A quaternion-based Branch-and-Prune algorithm for Discretizable Molecular Distance Geometry Problem with exact distance values

Felipe Fidalgo[1], Emerson V. Castelani[2] and Guilherme Philippi[3]

[1] *Department of Mathematics, Universidade Federal de Santa Catarina, Blumenau*
[2] *Department of Mathematics, Universidade Estadual de Maringá, Maringá*
[3] *Universidade Federal de Santa Catarina, Blumenau*

Abstract

Linha 01
Linha 02
Linha 03
Linha 04
Linha 05
Linha 06
Linha 07
Linha 08
Linha 09
Linha 10
Linha 11
Linha 12

Keywords:

1. Introduction

The *Distance Geometry Problem* (DGP) with exact distances is the class of problems of embedding simple and undirected graphs $G = (V, E, d)$, whose edges are weighed by a distance function $d : E \longrightarrow \mathbb{R}_+$, into a metric space $(\mathcal{M}, d_{\mathcal{M}})$ by preserving distances. That is, the goal of the DGP is to determine one-to-one applications $\mathbf{x}$ which maps G into $\mathcal{M}$ so that

$$\|\mathbf{x}_u - \mathbf{x}_v\|_{\mathcal{M}} \triangleq \|\mathbf{x}(u) - \mathbf{x}(v)\|_{\mathcal{M}} = d(\{u, v\}) \triangleq d_{u,v}, \quad \forall \{u, v\} \in E, \qquad (1)$$

assuming that $\|\cdot\|_{\mathcal{M}}$ is the norm provided by $d_{\mathcal{M}}$. Also, Saxe showed that the computational complexity of the this class of problems for an Euclidean metric space $\mathbb{R}^K$ is strongly **NP**–Complete for $K = 1$ and strongly **NP**–Hard for $K > 1$ (in this context, DGP is called *Embeddability Problem*) [1].

Applications in $\mathbb{R}^K$ have been arising in several areas of knowledge lately. Examples for $K \leq 3$, which are the most of them, include *Clock Syncronization Problem (CSyP)* ($K = 1$), *Sensor Network Localization Problem* (SNLP) ($K = 2$) and the position-analysis of the Assur Kinematic Chains ($K = 3$) [2, 3]. Also, some progress have been done for problems for $K >> 3$, specially, for the *Graph Embedding Subproblem (GES)* which arises in many areas of Machine Learning, including deep neural networks [4].

The pioneer application, widely spread, consists on finding conformations of molecules in $\mathbb{R}^3$ [5]. So, the *Molecular Distance Geometry Problem (MDGP)* [6] is derived, a subclass of the Euclidean DGP, to deal with this.

The present work deals with the *Discretizable Molecular Distance Geometry Problem* (DMDGP) with exact distances, a subclass of the MDGP whose vertices are displayed in a suitable order which guarantees that the search space can be modelled as a binary tree, *i.e.*, it always has a discrete search space. Therefore, such tree can be exploited using the Branch-and-Prune (BP) algorithm, a depth-first search method to find all feasible configurations for G in $\mathbb{R}^3$. The core of BP consists of a sequential homogeneous-matrix product which is stated using a translation and two reflections.

It is also known that Quaternion Algebra $\mathbb{H}$ provides a suitable environment to make rotations [7]. So, this paper aims to find the configurations of a protein as a DMDGP instance by using quaternion rotations in the core of BP in order to show that this makes the number of operations decrease. In the practice with proteins, some distances are given as real intervals (noisy data), because physical and chemical machinery produce errors in their measures [8, 9]. But, this paper focus to improve BP for exact cases, since all the interval models that uses BP in Euclidean Geometry (interval BP) reduce to the exact cases by sampling the interval. Some attempts to use other geometry models have been made, but yet computational expensive [10, 11].

The present work is organized as follows. Sec. 2 describes all the features of the DMDGP, the Branch-and-Prune algorithm and how proteins fit into this model according to the results from literature. Sec. 3 is the one which brings the real contribution of this paper, a quaternionic version of BP. Finally, Sec. 4 shows all computational issues and the results which validate the proposal and conclusions and future steps are driven from Sec. 5.

