is not meant to be a comprehensive survey. We rather focus the rest of this chapter on two particular methods: the one which is nowadays mostly used for the determination of the protein conformations deposited on the PDB (see Sect. 16.8.1) and another one that is more recent and potentially able to identify better-quality conformations of proteins (see Sect. 16.8.2). The reader who is interested in a wider discussion on global optimization methods for the MDGP is referred to recent surveys [30, 34, 36].

### *16.8.1 SA-Based Methods*

The SA [23] was introduced in 1983 by Kirkpatrick in order to solve nonlinear optimization problems. The basic idea is to simulate the annealing physical process, where a given system (such as a glass of water) is cooled down slowly for obtaining a low-energy structure (such as the crystalline structure of a piece of ice). In the simulation, particles of the physical system are represented by the variables of a certain objective function, while its energy is given by the objective function value. Randomly generated solutions to the problem are computed during the simulation, and, as the system is cooled down, the possibility to accept solutions that increase the system energy gets lower and lower. SA depends on a set of parameters, such as the initial temperature, the cooling schedule, and the number of solutions to be randomly generated. It belongs to the class of meta-heuristic approaches, which can be potentially applied to any optimization problem. As for all meta-heuristic searches, SA can give no guarantees to converge towards the global optimum.

In 1988, SA was proposed [46] as a valid alternative to the MMDG method outlined in Sect. 16.7. The MMDG, actually, has been reduced to a preprocessing step for generating initial candidate solutions to be given to SA. The employment of SA overcame some important issues, such as the one of finding embeddings in spaces with a dimension higher than 3 (see Sect. 16.7). In SA, indeed, the solution space is fixed and represented by a subregion of the Euclidean three-dimensional space. Shortly afterwards, the MMDG step was abandoned altogether [50].

To date, this is the method that is mostly used for the determination of protein conformations from NMR data (as a quick search on the PDB [2] can show). This can be due to the ease of implementation of meta-heuristics such as SA, as well as to the availability of software tools where SA is implemented together with other useful tools for managing NMR data. Available software include ARIA [48], CYANA [16], and UNIO [15]. In these implementations, SA represents one step of a more complex procedure, where, for example, ambiguous NOEs (see Sect. 16.4) are verified by exploiting partial solutions found by SA.

The whole procedure, however, and in particular the SA-based step, is heuristic. Decisions taken during the procedure, such as random modifications in candidate solutions or the rejection of some ambiguous distances on the basis of partially obtained solutions, can lead the search in the wrong direction, without any possibility to backtrack. It is important to remark that, even if the procedure can identify a solution for which all available distances are satisfied, this does not imply that it represents the actual protein structure. All possible conformations should be identified, and the ones having the most evident biological sense should be taken into consideration.

### *16.8.2 A Discrete and Exact Method for the MDGP*

In the formulated global optimization problem, the domain of the penalty function (such as, e.g., the function (16.4)) generally corresponds to a subregion of the three-dimensional Euclidean space. As a consequence, an infinite number of potential solutions are contained in this subregion, because it is continuous. The SA approach discussed in the previous section is based on a search in such a continuous space. Under certain assumptions, however, this subregion can be transformed in a discrete domain, where a finite number of potential solutions is contained.

Let $G$ be a weighted undirected graph representing an instance of the MDGP with exact distances. As discussed in Sect. 16.5, if $G$ is a trilateration graph, then there is information enough to solve the problem in polynomial time. For a given ordering on its vertex set, a trilateration graph is such that, for each $v \in V$ with $v > 3$, there are at least four vertices $u < v$ such that the distances between any $u$ and $v$ is known. We say that, in this case, there are four *reference distances* for the vertex $v$. In this hypothesis, the only feasible position for this vertex can be computed by intersecting the four spheres defined by the four available distances regarding $v$

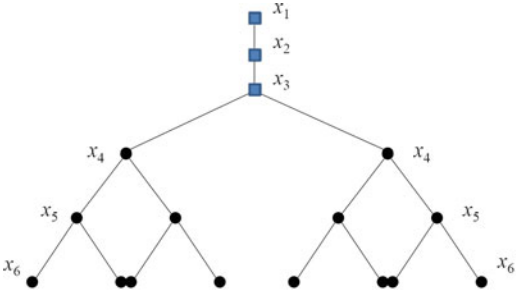


**Fig. 16.1** The search domain of a discretizable MDGP instance with exact distances

(see Sect. 16.5). By iterating this procedure from the vertex $v = 4$ until the last one in the ordering associated to the graph $G$, the molecular conformation can be obtained in only $|V| - 3$ steps.

This is an extreme case allowing for discretization. Instead of an infinite number of positions belonging to a continuous space, there is only one possible atomic position for each $v \in V$. The discretization is however still possible when weaker assumptions are satisfied [29]. If, for each $v > 3$, at least three (not four) vertices $u$ are available so that the distances between each $u$ and $v$ is known, then two possible positions for $v$ (not only one) can be computed by intersecting three spheres (see Sect. 16.5). In this case, the MDGP cannot be solved in polynomial time, because the new search domain is a binary tree organized in $n$ layers, each one containing the possible coordinates of a certain vertex in $G$ (see Fig. 16.1). On the last layer, there are $2^{n-3}$ possible atomic positions for the last vertex in the order associated to the graph $G$. As a consequence, this tree contains $2^{n-3}$ potential solutions to the MDGP.

On the basis of the consecutivity assumption for the reference distances, there are two classes of discretizable MDGP instances that can be defined. In the DMDGP [29], for each vertex $v > 3$, the three reference distances are between $v$ and, respectively, $v-1$, $v-2$, and $v-3$. In the DDGP [43], the reference distances can refer to any vertex which is smaller than $v$ in rank. As a consequence, it can be proved that the class of DMDGP instances is contained in the DDGP class. In the following, we will not make a precise distinction between the DMDGP and the DDGP. Therefore, we will say in general that an instance is *discretizable* if it belongs to one of these two subclasses of the MDGP.

The Branch and Prune (BP) algorithm [33] is based on the idea of efficiently exploring discrete search domains. It can be applied only in case the discretization assumptions are satisfied. The basic idea is to construct the binary tree step by step, i.e., atomic position by atomic position, and to verify the feasibility of such atomic positions as soon as they are computed. Suppose that a partial set of coordinates has already been computed for the vertices $u \in V$ which are smaller in rank than a certain given vertex $v$. By intersecting the three spheres as explained before, we can obtain the two corresponding possible positions for $v$. Then, by using some additional information on distances regarding $v$ that was not employed in the sphere intersection, the feasibility of such atomic positions can be verified. In case the position is not feasible (it does not satisfy at least one of the available distance constraints), then it can be removed from the tree. Moreover, the whole tree branch starting from this position can be pruned as well. The pruning phase of BP is its strong point.

Differently from meta-heuristic searches (such as the SA-based algorithm in Sect. 16.8.1) and methods which are based on an exploration of a continuous domain, the BP algorithm is a deterministic algorithm, which is potentially able to enumerate all solutions for a given instance of the MDGP. This point is crucial in protein structure determination. All possible conformations for a certain set of distances should be computed and successively analyzed.

As discussed in Sect. 16.4, NMR instances of the MDGP mostly contain interval distances (and not exact distances). In this case, however, even if the complexity of the problem increases with the uncertainty associated to the interval distances, the discretization is still possible. Let us suppose, for example, that, for a certain $v \in V$, two reference distances are exact, while the third distance is represented by an interval. In the sphere intersection, therefore, one of the spheres needs to be replaced by a spherical shell so that the new intersection consists, most likely, of two disjoint curves. In order to guarantee the discretization, a certain number of sample distances must be taken from the available interval, and a predetermined number of possible atomic positions on the two curves needs to be selected [32].

An instance of the MDGP is represented by a weighted undirected graph $G$ and by a vertex order for the vertices in $G$. Since the assumptions for the discretization strongly depend upon the given vertex order, changing the order can transform an MDGP instance into a discretizable instance, and vice versa. Therefore, given a graph $G$, it is interesting to verify whether there exist vertex orders that allow for the discretization [28].

This task is more complex when there are distances that are represented by intervals. In addition to the requirement on the presence of the distances necessary for performing the discretization, other conditions on the distance type (exact distance or interval) may need to be considered. When only one reference distance is an interval, indeed, the discretization can be performed by applying the strategy mentioned above (the intersection among two spheres and one spherical shell). When more than one reference distance is an interval, the intersection can give more complex Euclidean objects so that the definition of vertex orders avoiding for their generation can be necessary.

In [31, 32], instead of using an automatic tool, a vertex order has been handcrafted which allows for discretizing MDGPs concerning protein backbones and containing NMR data (see Fig. 16.2). This order is constructed so that, for each vertex $v > 3$, only one reference distance related to $v$ can be represented by an interval, whereas the other two are always exact. Similar orders for some protein side chains have been proposed in [6]. All these orders exploit distances derived from the chemical composition of proteins for the discretization process, whereas NMR distances are only employed for pruning purposes. This way, the discrete search domain cannot be affected by errors due to the NMR experiments. In order to

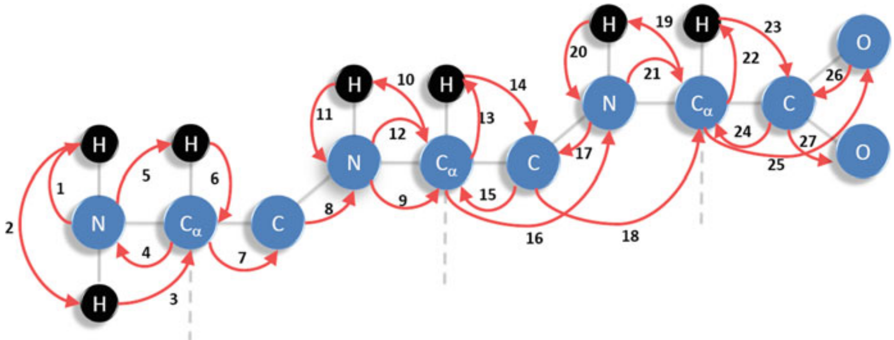


**Fig. 16.2** The handcrafted order for the discretization of protein backbones

consider NMR distances for pruning purposes only, cycling is possible in the orders, i.e., the same atom can be represented by more than one vertex of the graph $G$.

Vertex orders can also be generated so that the maximum width of the corresponding trees can be controlled. If pruning is performed by exploiting NMR distances (as in these hand-crafted orders), then it is important to place the hydrogen atoms (see Sect. 16.4) in strategic positions: if they are too far from each other in the order, pruning is not possible on too many consecutive layers of the tree, allowing for a consistent combinatorial explosion. A deep study on the width of BP trees can be found in [35] in the case all distances are exact.

The *interval* BP (*i*BP) [32] is an extension of the BP algorithm that is able to manage interval data. It has been conceived in order to manage the three following situations. First, the current vertex refers to a duplicated atom, i.e., to an atom which was already considered earlier in the order. In this case, the algorithm simply assigns to this vertex the same position of its previous copy (this implies that cycling does not increase the complexity of the problem). Second, the three reference distances for the current vertex are all exact, and the sphere intersection provides the only two possible positions. Third, one of the reference distances is represented by an interval. In this case, we need to intersect two spheres with a spherical shell, and this intersection provides two curves in the three-dimensional space. In order to discretize, we choose $D$ sample distances from the interval, and we intersect the corresponding three spheres $D$ times. As a consequence, $2 \times D$ possible atomic positions are determined for the current vertex.

Another important point in BP is the fact that it can manage wrongly assigned distances [39] in a deterministic way. Instead of pruning a tree branch as soon as an atomic position does not satisfy one of the distance constraints, the idea is to delay the pruning phase until a predefined number of violations are found. This approach, unfortunately, can be inefficient when the predefined maximum number of violations is large enough to significantly increase the tree width. Work is currently in progress for overcoming this issue.

## 16.9 New Perspectives in NMR Distance Geometry

Discovering the three-dimensional structure of molecules such as proteins is a very important and challenging problem in biology and biomedicine. In recent years, the research community actively worked on this problem, known as the MDGP. Despite this great effort, a lot of research still need to be performed in order to identify good-quality conformations of biological molecules.

The solution to an MDGP from NMR data can be mainly divided in two main steps. Firstly, the molecule is isolated and analyzed in solution by NMR spectroscopy (see Sect. 16.3); then, the distance information provided by the experiments is exploited for the construction of the molecular conformation (see Sect. 16.5). Both steps are strongly multidisciplinary so that biologists, chemists, physicists, mathematicians, and computer scientists can work in concert on efficient and reliable solution methods.

In spite of this fact, there are nowadays not so many interactions among these communities. In the biological community, the currently used methods for the solution of MDGPs containing NMR data are all based on the meta-heuristic SA (see Sect. 16.8.1), which can give no guarantees of optimality. On the other side, the operational research community developed several more sophisticated and accurate methods for the MDGP (see [30,34,36] for recent surveys). However, some methods rely on assumptions that may not be satisfied in biology, or their performances have never been evaluated on real NMR data. It is worth remarking that the SA-based methods for MDGP would not be able to provide any approximation to solutions if they were not coupled with appropriate tools for NMR management.

The BP algorithm (see Sect. 16.8.2) is a recent algorithm for the MDGP whose development is performed in a strong multidisciplinary collaboration. Firstly developed for solving artificial MDGP instances [33], the algorithm has been adapted successively for solving NMR instances [41]. It is very promising because of its deterministic nature. Differently from SA-based methods for the MDGP, the BP algorithm is potentially able to identify all the conformations satisfying the distance constraints. In other words, BP can enumerate all solutions to the mathematical problem, to be filtered later in order to discovering the most probable biological conformations. Any other conformation which is not contained in the BP solution set cannot be a solution to the problem (this statement is not true when meta-heuristic methods are employed). The development of BP is currently in progress and we believe it could be a valid alternative to currently employed methods.

Another interesting point for future research is the following. As discussed in Sect. 16.4, there is actually another problem to be solved prior the formulation of an MDGP with NMR data: the NOE assignment problem. In some cases this problem can be tough, especially in presence of ambiguous NOEs, and it can be the source of errors. We foresee therefore the possibility of integrating this assignment problem inside the BP algorithm. The employment of a completely deterministic method could allow for overcoming many of the issues causing errors in other methods.

**Acknowledgements** This work is partially supported by the ANR project ANR-10-BINF-03-01, “Bip:Bip”. TM and MN acknowledge support by the CNRS and the Institut Pasteur.

# References

1. Almeida, F., Moraes, A., Neto, F.G.: Overview on protein structure determination by NMR — Historical and future perspectives of the use of distance geometry. In: Mucherino et al, “Distance Geometry: Theory, Methods, and Applications”
2. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The Protein Data Bank. Nucleic Acid Res. **28**, 235–242 (2000)
3. Blumenthal, L.M.: Theory and Application of Distance Geometry. Chelsea, New York (1970)
4. Braun, W., Bösch, C., Brown, L.R., Gō, N., Wüthrich, K.: Combined use of proton-proton Overhauser enhancements and a distance geometry algorithm for determination of polypeptide conformations. Application to micelle-bound glucagon. Biochimica et Biophysica Acta **667**, 377–396 (1981)
5. Clore, G.M., Gronenborn, A.M.: Determination of three-dimensional structures of proteins and nucleic acids in solution by nuclear magnetic resonance spectroscopy. Crit. Rev. Biochem. Mol. Biol. **24**, 479–564 (1989)
6. Costa, V., Mucherino, A., Lavor, C., Carvalho, L.M., Maculan, N.: On suitable orders for discretizing molecular distance geometry problems related to protein side chains. In: IEEE Conference Proceedings, Federated Conference on Computer Science and Information Systems (FedCSIS12), Workshop on Computational Optimization (WCO12), Wroclaw, Poland, September 9–12 (2012)
7. Crippen, G.M.: A novel approach to calculation of conformation: distance geometry. J. Comput. Phys. **24**(1), 96–107 (1977)
8. Crippen, G.M., Havel, T.F.: Stable calculation of coordinates from distance information. Acta Crystallographica Section A**34**, 282–284 (1978)
9. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
10. Davis, R.T., Ernst, C., Wu, D.: Protein structure determination via an efficient geometric build-up algorithm. BMC Struct. Biol. **10**:S7 (2010)
11. Easthope, P.L., Havel, T.F.: Computational experience with an algorithm for tetrangle inequality bound smoothing. Bull. Math. Biol. **51**, 173–194 (1991)
12. Engh, R.A., Huber, R.: Accurate bond and angle parameters for X-ray structure refinement. Acta Crystallographica Section A **47**, 392–400 (1991)
13. Eren, T., Goldenberg, D.K., Whiteley, W., Yang, Y.R., Morse, A.S., Anderson, B.D.O., Belhumeur, P.N.: Rigidity, computation, and randomization in network localization. In: IEEE Infocom Proceedings, 2673–2684 (2004)
14. Grishaev, A., Llinas, M.: Protein structure elucidation from NMR proton densities. Proc. Nat. Acad. Sci. USA **99**, 6713–6718 (2002)
15. Guerry, P., Herrmann, T.: Comprehensive automation for NMR structure determination of proteins. Meth. Mol. Biol. **831**, 33–56 (1992)
16. Güntert, P.: Automated NMR structure calculation with CYANA. In: Downing, A.K. (ed.) Protein NMR Techniques. Meth. Mol. Biol. **278**, 353–378 (2004)
17. Güntert, P., Berndt, K.D., Wüthrich, K.: The program ASNO for computer-supported collection of NOE upper distance constraints as input for protein structure determination. J. Biomol. NMR **3**, 601–606 (1993)
18. Havel, T.F.: Distance geometry. In: Grant, D.M., Harris, R.K. (eds.) Encyclopedia of Nuclear Magnetic Resonance, pp. 1701–1710. Wiley, New York (1995)
19. Havel, T.F., Kunts, I.D., Crippen, G.M.: The theory and practice of distance geometry. Bull. Math. Biol. **45**, 665–720 (1983)

20. Horst, R., Pardalos, P.M.: Handbook of Global Optimization. Springer (1994), http://www.springer.com/mathematics/book/978-0-7923-3120-9
21. Huang, A., Stultz, C.M.: Finding order within disorder: elucidating the structure of proteins associated with neurodegenerative disease. Future Med. Chem. **1**, 467–482 (2009)
22. Janin, J.: Protein-protein docking tested in blind predictions: the CAPRI experiment. Mol. Biosyst. **6**, 2351–2362 (2010)
23. Kirkpatrick, S., Jr. Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
24. Kleywegt, G.J.: Validation of protein models from $C_\alpha$ coordinates alone. J. Mol. Biol. **273**(2), 371–376 (1997)
25. Kline, A.D., Braun, W., Wüthrich, K.: Studies by $^1$H nuclear magnetic resonance and distance geometry of the solution conformation of the a-amylase inhibitor Tendamistat. J. Mol. Biol. **189**, 377–382 (1986)
26. Koning, T.M., Davies, R.J., Kaptein, R.: The solution structure of the intramolecular photoproduct of d(TpA) derived with the use of NMR and a combination of distance geometry and molecular dynamics. Nucleic Acids Res. **18**, 277–284 (1990)
27. Kuszewski, J., Nilges, M., Brünger, A.T.: Sampling and efficiency of metric matrix distance geometry: a novel partial metrization algorithm. J. Biomol. NMR **2**, 33–56 (1992)
28. Lavor, C., Lee, J., Lee-St.John, A., Liberti, L., Mucherino, A., Sviridenko, M.: Discretization orders for distance geometry problems. Optim. Lett. **6**(4), 783–796 (2012)
29. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: The discretizable molecular distance geometry problem. Comput. Optim. Appl. **52**, 115–146 (2012)
30. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
31. Lavor, C., Liberti, L., Mucherino, A.: On the solution of molecular distance geometry problems with interval data. In: IEEE Conference Proceedings, International Workshop on Computational Proteomics (IWCP10), International Conference on Bioinformatics & Biomedicine (BIBM10), Hong Kong, 77–82 (2010)
32. Lavor, C., Liberti, L., Mucherino, A.: The *interval* Branch-and-Prune algorithm for the discretizable molecular distance geometry problem with inexact distances, to appear in J. Global Optim. (2012), http://link.springer.com/article/10.1007%2Fs10898-011-9799-6
33. Liberti, L., Lavor, C., Maculan, N.: A Branch-and-Prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
34. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular Distance Geometry Methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
35. Liberti, L., Masson, B., Lavor, C., Mucherino, A.: Branch-and-Prune trees with bounded width. In: Proceedings of the 10th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW11), Rome, Italy, 189–193 (2011)
36. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications, Tech. Rep. 1205.0349v1 [q-bio.QM], arXiv (2012)
37. Mucherino, A., Costantini, S., di Serafino, D., D'Apuzzo, M., Facchiano, A., Colonna, G.: Towards a Computational Description of the Structure of all-alpha Proteins as Emergent Behaviour of a Complex System. Comput. Biol. Chem. **32**(4), 233–239 (2008)
38. Mucherino, A., Seref, O.: Modeling and solving real life global optimization problems with meta-heuristic methods. In: Papajorgji, P.J., Pardalos, P.M. (eds.) Advances in Modeling Agricultural Systems, pp. 403–420 (2008) http://link.springer.com/chapter/10.1007%2F978-0-387-75181-8_19
39. Mucherino, A., Liberti, L., Lavor, C., Maculan, N.: Comparisons between an exact and a meta-heuristic algorithm for the molecular distance geometry problem. In: ACM Conference Proceedings, Genetic and Evolutionary Computation Conference (GECCO09), Montréal, Canada, 333–340 (2009)
40. Mucherino, A., Papajorgji, P., Pardalos, P.M.: Data Mining in Agriculture. Springer, New York (2009)

41. Mucherino, A., Lavor, C., Malliavin, T., Liberti, L., Nilges, M., Maculan, N.: Influence of pruning devices on the solution of molecular distance geometry problems. In: Pardalos, P.M., Rebennack, S. (eds.) Lecture Notes in Computer Science **6630**, Proceedings of the 10th International Symposium on Experimental Algorithms (SEA11), Crete, Greece, 206–217 (2011)
42. Mucherino, A., Lavor, C., Liberti, L.: Exploiting symmetry properties of the discretizable molecular distance geometry problem. J. Bioinformatics Comput. Biol. **10**(3), 1242009 (2012)
43. Mucherino, A., Lavor, C., Liberti, L.: The discretizable distance geometry problem, Optim. Lett. **6**(8), 1671–1686 (2012)
44. Nazareth, J.L.: Conjugate gradient method. Wiley Interdiscipl. Rev. Comput. Stat. **3**(1), 348–353 (2009)
45. Nilges, M.: Calculation of protein structures with ambiguous distance restraints. Automated assignment of ambiguous NOE crosspeaks and disulphide connectivities. J. Mol. Biol. **245**, 645–660 (1995)
46. Nilges, M., Clore, G.M., Gronenborn, A.M.: Determination of three-Dimensional structures of proteins from interproton distance data by hybrid distance geometry – dynamical simulated annealing calculations. Fed. Eur. Biochem. Soc. **229**, 317–324 (1988)
47. Nilges, M., Marcias, M.J., O'Donoghue, S.I.: Automated NOESY interpretation with ambiguous distance restraints: the refined NMR solution structure of the pleckstrin homology domain from $\beta$-spectrin. J. Mol. Biol. **269**, 408–422 (1997)
48. Rieping, W., Habeck, M., Bardiaux, B., Bernard, A., Malliavin, T.E., Nilges, M.: ARIA2: Automated NOE assignment and data integration in NMR structure calculations. Bioinformatics **23**(3), 381–382 (2007)
49. Sali, A., Blundell, T.L.: Comparative protein modelling by satisfaction of spatial restraints. J. Mol. Biol. **234**, 779–815 (1993)
50. Saxe, J.B.: Embeddability of weighted graphs in $k$-space is strongly NP-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
51. Shen, Y., Delaglio, F., Cornilescu, G., Bax, A.: TALOS+: a hybrid method for predicting protein backbone torsion angles from NMR chemical shifts. J. Biomol. NMR **44**, 213–223 (2009)
52. Spedding, M.: Resolution of controversies in drug/receptor interactions by protein structure. Limitations and pharmacological solutions. Neuropharmacology **60**, 3–6 (2011)
53. Weber, P.L., Morrison, R., Hare, D.: Determining stereo-specific $^1$H nuclear magnetic resonance assignments from distance geometry calculations. J. Mol. Biol. **204**, 483–487 (1988)
54. Wishart, D.S., Sykes, B.D.: The $^1$3C chemical-shift index: a simple method for the identification of protein secondary structure using 13C chemical-shift data. J. Biomol. NMR **4**, 171–180 (1994)
55. Wu, D., Wu, Z.: An updated geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global Optim. **37**, 661–673 (2007)
56. Wüthrich, K., Billeter, M., Braun, W.: Pseudo-structures for the 20 common amino acids for use in studies of protein conformations by measurements of intramolecular proton-proton distance constraints with nuclear magnetic resonance. J. Mol. Biol. **169**(4), 949–961 (1983)

# Chapter 17
# Using a Distributed SDP Approach to Solve Simulated Protein Molecular Conformation Problems

**Xingyuan Fang and Kim-Chuan Toh**

**Abstract** This chapter presents various enhancements to the DISCO algorithm (originally introduced by Leung and Toh (SIAM J. Sci. Comput. **31**:4351–4372, 2009) for anchor-free graph realization in $\mathbb{R}^d$) for applications to conformation of protein molecules in $\mathbb{R}^3$. In our enhanced DISCO algorithm for simulated protein molecular conformation problems, we have incorporated distance information derived from chemistry knowledge such as bond lengths and angles to improve the robustness of the algorithm. We also designed heuristics to detect whether a subgroup is well localized and significantly improved the robustness of the stitching process. Tests are performed on molecules taken from the Protein Data Bank. Given only 20% of the interatomic distances less than 6 Å that are corrupted by high level of noises (to simulate noisy distance restraints generated from nuclear magnetic resonance experiments), our improved algorithm is able to reliably and efficiently reconstruct the conformations of large molecules. For instance, given 20% of interatomic distances which are less than 6 Å and are corrupted with 20% multiplicative noise, a 5,600-atom conformation problem is solved in about 30 min with a root-mean-square deviation (RMSD) of less than 1 Å.

## 17.1 Introduction

Determining protein structure is an important problem in biology. Majority of the protein structures are obtained by X-ray crystallography. However, some proteins

X. Fang
Department of Operations Research and Financial Engineering, Princeton University, NJ, USA
e-mail: xingyuan@princeton.edu

K.-C. Toh (✉)
Department of Mathematics, National University of Singapore,
Singapore-MIT Alliance, Singapore
e-mail: mattohkc@nus.edu.sg

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_17,

could not be crystallized, and the information we have from its solution state is some pairwise atomic distance bounds [known as nuclear overhauser effect (NOE) distance restraints] estimated from nuclear magnetic resonance (NMR) spectroscopy experiments. From late 1970s, distance geometry algorithms have become increasingly used in the interpretation of experimental data on macromolecular conformation. Generally, we use these algorithms to determine the Cartesian coordinates of the atoms of a molecule which are consistent with a predetermined set of intramolecular distance constraints. Those constraints could come from experimental data and known chemistry knowledge such as bond lengths and angles. In 1984, the structure of the first protein determined in its native solution state from NMR data was computed by the algorithm DISGEO [14].

The mathematical setting of the molecular conformation problem is as follows. We wish to determine the coordinates of $n$ atoms $x_i \in \mathbb{R}^3$, $i = 1,\ldots,n$. The information that is available consists of measured distances or distance bounds for some of the pairwise distances $||x_i - x_j||$ with $(i,j) \in \mathcal{N}$, where $\mathcal{N}$ is the set of index pairs $(i,j)$ for which interatomic distance information is available. Note that in our convention, we only consider the pair $(i,j)$ with $i < j$. The molecular conformation problem is an instance of the graph-realization problem where the atoms are the vertices of the graph and the pairs $(i,j) \in \mathcal{N}$ are the edges with the weight of edge $(i,j)$ specified by the given distance data $\tilde{d}_{ij}$. In a $p$-dimensional graph-realization problem, one is interested in determining points in $\mathbb{R}^p$ such that $||x_i - x_j|| \approx \tilde{d}_{ij}$ for all $(i,j) \in \mathcal{N}$.

The molecular conformation problem is closely related to the *sensor network localization problem*, but much more challenging. In the sensor network localization 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 practice, the anchors and sensors are able to communicate with one another if they are not too far apart (say within a certain cutoff range), to obtain an estimate of the distance for each communicable pair. For the molecular conformation problem, there are no anchors. And more importantly, not all pairs of atoms within the cutoff range have distance estimates. In this chapter, we will use the term "conformation," "localization," and "realization" interchangeably.

Recently, semidefinite programming (SDP) relaxation techniques have been applied to the sensor network localization problem [5]. While this approach was successful for moderate-size problems with the number of sensors in the order of a few hundreds, it was unable to solve problems with a large number of sensors, due to computational limitations in SDP algorithms for solving large-scale problems. To localize larger networks, a distributed SDP-based algorithm for sensor network localization was proposed in [4]. The critical assumption required for the algorithm in [4] to work well is that there exist anchors distributed uniformly throughout the physical space. As a result, the algorithm cannot be applied to the molecular conformation problem, since the assumption of uniformly distributed anchors does not hold in the case of molecular conformation.

In [6], a distributed SDP-based algorithm (called DAFGL) was proposed and tested for the molecular conformation problem. The performance of the DAFGL algorithm is satisfactory when given 50% of pairwise distances less than 6 Å apart that are corrupted by 5% multiplicative noise. More recently, Leung and Toh [18] proposed a new distributed approach, the DISCO (for DIStributed COnformation) algorithm, with a view toward applications in molecular conformation. The DISCO algorithm was demonstrated to be efficient and robust in solving simulated molecular conformation problems when given only 30% of the pairwise distances less than 6 Å which are corrupted by 20% multiplicative noise. However, DISCO frequently fails to give good results when given only 20% pairwise distances less than 6 Å apart.

In this chapter, we describe the DISCO algorithm and the enhancements we made to make the algorithm work for the protein molecular conformation problem under the highly sparse distance data regime of using only 20% pairwise distances less than 6 Å that are corrupted by high level of noise. We should mention that in this work, real NOE distance data from NMR experiments are not considered, although that is our future goal. Instead, the distance data we considered are simulated from known protein conformations in order to validate the results obtained by DISCO by comparing the reconstructed conformations to the original ones. The protein molecules we used in our experiments are downloaded from the protein data bank (PDB). In our experiments, we discard all the hydrogen atoms in the molecules. The input distances (all in the unit of Å $= 10^{-10}$ m) are given as lying in intervals $[\underline{d}_{ij}, \overline{d}_{ij}]$ for $(i,j) \in \mathscr{N}$. Here $\mathscr{N}$ denotes the set of index pairs $(i,j)$ for which interatomic distance bounds are available. In our simulated protein molecular conformation problem, we consider two types of distance bounds. The first type of bounds comes from known chemistry information such as bond lengths and angles, and the interval is given in the form

$$\underline{d}_{ij} = (1-\varepsilon)d_{ij}, \quad \overline{d}_{ij} = (1+\varepsilon)d_{ij} \quad \forall\,(i,j) \in \mathscr{N}_{\mathrm{c}}, \tag{17.1}$$

where $\varepsilon \in (0, 0.1)$ is a parameter which can be chosen appropriately to reflect our confidence on the given distance $d_{ij}$ derived from the chemistry information of the molecule. The index set $\mathscr{N}_{\mathrm{c}}$ is used to denote the index pairs $(i,j)$ for which distance bounds based on chemistry information are given. The second type of bounds is designed to simulate NOE restraints, and they are given as follows:

$$\underline{d}_{ij} = \max\Big(2.0, (1-\sigma|\underline{z}_{ij}|)d_{ij}\Big), \quad \overline{d}_{ij} = (1+\sigma|\overline{z}_{ij}|)d_{ij} \quad (i,j) \in \mathscr{N}_{\mathrm{s}}, \tag{17.2}$$

where $d_{ij}$ is the true distance between atoms $i$ and $j$ and $\underline{z}_{ij}, \overline{z}_{ij}$ are independent random variables such that $|\underline{z}_{ij}|, |\overline{z}_{ij}|$ have unit mean value. The parameter $\sigma$ is the noise factor which we typically set to 20%. The number 2.0 in the expression for $\underline{d}_{ij}$ is a conservative lower bound on the shortest distance between two nonbonded (non-hydrogen) atoms. The index set $\mathscr{N}_{\mathrm{s}}$ is used to denote the index pairs $(i,j)$ for which the simulated distance bounds are given. Note that the overall index set $\mathscr{N}$ is the disjoint union of $\mathscr{N}_{\mathrm{c}}$ and $\mathscr{N}_{\mathrm{s}}$.

The main enhancements we made are as follows. First, we have included some distances derived from basic chemistry information such as bond lengths and bond angles into the distance data used in the conformation. We relied on the papers [1,17] for those basic chemistry information. To automatically derive those chemistry information for a protein molecule given its amino acids sequence, we find it convenient to design a structure array data structure to code the information pertaining to each atom in the molecule. Second, we also designed effective heuristics to detect whether a subgroup of atoms is well localized, as well as substantially improved the robustness of the stitching process in the DISCO algorithm. We demonstrate that our enhanced DISCO algorithm is efficient and robust, and it works well under the highly sparse distance data regime we have targeted. Compared to the original DISCO algorithm, the RMSD errors of some reconstructed molecules have been significantly improved when given only 20% of pairwise distances (corrupted by 20% multiplicative noise) less than 6 Å. For example, the average RMSD of the reconstructed conformations for the molecule `1RGS` is reduced from 4.5 Å to 1.3 Å over ten random instances. We should mention that the 20% distances less than 6 Å used to simulate NOE restraints are randomly selected from the set of all distances less than 6 Å excluding those derived from chemistry knowledge.

As nonrandom test problems for evaluating sensor network localization algorithms are of interest to the sensor network community, we plan to make the distance data we have generated in this chapter publicly available at the following website: http://www.math.nus.edu.sg/$\sim$mattohkc/disco.html

This chapter is organized as follows. Section 17.2 describes some existing molecular conformation algorithms; Sect. 17.3 details the mathematical models for molecular conformation based on SDP; Sect. 17.4 explains the design of DISCO and describes the enhancements we made; Sect. 17.5 describes the chemistry information we have incorporated into our simulated protein molecular conformation problems; Sect. 17.6 contains the experimental setup and numerical results; Sect. 17.7 gives the conclusion.

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

## 17.2 Related Work

Due to its great importance, there are quite a number of existing algorithms to tackle the molecular conformation problem. We discuss selected works in this section. Here we do not attempt to give a detailed survey of the existing work on the distance geometry approach for solving the molecular conformation problem. Our intention is only to highlight the most relevant work for which the experimental settings used bear the closest similarity with ours. For the existing algorithms which we mention in this section, we pay attention to each algorithm by the following aspects: the mathematics, its input data, and the results it is able to provide. In particular, we make a note of the largest molecule which each algorithm was able to solve and its error (mostly measured by RMSD) in the tests done by the authors. This information is summarized in Table 17.1.

Before we begin, we note that from the theory of distance geometry [25–27], there is a natural correspondence between inner product matrices and Euclidean distance matrices. Thus it is quite common to solve the molecular conformation problem by working with an inner product matrix. If we denote the atom coordinates by column vectors $x_i$, and let $X = [x_1 \ldots x_n]$, then the inner product matrix is given by $Y = X^{\mathrm{T}}X$. If we have a computed $\tilde{Y}$, then we can recover the approximate coordinates $\tilde{X}$ by taking the best rank-3 approximation based on the eigenvalue decomposition of $\tilde{Y}$.

The earliest distance geometry-based algorithm for molecular conformation is the EMBED algorithm [15] developed by Havel, Kuntz, and Crippen in 1983. The input data of EMBED consists of lower and upper bounds on some of the pairwise distances. EMBED uses the triangle and tetrangle inequalities to compute distance bounds for all pairs of points, followed by choosing random numbers within the bounds to form an estimated distance matrix $\tilde{D}$ (one should note that the triangle and tetrangle bounds are generally too weak to provide good estimates on the distances). It checks whether $\tilde{D}$ is close to a valid rank-3 Euclidean distance matrix by considering the three largest eigenvalues (in magnitude) of $\tilde{Y}$, the inner product matrix corresponding to $\tilde{D}$. In the fortunate case where 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 acceptable. 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.

Subsequently, Havel and Wüthrich developed the DISGEO package [14] to improve the performance of EMBED. As the EMBED algorithm is unable to compute a conformation of the whole protein structure, due to the high dimensionality of the problem, DISGEO uses two passes of EMBED to overcome the problem. In the first pass, coordinates are computed for a subset of atoms subject to constraints inherited from the whole structure. In this step, EMBED uses data not only from experiments but also from chemistry knowledge, including bond lengths, bond angles, and hybridization theory. This step forms a "skeleton" for the structure. The second pass of EMBED then computes coordinates for the remaining atoms, building upon the skeleton computed in the first pass. The authors tested the performance of DISGEO on the `BPTI` protein, which has 454 atoms. The input consists of distance (3,290) and chirality (450) constraints needed to fix the covalent structure and bounds (508) for distances between hydrogen atoms in different amino acid residues that are

**Table 17.1** A summary of protein conformation algorithms

| 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) | 1,870 | 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 2–3 Å(*) |
| MDS (02) | 700 | All distances less than 7 Å were given as lying in $[d_{ij} - 0.01, d_{ij} + 0.01]$ | Violations $< 0.01$ Å |
| StrainMin (06) | 5,147 | 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) | 1,849 | All distances between atoms in the same amino acid given exactly. All distances between pairs of hydrogen atoms less than 3.5 Å apart, given exactly | Exact |
| Geometric build-up (07) | 4,200 | All distances between atoms less than 5 Å apart given exactly | Exact |
| DAFGL (07) | 5,681 | 70% of the distances less than 6 Å were given as lying in $[\underline{d}_{ij}, \overline{d}_{ij}]$, where $\underline{d}_{ij} = (1 - 0.05\lvert\underline{z}_{ij}\rvert)_+ d_{ij}, \overline{d}_{ij} = (1 + 0.05\lvert\overline{z}_{ij}\rvert)$ $d_{ij}$, and $\underline{z}_{ij}, \overline{z}_{ij}$ are standard normal random variables with zero mean and unit variance | RMSD 3.16 Å |

(*) The RMSD of 1.07 Å reported by GNOMAD may be incorrect, and the true value should be about 2–3 Å since the number reported in Fig. 11 of [29] does not agree with that appears in Fig. 8

less than 4 Å apart, to simulate the distance constraints available from a nuclear Overhauser effect spectroscopy experiment. Using a pseudostructure representation, they were able to solve for 666 geometric points.[1] Havel's DG-II package [13], 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 work in this chapter is a continuation of the DISCO algorithm developed in 2009 [18]. DISCO differs from the previous methods in that it applies SDP relaxation methods to obtain the inner product matrix. In order to solve larger problems, it employs a divide-and-conquer approach for which each basis 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 to 5,600. The input data consists of 30% of the distances below 6 Å, given as lying in intervals $[\underline{d}_{ij}, \overline{d}_{ij}]$ which are generated from the true distances $d_{ij}$ with 20% multiplicative noises added. Given such input, DISCO is able to produce a conformation for most molecules with an RMSD of 2–3 Å.

Distributed algorithms (based on successive decomposition) similar to those in [6] were proposed for fast manifold learning in [30, 31]. In addition, those papers also considered recursive decomposition. The manifold learning problem is to seek a low-dimensional embedding of a manifold in a high-dimensional Euclidean space by modeling it as a graph-realization problem. The resulting problem has similar characteristics as the molecular conformation problem (which is an anchor-free graph-realization problem) we consider in this chapter, but there are some important differences which we should highlight. For the manifold learning problem, exact pairwise distances between any pairs of vertices (atoms in our case) are available, but for the molecular conformation problem, only a very sparse subset of pairwise distances are assumed to be given and are only known within a given range. Such a difference implies that for the former problem, any local patch will have a "unique" embedding (up to rigid motion and certain approximation errors) computable via an eigenvalue decomposition, and the strategy to decompose the graph into subgraphs is fairly straightforward. In contrast, for the latter problem, given the sparsity of the graph and the noise in the distances data, the embedding problem itself requires a new method, not to mention that sophisticated decomposition strategies also need to be devised.

The GNOMAD algorithm [29] by Williams, Dugan, and Altman is a gradient descent-based algorithm which attempts to satisfy the input distance constraints as well as minimum separation distance (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

[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, given 3,798 distance and 450 chirality constraints, with three computed structures having an average RMSD of 2.08 Å from the known crystal structure.

intelligently, they may trap the algorithm at an unsatisfactory local minimum. Since it is difficult to optimize all the atom positions simultaneously because of the high dimensionality of the problem, GNOMAD updates the positions of the atoms one atom at a time. The authors tested GNOMAD on the protein molecule 1TIM, which has 1,870 atoms. Given all the covalent distances and distances between atoms that share covalent bonds to the same atom, as well as 30% of pairwise distances less than 6 Å, they were able to compute a conformation with an RMSD of 2–3 Å. [2] The experimental setting we considered in this chapter is similar to that given in [29], but in the highly challenging regime of having very sparse and noisy distances. Also the algorithm we designed is completely different from GNOMAD. Our algorithm's input includes all the covalent distances and distances between atoms that share covalent bonds with the same atom. We also add 20% of pairwise distances less than 6 Å (which are corrupted by high level of noise) to simulate distances derived from an NMR experiment.

In this brief discussion on related work, we have not touched on approaches based on global optimization and discrete optimization methods. For example, in [19], the authors developed a branch-and-prune method. We refer the reader to [18,19,29] for more detailed review of various distance geometry-based methods including those in [9, 11, 16, 21], proposed for the molecular conformation problem.

## 17.3 Optimization Models for the Molecular Conformation Problem

We begin this section with the optimization models we consider for the molecular conformation problem. Then we introduce the SDP relaxations for these models and describe the gradient descent method we adopt for improving the positions of the atoms by using the SDP solution as the starting point. Finally, we present the alignment problem for stitching (sometimes we will also use the words "merging" and "combining") two groups of atoms together using the overlapping atoms in the groups.

### *17.3.1 Semidefinite Programming Models*

In the "measured distances" model, we have measured distances $\tilde{d}_{ij}$ for certain pairs of atoms, i.e.,

$$\tilde{d}_{ij} \approx ||x_i - x_j|| \qquad (i,j) \in \mathscr{N}. \tag{17.3}$$

[2] The RMSD of 1.07 Å reported in Fig. 11 in [29] is inconsistent with that appearing in Fig. 8. It seems that the correct RMSD should be about 2–3 Å.

In this model, the unknown positions $\{x_i\}_{i=1}^n$ represent the best fit to the measured distances, obtained by solving the following nonconvex minimization problem:

$$\min\left\{\sum_{(i,j)\in\mathcal{N}}\left|||x_i-x_j||^2-(\tilde{d}_{ij})^2\right|\right\}. \tag{17.4}$$

For convenience, we denote the measured interatomic distance matrix by $\tilde{D}$. In the “distance bounds” model, we have lower and upper bounds on the distances between certain pairs of atoms, i.e.,

$$\underline{d}_{ij} \le ||x_i-x_j|| \le \overline{d}_{ij} \qquad (i,j)\in\mathcal{N}. \tag{17.5}$$

In this model, the unknown positions $\{x_i\}_{i=1}^n$ represent any feasible solution $X = [x_1,\dots,x_n]$ satisfying the bound constraints. We denote the lower and upper bound distance matrices by $\underline{D}$, $\overline{D}$. Note that for the “distance bounds” model, we can naturally convert it to the “measured distance” model by taking $\tilde{d}_{ij} = (\underline{d}_{ij}+\overline{d}_{ij})/2$.

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

$$Y := X^{\mathrm{T}}X, \quad \text{where } X = [x_1 \dots x_n]. \tag{17.6}$$

Let $\{e_i\}_{i=1}^n$ be set of standard unit vectors in $\mathbb{R}^n$. By denoting $e_{ij} = e_i - e_j$, we note that

$$||x_i-x_j||^2 = e_{ij}^{\mathrm{T}}Ye_{ij}.$$

We can therefore conveniently express the constraints in Eqs. (17.3) and (17.5), respectively, as

$$(\tilde{d}_{ij})^2 \approx e_{ij}^{\mathrm{T}}Ye_{ij} \qquad (i,j)\in\mathcal{N},$$

$$(\underline{d}_{ij})^2 \le e_{ij}^{\mathrm{T}}Ye_{ij} \le (\overline{d}_{ij})^2 \quad (i,j)\in\mathcal{N}.$$

The SDP relaxation then consists in relaxing the constraint “$Y = X^{\mathrm{T}}X$” in Eq. (17.6) into the constraint “$Y \succeq 0$,” where the notation means that the $n\times n$ symmetric matrix $Y$ is positive semidefinite.

The SDP relaxation of the measured distances model (17.4) is given by

$$\min\left\{\sum_{(i,j)\in\mathcal{N}}\left|e_{ij}^{\mathrm{T}}Ye_{ij}-(\tilde{d}_{ij})^2\right| \mid Y\succeq 0\right\}. \tag{17.7}$$

Similarly we can express the SDP relaxation of the distance bounds model (17.5) as finding an element in the following set:

$$\left\{Y \mid (\underline{d}_{ij})^2 \le e_{ij}^{\mathrm{T}}Ye_{ij} \le (\overline{d}_{ij})^2 \quad \forall\,(i,j)\in\mathcal{N},\, Y\succeq 0\right\}. \tag{17.8}$$

Once we have obtained a matrix $Y$ by solving either Eq. (17.7) or Eq. (17.8), we can estimate the atom positions $X = [x_1 \ldots x_n]$ by setting $X$ to be the best rank-3 approximation of $Y$.

In [6], it has been shown that if the distance data is exact and the conformation problem is uniquely localizable, then the SDP relaxation (17.7) is able to produce the exact atom coordinates up to a rigid motion. We refer the reader to [6] for the definition of "uniquely localizable." Intuitively, it means that there is only one configuration in $\mathbb{R}^3$ (up to a rigid motion) that satisfies all the distance constraints. The result (which is a variant of the result established by So and Ye [22] for graph realization with anchors) gives a strong indication that the SDP relaxation technique is a powerful relaxation. We can therefore hope that applying SDP relaxation to problems with sparse and noisy distance data will be effective.

We now discuss what typically would happen 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 recover the true coordinates. Furthermore, the solution $Y$ of the SDP (17.7) or (17.8) will generally have rank greater than 3, as we shall explain next. 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, for noisy distance data, $Y$ will tend to have a rank higher than 3. Another reason for $Y$ to have a higher rank is that, if there are multiple optimal solutions in an SDP problem, interior-point methods used by many SDP solvers would converge to a solution with maximal rank [12].