2. DMDGP, Branch-and-Prune and Proteins

2.1. Graph Rigidity and Vertex Ordering

Each function $\mathbf{x} : V \longrightarrow \mathbb{R}^3$ is named a *Realization* of G, which is said to be *valid*, if Equations (1) are all satisfied. In addition, the pair $(G, \mathbf{x})$ is, then, known as a *Framework* [12]. A piece of the theory of *Graph Rigidity* follows, based on Jackson *et al.* [13]. Two frameworks $(G, \mathbf{x})$ and $(G, \mathbf{y})$ are said to be *equivalent*, if $\|\mathbf{x}_u - \mathbf{x}_v\| = \|\mathbf{y}_u - \mathbf{y}_v\|$ (for all $\{u, v\} \in E$), which is denoted as $(G, \mathbf{x}) \sim (G, \mathbf{y})$, and *congruent*, if $\|\mathbf{x}_u - \mathbf{x}_v\| = \|\mathbf{y}_u - \mathbf{y}_v\|$ (for all $u \neq v \in V$), which is denoted as $(G, \mathbf{x}) \equiv (G, \mathbf{y})$. There are two important remarks: (i) in the context of distance-weighted graphs, equivalent means isometric [12]; and (ii) any congruence is a composition of reflections, translations and/or rotations. We also say that $(G, \mathbf{x})$ is a *Rigid Framework* if it is congruent to any other framework $(G, \mathbf{y})$, where $\mathbf{y}$ is another realization for G. Now, let $(G, \mathbf{x})$ be a framework in $\mathbb{R}^3$, where $|V| = n$ and $|E| = m$, and $R\lambda = 0$ be the homogeneous linear system such that $\lambda \in \mathbb{R}^{3n}$ and $R \in \mathbb{R}^{m \times 3n}$, where each row is assigned to an edge $\{u, v\} \in E$ and the exactly six non-zero entries are given by $\mathbf{x}_i(u) - \mathbf{x}_i(v)$ and $\mathbf{x}_i(v) - \mathbf{x}_i(u)$, for $i = 1, 2, 3$, as $\mathbf{x}_i(u)$ denotes the i-th coordinate of $\mathbf{x}_u$ in $\mathbb{R}^3$. If the solutions of $R\lambda = 0$ are only translations and rotations, such framework is called *infinitesimally rigid* and, therefore, a *rigid graph* is such that it has an infinitesimally rigid framework [14]. And, modulo congruences, a DGP graph G is said to be *globally rigid* whether it has only one valid realization $\mathbf{x}$ [12].

Furthermore, Lebrecht Henneberg rose the question of using vertex orders in a graph to study rigidity in 1886 [15], which proved to be imperative to deal with problems in this area [16]. An *order* in a graph $G = (V, E)$ is given by a bijective function $\rho : V \longrightarrow \{1, 2, \ldots, |V|\}$, inducing the existence of a total order relation $<$ in V. Also, the number $\rho(v)$ is named the *rank* of vertex v. So, given an order for a DGP graph, it is known that if there is a same-number of adjacent predecessors for each vertex, such number is intimately connected to the cardinality of the solution set for a DGP, as described in [17].

2.2. The Discretizable Molecular Distance Geometry Problem

As the aim is to deal with proteins and we do not have a trilateration order for protein graphs in general [18], this paper will deal with the order associated to the *Discretizable Molecular Distance Geometry Problem* (DMDGP).

The DMDGP is a subclass of the DGP for whom there exists a total order relation ($<$) in V such that it satisfies the following assumptions:

(i) (discretization)
every pair of vertices $u, v \in V$, respectively with ranks i and j given from $<$, that meets $1 \leq |i-j| \leq 3$ determines an edge $\{u, v\} \in E$ and

(ii) (non-collinearity)
the distance values among each triplet of consecutive vertices with ranks $i-2, i-1$ and i, according to $<$, satisfy the strict triangular inequality

$$d_{i-2,i} < d_{i-2,i-1} + d_{i-1,i}, \qquad \text{for all } i \geq 3. \tag{2}$$

Assumption (i) asserts that each point in the realization lies, at least, in the intersection of two spheres and Assumption (ii) guarantees that the centers of such spheres are not collinear. As it is represented in Figure 1, both of the assumptions together imply that there are two possible feasible positions for each vertex at most [18]. Aiming to be practical, the the edge set is splitted into $E = E_D \cup E_P$, where $E_D := \{\{i, j\} \in E : 1 \leq |i-j| \leq 3\}$ stores the *Discretization Edges* and $E_P = E \setminus E_D$ allocates the *Pruning Edges*, whose names will make sense later on.

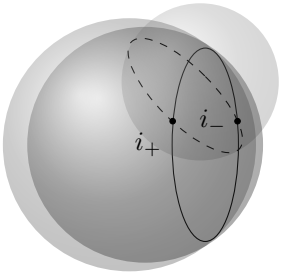

Figure 1: Intersection of three spheres with non-collinear centers.

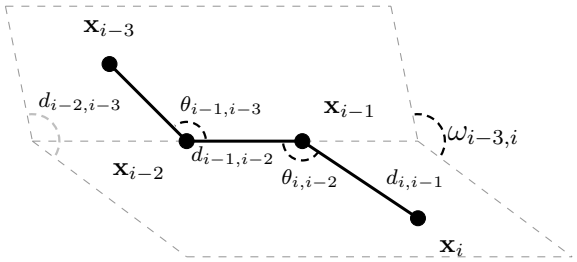

Figure 2: DMDGP fundamental quatruplet and internal coordinates.

2.3. The classic version of the Branch-and-Prune algorithm

The *Branch-and-Prune* (BP) algorithm is a combinatorial method which is able to find all possible configurations of a DMDGP graph $G = (V, E, d)$ with $|V| = n$, modulo translations/rotations, using distances and angles.

This section is based on the developments of BP which are displayed in Lavor *et al.* [18]. An important remark is that the authors ask for excuses

for detailing all the transformation issues, but it will be very important for understanding the geometric behavior in order to apply quaternions.