This situation leads to potential issues. If $Y$ has a rank higher than 3, then the best rank-3 approximation of $Y$ is unlikely to give accurate positions for the atoms. To ameliorate this difficulty, we add the following regularization term into the objective function, i.e.,

$$-\gamma\langle I, Y\rangle, \tag{17.9}$$

where $\gamma$ is a positive regularization parameter. The motivation for introducing this term is to spread the atoms further apart so as to induce them to lie in a lower-dimensional space. Indeed, under the condition that $0 = \sum_{i=1}^n x_i = Xe$, where $e \in \mathbb{R}^n$ is the vector of all ones, we can easily show that $\sum_{i=1}^n \sum_{j=1}^n ||x_i - x_j||^2 = \langle I, Y\rangle/(2n)$ by using the definition that $Y = X^{\mathrm{T}}X$. We refer interested readers to [5] for details on the derivation of the regularization term. Thus, for the measured distances model, the regularized SDP model for Eq. (17.7) becomes

$$\min \left\{ \sum_{(i,j)\in\mathcal{N}} |e_{ij}^{\mathrm{T}} Y e_{ij} - (\tilde{d}_{ij})^2| - \gamma\langle I, Y\rangle \;\middle|\; e^{\mathrm{T}} Y e = 0,\; Y \succeq 0 \right\} \tag{17.10}$$

and the one related to the distance bounds model (17.8) becomes

$$\min \left\{ -\langle I, Y\rangle \mid (\underline{d}_{ij})^2 \leq e_{ij}^{\mathrm{T}} Y e_{ij} \leq (\overline{d}_{ij})^2 \;\forall\, (i,j) \in \mathcal{N},\; e^{\mathrm{T}} Y e = 0,\; Y \succeq 0 \right\}. \tag{17.11}$$

Note that we have added the constraint "$e^{\mathrm{T}}Ye = 0$" to reflect the requirement for which the center of mass $Xe$ is fixed to the origin.

We should emphasize that as observed in [5], the inclusion of the regularization term in the SDP model can greatly improve the quality of the conformation solution $X$ generated by the SDP model together with a subsequent gradient descent refinement. However, the choice of the regularization parameter $\gamma$ is crucial. In our implementation of the enhanced DISCO algorithm, we adaptively adjust the value of $\gamma$ based on the following separation ratio:

$$\texttt{sep\_ratio} = \frac{1}{|\mathcal{N}|} \sum_{(i,j)\in\mathcal{N}} s_{ij}, \quad \text{where } s_{ij} = \frac{||x_i - x_j||}{\tilde{d}_{ij}}. \tag{17.12}$$

Observe that, if the solution $X$ generated by the SDP model is the correct conformation and $\tilde{d}_{ij}$ is the exact distance between atoms $i$ and $j$ for all $(i,j) \in \mathcal{N}$, then we must have `sep_ratio`=1. Of course, for noisy distances, `sep_ratio` would not be exactly 1, but we expect the value to be close to 1 if the computed conformation is not too different from the true one. In fact, from our extensive numerical experiments, we find that the value of `sep_ratio` should generally lie in the interval $[0.85, 1.1]$ in order for the conformation solution (generated from the SDP model) to have a reasonably good quality (measured in terms of the RMSD with respect to the true conformation). Based on such a criterion, we increase $\gamma$ if `sep_ratio` is too small and decrease it if `sep_ratio` is too big. If after 5 trials, we cannot get `sep_ratio` to lie in the required interval, we declare that the underlying conformation problem $\{\tilde{d}_{ij} | (i,j) \in \mathcal{N}\}$ is not localizable and flag it as "bad."

Note that, in a distributed algorithm like DISCO, it is very important for us to design a reasonably good heuristic to detect whether a configuration is bad. This is because a bad configuration should not be stitched to a good (well-localized) configuration for otherwise it will destroy the good one after the two configurations are aligned and stitched. A bad configuration should be separately handled after the majority of the atoms in the molecule have been localized.

### 17.3.2 Coordinate Refinement via Gradient Descent

If we are given measured pairwise distances $\tilde{d}_{ij}$, then the atoms' coordinates can also be computed as the minimizer of the following nonconvex minimization problem:

$$\min f(X) := \sum_{(i,j)\in\mathcal{N}} \left(||x_i - x_j|| - \tilde{d}_{ij}\right)^2 - \beta \sum_{i=1}^{n} \sum_{j=1}^{n} ||x_i - x_j||^2. \tag{17.13}$$

Note that the above objective function is different from that of Eq. (17.4) because we want it to be differentiable. Observe that as in our SDP model, we have added a regularization term (with positive parameter $\beta$) in the objective function in Eq. (17.13). Similarly, if we are given bounds $\underline{d}_{ij}, \overline{d}_{ij}$ for pairwise distances, then the configuration can be computed as the solution of the following problem:

$$\min \sum_{(i,j)\in\mathcal{N}} \left(||x_i - x_j|| - \underline{d}_{ij}\right)_-^2 + \left(||x_i - x_j|| - \overline{d}_{ij}\right)_+^2 - \beta \sum_{i=1}^{n}\sum_{j=1}^{n} ||x_i - x_j||^2. \quad (17.14)$$

We can solve Eq. (17.13) or Eq. (17.14) by applying local optimization methods. For simplicity and computational efficiency, we choose to use a gradient descent method with backtracking line search. The algorithmic framework of this method is rather straightforward, so we shall omit the details. It is a simple exercise in calculus to find the gradient of $f$ with respect to the coordinate vector $x_i$. However, we should emphasize that, to compute the gradient efficiently, the computation must be designed appropriately. In our implementation, we find it convenient to construct the $n \times |\mathcal{N}|$ sparse node-arc incidence matrix $E$ for which the $(i,j)$th column contains only two nonzero entries with 1 and $-1$ at row $i$ and $j$, respectively. With $E$, we have that $XE = [x_i - x_j \mid (i,j) \in \mathcal{N}]$. Thus to calculate $[||x_i - x_j|| \mid (i,j) \in \mathcal{N}]$, one just needs to take the norm of the columns of $XE$.

The problems (17.13) and (17.14) are highly nonconvex problems with many local minimizers. Thus, if the initial iterate $X^0$ is not close to a good local minimizer, then it is extremely unlikely that the resulting $X$ 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 $X$ which improves upon the solution $X^0$ obtained from the SDP relaxation.

We should note that, while adding the regularization term in Eqs. (17.13) and (17.14) (with a suitably chosen parameter $\beta$) would generally lead to a more accurate conformation solution $X$, it can sometimes (though rarely happen) give a much worse solution if the parameter $\beta$ is chosen to be too large. In our implementation of the enhanced DISCO algorithm, we guard against such a bad case by examining the following expansion ratio:

$$\texttt{expansion\_ratio} = \max\{||x_i||/||x_i^0|| \mid i = 1,\ldots,n\},$$

where the columns of $X^0$ and $X$ are translated to have their respective center of mass at 0. If the expansion ratio is larger than 3, we declare that the solution $X$ is worse than $X^0$, and we discard the solution $X$ from the refinement process. In our more sophisticated implementation, we would perform another pass of the gradient descent refinement by deleting the potentially "bad" atoms and their corresponding distances from the optimization model while also reducing the parameter $\beta$. By doing so, one hopes that the coordinates of the remaining atoms can be improved.

### 17.3.3 Alignment of Configurations

A molecular configuration has translational, rotational, and reflective freedom. Nevertheless, we need to be able to compare two configurations to determine how similar they are. In order to do so, it is necessary to align them in a common coordinate system. Given $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$, we can define the "best" alignment as the affine transformation $T$ that minimizes the following problem:

$$\min_{c\in\mathbb{R}^3, Q\in\mathbb{R}^{3\times 3}} \left\{ \sum_{i=1}^{n} ||T(x_i) - y_i||^2 \mid T(x_i) = c + Q(x_i)\ \forall\, i,\ Q \text{ is orthogonal} \right\}. \tag{17.15}$$

The functional form of $T$ restricts it to be a combination of translation, rotation, and reflection. In the special case when the columns of $X = [x_1, \ldots, x_n]$ and $Y = [y_1, \ldots, y_n]$ are centered at the origin, Eq. (17.15) reduces to the following orthogonal Procrustes problem:

$$\min_{Q\in\mathbb{R}^{3\times 3}} \{ ||QX - Y||_F \mid Q \text{ is orthogonal} \}.$$

It is well known that the optimal $Q$ can be computed from the singular value decomposition of $XY^{\mathrm{T}}$.

## 17.4 The Basic Ideas of the DISCO Algorithm

Here we present the basic ideas of the DISCO algorithm (for DIStributed COnformation). For the detail algorithm, we refer the reader to [18].

Before having DISCO, we could solve the molecular conformation problem by using the SDP relaxation technique and gradient descent refinement if the molecule was not too big (say the number of atoms was below 500). The aim of DISCO is to solve large-scale problems.

A natural idea to solve a large conformation problem is to employ a divide-and-conquer approach, which applies the following general framework. If the number of atoms is not too large, then solve the conformation problem via SDP, and apply gradient descent refinement to improve the coordinates; otherwise, break the atoms into two subgroups, solve each subgroup recursively, and align and stitch them back together subsequently, again postprocessing the coordinates by applying gradient descent refinement after each stitching step.

We find that the use of the divide-and-conquer approach not only allow us to solve larger problem. It also allows us to design a more robust algorithm. For example, the repeated use of gradient descent refinement after each stitching step has certainly helped DISCO to become much more successful in producing accurate conformations. As a by-product of the divide-and-conquer process, we find that certain information collected during the SDP localization and stitching steps in DISCO can be used to help us to design a more robust algorithm.

Next we describe how DISCO (a) recursively divides a molecule into two subgroups of atoms and (b) how to stitch (as well as how to decide whether to stitch) two processed subgroups together. The idea DISCO uses to tackle the first issue is to minimize the number of edges between the two subgroups. The reason is that when a group is split into two disjoint subgroups, the edges (distances) between the two subgroups are lost. In other words, some distance information is lost. Thus DISCO tries to minimize the information lost. For the second issue, DISCO's strategy is for the two subgroups to have overlapping atoms. If the overlapping atoms are accurately localized in the two subgroups, then they can be aligned for the purpose of stitching the two subgroups together. If not, it would not be a good idea to align and stitch them since a bad subgroup can destroy the good one when they are stitched together. Therefore, DISCO designed a heuristic criteria for determining whether the overlapping atoms are accurately localized. In order to have a reliable alignment based on the overlapping atoms in the two subgroups, one of the most obvious criterion is that the RMSD of the coordinates of the overlapping atoms contained in the two subgroups must not be large, say less than 3 Å; otherwise, it gives a strong indication that at least one of the two subgroups is not well localized. We have observed that the RMSD of the overlapping atoms used in the stitching of two subgroups provides valuable information on the quality of the larger stitched configuration. Thus, for each stitched configuration, we assign a quality index (q_index) as follows:

$$\texttt{q\_index} = \max \left\{ \begin{array}{l} \text{RMSD of} \\ \text{overlapping} \\ \text{atoms} \end{array}, \; \frac{1}{|\mathcal{N}'|} \sum_{(i,j)\in\mathcal{N}'} \max\left(s_{ij}^5, s_{ij}^{-5}\right) \right\}, \qquad (17.16)$$

where $s_{ij}$ is defined as in (17.12) and $\mathcal{N}'$ is the subset of $\mathcal{N}$ involved in the stitched configuration, but it excludes the large outlier values of more than 100 in $\max(s_{ij}^5, s_{ij}^{-5})$. The power of 5 in (17.16) is chosen based on empirical experience. At the basis level, the quality index is assigned based on the SDP solution $Y$ obtained from (17.10) or (17.11) as follows:

$$\texttt{q\_index} = \left( \frac{1}{|\mathcal{N}'|} \sum_{(i,j)\in\mathcal{N}'} \max\left(s_{ij}^5, s_{ij}^{-5}\right) \right)^{1/2},$$

where $s_{ij}$ is calculated based on the best rank-3 approximation $X = [x_1, \ldots, x_{n'}]$ of the SDP solution $Y$. If a basis configuration is already declared as "bad," we set its q_index to be $\infty$. Note that in the case of sensor network localization with exact distance data, it has been demonstrated in [3] that the quantity $Y_{ii} - ||x_i||^2$ gives an error measure of the estimated position for the $i$th point. Unfortunately, when the distance data is highly noisy, that error measure has no obvious correlation to the underlying accuracy of the computed position $\boldsymbol{x_i}$. Thus we cannot use $\{Y_{ii} - ||\boldsymbol{x_i}||^2 \,|\, i = 1, \ldots, n\}$ to assign a value for q_index.

**Algorithm 9** The DISCO algorithm

```
procedure Disco($L, U$)
if number of atoms < basis size then
    [$cAest, cI$] ← DiscoBasis($L, U$)
else
    [$cAest, cI$] ← DiscoRecursive($L, U$)
end if
[$cAest, cI$] ← DiscoPatch($L, U$)
return $cAest, cI$
end procedure

procedure DiscoBasis($L, U$)
$cI$ ← LikelyLocalizableComponents($L, U$)
for $i = 1, \ldots,$Length($cI$) do
    $cAest\{i\}$ ← SdpLocalize($cI\{i\}, L, U$)
    $cAest\{i\}$ ← Refine($cAest\{i\}, cI\{i\}, L, U$)
end for
return $cAest, cI$
end procedure

procedure DiscoRecursive($L, U$)
[$L_1, U_1, L_2, U_2$] ← Partition($L, U$)
[$cAest_1, cI_1$] ← Disco($L_1, U_1$)
[$cAest_2, cI_2$] ← Disco($L_2, U_2$)
$cAest$ ← [$cAest_1, cAest_2$]
$cI$ ← [$cI_1, cI_2$]
repeat
    [$cAest, cI$] ← CombineChunks($cAest, cI$)
    [$cAest, cI$] ← Refine($cAest, cI, L, U$)
until no change
return $cAest, cI$
end procedure
```

To summarize, suppose we have two subgroups with quality indices, $\texttt{q\_index}^1$ and $\texttt{q\_index}^2$, and that they have sufficient number (which we set a threshold of 8 atoms) of overlapping atoms with a dense underlying subgraph, roughly speaking, we will stitch the two subgroups together only if $\max\{\texttt{q\_index}^1, \texttt{q\_index}^2\} < 3$ and that the RMSD of the overlapping atoms in the two subgroups is less than 3 Å.

Despite our rather effective heuristics to detect whether a subgroup is well localized and to decide whether two subgroups should be stitched, we should point out that DISCO may still fail to work for some cases (we refer the reader to [18] for details). Thus, it is necessary to invest more effort to partition a group of atoms into localizable subgroups and improve the heuristics for detecting badly localized subgroups. In our work, after incorporating chemistry information to the input distance data, we observe that it becomes slightly easier to partition a group of atoms into two localizable subgroups.

The pseudocode of the DISCO algorithm is presented in Algorithm 9. We illustrate how the DISCO algorithm solves a small molecule in Fig. 17.1.

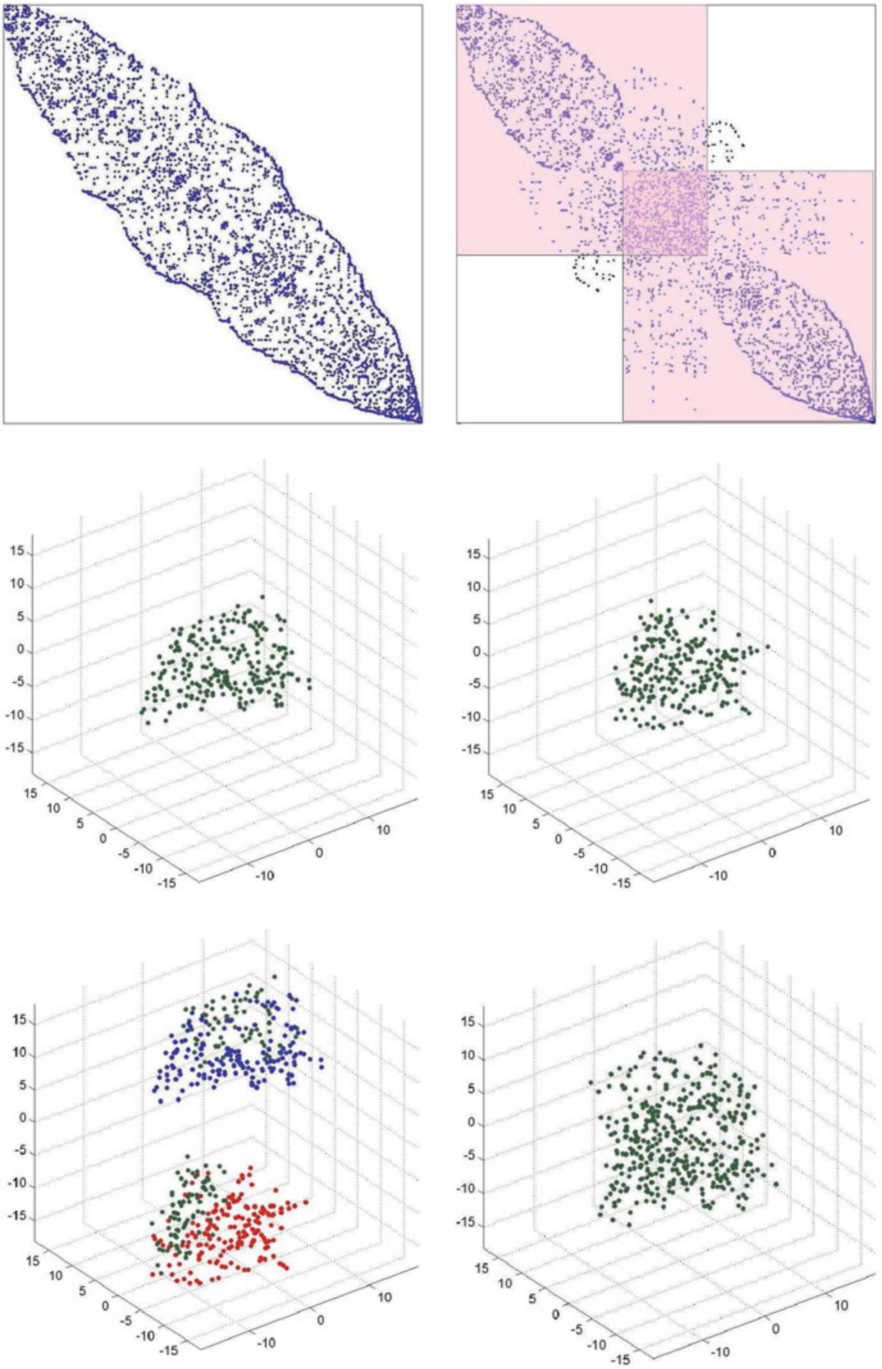


**Fig. 17.1** (*top left* and *right*) Since the number of atoms is too large ($n = 402 >$ basis size $= 300$), we divide the atoms into two subgroups. (*middle left* and *right*) We solve the subgroups independently. (*bottom left*) The subgroups have overlapping atoms, which are colored in *green*. (*bottom right*) The overlapping atoms allow us to align the two subgroups to form a conformation of the molecule

## 17.5 Chemistry Information

In this section, we describe the chemistry information we have added to the input distance data for DISCO.

### *17.5.1 Backbone Distances*

It is well known that a protein molecule contains a backbone (which serves as the main "skeleton" of the molecule) from which the general shape of the molecule is determined; see Fig. 17.2 for a schematic diagram of a backbone.

The most commonly known distance information between atoms in a molecule are bond lengths and bond angles. Given bond lengths and bond angles, we can easily calculate the distance between two atoms which are bonded to the same atom by the cosine law. Our first attempt is to incorporate known distance information for atoms in the protein molecule into our simulated protein structure determination problem. Specifically, we incorporate the following known distances:

(a) Bond lengths of bonded pairs of atoms along the backbone and distances between nonbonded atom pairs along the backbone which are derivable based on known bond lengths and bond angles by using the cosine law. The information we use comes mainly from the paper by Laskowski and Moss [17] and cross-checked with the data in [10]. The mean distances between various atom pairs along the backbone are given in Table 17.2. Since the bond lengths and bond angles are not known perfectly, but within small standard deviations around some mean values, we add the distance information in the form of lower and upper bounds as follows:

$$\underline{d}_{ij} = d_{ij}(1-r), \quad \overline{d}_{ij} = d_{ij}(1+r), \tag{17.17}$$

where $d_{ij}$ is the mean distance and $r$ is the standard deviation. For the distance coming from a bonded atom pair, we take $r$ to be 1%; for a nonbonded pair, we take $r$ to be 3%.

(b) Bond lengths of bonded pairs of atoms in the side chains. The main information we use is from a standard organic chemistry textbook [20] and a paper by

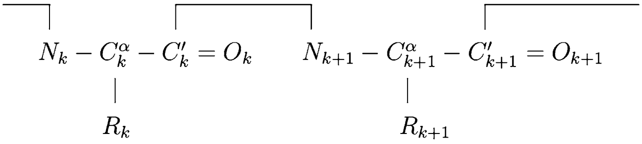

$N_k - C_k^\alpha - C_k' = O_k$
$N_{k+1} - C_{k+1}^\alpha - C_{k+1}' = O_{k+1}$
$R_k$
$R_{k+1}$

**Fig. 17.2** A schematic diagram of a protein backbone, where $R_k$ denotes the side chain of the $k$th amino acid

**Table 17.2** Bond lengths and bond angles for atoms on a protein backbone [17]

| | | | |
|---|---|---|---|
| $d(C' - N)$ | 1.32 | Bond length | [17] |
| $d(C = O)$ | 1.24 | Bond length | [17] |
| $d(C' - C^{\alpha})$ | 1.52 | Bond length | [17] |
| $d(C^{\alpha} - N)$ | 1.46 | Bond length | [17] |
| $d(C^{\alpha} - C^{\beta})$ | 1.53 | Bond length | [17] |
| $\tau(O = C' - N)$ | 123° | Bond angle | [17] |
| $\tau(O = C' - C^{\alpha})$ | 120° | Bond angle | [17] |
| $\tau(N - C' - C^{\alpha})$ | 116° | Bond angle | [17] |
| $\tau(N - C^{\alpha} - C')$ | 111° | Bond angle | [17] |
| $\tau(N - C^{\alpha} - C^{\beta})$ | 110° | Bond angle | [17] |
| $\tau(C^{\beta} - C^{\alpha} - C')$ | 111° | Bond angle | [17] |
| $\tau(C' - N - C^{\alpha})$ | 121° | Bond angle | [17] |
| $d(C_k^{\alpha}, N_{k+1})$ | 2.41 | Cosine law | |
| $d(C_k', C_{k+1}^{\alpha})$ | 2.42 | Cosine law | |
| $d(O_k, C_{k+1}^{\alpha})$ | 2.76 | Cosine law | |
| $d(O_k, N_{k+1})$ | 2.25 | Cosine law | |
| $d(C_k^{\beta}, N_{k+1})$ | 3.27 | Cosine law | |

The table also includes pairwise distances derivable from the known data

The subscript "*k*" refers to the *k*th amino acid on the backbone

**Table 17.3** A summary of the main bond lengths information we used for the side-chain atoms

| Bond | Length (Å) |
|---|---|
| C–C | 1.54 |
| C=C | 1.47 |
| C=O | 1.43 |
| C–O | 2.15 |
| C–N | 2.10 |

Cornell and Cieplak [7]. The values we use are shown in Table 17.3. As there are too many kinds of bonds, we do not show all of them in the table. We also conduct experiments to find some bond lengths which are not found in the literature. The way we did the experiments is as follows: for a particular bond, we calculate several such bond lengths from known conformations of molecules in PDB and then take the average. The way we add the information to DISCO's input data is similar to Eq. (17.17). For the atoms which are mutually bonded, we take the standard deviation $r$ to be 2%; for the atoms which are not mutually bonded, we take $r$ to be 6%.

As mentioned in the Introduction, to automatically derive the chemistry information for a protein molecule given its amino acids sequence, we find it convenient to design a structure array data structure to code the information pertaining to each atom in the molecule. As an example, for the 402-atom protein molecule 1PTQ, we use a 1 × 402 structure array (say p) with fields 'c', 'a', 'aa', and 'am' to store the information pertaining to the molecule. The first two elements of p are shown below:

```
p(1).c=[5.7208,-2.5088,10.2270],

p(1).a='N',p(1).aa='HIS',p(1).am='N'

p(2).c=[5.2388,-1.4878,9.2950],

p(2).a='C',p(2).aa='HIS',p(2).am='CA'
```

Here p(1).c refers to the known coordinates of the first atom; p(1).a refers to the type of atom; p(1).aa refers to the amino acid for which the first atom resides in; p(1).am refers to the atomic label of the first atom with respect to the amino acid it resides in.

### *17.5.2 Van der Waals Radii*

We have also tried to add more lower bounds to our input data based on van der Waals radii. The van der Waals radius of an atom is half the MSD between two atoms (of the same type) which are not chemically related to each other. By carrying out empirical study using proteins from PDB, we found that van der Waals radii provide a good lower bound for the pairwise distance of atoms which are at least three bonds away in the molecule. Note that atomic radii can also be used to generate lower bounds for pairwise distances. But the van der Waals radii give better lower bounds as they are normally twice as large as atomic radii.

The van der Waals radii we used are from a standard inorganic textbook [2], which are given as follows: C (1.70 Å), N (1.55 Å), O (1.52 Å), and S (1.80 Å).

However, after experimenting with additional lower bounds generated from van der Waals radii, we found that the results usually do not improve significantly. In addition, the time taken to solve the conformation problem becomes significantly longer because of the large number of additional lower bounds we have to handle.

The reason for not getting better results after adding the van der Waals radii might be as follows. For the input pairwise distances, though they are not exact, they are estimators of the true pairwise distances. But for van der Waals radii-generated lower bounds, they are generally too weak to give useful information on the pairwise distances. Thus, adding van der Waals radii-generated lower bounds is not really useful. Since it also increases the computational cost by doing so, we have decided not to add van der Waals radii-generated lower bounds into our algorithm.

## 17.6 Numerical Experiments

Here we explain the computational issues in the DISCO algorithm. In Sect. 17.6.1, we present the experimental setup. In Sect. 17.6.2, we discuss the numerical results.

### *17.6.1 Experimental Setup*

The source codes for our DISCO algorithm are written in MATLAB, and the SDPT3 software package of Toh et al. [23, 24, 28] is used to solve the SDP problems arising in the DISCOBASIS step of the algorithm.

We perform the numerical experiments on a dual-processor machine (3.2 GHz Intel Core i5) with 4 GB RAM, running MATLAB version 7.8 using only one processor.

We tested our algorithm using input distance data obtained from a set of seven molecules taken from the PDB. The conformations of these molecules are already known, so we can compare our computed conformations to the true conformations.

For the input distance data, we have two types of distance bounds. The first type of bounds comes from chemistry information pertaining to the molecule, and the second type of bounds is generated randomly to simulate NOE restraints. The sparsity of the simulated NOE distance bounds was modeled by choosing at random a proportion of all the short-range pairwise distances less than the cutoff range of 6 Å, subject to the condition that the distance graph is connected.[3] The cutoff range of 6 Å was selected because NMR techniques are able to give us distance information between some pairs of atoms only if they are less than approximately 6 Å apart. We have adopted this particular input data model because it is simple and fairly realistic [6, 29]. In realistic molecular conformation problems, exact interatomic distances are not given, but only lower and upper bounds on the interatomic distances are known. Thus, after selecting a certain proportion of short-range interatomic distances, we add noise to the distances to give us lower and upper bounds. In this chapter, we have experimented with a "normal" and a "uniform" noise model. The noise level is specified by a parameter $\sigma$, which indicates the expected value of the noise. When we say we have a noise level of 20%, what that means is that $\sigma = 0.2$. In the normal noise model, the bounds are specified by

$$\underline{d}_{ij} = \max\Big(\alpha_{ij}, (1 - \sigma|\underline{z}_{ij}|)d_{ij}\Big), \quad \overline{d}_{ij} = (1 + \sigma|\overline{z}_{ij}|)d_{ij},$$

where $\underline{z}_{ij}, \overline{z}_{ij}$ are independent normal random variables with zero mean and standard deviation $\sqrt{\pi/2}$. Consequently, the expected value of $|\underline{z}_{ij}|, |\overline{z}_{ij}|$ is 1, and the variance is $\pi/2 - 1$. The positive scalar $\alpha_{ij}$ in $\underline{d}_{ij}$ is the MSD between atoms $i$ and $j$, and we will discuss how it is chosen in the next paragraph.

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

**Table 17.4** A conformation problem with sparse and noisy distance data solved in a centralized fashion without divide and- conquer

Input data: 20% distances $\leq 6$ Å

| Molecule | $n$ | 20% normal noise | | 20% uniform noise | |
|---|---|---|---|---|---|
| | | RMSD (Å) | $\ell$ | RMSD (Å) | $\ell$ |
| 1PTQ | 402 | 1.08 | 4 | 0.84 | 4 |

In the table, $\ell$ is the number of atoms with degree less than 4

In addition to the lower and upper bounds, which are available only for some atom pairs, we have MSDs between all pairs of atoms . Due to physical reasons, two atoms $i$ and $j$ must be separated by an MSD $\alpha_{ij}$, which depends on particular details such as the type of atoms (e.g., C–N, N–O) and whether they are covalently bonded. The MSD gives a lower bound for the distance between the two atoms . In our input distance data, for simplicity, we set $\alpha_{ij} = 1$ Å for all covalently bonded atom pairs, regardless of the types of atoms , and $\alpha_{ij} = 2$ Å for all nonbonded pairs. If we wished, we could also set $\alpha_{ij}$ to be the sum of the van der Waals radii (given in Sect. 17.5.2) of the corresponding atom pair, in the case in which the atoms are at least three bonds away in the molecule.

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 Sect. 17.3.3, then the RMSD is defined by the following formula:

$$\mathrm{RMSD} = \frac{1}{\sqrt{n}} \left( \sum_{i=1}^{n} ||x_i - x_i^*||^2 \right)^{1/2} .$$

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

### 17.6.2 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 a small conformation problem using sparse and noisy distances. This information is presented in Table 17.4. Even if we solve the conformation problem in a centralized fashion without divide and conquer, due to the sparsity and noise in the distance data, we can only get an approximate solution.

The performance of our enhanced DISCO algorithm is listed in Tables 17.5 and 17.6. We report the average RMSDs of the conformations obtained for various molecules over ten random instances of input distance data.

**Table 17.5** The average RMSDs of the computed conformations for various molecules corresponding to ten random instances of input distance data generated by the normal noise model

| Input data: 20% distances ≤ 6 Å, corrupted by 20% normal noise | | | | | |
|---|---|---|---|---|---|
| Molecule | $n$ ($l$) | RMSD (Å) | Time (s) | `nnz_chem`/$n$ | `nnz_noe`/$n$ |
| 1PTQ | 402 (5) | 0.86 | 23.3 | 2.6 | 3.0 |
| 1AX8 | 1,003 (2) | 1.48 | 110.9 | 2.6 | 3.2 |
| 1F39 | 1,534 (5) | 1.25 | 182.6 | 2.7 | 3.2 |
| 1RGS | 2,015 (10) | 1.33 | 386.6 | 2.7 | 3.2 |
| 1KDH | 2,923 (14) | 1.35 | 515.8 | 2.6 | 3.4 |
| 1BPM | 3,672 (9) | 0.99 | 764.3 | 2.6 | 3.6 |
| 1MQQ | 5,681 (29) | 0.87 | 1,665.6 | 2.6 | 3.7 |

In the table, $l$ is the average number of atoms with less than 4 neighbors; `nnz_chem` is the number of distance bounds generated based on chemistry information; `nnz_noe` is the number of distance bounds generated randomly to simulate the NOESY distance restraints

**Table 17.6** Same as Table 17.5 but for input distance data generated by the uniform noise model

| Input data: 20% distances ≤ 6 Å, corrupted by 20% uniform noise | | | | | |
|---|---|---|---|---|---|
| Molecule | $n$ ($l$) | RMSD (Å) | Time (s) | `nnz_chem`/$n$ | `nnz_noe`/$n$ |
| 1PTQ | 402 (5) | 0.77 | 21.4 | 2.6 | 3.0 |
| 1AX8 | 1,003 (2) | 1.37 | 106.0 | 2.6 | 3.2 |
| 1F39 | 1,534 (5) | 1.12 | 168.7 | 2.7 | 3.2 |
| 1RGS | 2,015 (10) | 1.22 | 360.5 | 2.7 | 3.2 |
| 1KDH | 2,923 (14) | 1.12 | 473.1 | 2.6 | 3.4 |
| 1BPM | 3,672 (9) | 0.83 | 696.5 | 2.6 | 3.6 |
| 1MQQ | 5,681 (29) | 0.75 | 1,589.1 | 2.6 | 3.7 |

**Table 17.7** Results obtained by the enhanced DISCO algorithm on the "bridge-donut" and "PACM" 3D graph-realization problems considered in [8]

| | Bridge-donut ($n = 500$) | | | PACM ($n = 799$) | | |
|---|---|---|---|---|---|---|
| Noise level | ANE | RMSD | Time (s) | ANE | RMSD | Time (s) |
| 0 | 6.11e−03 | 1.78e−02 | 42.67 | 1.77e−02 | 9.42e−02 | 91.42 |
| 5 | 1.02e−02 | 2.95e−02 | 42.77 | 2.37e−02 | 1.26e−01 | 97.27 |
| 10 | 3.28e−02 | 9.53e−02 | 40.79 | 8.00e−02 | 4.27e−01 | 93.46 |
| 15 | 4.11e−02 | 1.19e−01 | 42.01 | 4.43e−02 | 2.36e−01 | 104.22 |
| 20 | 4.52e−02 | 1.31e−01 | 45.96 | 4.95e−02 | 2.64e−01 | 98.75 |
| 25 | 8.42e−02 | 2.45e−01 | 44.58 | 7.91e−02 | 4.22e−01 | 98.94 |
| 30 | 9.39e−02 | 2.73e−01 | 55.33 | 9.24e−02 | 4.93e−01 | 99.51 |
| 35 | 8.53e−02 | 2.48e−01 | 69.11 | 2.00e−01 | 1.07e−00 | 136.53 |
| 40 | 1.79e−01 | 5.21e−01 | 73.04 | 9.52e−02 | 5.08e−01 | 161.13 |
| 45 | 1.43e−01 | 4.16e−01 | 60.40 | 1.95e−01 | 1.04e−00 | 198.45 |
| 50 | 2.13e−01 | 6.19e−01 | 46.74 | 2.14e−01 | 1.14e−00 | 246.29 |

Finally, we would like to add that the enhanced DISCO algorithm can also improve the performance of DISCO on the 3D anchor-free graph-realization problems considered in [8]. For the "bridge-donut" and "PACM" graphs considered in that paper, we are able to obtain the results shown in Table 17.7, which are

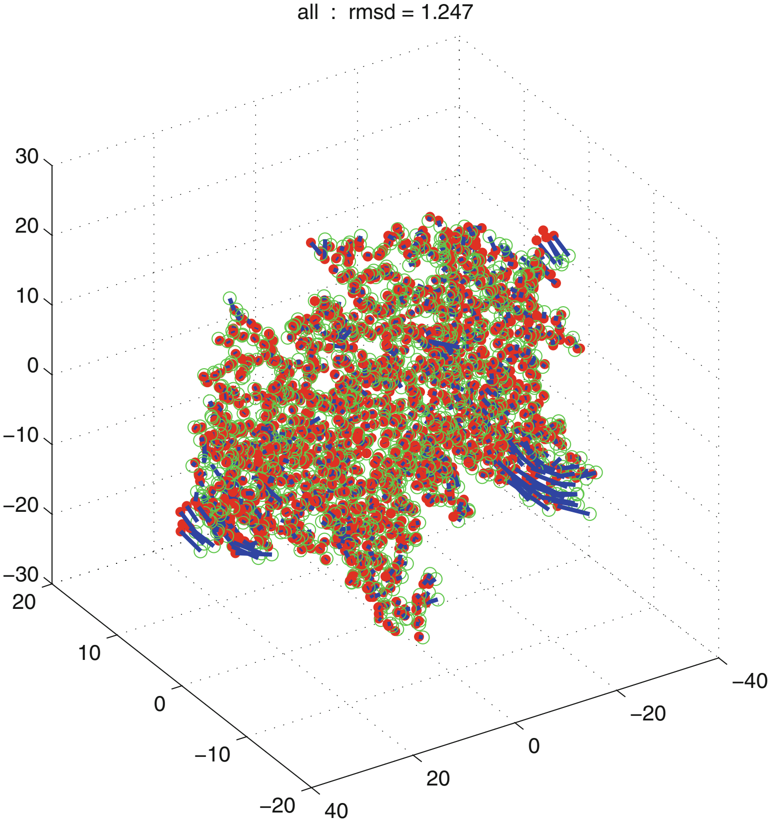


**Fig. 17.3** For each molecule, ten random inputs were generated with different random number seeds. We plot the RMSDs of the ten structures produced by DISCO against the molecule number. (*left*) 20% short-range distances, 20% normal noise; (*right*) 20% short-range distances, 20% uniform noise

comparable or better than the reconstruction results obtained by the 3D-ASAP divide-and-conquer algorithm in [8]. In Table 17.7, "ANE" denotes the average normalized error which is defined by $\sqrt{\sum_{i=1}^n ||x_i - x_i^*||^2}/\sqrt{\sum_{i=1}^n ||x_i^*||^2}$, assuming that the true configuration $\{x_i^* \mid i = 1,\ldots,n\}$ has center of mass at the origin.

The RMSD plots across the molecules, with ten runs given different random distance data, are shown in Fig. 17.3. The plots show that our enhanced DISCO algorithm is able to produce accurate conformations ($< 2\mathring{A}$) for all the molecules over different random inputs. Note that for each molecule, we only generate about 3.0–3.7$n$ simulated distance bounds (to simulate the NOESY distance restraints) to be used in order to construct the conformations. Thus, the number of simulated distance bounds supplied is extremely sparse compared to the total number $(n(n-2)/2)$ of possible pairwise distances.

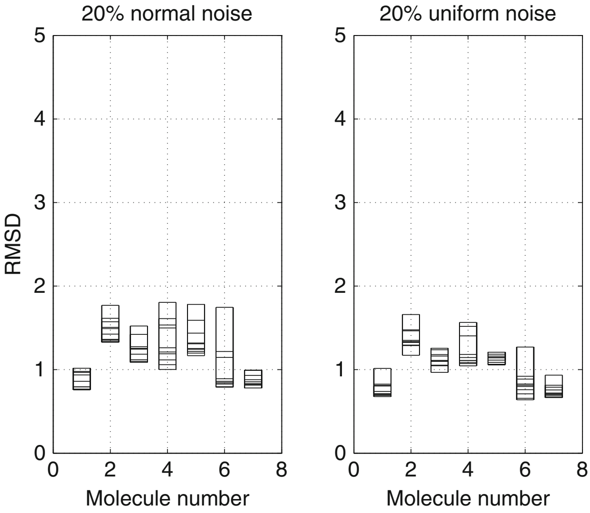


**Fig. 17.4** The conformation of the molecule `1F39` corresponding to the first random input distance data in Table 17.5. In the plot, *green circles* depict the true positions, *red dots* give the computed positions, and *blue line* segments are the error vectors

Before the current enhancements, DISCO did not perform so well, for example, on the molecule `1RGS`, which has a less rigid structure. But now we can see from the plots in Fig. 17.3 that our enhanced DISCO algorithm is able to solve the problems robustly and accurately. When given 20% of the short-range distances, corrupted by 20% of noise, the computed conformations have RMSD between 1.0 and 1.8 Å. We believe the RMSDs we obtained 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 3.75 Å. This is reasonable because the maximum given distance is about 6 Å and the smallest distance is about 1.5 Å. Given 20% noise, we give a bound of about 3.0–4.5 Å for that distance. Thus the true distance is only estimated to within the range of 0.75 Å. Therefore we should expect the ideal RMSD to be about 0.75 Å.

To give the reader an idea of how the computed conformations look like generally, we show in Fig. 17.4 the conformation of the molecule `1F39` corresponding to the input data in Table 17.5. As we may observe from the plot, the atoms in the core region are accurately localized, but those on the peripheral region are less well localized.

## 17.7 Conclusion

We have proposed a novel divide-and-conquer, SDP-based algorithm for the molecular conformation problem. Our numerical experiments demonstrate that the algorithm is able to solve very sparse and highly noisy protein molecular conformation problems with simulated data accurately and efficiently. The largest molecule with more than 5,000 atoms was solved in about 30 min to an RMSD of 1.0 Å, given only 20% of pairwise distances less than 6 Å which are corrupted by 20% multiplicative noise.

In this work, we have only dealt with simulated data. The next step forward would be to adapt our enhanced DISCO algorithm to tackle molecular conformation problems with real MNR experimental data, as was done in [14].

## References

1. Ashida, T., Tsunogae, Y., Tanaka, I., Yamane, T.: Peptide chain structure parameters, bond angles and conformationai angles from the cambridge structural database. Acta Crystallographica **B43**, 212–218 (1987)
2. Atkins, P.: Inorganic Chemistry. Oxford University Press, New York (2006)
3. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: Proceedings of the third international symposium on Information processing in sensor networks, ACM Press, 46–54 (2004)
4. Biswas, P., Ye, Y.: A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization. In: Hager, W.W. (ed.) Multiscale Optimization Methods and Applications, pp. 69–84. Springer, New York (2006)
5. Biswas, P., Liang, T.-C., Toh, K.-C., Wang, T.-C., Ye, Y.: Semidefinite programming approaches for sensor network localization with noisy distance measurements. IEEE Trans. Autom. Sci. Eng. **3**, 360–371 (2006)
6. Biswas, P., Toh, K.-C., Ye, Y.: A distributed SDP approach for large scale noisy anchor-free graph realization with applications to molecular conformation. SIAM J. Sci. Comput. **30**, 1251–1277 (2008)
7. Cornell, W., Cieplak, P.: A second generation force field for the simulation of proteins, nucleic acids and organic molecules. J. Am. Chem. Soc. **117**, 5179–5197 (1995)
8. Cucuringu, M., Singer, A., Cowburn, D.: Eigenvector synchronization, graph rigidity and the molecule problem. arXiv:1111.3304v3 (2012)
9. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data. J. Global Optim. **26**, 321–333 (2003)
10. Engh, R., Huber, R.: Accurate bond and angle parameters for x-ray protein structure refinement. Acta Crystallographica **A47**, 392–400. 1991.
11. Grooms, I.G., Lewis, R.M., Trosset, M.W.: Molecular embedding via a second-order dissimilarity parameterized approach. SIAM J. Sci. Comput. **31**, 2733–2756 (2009)
12. Güler, O., Ye, Y.: Convergence behavior of interior point algorithms. Math. Program. **60**, 215–228 (1993)
13. Havel, T.F.: A evaluation of computational strategies for use in the determination of protein structure from distance constraints obtained by nuclear magnetic resonance. Progr. Biophys. Mol. Biol. **56**, 43–78 (1991)
14. Havel, T.F., Wüthrich, K.: 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. Math. Biol. **46**, 673–698 (1984)
15. Havel, T.F., Kuntz, I.D., Crippen, G.M.: The combinatorial distance geometry approach to the calculation of molecular conformation. J. Theor. Biol. **104**, 359–381 (1983)
16. Hendrickson, B.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
17. Laskowski, R., Moss, D.: Main-chain bond lengths and bond angles in protein structures. J. Mol. Biol. **231**, 1049–1067 (1993)

18. Leung, N.-H., Toh, K.-C.: An sdp-based divide-and-conquer algorithm for large scale noisy anchor-free graph realization. SIAM J. Sci. Comput. **31**, 4351–4372 (2009)
19. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
20. McMurry, J.: Organic Chemistry. Thompson, Belmont (2008)
21. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. J. Global Optim. **15**, 219–234 (1999)
22. So, A.M.-C., Ye, Y.: Theory of semidefinite programming for sensor network localization. In: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 405–414 (2005)
23. Toh, K.-C., Todd, M.J., Tutuncu, R.H.: The SDPT3 web page: http://www.math.nus.edu.sg/~mattohkc/sdpt3.html.
24. Toh, K.C., Todd, M.J., Tutuncu, R.H.: SDPT3—a MATLAB software package for semidefinite programming. Optim. Meth. Software **11**, 545–581 (1999)
25. Trosset, M.W.: Applications of multidimensional scaling to molecular conformation. Comput. Sci. Stat. **29**, 148–152 (1998)
26. Trosset, M.W.: Distance matrix completion by numerical optimization. Comput. Optim. Appl. **17**, 11–22 (2000)
27. Trosset, M.W.: Extensions of classical multidimensional scaling via variable reduction. Comput. Stat. **17**, 147–163 (2002)
28. Tutuncu, R.H., Toh, K.-C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. Math. Program. Ser. B **95**, 189–217 (2003)
29. Williams, G.A., Dugan, J.M., Altman, R.B.: Constrained global optimization for estimating molecular structure from atomic distances. J. Comput. Biol. **8**, 523–547 (2001)
30. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. SIAM J. Sci. Comput. **26**, 313–338 (2004)
31. Zhang, Z., Zha, H.: A domain decomposition method for fast manifold learning. In: Proceedings of Advances in Neural Information Processing Systems, 18 (2006)

# Chapter 18
# An Overview on Protein Structure Determination by NMR: Historical and Future Perspectives of the use of Distance Geometry Methods

**Fabio C.L. Almeida, Adolfo H. Moraes, and Francisco Gomes-Neto**

**Abstract** Determination of the protein high-resolution structures is essential for the understanding of complex biological mechanisms, for the development of biotechnological methods, and for other applications such as drug discovery. Protein structures solved by nuclear magnetic resonance (NMR) rely on a set of semiquantitative short-range distances and angles information. The exploration of the whole conformational space imposed by the experimental restraints is not a computationally simple problem. The lack of precise distances and angles does not allow to find solutions to this problem by fast geometric algorithms. The main idea is to define an atomic model for the protein structure and to exploit all known geometric angle and distance information along with the semi-quantitative short-range experimental information from NMR. We give an overview of the development of computational methods aimed at solving the problem either by metric matrix distance geometry or using other methods such as simulated annealing. We also discuss future demands and perspectives for structural calculations using NMR data. The need of determining larger and more complex protein structures implies the strong necessity of developing new methods for structural calculation with sparse data.

> *We are like dwarfs sitting on the shoulders of giants. We see more, and things that are more distant, than they did, not because our sight is superior or because we are taller than they, but because they raise us up, and by their great stature add to ours . . .* This sentence was written (in Latin) in the logic treatise Metalogicon by John of Salisbury in 1159. Salisbury attributed this sentence to Bernard of Chartres. It was reused later by Isaac Newton to explain the development of western science.

F.C.L. Almeida (✉) • A.H. Moraes • F. Gomes-Neto
National Center of Nuclear Magnetic Resonance, Institute of Medical Biochemistry,
Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil
e-mail: falmeida@cnrmn.bioqmed.ufrj.br; amoraes@cnrmn.bioqmed.ufrj.br;
fgomes@cnrmn.bioqmed.ufrj.br

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0_18,

## 18.1 Introduction

The goal of this chapter is to give an overview of protein structure determination by nuclear magnetic resonance(NMR). We will give a historical perspective that illustrates the necessity of solving distance geometry problems in order to determine protein structure. Briefly, the problem consists of exploiting experimental information that is obtained from NMR experiments and that mainly concerns distances between hydrogen atoms, in order to find the three-dimensional structure of a protein. Together with the NMR information, we can also use additional information deduced from the knowledge accumulated during the twentieth century on molecular structures. For this reason, the sentence by John of Salisbury that we quoted perfectly applies to protein structure determination.

This chapter is organized as follows. In Sect. 18.2, we briefly introduce protein structures. In Sect. 18.3, we give a description of the conformational space of protein structures, while we discuss about molecular dynamics in Sect. 18.4. In Sect. 18.5 we briefly describe NMR experiments, and Sect. 18.6 is devoted to the problem of deriving some atomic distance restraints from NMR data. Section 18.7 is devoted to some pseudo-potentials that can be used for modeling the distance restraints, while the distance geometry problem with NMR data is discussed in Sect. 18.8, where the first implemented computational method for protein structural calculation from NMR data is presented. Nowadays, the most used method for solving distance geometry problems with NMR data is the meta-heuristic simulated annealing (SA): we present two variants of this algorithm in Sect. 18.9, one that is based on the Cartesian representation of the protein structures and the other one that is based on the torsion angle representation. We conclude our chapter in Sect. 18.10, where we discuss some future demands for protein structure determination.

## 18.2 Introduction to Protein Structure

The determination of protein high resolution structures is essential for the understanding of complex biological mechanisms, for the development of biotechnological methods, drug design, and many other applications. Requesting the protein structure to have a high resolution implies that the position of each of its atom is identified precisely (uncertainty smaller than 1 Å).

Proteins are polymeric chains in which the units are the 20 natural L-$\alpha$-aminoacids that are connected by peptide bonds. Several structures of dipeptides, which have been solved by X-ray crystallography in the early 1930s by the group led by Linus Pauling, demonstrated that the peptide bond can have two configurations: cis and trans [34, 57]. The trans configuration has lower energy and it represents the most abundant configuration in proteins. There are however exceptions, such as cis-prolines which are important for thioredoxin activity [13, 29, 30]. The peptide bond

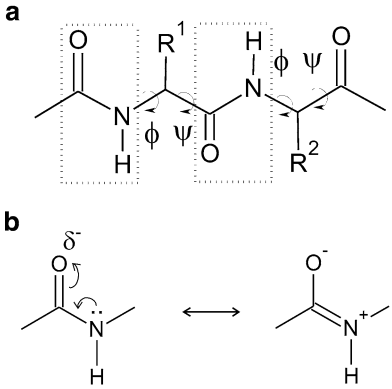


**Fig. 18.1** Illustration of a peptide bond planar structure: (**a**) the planar peptide bond (*dotted rectangle*); the dihedral angles that give torsion freedom for the peptide backbone ($\Phi$ and $\Psi$ angles) are indicated by *arrows*; the side chains are represented by $R^1$ and $R^2$; (**b**) resonance forms of the peptide bond and its double-bond character

is planar because of the resonance effect that gives to it a double-bond character. Figure 18.1 illustrates a polypeptide chain and the planar character of the peptide bond.

Several other geometrical properties of proteins were defined before the first protein structure was solved by Kauzmann in 1964 [34]. Maybe the most important property is given by the presence of secondary structure elements, such as $\alpha$-helices, that was firstly proposed by Linus Pauling [56–59].

The amino acid sequence is also called primary structure. An amino acid included in a polypeptide chain is called amino acid residue. Secondary structures represent local structural organizations that are stabilized by hydrogen bonds in the main chain. They can be observed in several proteins. The main polypeptide chain, also called protein backbone, is the protein sequence without the radicals of each L-$\alpha$-amino acids, i.e., without side chains. The backbone chains contain only one hydrogen donor to a hydrogen bond, the amidic hydrogen (N–H), and only one electron pair, which serves as hydrogen acceptor in a hydrogen bond, the free electron pair of the carbonyl (CO). Recall that electron pairs of amide nitrogen on the main chain is "busy" because it is part of the double bond related to one of the resonance forms of the peptide bond (see Fig. 18.1). This means that, in proteins, the only hydrogen bonds stabilizing secondary structures are the ones between the amidic N–H and the carbonyl.

The protein backbone needs to bend in order to stabilize secondary structures, because this is needed for forming hydrogen bonds between amino acids. There are two degrees of freedom that leads to the bending of the main chain. These degrees of freedom are defined by the dihedral angles $\Phi$ and $\Psi$. The dihedral angle $\Phi$ among the atoms $C_\alpha^{i-1}$, N, $C_\alpha$, and $C'$ defines the torsion of the bond N–$C_\alpha$. The dihedral angle $\Psi$ among N, $C_\alpha$, $C'$, and $N^{(i+1)}$ defines the torsion of the bond $C_\alpha$–$C'$ (where $C'$ is the carbonyl carbon). It is important to note that the two-dimensional plot of $\Phi$ versus $\Psi$, known as Ramachandran plot, can describe the folding of a protein in the sense that particular pairs ($\Phi$,$\Psi$) can be identified for each amino acid residue forming the protein. It is also remarkable that the Ramachandran plot defines all the conformational space for the backbone structure of a protein [61]. Note that the

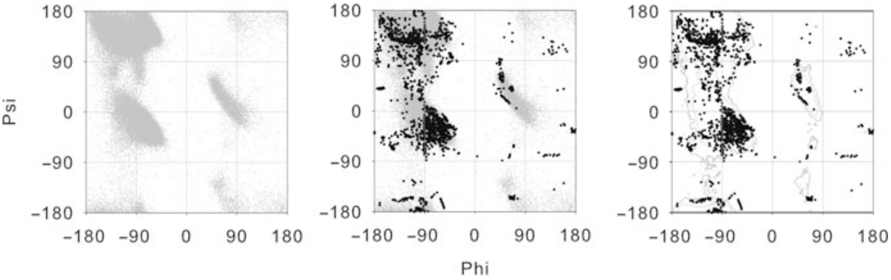


**Fig. 18.2** Ramachandran plot. *Leftmost plot*: $\varPhi$ and $\varPsi$ angles for 500 high-resolution crystal structures selected from PDB: the plot shows a general dihedral freedom adopted in proteins [48]. *Plot in the center*: superposition of the angles $\varPhi$ and $\varPsi$ extracted from the 20 lower-energy structures of thioredoxin 1 (PDB id:2I9H) solved by solution NMR [60]. *Rightmost plot*: contour plot showing allowed and generously allowed regions for the angles $\varPhi$ and $\varPsi$ calculated from the initial data set and the superposition of dihedral angles from 2I9H structures

torsion of the peptide bond is not considered as a degree of freedom because of its planarity.

In the 1960s, Ramachandran performed some computational calculations on small peptides and showed that not all combinations of $\varPhi$ and $\varPsi$ are possible in proteins. Moreover, there are high-energetical conformations that can be considered as forbidden [61]. On the other hand, $\varPhi$ and $\varPsi$ combinations that can be observed in secondary structure define the lowest-energy conformations. High energetical states are due to steric effects between large side chains. We can say that, for a given amino acid residue, the larger is the side chain, the smaller are the possible low-energy areas in the Ramachandran plot. Figure 18.2 shows the Ramachandran plot of yeast thioredoxin 1 (PDB id:2I9H, [2]) and the location of the main secondary structures.

The two most frequent secondary structure elements are $\alpha$-helices and $\beta$-sheets (parallel and antiparallel). It is not in the scope of this chapter to describe all the secondary structure elements but to contextualize in regard to the protein structural determination problem. The two secondary structure elements define the lowest-energy regions of the Ramachandran plot, in which the $\alpha$-helix region is near $\varPhi = -60^\circ$ and $\varPsi = -30^\circ$, and the $\beta$-sheet region is near $\varPhi = -120^\circ$ and $\varPsi = 135^\circ$ (see Fig. 18.2).

## 18.3 The Problem of Conformational Space

As we have seen in the previous section, the Ramachandran plot is related to the backbone conformation of a protein. If one knows the dihedral angles $\varPhi$ and $\varPsi$ for all residues, not only the secondary structure is determined, but also the tertiary structure can be derived from this information. The tertiary structure is given by all three-dimensional coordinates of the atoms forming the protein. The quaternary structure defines the structural organization of proteins in oligomers. The oligomerization of a protein can be homo-oligomerization, where the association involves the same amino acid chain, or hetero-oligomerization, where the association occurs with different chains. There are several levels of oligomerization: dimers, decamers/dodecamers, and virus structures, which may contain thousands of chains.

It is important to discuss the forces that stabilize the tertiary and the quaternary structures of proteins. They are mainly represented by intermolecular non-covalent interactions between atoms belonging to the protein backbone or to the side chains of the amino acids. These interactions are generally called *tertiary contacts*. We give, in the following, some details about the main interaction forces in proteins.

### *18.3.1 Hydrogen Bonds*

Besides the hydrogen bonding between the amidic group N–H and the carbonyl group (CO) of the backbone that stabilizes the secondary structures, there are many others amino acid side chains that can form hydrogen bonds. Amino acid residues serine, threonine, and tyrosine contain a hydroxyl group (–OH) that can be either donor or acceptor of a hydrogen in a hydrogen bond. Moreover, aspartate and glutamate are carboxylic acids that contain a hydroxyl and a carbonyl (donor and acceptor). Asparagine and glutamine contain amide (–NH2, mainly donor) and a carbonyl (acceptor). Finally, lysine (amine) and arginine (guanidinium group) are also good donors and acceptors for hydrogen bond.

In proteins, there is always competition between intramolecular and intermolecular (the protein solvent is water) hydrogen bonding. The more is the residue exposed to water (near the protein surface), the smaller is the contribution of the intramolecular hydrogen bond to the stabilization of tertiary and quaternary structures.

### *18.3.2 Coulomb Interactions*

Several side chains can ionize in water, associating or dissociating protons that become charged. At neutral pH, aspartate, glutamate, and the carboxy terminus are negatively charged, whereas lysine, arginine, histidine, and the amino-terminus are positively charged. The proximity of two opposite charges leads to Coulombic interactions that, when present, strongly contribute to the stabilization of tertiary and quaternary structures.

Charged residues are solvated by water. The dipole of water neutralizes the charge. Coulombic interaction, also known as salt bridge, is restricted to protein microenvironments where the water access is limited. Similar to the dependence of the strength of hydrogen bonds on water access, the more exposed water is to the charged residue, the weaker is the intramolecular Coulombic interaction.

### *18.3.3 Van der Waals*

van der Waals (vdW) interactions are dipolar–dipolar interactions that occur at very short distances ($r < 5$ Å). Although they are the weakest forces involved in the protein structure stabilization, they are the most important for the tertiary and quaternary structures of proteins because of their high abundance. Every dipole that is close to each other contributes to the protein stabilization.

Water contributes favorably for VdW because the apolar hydrophobic side chains tend to avoid the exposure to the solvent, the so-called hydrophobic effect. In this way, they become part of a hydrophobic core. The exposure of hydrophobic side chains to the bulk water leads to high entropic penalty. The VdW force has two components, one is repulsive, at very short distances ($r < 1.8$ Å), which decays proportionally to $r^{-12}$ and the other one is attractive ($1.8 < r < 5$ Å), which decays proportionally to $r^{-6}$.

VdW is the "glue" that sticks together the protein structure. Hydrophobic residues are packed in the protein and kept by VdW interactions.

The water contribution is also very important. The exposure of each residue to water determines what kind of interaction is more important for the structure stabilization. Polar residues tend to be found on the surface of proteins and this is the reason why the polar interaction contribution, such as hydrogen bonds and Coulomb interaction, needs to be pondered by the water access.

Water access is also essential for protein dynamics. Polar side chains on the surface of globular proteins have structures that fluctuates among several conformational states. On the other hand, polar side chains, which are packed in the protein core, strongly contributes to the stabilization of the protein structure and are subject to restricted motions and well-defined configurations. The limited access of water increases the interaction energy of intermolecular hydrogen bonds and salt bridges. Apolar side chains in the protein core are packed with restricted motions due to the VdW interactions.

Apolar side chains on the surface of a protein are exposed to water. Any exposure of apolar surface to water leads to entropic penalties due to the super-organization of the water molecules. In order to avoid the entropic penalty, the protein tends to find an alternate organization where the apolar surface is hidden from water. Proteins that contain hydrophobic patches are less soluble in water and/or tend to oligomerize or interact with other proteins.

## 18.4 Protein Geometry and Introduction to Molecular Dynamics Simulation

It is not our goal to describe all the details of protein geometry and molecular dynamics simulation, but rather emphasize some of its aspects, that are important in the context of protein structural calculation.

Nowadays, we have the possibility of considering the knowledge accumulated over the last century on molecular structures, and particularly the knowledge about protein structure. Force fields are generally based on simplified versions of the classical mechanical equations that can be defined for each geometry element in the molecule and by each interaction force. The creation, and the continuous improvement, of the force fields enables the simulation of the protein geometry, of the intra- and inter-molecular interactions, and of the protein dynamics.

Simulations of molecular dynamics can be performed by solving Newton's equation in discrete time steps (known as integration time). The time step must be small enough to not overcome any polypeptide dynamic event, such as vibrations. Typically, the time step is smaller than 5 femtoseconds (fs, that is $5 \times 10^{-15}$s). The mass of each atom, the equilibrium distances, and angles are parameterized in the available force fields [9, 32, 65].

To compute the trajectory at each integration time ($dt$), the motion equations are obtained using Newton's second law ($\mathbf{F} = m\mathbf{a}$). The resulting external forces can be written as the gradient of the potential energy:

$$\mathbf{F} = -\nabla V. \tag{18.1}$$

The gradient ($\nabla$) is a vector operator that, when applied on a function, such as $V(x,y,z)$, results in a vector $\mathbf{F}$:

$$\nabla V(x,y,z) = \frac{\partial V}{\partial x}\mathbf{e_x} + \frac{\partial V}{\partial y}\mathbf{e_y} + \frac{\partial V}{\partial z}\mathbf{e_z}.$$

The combination of the equations above results in a differential equation that is integrated at each time step in order to obtain the trajectory of motion:

$$\nabla V = -m\frac{\mathrm{d}^2\mathbf{r}}{\mathrm{d}t^2}. \tag{18.2}$$

Note that the vector force $\mathbf{F}$ is obtained for a potential field $V(x,y,z)$. This is the reason why the set of parameters is also called "force field". The protein structure geometry is defined in the force field by the bond lengths, the bond angles, and by the proper and improper dihedral angles. The nonbonded intramolecular interaction is defined by the nonbonded potential, which mainly considers Coulomb and VdW interactions:

$$V_{\text{total}} = V_{\text{bonds}} + V_{\text{angles}} + V_{\text{dihedrals}} + V_{\text{impropers}} + V_{\text{nonbonded}}.$$

The intramolecular interactions with the solvent are also defined by nonbonded terms. The bond and angle potentials are harmonic potentials that model the vibration motion according to Hooke's law:

$$V_{\text{bonds}} = \sum_{\text{bonds}} K_{\text{b}}(r - r_0)^2,$$

$$V_{\text{angles}} = \sum_{\text{angles}} K_{\theta}(\theta - \theta_0)^2,$$

where $K_{\text{b}}$ and $K_{\theta}$ are spring constants for bonds and angles, respectively. $r$ is the generic bond length, while $r_0$ is bond length at equilibrium. Similarly, $\theta$ is the generic bond angle, whereas $\theta_0$ is the bond angle at equilibrium.

A proper dihedral defines torsion angles which are formed by four atoms joined contiguously through bonds. It defines the geometry of real dihedrals of the protein. Improper dihedrals define the planarity of aromatic rings and peptide bonds, and they avoid stereo centers to interconvert. They also express torsion angles formed by atoms that are not necessarily connected through bonds. Proper dihedrals are usually expressed as periodic potentials:

$$V_{\text{dihedrals}} = \sum_{\text{dihedrals}} K_{\omega}\left[1 + \cos(n\omega - \gamma)\right],$$

$$V_{\text{impropers}} = \sum_{\text{impropers}} \frac{1}{2} K_{\xi}\left[\xi_{ijkl} - \xi_0\right].$$

$K_{\omega}$ and $K_{\xi}$ are force constants. $\omega$ is the proper dihedral angle and $\gamma$ is a phase of the periodic potential. $\xi_{ijkl}$ is the generic improper dihedral angle and $\xi_0$ is the improper dihedral at equilibrium.

The nonbonded potentials are defined as following (the first term represents the Coloumb forces, while the second one represents the VdW forces):

$$V_{\text{nonbonded}} = \sum_{i,j pairs} \frac{q_i q_j}{\varepsilon r_{ij}} + \sum_{i,j pairs} \left( \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right),$$

where $q_i$ is the charge of the atom, $\varepsilon$ is the electrical permittivity constant, $r_{ij}$ is the distance between the two atoms $i$ and $j$, and $A_{ij}$ and $B_{ij}$ are two constants related to the Lennard-Jones potential, modeling the VdW forces. We remark that other potentials, modeling, for example, the hydrogen bonds, can also be defined in force fields.

Tables 18.1 and 18.2 show, as an example, the force field and the topology implemented in the XPLOR-NIH and CNS.

Note that the topology of each amino acid (we consider the serine in the tables) is defined by the atomic weight, by the charge, and by the covalent connection of each atom. The force field is defined by the parameters (bond, angle, proper and improper dihedrals, and nonbonded interaction) that enables the calculation of all the potentials listed above.

**Table 18.1** Selected parts of the topology table used by XPLOR-NIH and CNS. We consider the topology of the serine and report the atom type, charge, bonds description and atoms involved in proper and improper dihedral angle definitions. Note that the improper torsion angles define chirality and stereoisomery of the aminoacid.

| atoms | | | bonds | |
|---|---|---|---|---|
| atom | type | charge | atom1 | atom2 |
| N | NH1 | −0.36 | N | HN |
| HN | H | 0.26 | N | CA |
| CA | CH1E | 0.00 | CA | HA |
| HA | HA | 0.10 | CA | CB |
| CB | CH2E | 0.08 | CB | HB1 |
| HB1 | HA | 0.10 | CB | HB2 |
| HB2 | HA | 0.10 | CB | OG |
| OG | OH1 | −0.68 | OG | HG |
| HG | H | 0.40 | O | C |
| C | C | 0.48 | C | CA |
| O | O | −0.48 | | |

| angles | | | | | | |
|---|---|---|---|---|---|---|
| *improper* | HA | N | C | CB | *chirality* | CA |
| *improper* | HB1 | HB2 | CA | OG | *stereo* | CB |
| *dihedral* | OG | CB | CA | N | – | – |

**Table 18.2** Selected parts of the PARALLHDG force field (parallhdg5.1.param) [45]

```
BOND C CH1E 1000.000 sd = 0.001 1.525
BOND C CH2E 1000.000 sd = 0.001 1.516
BOND C CH2G 1000.000 sd = 0.001 1.516
...
ANGLe C CH1E CH1E 500.00 sd = 0.031 109.0754
ANGLe C CH1E CH2E 500.00 sd = 0.031 110.1094
ANGLe C CH1E CH3E 500.00 sd = 0.031 110.4838
...
IMPRoper C CH1E HA HA 500.00 sd = 0.031 0 -70.4072
IMPRoper C CH1E N CH1E 500.00 sd = 0.031 0 -179.9829
IMPRoper C CH1E NH1 CH1E 500.00 sd = 0.031 0 180.0000
...
DIHEdral C CH2E CH2E CH1E 5.00 sd = 0.031 3 0.0000
DIHEdral CH1E CH1E CH2E CH3E 5.00 sd = 0.031 3 0.0000
DIHEdral CH1E CH2E CH2E CH2E 5.00 sd = 0.031 3 0.0000
...
NONBonded HA 0.0498 1.4254 0.0450 2.6157 !- charged group.
NONBonded HC 0.0498 1.0691 0.0498 1.0691 ! Reduced vdw radius
NONBonded C 0.1200 3.7418 0.1000 3.3854 ! carbonyl carbon
```

We report the CNS force field description for bonds, angles, and dihedrals, where the atom type, the type of potential, and the spring constant are given [8, 18]

In the next section, we briefly describe conceptual aspects of NMR that help in understanding how to use and convert NMR experimental data in distance restraints. NMR and molecular dynamics simulation, along with other computational methods, can be considered as good partners, in the sense they are complimentary. NMR experiments provide essential structural and dynamical information for parameterization and improvements of the computational methods, while the computational methods provide a unique way to interpret the experimental data.

## 18.5 Introduction to Nuclear Magnetic Resonance

NMR is a spectroscopy that deals with the nuclear spin and its interaction with magnetic field. Several nuclei are magnetically active, in the sense that they have an associated magnetic moment. Among the magnetically active nuclei, $^{1}$H, $^{13}$C, and $^{15}$N are the most important probes for protein NMR (see Table 18.3). A small protein containing about 100 amino acids approximately contains 2,000 hydrogens, 500 carbons, and 130 nitrogens. Each of these nuclei can be unambiguously assigned, providing precious information. The main physical properties obtained from NMR experiments are chemical shift, scalar coupling, and dipolar interaction (from dipolar coupling).

In practice, proteins prepared for structural determination are enriched with the nuclei presented in Table 18.3. To this purpose, the protein is biosynthesized by a bacterium (among other cells) and grown in an isotope-labeled medium [20].

The magnetism is a consequence of the spin angular momentum. Nuclear magnetism is caused by the nuclear spin. Magnetic active nuclei has a magnetic moment $\mu$, which is associated to the nucleus that is described by the nuclear spin angular momentum **I**. They are collinear and proportional to each other:

$$\mu = \gamma \hbar \mathbf{I}.$$

The proportionality constant is the magnetogyric ratio $\gamma$ multiplied by the Planck constant $\hbar = h/2\pi$. See Table 18.3.

The nuclear spin angular momentum, the vector **I**, has the following magnitude:

$$|\mathbf{I}^2| = \mathbf{I} \cdot \mathbf{I} = \hbar^2 \left[ I(I+1) \right],$$

where $I$ is the spin angular momentum quantum number.

The spin is a quantum entity without classical analog. Nevertheless, it is useful to use a semiclassical representation based on classical angular momentum to build up a geometric representation of the spin (see Fig. 18.3).

Only one component of the angular momentum **I**, $I_x$, $I_y$, or $I_z$, can be determined simultaneously with its magnitude $|\mathbf{I}^2|$. By convention, the value of the z component $I_z$ is specified by the equation

$$I_z = \hbar m,$$

**Table 18.3** Physical properties of some magnetically active nuclei commonly used in protein NMR

| Nucleus | Nuclear spin quantum number ($I$) | Magnetogyric ratio | Natural abundance |
|---|---|---|---|
| $^{1}$H | 1/2 | 267.513 | 100% |
| $^{13}$C | 1/2 | 67.262 | 1% |
| $^{15}$N | 1/2 | 27.116 | 0.377% |
| $^{31}$P | 1/2 | 108.291 | 100% |

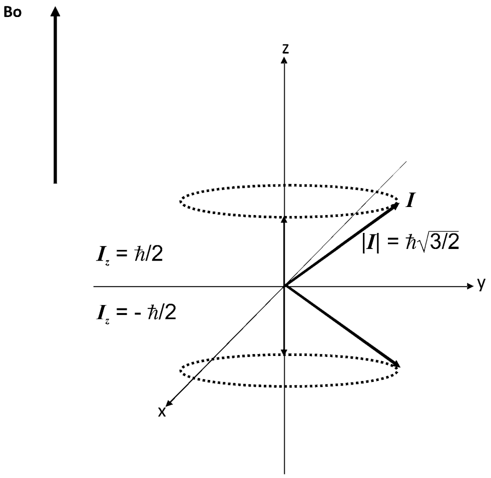


**Fig. 18.3** A schematic representation of the angular momentum of nuclei with nuclear spin angular momentum $I = 1/2$. The vector **I**, in *black*, shows the two quantum states, while the vectors in *grey* represent its projection on the $z$ axis. The projection on $z$ can be determined when there is uncertainty in the projection on the $xy$ plane. The uncertainty is represented by the *dotted grey line*. It implies that **I** can be projected in any position of the $xy$ plane

where $m$ is the magnetic quantum number that can have the following values:

$$m \in \{-I, -I+1, -I+2, \ldots, I-2, I-1, I\}.$$

For a nucleus with $I = 1/2$, **I** adopts two orientation. There is certainty in the projection $I_z$ and uncertainty in $I_x$ and $I_y$. $I_z$ can be either in $+z$ ($m = 1/2$) or in $-z$ ($m = -1/2$). The magnitude of **I** and $I_z$ are

$$|\mathbf{I}| = \frac{\hbar\sqrt{3}}{2}, \quad I_z = \frac{\hbar}{2}, \quad I_z = -\frac{\hbar}{2}.$$

The energy of the interaction of the magnetic moment ($\mu = \gamma\mathbf{I}$) in the presence of an external static magnetic field (**B**) is proportional to the scalar product of $\mu$ and **B**:

$$E = -\mu \cdot \mathbf{B}.$$

Both are vectorial quantities and the energy is dependent on the relative orientation of these two vectors. Figure 18.4 shows the energy diagram for a spin $I = 1/2$.

When field **B** is applied along the $z$ direction, the energy becomes

$$E = -\gamma Bo I_z = -m\gamma\hbar Bo,$$

where $Bo$ is the magnitude of the magnetic field **B** along $z$ direction. So, for a spin $I = 1/2$:

- The quantum state $m = 1/2$, which is parallel with $Bo$, is the minimum energy state ($|\alpha >$ state) with $E = -\gamma\hbar Bo/2$.
- The quantum state $m = -1/2$, which is antiparallel with $Bo$, is the maximum energy state ($|\beta >$ state) with $E = \gamma\hbar Bo/2$.

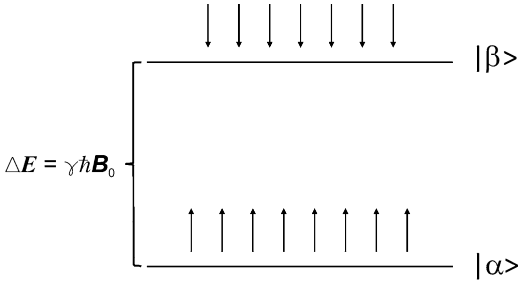


**Fig. 18.4** Diagram representing the energy levels of a nuclear spin $I = 1/2$. Note that, at equilibrium, the high-energy level is less populated than the low-energy one. The *arrows* represent the projections on $z$ (*up* and *down*). The *up arrows* indicate spins at the lower-energy state (the $z$-projection is parallel to the main magnetic field) while the *down arrows* are antiparallel to the static magnetic field (high-energy state)

The difference in energy is

$$\Delta E = \hbar\gamma Bo.$$

Note that the energy difference is proportional to $Bo$. The energy states are degenerate ($\Delta E = 0$) in absence of the magnetic field.

We have so far discussed about isolated spins only. For an ensemble of spins, we need to consider the vectorial sum of the magnetic moment for each spin in the ensemble. In an ensemble, the $x$ and $y$ components of the magnetic moment are canceled. At thermal equilibrium, the lowest energy state is more important: following a Boltzmann distribution as $\Delta E > 0$ in presence of a static magnetic field. This gives rise to a macroscopic magnetic component along the $z$ axis that is the result of the sum over all spins of the ensemble. This is called magnetization vector **M** (see Fig. 18.5). Note that **M** is zero in absence of an external magnetic field and gets polarized ($|\mathbf{M}| > 0$) in presence of the magnetic field.

The NMR experiment consists of applying a radiofrequency pulse with one quantum of energy ($\Delta E = \hbar\omega = \hbar\gamma Bo$) and consequently of changing the population balance of the energy states. The magnetic component of the radiofrequency pulse $\mathbf{B_1}$ is applied on the $xy$ plane. Figure 18.5 illustrates the magnetic component of the pulse causing the nutation of **M** at the rotating frame. Nutation consists of the evolution of **M** around $\mathbf{B_1}$.

The energy of the radiofrequency pulse is

$$E_{\mathrm{rf}} = \hbar\omega_0.$$

The resonance condition is

$$E_{\mathrm{rf}} = \Delta E \Rightarrow \hbar\omega_0 = \hbar\gamma Bo \Rightarrow \omega_0 = \gamma Bo,$$

where $\omega_0$ is the Larmor frequency.

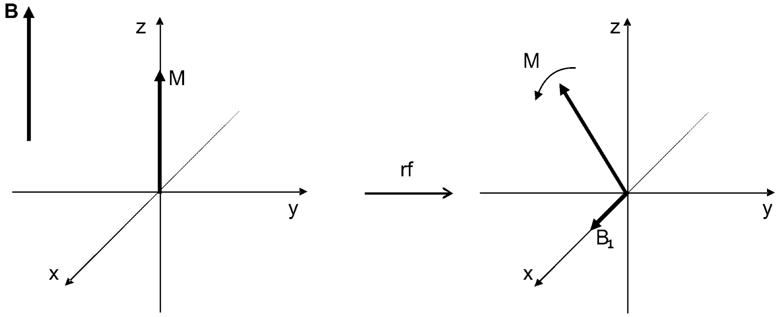


**Fig. 18.5** Effect of a radiofrequency pulse represented by its magnetic component $\mathbf{B_1}$ on the magnetization vector **M**. The figure illustrates the nutation of the **M** around $\mathbf{B_1}$ at the rotating frame. Since the pulse is applied in x, the nutation occurs in the zy plane. At the laboratory frame $\mathbf{B_1}$ rotates in the xy plane at the frequency of the applied pulse. The rotating frame is a frame of reference that rotates around the $z$ axis at the same frequency of the applied rf pulse ($\omega_0$). At the rotating frame $\mathbf{B_1}$ is static

The nutation angle of the magnetization is controlled by the rf irradiation time. The spectroscopist calibrates the time necessary for nutating the magnetization at 90° ($M_z = 0, M_{xy} = 1$) or at 180° ($M_z = -1, M_{xy} = 0$), or at any other nutation angle. The calibrated pulse width is then used to set up the pulse sequences necessary for data collection for structure determination.

After excitation with the rf pulse, the transmitter is turned off. The magnetization is free to evolve back to equilibrium, precessing at the Larmor frequency around *Bo*. The frequency of evolution is detected by the receiver, transformed from time to frequency domain by a Fourier transform, which generates the NMR spectrum. Each spin in the ensemble displays in the spectrum. The NMR spectrum contains information of each spin present in the sample (see Fig. 18.6).

The differences in the electronic density in different molecules or parts of those structures cause the magnetic field to vary on a submolecular distance scale. This effect is called chemical shift and is extremely important for the application of NMR spectroscopy to study the molecules. In order to understand this effect, it is important to know how the electronic density of a molecule responds to the application of a static field **B**.

As showed in Fig. 18.7, the mechanism that leads to chemical shift can be simplified in a two-step process:

1. The external magnetic field induces currents in the electron clouds of the molecule.
2. These generated currents induce a magnetic field which can be added vectorially to the static field $\mathbf{B}^{\text{ind}}$:

$$\mathbf{B}^{\text{loc}} = \mathbf{B} + \mathbf{B}^{\text{ind}}.$$

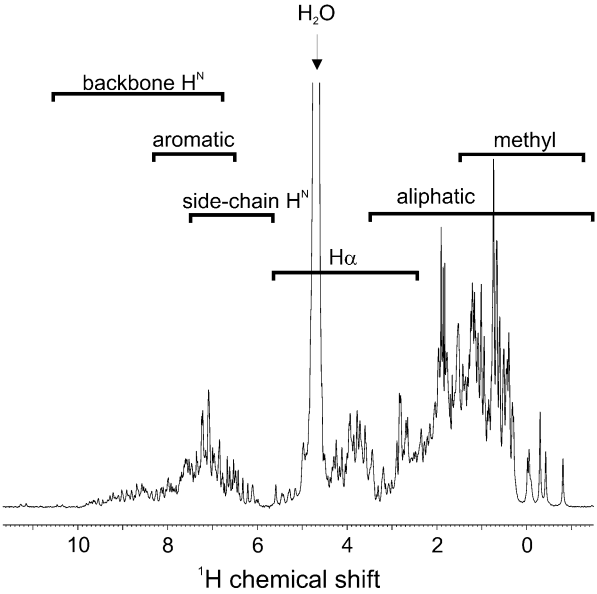
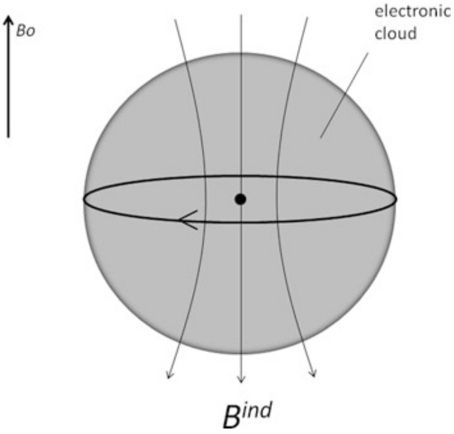


**Fig. 18.6** Typical NMR spectrum of a protein. Ranges of chemical shifts expected for the various types of $^1H$ resonances

**Fig. 18.7** A schematic representation of an atom, which illustrates the nucleus and the effect of the rotation of the electrons inducing a magnetic field $B^{\mathrm{ind}}$ which is antiparallel to the static magnetic field



Some important information about $\mathbf{B}^{\mathrm{ind}}$ follows. First, the induced field is approximately linearly dependent on the applied field. Second, the magnitude and direction of some induced magnetic field is dependent on the shape of the molecule and on the location of the nuclear spin in the protein. Assuming these facts, we can write the induced magnetic field as follows:

$$\mathbf{B}^{\mathrm{ind}} = -\boldsymbol{\sigma} \cdot \mathbf{B},$$

where $\boldsymbol{\sigma}$ is called shielding tensor, represented by a 3 × 3 square matrix. Note that $\boldsymbol{\sigma}$ is not a vector.

## 18.6 Experimental Restraints Generated by NMR

The main information for protein structural calculation is the nuclear Overhauser effect (NOE). NOE was first observed by Albert Overhauser in 1953 [54]. As previously observed, ensembles of spins get polarized in the presence of an external magnetic field. When two or more spins are near in space, only few angstroms apart, they become coupled (dipolar coupling). Under this condition, they can exchange polarization, affecting the intensities of the resonances of each of the spins. The dipolar coupled spins do not relax independently. The polarization transfer occurs via auto-relaxation but also through cross-relaxation.

Cross-relaxation mix populations between the two spins. The NOE is used to correlate spins through space [36]. The pulse sequence Nuclear Overhauser Effect SpectroscopY (NOESY) is the most important source of restraints [76]. The cross-peaks in a NOESY spectrum provide the distance information between two hydrogens in a protein. The intensity of the NOE cross-peak ($I_{\mathrm{NOE}}$) is proportional to the distance between two hydrogens (the atoms $i$ and $j$) and depends on the cross-relaxation rate:

$$I_{\mathrm{NOE}} = \alpha \frac{1}{\langle D_{ij} \rangle^6},$$

where $\alpha$ is the proportionality constant and $\langle D_{ij} \rangle$ is the time averaged distance between the two hydrogens. Note that the intensity drops with the sixth power of the distance. Only distances smaller than 6 Å can be therefore measured.

The parameter $\alpha$ contains information on the dynamics of the system ($\alpha = f(\tau)$). $\tau$ is the effective correlation time of the nuclei and contains the information about the internal dynamics of each hydrogen, as well as the global dynamics of the protein, such as the overall rotational correlation time. $\tau$ cannot be quantitatively treated for each individual hydrogen, and, thus, the NOE information is used in a semiquantitative way. Instead of giving exact distance information, NOEs give ranges of distance, i.e., a lower and upper bound on the actual distance.

There are methods following the local dynamics using a relaxation matrix. These methods provide better-quality distance information, but they still give only time-averaged distances [46].

The step of transforming the NOE intensities into ranges of distances is known as calibration. There are several ways to calibrate NOEs. The most frequent way is to use NOE intensities (or volumes) of hydrogen pairs of known secondary structure elements. The distances of those pairs are indeed well known. One can calculate a certain parameter on the basis of these distances and use the same parameter for all NOEs. This method is the most used for initial protein calculation.

A different NOE calibration method can be used during refinements. At this stage of protein calculations, the structure is already known. Thus, the distances extracted from the structures can be used for NOE calibration.

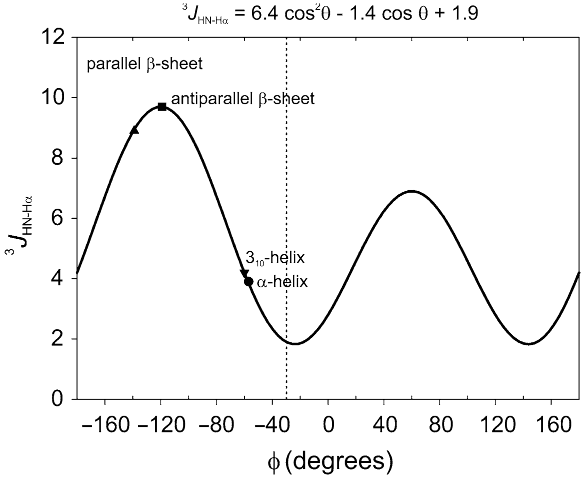


**Fig. 18.8** Karplus Plot of $^3J_{\mathrm{H}_N-\mathrm{H}_\alpha}$ (in Hz) versus the torsion angle $\Phi$. The *grey solid curve* is the best fit of equation parameters (*top of the figure*) where $\theta = |\Phi - 60|$. Values for regular secondary structures are indicated for $\alpha$-helix (*circle* at $-57°$, 3.9 Hz), $3_{10}$ helix (*inverted triangle* at $-60°$, 4.2 Hz), antiparallel $\beta$-sheet (*square* at $-139°$, 8.9 Hz), and antiparallel $\beta$-sheet (*triangle* at $-119°$, 9.7 Hz) [55]. The region on the *left* delimited by the *dotted green line* ($-30°, -180°$) concentrates dihedral angles ($\Phi$) of all amino acids (exception made for the glycines) [75]

### 18.6.1 Scalar Coupling (J)

The other source of information in the NMR experiments is the scalar couplings ($J$). Differently from the dipolar coupling that occurs through space, the scalar coupling occurs through bonds. $J$ coupling can be through one, two, or three bonds ($^1J$, $^2J$, $^3J$). One-bond $J$ coupling are typically heteronuclear, such as the coupling between amidic nitrogen and hydrogen ($^1J_{^{15}\mathrm{N}-^{1}\mathrm{H}}$). Two-bond $J$ coupling occurs between geminal hydrogens, such as $CH_2$.

Finally, three-bond $J$ coupling are the most important for structural information. Their value gives information about dihedral angles. For instance, the coupling between the amidic hydrogen and alpha hydrogen ($^3J_{\mathrm{H}_N-\mathrm{H}_\alpha}$) depends on the $\Phi$ angle of the Ramachandran plot. Figure 18.8 shows the Karplus relation [33] of the dependence of $^3J_{\mathrm{H}_N-\mathrm{H}_\alpha}$ with $\Phi$. There are several NMR experiments designed to measure several dihedrals of a protein.

### 18.6.2 Chemical Shift

As previously shown, chemical shifts are dependent on the microenvironment. They are very sensitive to small changes. A correlation between chemical shifts of

**Table 18.4** The correlation between chemical shifts and secondary structures of proteins

| Residue | Random coil value of chemical shift (rc, ppm) | | | |
|---|---|---|---|---|
| | C′ | Ha | CA | CB |
| | Condition to assign a secondary structure | | | |
| | $\alpha$-helix $- >$ rc $+ 0.5$ | $\alpha$-helix $- >$ rc $- 0.1$ | $\alpha$-helix $- >$ rc $+ 0.7$ | $\alpha$-helix $- >$ rc $+ 0.7$ |
| | $\beta$-sheet $- <$ rc $- 0.5$ | $\beta$-sheet $- <$ rc $+ 0.1$ | $\beta$-sheet $- <$ rc $- 0.7$ | $\beta$-sheet $- <$ rc $- 0.7$ |
| Ala | 177.1 | 4.19 | 52.5 | 19 |
| Cys | 174.8 | 4.52 | 58.3 | 28.6 |
| Asp | 177.2 | 4.63 | 54.1 | 40.8 |
| Glu | 176.1 | 4.24 | 56.7 | 29.7 |
| Phe | 175.8 | 4.42 | 57.9 | 39.3 |
| Gly | 173.6 | 4.11 | 45 | 0 |
| His | 175.1 | 4.59 | 55.8 | 32 |
| Ile | 176.9 | 4.09 | 62.6 | 37.5 |
| Lys | 176.5 | 4.23 | 56.7 | 32.3 |
| Leu | 177.1 | 4.35 | 55.7 | 41.9 |
| Met | 175.8 | 4.32 | 56.6 | 32.8 |
| Asn | 175.1 | 4.62 | 53.6 | 39 |
| Pro | 176 | 4.33 | 62.9 | 31.7 |
| Gln | 176.3 | 4.28 | 56.2 | 30.1 |
| Arg | 176.5 | 4.32 | 56.3 | 30.3 |
| Ser | 173.7 | 4.38 | 58.3 | 62.7 |
| Thr | 175.2 | 4.37 | 63.1 | 68.1 |
| Val | 177.1 | 4.11 | 63 | 31.7 |
| Trp | 175.8 | 4.42 | 57.8 | 28.3 |
| Tyr | 175.7 | 4.43 | 58.6 | 38.7 |

The random coil (rc) chemical shift value for each nuclei is presented for each amino acid residue
The condition for assigning a secondary structure element on the basis of the chemical shift is given for each nucleus

hydrogen alpha (H$\alpha$), carbon alpha ($^{13}C\alpha$), carbon beta ($^{13}C_{\beta}$), and the carbonyl ($^{13}C'$) and the secondary structure has been established. It consists in a very important structural information, because after resonance assignments of a protein, it becomes straightforward to determine its secondary structure elements based solely on chemical shifts. Table 18.4 summarizes the correlation between each of the nuclei and the chemical shift.

### *18.6.3 Residual Dipolar Couplings*

As previously observed, the dipolar coupling is responsible for the mechanism of polarization transfer through cross-relaxation, which leads to the NOEs. However, dipolar couplings cannot be measured in the NMR spectra because of the isotropic molecular tumbling.

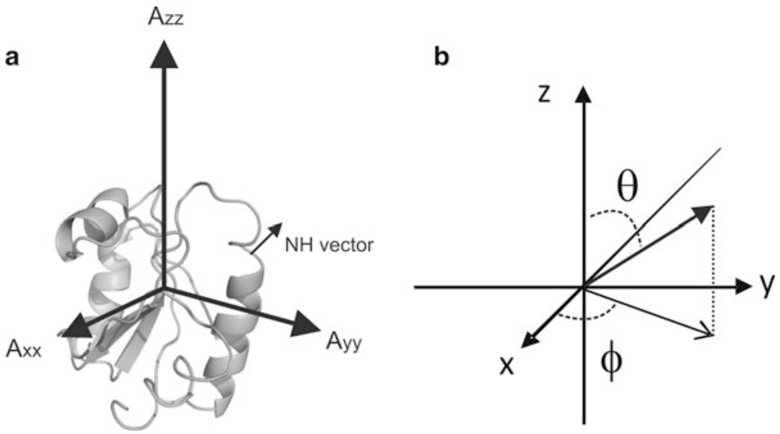


**Fig. 18.9** (**a**) A protein structure (yeast thioredoxin, PDB id: 2I9H) showing the calculated molecular alignment tensors $A_{xx}$, $A_{yy}$, $A_{zz}$, as well as the representation of a dipolar vector (the NH vector in this case). By definition, $A_{zz} > A_{yy} > A_{xx}$. The principal molecular alignment tensor is therefore Azz. (**b**) Representation of the dipolar vector (the NH vector) in the molecular orientation frame of reference

In the 1990s, Prestegards and collaborators solubilized proteins in anisotropic media and showed that the residual orientation of the protein was able to recover dipolar coupling information. Anisotropic media consist of colloidal phases, such as bicelles and liquid crystals, or bacteriophages, such as Pf1, which are spontaneously oriented in the magnetic field. They restrict the Brownian motion of proteins in a way that induces a residual orientation due to the intrinsic anisotropic shape of the protein (see Fig. 18.9). Still the proteins keep tumbling fast, maintaining all the good behavior in of sharp lines, necessary for solution NMR.

Still the proteins keep tumbling fast in solution, maintaining all the good-behavior in solution of sharp lines, necessary for solution NMR. The residual orientation induces the reappearance of the dipolar coupling in solution. The residual dipolar coupling constant depends on the degree of orientation of the protein in the anisotropic media. The spectroscopist is able to tune the line shape and the degree of orientation, changing the concentration and other properties of the anisotropic media.

Dipolar coupling depends on the angle between the dipolar vectors with the main static magnetic field. This is true for a static oriented sample. Proteins dissolved in anisotropic media are not static. In this case, the residual dipolar coupling (RDC) does not depend directly on the angle of the dipolar vector with the static magnetic field, but RDCs are the measure of the angle of the dipolar vector with the principal molecular alignment tensor.

The principal molecular alignment tensors can be measured experimentally and also calculated from the molecule shape (Fig. 18.9). Thus, RDCs can be considered as an experimental restraint. This is a good quality restraint because it is a long-range angular restraint. RDCs have been used extensively as a refinement tool and their use allows for improving the geometric quality of the structures [44].

## 18.7 Experimental Pseudo-potentials

We introduce in this section some experimental pseudo-potentials based on the information obtained by NMR experiments. We describe NOEs as distance restraints, scalar coupling and chemical shifts as short-range angular restraints (proper dihedrals), and RDCs as long-range angular restraints. There are other sources of restraints that we do not discuss here: paramagnetic restraint, which are long-range distance restraints [15], chemical shift anisotropy restraints [42, 43, 74], among others.

The general strategy is to transform the experimental information into pseudo-potentials that can be used in the structural calculations. Next, we describe some pseudo-potential for each information obtained experimentally.

### *18.7.1 NOEs: Distance Restraints*

After NOE calibration, the list of NOEs serves as an input for structural calculation. The NOE assignment list contains the specification of the hydrogen pair and the distance information, determining a lower ($L_{ij}$) and upper bound distances ($U_{ij}$). The lower bound is approximately 1.8 Å, which is the shortest possible distance between two hydrogens, accordingly to their atomic VdW radii. The upper bound distance depends on the target distance calculated from NOE calibration. Typically the distance restraints are assigned in classes: weak ($U_{ij} = 6$ Å), medium ($U_{ij} = 3.4$ Å), and strong ($U_{ij} = 2.8$ Å). The interval for each class is somewhat arbitrary and can vary from author to author.

**Quadratic Pseudo-potential** The pseudo-potential for NOE can be defined as follows. It gives no energy penalty when the distance between the two hydrogens ($i$ and $j$) is contained in the interval $[L_{ij}, U_{ij}]$. The potential increases quadratically when $r$ does not belong to the given interval:

$$V_{ij} = \begin{cases} C_1(r - L_{ij})^2, & \text{if } r < L_{ij} \\ 0, & \text{if } L_{ij} < r < U_{ij} \\ C_2(r - U_{ij})^2, & \text{if } r > U_{ij}, \end{cases} \tag{18.3}$$

where $C_1$ and $C_2$ are force constants that control the steepness of the energy pseudo-potential.

**Biharmonic Pseudo-potential** The pseudo-potential for NOE can also be defined as a function of a unique target distance $D_{ij}$ that can be calibrated from NOE intensities. In this case, the pseudo-potential is defined as follows:

$$V_{ij} = \begin{cases} C_1(r - D_{ij})^2, & \text{if } r > D_{ij} \\ C_2(r - D_{ij})^2, & \text{if } r < D_{ij}, \end{cases}$$

where $C_1$ and $C_2$ are force constants that are weighed by the thermal energy ($K_bT$) available in the computational system:

$$C_1 = S_1 \frac{K_b T}{2} \quad \text{and} \quad C_2 = S_2 \frac{K_b T}{2}$$

where $K_b$ is the Boltzmann constant and $T$ is the absolute temperature of the system. Note that the potential is not zero when $r$ is within the interval defined by a lower and an upper bound. $S_1$ and $S_2$ are scale factors.

### *18.7.2 Dihedral Restraints*

Dihedral restraints can be incorporated in the structural calculation. They are obtained from scalar coupling measurements and chemical shift information. For each dihedral restraint, we have the target dihedral $\theta_{\text{target}}$ and the permitted variation $\Delta\theta$, which is usually relatively large. This way, it allows the dihedral conformational space to vary freely within the low-energy Ramachandran area.

Pseudo-potential for dihedral angle is defined as follows:

$$V_{\text{dihedral}} = \begin{cases} C_1(\theta - \theta_{\text{target}})^2, & \text{if } \theta < \theta_{\text{target}} - \Delta\theta \\ 0, & \text{if } \theta_{\text{target}} - \Delta\theta < \theta < \theta_{\text{target}} + \Delta\theta \\ C_2(\theta - \theta_{\text{target}})^2, & \text{if } \theta > \theta_{\text{target}} + \Delta\theta, \end{cases}$$

where $C_1$ and $C_2$ are the two force constants.

### *18.7.3 Scalar J-Coupling Restraints*

The pseudo-potential energy term for scalar coupling makes use of the Karplus relation. This equation uses the dihedral angle $\theta$ obtained at each time step of structure calculation to obtain the calculated scalar coupling ($J_{\text{calculated}}$).

$$J = A\cos^2(\theta + P) + B\cos(\theta + P) + C,$$

where $A$, $B$, and $C$ are the Karplus coefficients and $P$ is a phase. It then uses $J_{\text{calculated}}$ to create a pseudo-potential $V_J$ by comparing it to the experimental J coupling ($J_{\text{observed}}$). The pseudo-potential is defined as follows:

$$V_J = C(J_{\text{calculated}} - J_{\text{observed}})^2,$$

where $C$ is the force constant.

### 18.7.4 Chemical Shift Restraints

$^1$H and $^{13}$C chemical shifts correlate with the angles $\Phi$ and $\Psi$ and can define secondary structure elements. Several implementations on protein structural calculation include harmonic potentials for chemical shifts. The X-PLOR-NIH package for protein structural calculation [64] includes pseudo-potentials for $C_\alpha$ and $C_\beta$ chemical shifts [37]. It also includes pseudo-potentials for non-exchangeable hydrogens. Chemical shifts are calculated on the basis of semiempirical methods, where random coil values, ring currents, magnetic anisotropy, and electric-field chemical shifts are considered. The experimental chemical shift is compared to the predicted one from the structure, and the pseudo-potential takes care of refining the structure to agree with chemical shifts [37, 38].

The most used strategy to take into account chemical shifts is through the prediction of the $\Phi$ and $\Psi$ dihedral angles. The program TALOS [66] uses a combination of six chemical shifts information: $\delta_{H_N}$, $\delta_{H\alpha}$, $\delta_{C\alpha}$, $\delta_{C_\beta}$, $\delta_{C'}$, and $\delta_N$. The program is based on a search on a database containing 200 high-resolution protein structures, containing sequence information, $\Phi$ and $\Psi$ torsion angles, and chemical shift assignments. It looks for chemical shift similarities between a certain residue and the two adjacent residues (triplets of residues). It always uses triplets of residues to predict backbone torsion angles of a given residue. If there is a consensus of $\Phi$ and $\Psi$ angles among the ten best database matches, then TALOS uses these database triplet structures to form a prediction for the backbone angles of the target residue.

Based on the matches, TALOS calculates a consensus for $\Phi$ and $\Psi$ angles ($\Phi_{\text{target}}$ and $\Psi_{\text{target}}$). The values of $\Phi_{\text{target}}$ and $\Delta\Phi$ and $\Psi_{\text{target}}$ and $\Delta\Psi$ are included as dihedral angle restraints. The accuracy of TALOS predictions is about 89%. Most of the errors occur in regions of the Ramachandran that does not define secondary structure elements. TALOS prediction can thus be used reliably for secondary structure elements.

### 18.7.5 Residual Dipolar Coupling Restraints

As observed before, partial orientation of macromolecules in anisotropic media allowed the detection of RDCs. RDCs are good quality restraints because they define angles between a bond vector and the principal molecular alignment tensor (see Fig. 18.9). In order to compute RDCs, it is necessary to use an external orientational axis that is the reference for the angle measurement between the bond vectors. The implementations of RDC pseudo-potentials in the program Xplor-NIH can take into account dipolar vectors between atoms that are directly bonded (such as N–H or C–H bonds), or more flexible situations where the dipolar vector is between atoms not directly bonded, such as $^1$H-$^1$H dipolar couplings. $^1$H–$^1$H dipolar couplings are more difficult since $^1$H-$^1$H distances can vary. In this chapter, we describe only the directly bonded RDCs. For more detailed information on other implementations, the reader is referred to [1, 10–12, 49, 63, 70, 71].

A necessary step is the calculation from the structure of the rhombicity and of the amplitude of the molecular alignment tensor. This is accomplished from the shape of the molecule. The molecular alignment tensors from experimental RDC are obtained from the following equation:

$$\mathrm{RDC}(\theta,\Phi) = A_{\mathrm{a}}\left\{(3\cos^2\theta - 1) + \frac{3}{2}R(\sin^2\theta\cos^2\Phi)\right\},$$

where $\theta$ and $\Phi$ are the polar angles of the dipolar vector in the molecular frame of reference (see Fig. 18.9), the axial $A_{\mathrm{a}}$ and radial $A_{\mathrm{r}}$ components, and rhombicity $R$ are defined as follows:

$$A_{\mathrm{a}} = \frac{1}{3}\left\{\frac{A_{zz} - (A_{yy} + A_{xx})}{2}\right\}, \quad A_{\mathrm{r}} = \frac{A_{xx} - A_{yy}}{3}, \quad R = \frac{A_{\mathrm{r}}}{A_{\mathrm{a}}}.$$

The pseudo-potential is defined as a quadratic harmonic potential:

$$V_{\mathrm{RDC}} = K_{\mathrm{RDC}}(\mathrm{RDC}_{\mathrm{calculated}} - \mathrm{RDC}_{\mathrm{observed}})^2.$$

More frequently, $\theta$, the angle between the internuclear dipolar vector and the reference external vector, which represents $A_{zz}$ in the calculation, is obtained with a good precision. The rhombic component is usually not precise enough to be used in the calculation. Thus, in practice, RDCs are able to define a cone with angle $\pm\theta$ around the principal component of the molecular axis. Of course, the lack of precision in $\Phi$ limits the restraining ability of RDCs.

So far, we provided a description of pseudo-potentials which are based on experimental restraints obtained by NMR. In the next sections, we describe some computational solutions for calculating protein structures by using the NMR experimental information.

## 18.8 Distance Geometry Methods

The most important aspect for protein structure determination by NMR is the exploration of the conformational space imposed by the experimental restraints. X-ray diffraction of a single crystal generates an electron density map, which directly provides structural information. In contraposition, NMR experimental restraints are not able to give structural information, but rather short-range distances and dihedral angles restraints. The result of such a calculation is not a single structure, as for X-ray diffraction, but a set of structures that are all able to satisfy the experimental restraints.

As discussed earlier, NMR experimental restraints consist of semi-quantitative short-range distances and angles information. The structural calculation uses ranges of distances and angles, rather than precise measurements. NMR distance and angle restraints provide upper and lower bounds for both distances and angles.

Ideally, the measurement of precise long-range (in the order of the radius of gyration) distances or angles generates higher-quality restraints. However, this kind of restraints is difficult to measure by NMR. RDCs are better-quality restraints because they give information about long-range angles, but their application is restricted. In fact, only $\theta$ angles can be measured with precision. Nevertheless, the inclusion of RDCs in the structure calculation has a dramatic effect on the geometric quality [68]. Recent advances in solid state NMR and paramagnetic relaxation enhancement experiments (PRE) in solution introduced some better-quality long-range distance restraints [21,31,40].

What makes structure determination by NMR possible is the fact that the number of short-range distance restraints is generally much larger than the degrees of freedom. There are two degrees of freedom per amino acid residue in the protein backbone ($\Phi$ and $\Psi$ dihedral angles), and, typically, good NMR experiments are able to provide more than 15 short-range restraints per amino acid residue.

NMR structure determination is not a computationally simple problem. The lack of precise distances and angles avoid the solution by fast geometric algorithms [3–5]. The computational solution was the inclusion of an all-atom model with all the known protein geometric angle and distances information along with the semiquantitative short-range experimental information. This approach made it possible to obtain the structures of globular proteins.

In the following, we briefly introduce the computational tools that have been particularly conceived in order to tackle with the problem of exploring the whole conformational space imposed by the imprecise experimental restraints.

The most naive way to explore the whole conformational space is to build a systematic grid of potential conformations and exhaustively explore it. However, this method can be applied only to small peptides [67]. Later we consider again this idea in the context of torsion angle simulated annealing.

The problem of finding the structure of a molecule from some distance and angle restraints is known in the scientific literature as the (molecular) *distance geometry* problem. Many methods and algorithms have been developed over the past last years for an efficient solution of this problem. The first method for distance geometry dates back to the 1970s. The basic idea is to define a penalty function which is able to measure the satisfaction of the available restraints, and to optimize this penalty function. One of the advantages is that the minimum value of the penalty function (corresponding to the optimal structure satisfying all restraints) is known a priori, because, when the data are correct, it must be ideally zero. If there is no geometric solution with error near zero, it is a strong evidence of systematic errors in the experimental data [26,27].

The first method for distance geometry makes use of the metric matrix **G**, from which it is possible to obtain the Cartesian coordinates of the atoms of the molecule by exploiting the available set of distances between some pairs of atoms. The relation between the elements $G_{ij}$ of the metric matrix **G** and the Cartesian coordinates of the two atoms $i$ and $j$ is given by

$$G_{ij} = \mathbf{r_i} \cdot \mathbf{r_j}. \tag{18.4}$$

In the matrix **G**, the diagonal elements are the squares of the Cartesian coordinates of the atom $i$, whereas the off-diagonal elements represent the projection of $\mathbf{r_i}$ over $\mathbf{r_j}$. The square of the Cartesian coordinates of the atom $i$ can be viewed as an vector, defined by the position of $i$ and the origin $(0,0)$. The diagonal elements can be seen the norm of the vector $\mathbf{r_i}$, which defines the position of each atom in relation to the origin.

As it is well known, the dot product can be written as

$$G_{ij} = |r_i||r_j|\cos\theta,$$

where $\theta$ is the angle between the two vectors. Such an angle is 0 for diagonal elements, nonzero for off-diagonal elements.

The metric matrix **G** is built by considering all $N \times N$ possible distances for the set of $N$ atoms. The elements of the metric matrix are obtained through the relations

$$G_{ii} = \frac{1}{N}\sum_{j}^{N} D_{ij}^2 - \frac{1}{2N^2}\sum_{jk}^{N} D_{jk}^2, \quad G_{ij} = \frac{1}{2}\left(G_{ii} + G_{jj} - D_{ij}^2\right),$$

where $D_{ij}$ is the distance between the atoms $i$ and $j$, and $N$ is the total number of atoms. The metric matrix is positive semi-definite and has rank 3. All eigenvalues are positive or zero and at most three eigenvalues are different from zero.

The general metric matrix decomposition equation is used for the diagonalization, which is necessary to find the coordinates of each atom:

$$G_{ij} = \sum_{\alpha=1}^{n} \lambda_\alpha E_i^\alpha E_j^\alpha. \tag{18.5}$$

$E_i^\alpha$ and $E_j^\alpha$ are the eigenvectors and $\lambda_\alpha$ is the eigenvalue of the matrix; $n$ is the dimensionality of the system.

The combination of Eqs. (18.4) and (18.5) leads to the following equation, which enable the calculation of the three-dimensional coordinates of the points of the system from the metric matrix elements:

$$r_i^\alpha = \sqrt{\lambda_\alpha} E_i^\alpha.$$

It is implicit in the equations the assumption that every distance is referenced to the origin $(0,0)$. In general, one of the atoms, say the one labeled with 1, is set to the origin.

As discussed before, the distance information is generally given by a list of lower and upper bounds:

$$L_{ij} < D_{ij} < U_{ij}.$$

The basic steps of the first method for distance geometry are [25]:

1. *Bound smoothing*—consists of extrapolating the tightest possible bounds on the incomplete list of interatomic distances

2. *Metrization*—tries to find a matrix of exact values within the lower and upper bound
3. *Embedding*—computes the coordinates of all atoms of the protein
4. *Optimization*—minimizes the penalty function value, i.e., the measure of the violation of both lower and upper bounds on the distances, where some geometric constraints of proteins are also considered

We give the details of these four main steps in the following.

### 18.8.1 Bound Smoothing

Metric matrix distance geometry algorithms work with exact distances (derived from bond lengths and angles) and NMR experimental data, which are non-exact distances. In the first implementation of algorithms for distance geometry, the distances were chosen independently and randomly within the available lower and upper bounds.

Successively, a bound smoothing was developed for choosing better distances. The technique is based on the fact that interatomic distances always obey triangle inequalities. In fact, the triangle inequality theorem states that any side of a triangle is always shorter than the sum of the two other sides. For a triplet of atoms $(i, j, k)$, it follows that

$$L_{ik} - U_{kj} \leq D_{ij} \leq U_{ik} + U_{kj}.$$

Note that triangle inequality theorem imposes some constraints on $D_{ij}$. Many algorithms for distance geometry consider these inequalities for all possible triplets $(i, j, k)$ in order to obtain the so-called triangle inequalities bounds.

Another relation that could be used for bound smoothing is given by the tetrangle inequalities. The tetrangle inequality is similar to the triangle inequality, but it considers quadruplets of atoms, not triplets. It is able, in general, to provide tighter bounds on $D_{ij}$, but it is much more expensive from a computational point of view.

### 18.8.2 Metrization

The metrization procedure can be used to improve the geometrical consistency of the randomly chosen distances. We suppose that all distances were chosen from bounds previously processed by a bound smoothing technique (based on triangle and/or tetrangle inequalities). The metrization is based on the construction of distance matrices whose elements respect two rules:

1. Their lower and upper bounds satisfy the triangle and the tetrangle inequalities.
2. The chosen distances satisfy the triangle inequality.

The second rule ensures that later interatomic distance choices are consistent with earlier ones. The metrization imposes interdependency between the randomly chosen distances (they are, in fact, not completely independent to each other).

### 18.8.3 Embedding

The initial distances are chosen as an exact distance contained in the interval defined by the corresponding lower and upper bounds. The metric matrix is calculated, and it frequently results in a non-embeddable matrix in the three-dimensional space. This means that the matrix is not positive semidefinite, i.e., the solution is inconsistent with any conformation in the three-dimensional space.

The main aim is to identify an embeddable metric matrix in three dimensions. Within the bound distances, there is a metric matrix in which the absolute values of the three largest eigenvalues are positive, and their corresponding eigenvectors contain the Cartesian coordinates of the atoms of the molecule. If these values are not positive, the chosen distances are not consistent, and the embedding cannot be performed.

### 18.8.4 Optimization

This step consists in improving the quality of the protein structure found during the embedding. To this aim, a penalty function (measuring the violations of lower and upper bounds, as well as some geometrical deviations) is defined and optimized. This penalty function must obey to the following rules:

1. Must be nonnegative
2. Must be zero when all the geometric constraints are satisfied
3. Must be twice differentiable in its whole domain

An example of penalty function is

$$F(x) = \sum_{ij} A_{ij}^2(x) + \sum_{ij} B_{ij}^2(x) + \sum_{ijkm} C_{ijkm}^2(x),$$

where:

- $A_{ij}^2(x) = 0$ if and only if the distance between nonbonded pairs of atoms $(i, j)$ is larger than their hard VdW sphere radii.
- $B_{ij}^2(x) = 0$ if and only if the distance between the pair of atoms $(i, j)$ restrained by experimental data lies within the corresponding lower and upper bound.
- $C_{ijkm}^2(x) = 0$ if and only if the angle $(i, j, k, l)$ respects the absolute chirality.

In order to minimize the penalty function, a conjugate gradient minimization method can be used. Different penalty functions have been defined in different distance geometry approaches [25].

### *18.8.5 Scaling*

At the very end, the obtained protein structure can be scaled so that it represents a globular protein. To this purpose, the expected radius of gyration of the structure is calculated. This expected radius can be larger or smaller than the radius of gyration calculated from the embedded coordinates. Therefore, a scaling factor equal to the ratio between expected and actual radius of gyration is computed. The embedded coordinates are then multiplied by this factor, because it makes any successive regularizations easier to perform.

## 18.9 Simulated Annealing

### *18.9.1 SA in Cartesian Space*

As discussed in the previous section, the first method for distance geometry problems arising in the molecular context makes use of gradient conjugate minimizations of a given penalty function. We remark that such penalty functions do not consider many molecular forces that are instead used in molecular dynamics simulations. As a consequence, structures obtained by this method can produce correct overall folds, but they have poor local geometry. It was realized then that these structure were a very good input for restrained molecular dynamics simulation.

The first approach using restrained molecular dynamics simulation was employed to refine structures calculated from distance matrix distance geometry. The group of Clore and Gronenborn [50–52] used a simulated annealing (SA) algorithm in order to find solutions for multiple variable systems. SA was derived from a metallurgic process where the system is heated at extremely high temperatures and let cooling down slowly. The simulation of this process could allow the atoms of a molecule to assume a low-energy configuration [35].

Standard molecular dynamics simulation force fields are built in order to reproduce the behavior of a molecular system in thermal equilibrium (constant temperatures). High-energy transitions such as cis/trans isomerization and steric hindrance cannot be surpassed using these force fields. For standard molecular dynamics simulation, the calculated structures do not change so much from their initial conformation, or they get stuck at a local minima. In order to partially solve the problem of sampling the conformational space given by the experimental restraints, a set of simplifications was proposed.

The first simplification consists in associating to every atom the same molecular weight (typically 100). This avoids high-frequency bond and angle vibrations, enabling a significant reduction in the number of thermalization steps. If the thermalization is too fast, with a reduced number of integration steps, then high-frequency vibrations, which affect mostly low atomic weight atoms such as hydrogens, can generate strong forces that could break covalent bonds. This simplification is especially important in the SA protocol, where the bath temperature increases up to 2,000 K and the thermalization is essential for the success of the process.

Another simplification is the turning off of attraction nonbonded interaction during the hot phase of SA. The Coulomb term is turned off and the van der Waals potential is replaced by the simplified term (REPEL) [51]:

$$F_{\mathrm{REPEL}} = \begin{cases} 0, & \text{if } r \geq s.r_{\min} \\ k_{\mathrm{rep}}(s^2 r_{\min}^2 - r^2), & \text{if } r < s.r_{\min}, \end{cases}$$

where the values of $r_{\min}$ are the standard values for van der Waals radii (defined in the force fields) [6]. The scale factor $s$ is set to 1.0 in the hot phase and to 0.825 in the cooling phases. In REPEL, only the repulsive term of the Lennard-Jones potential is maintained, reducing in this way the computational cost. This allows for surpassing high-energy barriers, which are due, in many situations, to the attractive forces imposed by Coulomb and VdW interaction, which aid the conformational space sampling.

Additionally, the force field is modified by increasing the penalty for bond and angle geometry violations. Finally, the distance restraint quadratic potential [Eq. (18.3)] is replaced by a simplified linear term, where the penalties increase linearly with the distance restraint violation. It was shown that this modification allows for correcting faster the geometry of the molecule.

During SA, the weight of force field parameters is adjusted to favor the conformational sampling. A typical sequence of events in an SA protocol is showed in Fig. 18.10, where the distance restraint potential (NOE) is weighted high during all phases, while the Coulomb term is turned off.

This new method was included in the structure calculation program XPLOR [8], where a hybrid approach to distance geometry was implemented: both target function minimization and simulated annealing in the structure calculation.

The starting structure is calculated using the distance matrix distance geometry algorithm [73]. Successively, target function minimization is performed and finally a series of cycles of simulated annealing calculation are executed. It is common to compute hundreds of structures. However, only the 20 lower-energy structures are selected to represent the protein.

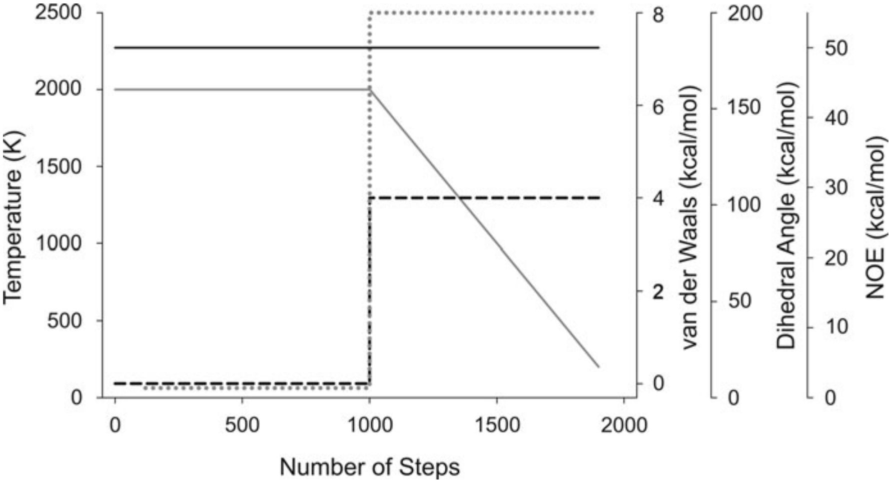


**Fig. 18.10** Illustration of a typical Cartesian space SA protocol used for protein structure calculation. Scheduled changes in the parameter values are plotted as a function of the time steps. The bath temperature is represented as the *solid grey line*, the dihedral angle potential as a *dotted grey line*, the distance restraint potentials as *solid black lines*, and the VdW potential by the *dashed black lines* [7, 8]

## 18.9.2 SA in Torsion Angle Space

Molecular dynamics simulations (as well as SA) in the Cartesian space uses Newton mechanics at discrete time steps in order to describe the protein motion [Eqs. (18.1) and (18.2)]. Newton equations deduce the motion equations of a system from the knowledge of all external forces acting on it.

Another way to approach the molecular mechanics is by solving Lagrange equations. Lagrange mechanics uses scalar equations, which avoid the need to describe all the external forces that act on the system in a vectorial formalism. The Lagrangian function is defined by the difference among kinetic and potential energy:

$$L = T - V,$$

where $T$ is the kinetic energy and $V$ is the potential energy of the system. The motion equations are obtained from the Lagrangian function by the following differential equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0, \tag{18.6}$$

where the $q_i$'s represent the coordinates of the system and $\dot{q}_i$ is the time derivative of the system coordinates (velocity). Note that this equation is not vectorial.

In order to illustrate the Lagrangian mechanics, we consider a simple system consisting of a linear spring-mass system on a frictionless table. The Lagrangian function becomes

$$L = T - V = \frac{1}{2}m\dot{x}^2 - \frac{1}{2}kx^2,$$

where $m$ is the mass, $x$ is the linear coordinate, and $k$ is the spring constant. The conservative system (18.6) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}(m\dot{x}) + kx = 0 \quad \Rightarrow \quad m\ddot{x} + kx = 0.$$

Note that the differentiation led to the equation of motion of the system in the same form as for the Newtonian formalism of classical mechanics, but without the need of figuring out all external vectorial forces on the system.

The same can be done for simulating the motions of a protein. The great advantage is that we can compute positions and the movements (acceleration) of the atoms by simplifying the coordinate system. The variables are only the torsion angles of a protein. The degree of freedom is decreased about tenfold, because the geometrical parameters, such as bond lengths, bond angles, and improper dihedrals (chirality and planarity), are fixed to their optimal values during the simulation.

As discussed before, what makes the search for conformational space by methods such as SA difficult is the rough energy landscape for a protein. There are many local minima to be avoided by computational methods. The strategies to reach the global minimum and avoid kinetic traps demand high computational time and special algorithms.

In Cartesian SA, much of the computational time is focused on calculations of geometrical parameters that almost do not change. The deviations from optimal geometry of bond lengths, bond angles, chirality, and planarity are small because they are parameterized to be as small as possible. In torsion angle dynamics, instead, these are fixed and so is the number of local minima. This is the main reason why torsion angle dynamics increase the efficiency of the search for conformational space imposed by the NMR experimental restraints.

The force field which is used in Cartesian dynamics considers strong potentials in order to keep the covalent structures. In torsion angle dynamics, the parameters are much simplified. One important aspect is that the time step for numerical integration in Cartesian dynamics must be very small ($<5$ fs), and there is therefore the risk of breaking some covalent structures because of bond and angle with high-frequency vibrations. In torsion angle dynamics, time steps can be three times longer because the covalent structures are fixed and such vibrations are inexistent.

In the implementation of torsion angle dynamics, the protein is described as a tree of rigid bodies connected by single bonds. The only degrees of freedom are rotations around the single bonds. The tree structure starts with a base, typically at the N-terminus and ends with the “leaves” that are the end of the side chains and the

C-terminus. The rigid bodies are labeled from 0 to $n$. The base is number 0 and each torsion angle is represented as $\theta_k$, where $k \geq 1$. The conformation of the molecule can be uniquely specified by its torsion angle $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$.

The potential energy is defined as

$$V = \begin{cases} 0 & \text{if distances and angles are within the bounds and atoms are} \\ & \text{not overlapped} \\ V_{\text{target}} & \text{otherwise,} \end{cases}$$

where $V_{\text{target}}$ is the target function that is dependent on the upper and lower bounds for the distance and on the angular restraints. $\omega_0$ is a weighting factor. Note that $V > 0$ if the experimental bound are not satisfied or atoms are overlapped. Motion occurs when $V > 0$. The kinetic energy and the inertia tensor are calculated recursively at each time step of numerical integration. For details on the algorithms, see [22, 24].

The Lagrange equation takes the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\theta}_i}\right) - \frac{\partial L}{\partial \theta_i} = 0.$$

The differentiation leads to equation of motions that takes the form:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) = 0, \tag{18.7}$$

where $M(\theta)$ is the mass matrix and $C(\theta, \dot{\theta})$ is a constant $n$-dimensional vector. Note that Eq. (18.7) was obtained by using a similar mathematical procedure presented in the simple system of linear spring-mass system in a frictionless table [Eq. (18.6)]. For a detailed description, see [22].

Torsion angle space SA is efficient for searching conformational space because it smoothes the protein energy landscape, avoiding local minima. It also enables the hot phase of SA at very high temperature, such as 50,000 K. However, we have to mention that it is a statistical method and there is no mathematical proof that the global minimum could be actually found.

The introduction of the torsion angle space SA solved the problem of searching the conformational space given by NMR experimental restraints. It is the most frequently used method, and it is implemented in all programs developed for structural determinations, such as XPLOR-NIH, CNS, and CYANA. The algorithm is very efficient and enables the calculation of a protein structure in minutes.

## 18.10 Future Demands for Protein Structure Determination

This present chapter showed the evolution of computational methods for protein structural determination using NMR experimental data. It is clear that structural determination by NMR does not rely on direct spatial data but on a set of small-range experimental distance and angle restraints that, combined with some structural geometrical information on proteins, can be exploited for producing structural models. Over the years the NMR structures determined by the methods discussed in this chapter have been accepted by the scientific community as realistic and useful for studying biochemical mechanistic problems.

The torsion angle space SA protocols can be very efficient for searching the conformational space under the constraints given by NMR experiments. All semiautomated methods for structural determination, such as ARIA [62] and UNIO [17, 19, 28, 72], make use of torsion angle space SA. It is also implemented in the software tools for structure determination by NMR, such as XPLOR-NIH [64], CNS [7, 8], and CYANA [23, 24, 28, 47].

Although distance geometry combined with simulated annealing (DGSA) is not the most usual method, it offers many advantages: (1) distance geometry is not a statistical method and can offer mathematical proof that the global minimum has been achieved; (2) DGSA is as fast and efficient in the search of conformational space as it is torsion angle space simulated annealing; (3) since DGSA relies on a geometrical method for the search of conformational space, it can be used for large proteins and complexes. Statistical methods, on the other hand, can become inefficient when the size of the protein is large.

The increase in protein size also imposes a more restricted number of restraints. NMR spectroscopy can nowadays generate structural information for large proteins and protein complexes. However, such a structural information is sparse and new methods for structural calculation with sparse date are becoming increasingly important.

Standley [69] proposed in 1999 a branch-and-bound algorithm for protein refinements with sparse data. They used distance geometry methods to minimize an error function which is based on the experimental restraints, as well as a residue-based protein folding potential. This algorithm is able to identify more compact structures. The protein folding term is based on the idea of using long-range potentials so that the dependence of long-range distance restraints is reduced.

Dong and Wu [16] in 2003 introduced a geometrical method for solving NMR structure with sparse data. In general, NMR spectroscopy generates experimental data that are not complete. They have used geometrical information, in a similar way as bound smoothing and metrization uses triangle and tetrangle inequalities, to build up the "missing" information. The algorithm calculates the coordinates of a given atom on the basis of the coordinates of the previously computed atoms and of the distances between the current and the previous atoms. Some assumptions need to be satisfied in order to use this algorithm. Davis et al. [14] proposed an improved algorithm, called *revised updated geometric build-up algorithm* (RUGB), to build up missing information.

Liberti et al. [41] proposed the use of a discrete search occurring in continuous space for solving protein structure. The main idea is to use distance information between atoms that are contiguous (sequential) in order to discretize the search space (which has the structure of a tree), and to employ a branch-and-prune algorithm for solving the discretized problem. In the branch-and-prune, new candidate atomic positions are generated at each iteration (branching), and their feasibility is verified immediately so that branches of the tree which do not contain solutions can be removed (pruning). The branch-and-prune can work with both exact or interval data [39] and also in the hypothesis in which only distances between hydrogen atoms are available [53].

In conclusion, structural determination using NMR experimental data needs the use of efficient computational methods. The continuous development of NMR and of computational methods can improve the quality, efficiency, and limits for structural determination by NMR.

**Acknowledgments** We are grateful to Prof. Antonio Mucherino and Prof. Carlile Lavor for the edition and revision of this chapter.

## References

1. Bax, A., Kontaxis, G., Tjandra, N.: Dipolar couplings in macromolecular structure determination. Nucl. Magn. Reson. Biolog. Macromol., Pt B **339**, 127–174 (2001)
2. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The Protein Data Bank. Nucleic Acids Res. **28**, 235–242 (2000)
3. Braun, W.: Distance geometry and related methods for protein-structure determination from NMR data. Q. Rev. Biophys. **19**(3–4), 115–157 (1987)
4. Braun, W., Bösch, C., Brown, L.R., Go, N., Wüthrich, K.: Combined use of proton-proton Overhauser enhancements and a distance geometry algorithm for determination of polypeptide conformations. Application to micelle-bound glucagon. Biochimica Et Biophysica Acta **667**(2), 377–396 (1981)
5. Braun, W., Go, N.: Calculation of protein conformations by proton proton distance constraints – A new efficient algorithm. J. Mol. Biol. **186**(3), 611–626 (1985)
6. Brooks, B.R., Bruccoleri, R.E., Olafson, B.D., States, D.J., Swaminathan, S., Karplus, M.: CHARMM – a program for macromolecular energy, minimization, and dynamics calculations. J. Comput. Chem. **4**(2), 187–217 (1983)
7. Brünger, A.T.: Version 1.2 of the crystallography and NMR system. Nat. Protocol. **2**(11), 2728–2733 (2007)
8. Brünger, A.T., Adams, P.D., Clore, G.M., DeLano, W.L., Gros, P., Grosse-Kunstleve, R.W., Jiang, J.S., Kuszewski, J., Nilges, M., Pannu, N.S., Read, R.J., Rice, L.M., Simonson, T., Warren, G.L.: Crystallography & NMR system: a new software suite for macromolecular structure determination. Acta Crystallographica, Section D, Biological Crystallography **54**, 905–921 (1998)
9. Case, D.A., Cheatham, T.E., Darden, T., Gohlke, H., Luo, R., Merz, K.M. Jr., Onufriev, A., Simmerling, C., Wang, B., Woods, R.J.: The Amber biomolecular simulation programs. J. Comput. Chem. **26**(16), 1668–1688 (2005)
10. Clore, G.M., Gronenborn, A.M., Bax, A.: A robust method for determining the magnitude of the fully asymmetric alignment tensor of oriented macromolecules in the absence of structural information. J. Mag. Reson. **133**(1), 216–221 (1998)
11. Clore, G.M., Gronenborn, A.M., Tjandra, N.: Direct structure refinement against residual dipolar couplings in the presence of rhombicity of unknown magnitude. J. Mag. Reson. **131**(1), 159–162 (1998)
12. Clore, G.M., Starich, M.R., Bewley, C.A., Cai, M., Kuszewski, J.: Impact of residual dipolar couplings on the accuracy of NMR structures determined from a minimal number of NOE restraints. J. Am. Chem. Soc. **121**(27), 6513–6514 (1999)

13. Collet, J.F., Messens, J.: Structure, function, and mechanism of thioredoxin proteins. Antioxidants & Redox Signaling **13**(8), 1205–1216 (2010)
14. Davis, R.T., Ernst, C., Wu, D.: Protein structure determination via an efficient geometric build-up algorithm. BMC Struct. Biol. **10**(1):S7 (2010)
15. Donaldson, L.W., Skrynnikov, N.R., Choy, W.-Y., Muhandiram, D.R., Sarkar, B., Forman-Kay, J.D., Kay, L.E.: Structural characterization of proteins with an attached ATCUN motif by paramagnetic relaxation enhancement NMR spectroscopy. J. Am. Chem. Soc. **123**(40), 9843–9847 (2001)
16. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Global Optim. **26**, 321–333 (2003)
17. Ellgaard, L., Bettendorff, P., Braun, D., Herrmann, T., Fiorito, F., Jelesarov, I., Guntert, P., Helenius, A., Wüthrich, K.: NMR structures of 36 and 73-residue fragments of the calreticulin P-domain. J. Mol. Biol. **322**(4), 773–784 (2002)
18. Engh, R.A., Huber, R.: Accurate bond and angle parameters for X-ray protein-structure refinement. Acta Crystallographica, Section A **47**, 392–400 (1991)
19. Fiorito, F., Herrmann, T., Damberger, F.F., Wüthrich, K.: Automated amino acid side-chain NMR assignment of proteins using (13)C- and (15)N-resolved 3D (1)H,(1)H -NOESY. J. Biomol. NMR **42**(1), 23–33 (2008)
20. Galvão-Botton, L.M.P., Katsuyama, A.M., Guzzo, C.R., Almeida, F.C., Farah, C.S., Valente, A.P.: High-throughput screening of structural proteomics targets using NMR. FEBS Lett. **552**(2–3), 207–213 (2003)
21. Gillespie, J.R., Shortle, D.: Characterization of long-range structure in the denatured state of staphylococcal nuclease; 2. Distance restraints from paramagnetic relaxation and calculation of an ensemble of structures. J. Mol. Biol. **268**(1), 170–184 (1997)
22. Guntert, P.: Structure calculation of biological macromolecules from NMR data. Q. Rev. Biophys. **31**(2), 145–237 (1988)
23. Guntert, P., Braun, W., Wüthrich, K.: Efficient computation of 3-dimensional protein structures in solution from Nuclear-Magnetic-Resonance data using the program DIANA and the supporting programs CALIBA, HABAS and GLOMSA. J. Mol. Biol. **217**(3), 517–530 (1991)
24. Guntert, P., Mumenthaler, C., Wüthrich, K.: Torsion angle dynamics for NMR structure calculation with the new program DYANA. J. Mol. Biol. **273**(1), 283–298 (1997)
25. Havel, T.F.: An evaluation of computational strategies for use in the determination of protein-structure from distance constraints obtained by Nuclear-Magnetic-Resonance. Progr. Biophys. Mol. Biol. **56**(1), 43–78 (1991)
26. Havel, T.F., Wüthrich, K.: A distance geometry program for determining the structures of small proteins and other macromolecules from nuclear magnetic resonance measurements of intramolecular H-1-H-1 proximities in solution. Bull. Math. Biol. **46**(4), 673–698 (1984)
27. Havel, T.F., Wagner, G., Wüthrich, K.: Spatial structures for BPTI in the crystal and in solution from distance geometry calculations. Experientia **40**(6), 608–608 (1984)
28. Herrmann, T., Guntert, P., Wüthrich, K.: Protein NMR structure determination with automated NOE-identification in the NOESY spectra using the new software ATNOS. J. Biomol. NMR **24**(3), 171–189 (2002)
29. Holmgren, A.: Antioxidant function of thioredoxin and glutaredoxin systems. Antioxidants & Redox Signaling **2**(4), 811–820 (2000)
30. Holmgren, A., Bjornstedt, M.: Thioredoxin and thioredoxin reductase. Methods Enzymol **252**, 199–208 (1995)
31. Iwahara, J., Schwieters, C.D., Clore, G.M.: Ensemble approach for NMR structure refinement against H-1 paramagnetic relaxation enhancement data arising from a flexible paramagnetic group attached to a macromolecule. J. Am. Chem. Soc. **126**(18), 5879–5896 (2004)
32. Jorgensen, W.L., Maxwell, D.S., TiradoRives, J.: Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. J. Am. Chem. Soc. **118**(45), 11225–11236 (1996)
33. Karplus, M.: Vicinal proton coupling in nuclear magnetic resonance. J. Am. Chem. Soc. **85**(18), 2870–2871 (1963)

34. Kauzmann, W.: Three-dimensional structures of proteins. Biophys. J. **4**, 43–54 (1964)
35. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
36. Kumar, A., Wagner, G., Ernst, R.R., Wuethrich, K.: Buildup rates of the nuclear Overhauser effect measured by two-dimensional proton magnetic-resonance spectroscopy – Implications for studies of protein conformation. J. Am. Chem. Soc. **103**(13), 3654–3658 (1991)
37. Kuszewski, J., Qin, J., Gronenborn, A.M., Clore, G.M.: The impact of direct refinement against C-13(Alpha) and C-13(Beta) chemical-shifts on protein-structure determination by NMR. J. Mag. Reson. Ser. B, **106**(1), 92–96 (1995)
38. Kuszewski, J., Gronenborn, A.M., Clore, G.M.: Improving the quality of NMR and crystallographic protein structures by means of a conformational database potential derived from structure databases. Protein Sci. **5**(6), 1067–1080 (1996)
39. Lavor, C., Liberti, L., Mucherino, A.: The interval Branch-and-Prune Algorithm for the Discretizable Molecular Distance Geometry Problem with Inexact Distances. to appear in J. Global Optim. (2012) DOI: 10.1007/s10898-011-9799-6
40. Liang, B.Y., Bushweller, J.H., Tamm, L.K.: Site-directed parallel spin-labeling and paramagnetic relaxation enhancement in structure determination of membrane proteins by solution NMR spectroscopy. J. Am. Chem. Soc. **128**(13), 4389–4397 (2006)
41. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
42. Lipsitz, R.S., Tjandra, N.: Carbonyl CSA restraints from solution NMR for protein structure refinement. J. Am. Chem. Soc. **123**(44), 11065–11066 (2001)
43. Lipsitz, R.S., Tjandra, N.: N-15 chemical shift anisotropy in protein structure refinement and comparison with NH residual dipolar couplings. J. Mag. Reson. **164**(1), 171–176 (2003)
44. Lipsitz, R.S., Tjandra, N.: Residual dipolar couplings in NMR structure analysis. Ann. Rev. Biophys. Biomol. Struct. **33**, 387–413 (2004)
45. Linge, J.P., Nilges, M.: Influence of non-bonded parameters on the quality of NMR structures: A new force field for NMR structure calculation. J. Biomol. NMR **13**(1), 51–59 (1999)
46. Linge, J.P., Habeck, M., Rieping, W., Nilges, M.: Correction of spin diffusion during iterative automated NOE assignment. J. Mag. Reson. **167**(2), 334–342 (2004)
47. Lopez-Mendez, B., Guntert, P.: Automated protein structure determination from NMR spectra. J. Am. Chem. Soc. **128**(40), 13112–13122 (2006)
48. Lovell, S.C., Davis, I.W., Arendall, W.B., de Bakker, P.I.W., Word, J.M., Prisant, M.G., Richardson, J.S., Richardson, D.C.: Structure validation by $C_\alpha$ geometry: $\Phi$, $\Psi$ and $C_\beta$ deviation. Protein. Struct. Funct. Genet. **50**(3), 437–450 (2003)
49. Meiler, J., Blomberg, N., Nilges, M., Griesinger, C.: A new approach for applying residual dipolar couplings as restraints in structure elucidation. J. Biomol. NMR **16**(3), 245–252 (2000)
50. Nilges, M., Clore, G.M., Gronenborn, A.M.: Determination of 3-dimensional structures of proteins from interproton distance data by dynamical simulated annealing from a random array of atoms – circumventing problems associated with folding. FEBS Lett. **239**(1), 129–136 (1988)
51. Nilges, M., Clore, G.M., Gronenborn, A.M.: Determination of 3-dimensional structures of proteins from interproton distance data by hybrid distance geometry – Dynamical simulated annealing calculations. FEBS Lett. **229**(2), 317–324 (1988)
52. Nilges, M., Gronenborn, A.M., Brünger, A.T., Clore, G.M.: Determination of 3-dimensional structures of proteins by simulated annealing with interproton distance restraints – Application to crambin, potato carboxypeptidase inhibitor and barley serine proteinase inhibitor-2. Protein Eng. **2**(1), 27–38 (1988)
53. Nucci, P., Nogueira, L.T., Lavor, C.: Determining protein backbone from H and H-alpha short interatomic distances, In: Mastorakis, N.E., Mladenov, Demiralp, M, V, Bojkovic Z (eds.) WSEAS Press, Athens, Advances in Biology, Bioengineering and Environment, 43–48 (2010)
54. Overhauser, A.W.: Polarization of nuclei in metals. Phys. Rev.**92**(2), 411–415 (1953)

55. Pardi, A., Billeter, M., Wüthrich, K.: Calibration of the angular-dependence of the amide proton-C-alpha proton coupling-constants, 3JHN-alpha, in a globular protein – Use of 3JHN-alpha for identification of helical secondary structure. J. Mol. Biol. **180**(3), 741–751 (1984)
56. Pauling, L., Corey, R.B.: 2 hydrogen-bonded spiral configurations of the polypeptide chain. J. Am. Chem. Soc. **72**(11), 5349–5349 (1950)
57. Pauling, L., Corey, R.B.: The structure of synthetic polypeptides. Proc. Nat. Acad. Sci. Unit. States Am. **37**(5), 241–250 (1951)
58. Pauling, L., Corey, R.B.: Configuration of polypeptide chains. Nature **168**, 550–551 (1951)
59. Pauling, L., Corey, R.B., Branson, H.R.: The structure of proteins – 2 hydrogen-bonded helical configurations of the polypeptide chain. Proc. Nat. Acad. Sci. Unit. States Am. **37**, 205–211 (1951)
60. Pinheiro, A.S., Amorim, G.C., Netto, L.E., Almeida, F.C., Valente, A.P.: NMR solution structure of the reduced form of thioredoxin 1 from Sacharomyces cerevisiae. Protein. Struct. Funct. Bioinformatics **70**(2), 584–587 (2008)
61. Ramachandran, G.N., Ramakrishnan, C., Sasisekharan, V.: Stereochemistry of polypeptide chain configurations. J. Mol. Biol. **7**(1), 95–99 (1963)
62. Rieping, W., Habeck, M., Bardiaux, B., Bernard, A., Malliavin, T.E., Nilges, M.: ARIA2: Automated NOE assignment and data integration in NMR structure calculation. Bioinformatics **23**(3), 381–382 (2007)
63. Sass, H.J., Musco, G., Stahl, S.J., Wingfield, P.T., Grzesiek, S.: An easy way to include weak alignment constraints into NMR structure calculations. J. Biomol. NMR **21**(3), 275–280 (2001)
64. Schwieters, C.D., Kuszewski, J.J., Clore, G.M.: Using Xplor-NIH for NMR molecular structure determination. Progr. Nucl. Mag. Reson. Spectros. **48**(1), 47–62 (2006)
65. Scott, W.R.P., Hürger, Ph.H., Tironi, I.G., Mark, A.E., Billeter, S.R., Fennen, J., Torda, A.E., Huber, T., van Gunsteren, P.K.W.F.: The GROMOS biomolecular simulation program package. J. Phys. Chem. A **103**(19), 3596–3607 (1999)
66. Shen, Y., Delaglio, F., Cornilescu, G., Bax, A.: TALOS plus: a hybrid method for predicting protein backbone torsion angles from NMR chemical shifts. J. Biomol. NMR **44**(4), 213–223 (2009)
67. Smith, G.M., Veber, D.F.: Computer-aided, systematic search of peptide conformations constrained by NMR data. Biochem. Biophys. Res. Commun. **134**(2), 907–914 (1986)
68. Spronk, C.A.E.M., Linge, J.P., Hilbers, C.W., Vuister, G.W.: Improving the quality of protein structures derived by NMR spectroscopy. J. Biomol. NMR **22**(3), 281–289 (2002)
69. Standley, D.M., Eyrich, V.A., Felts, A.K., Friesner, R.A., McDermott, A.E.: A branch and bound algorithm for protein structure refinement from sparse NMR data sets. J. Mol. Biol. **285**(4), 1691–1710 (1999)
70. Tjandra, N., Bax, A.: Direct measurement of distances and angles in biomolecules by NMR in a dilute liquid crystalline medium. Science **278**(5340), 1111–1114 (1997)
71. Tjandra, N., Marquardt, J., Clore, G.M.: Direct refinement against proton-proton dipolar couplings in NMR structure determination of macromolecules. J. Mag. Reson. **142**(2), 393–396 (2000)
72. Volk, J., Herrmann, T., Wüthrich, K.: Automated sequence-specific protein NMR assignment using the memetic algorithm MATCH. J. Biomol. NMR **41**(3), 127–138 (2008)
73. Wagner, G., Braun, W., Havel, H.T., Schaumann, T., Go, N., Wüthrich, K.: Protein structures in solution by Nuclear-Magnetic-Resonance and Distance Geometry – the polypeptide fold of the basic pancreatic trypsin-inhibitor determined using 2 different algorithms, Disgeo and Disman. J. Mol. Biol. **196**(3), 611–639 (1987)
74. Wu, Z.R., Tjandra, N., Bax, A.: P-31 chemical shift anisotropy as an aid in determining nucleic acid structure in liquid crystals. J. Am. Chem. Soc. **123**(15), 3617–3618 (2001)
75. Wüthrich, K.W.T.: NMR of Proteins and Nucleic Acids, vol. 1. Wiley-Interscience, New York (1986)
76. Wüthrich, K., Billeter, M., Braun, W.: Polypeptide secondary structure determination by nuclear magnetic resonance observation of short proton-proton distances. J. Mol. Biol. **180**(3), 715–740 (1984)

# Index

A. Mucherino et al. (eds.), *Distance Geometry: Theory, Methods, and Applications*,
DOI 10.1007/978-1-4614-5128-0,