Now, for describing BP machinery, let us denote v_i as the vertex of rank $\rho(v_i) = i$. From the available discretization edges E_D, the so-called *internal coordinates* for each vertex can be calculated. All the computations during BP execution uses those data, whose names are inspired in molecular issues. They are described below: for $i > 4$, each quadruplet $\{i-3, i-2, i-1, i\}$ (depicted in Figure 2) determines

- the *bond lengths* are the given distances $d_{i-1,i}$, that is, the lengths of the edges consisting of consecutive vertices $\{v_{i-1}, v_i\}$, for $i = 2, \ldots, n$;
- the *bond angles* $\theta_{i-2,i}$ are defined between the pair of consecutive bonds $\{v_{i-2}, v_{i-1}\}$ and $\{v_{i-1}, v_i\}$, for $i = 3, \ldots, n$, which can be calculated by

$$\theta_{i-2,i} = \cos^{-1}\left(\frac{d_{i-2,i-1}^2 + d_{i-i,i}^2 - d_{i-2,i}^2}{2d_{i-2,i-1}d_{i-1,i}}\right); \tag{3}$$

- and the *torsion angles* $\omega_{i-3,i}$ are defined by the respective planes determined by $v_{i-3}, v_{i-2}, v_{i-1}$ and v_{i-2}, v_{i-1}, v_i, for $i = 4, \ldots, n$.

$$\omega_{i-3,i} = \cos^{-1}\left(\frac{2d_{i-2,i-1}^2\left(d_{i-3,i-2}^2 + d_{i-2,i}^2 - d_{i-3,i}^2\right) - (\tilde{d}_{i-3,i-1})(\tilde{d}_{i-2,i})}{\sqrt{\left(4d_{i-3,i-2}^2 d_{i-2,i-1}^2 - \tilde{d}_{i-3,i-1}^2\right)\left(4d_{i-2,i-1}^2 d_{i-1,i}^2 - \tilde{d}_{i-2,i}^2\right)}}\right), \tag{4}$$

where

$$\tilde{d}_{i-3,i-1} = d_{i-3,i-2}^2 + d_{i-2,i-1}^2 - d_{i-3,i-1}^2 \quad \text{and} \quad \tilde{d}_{i-2,i} = d_{i-2,i-1}^2 + d_{i-i,i}^2 - d_{i-2,i}^2.$$

The search space for G can be designed as a binary tree $\mathcal{T}(G)$ and BP is able to explore it using combinatorial tools, as explained in the following. Let us take $\mathbf{x} : V \to \mathbb{R}^3$ as a conformation for G, which is associated to a path in $\mathcal{T}(G)$, from the root to the leaf. And to find it, BP performs a depth-first search in this tree by undergoing three stages.

(i) *Initialization Stage*

In the initial step, BP fixes the first plane by determining the first three vertices in unique places, according to [19]. It represents the unique beginning for all configurations modulo translations and/or rotations. As in Figure 3, some characteristics have been observed:

- the position of vertex v_1 is considered as the origin of a local Cartesian frame $x_1y_1z_1$ such that the position of vertex v_2 is defined in the negative x_1-axis and the position of vertex v_3 lies in the first or second quadrant of the x_1y_1-plane;
- the position of vertex v_2 is also the origin of a coordinate system $x_2y_2z_2$, whose negative x_2-axis passes through the position of vertex v_1 and, again, the position of vertex v_3 lies in the first or second quadrant of the x_2y_2-plane;
- at last, the position of vertex v_3 is the origin of a coordinate frame $x_3y_3z_3$, whose negative x_3-axis passes through the position of vertex v_2 and the position of vertex v_1 lies on the third or fourth quadrant of x_3y_3-plane.

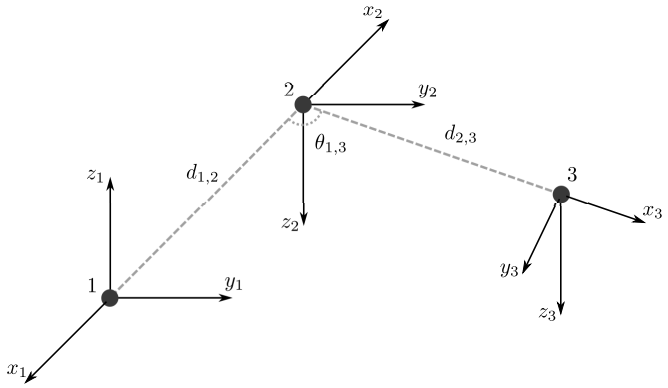

Figure 3: The first three vertices are univocally determined - reference frame

Therefore, considering all the first three vertices in the same frame, the positions can be calculated by walking in one of the feasible paths of the tree from the root to the third level as follows:

(a) the first vertex is placed in the origin $\mathbf{x}_1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$;

(b) the first position is transformed into the second position by a translation by the vector $\mathbf{t}_2 = \begin{bmatrix} d_{1,2} & 0 & 0 \end{bmatrix}^T$, a rotation in $\theta = \pi$ around the axis spanned by the canonical vector $\mathbf{k}$ and a rotation in $\omega = \pi$ about the axis spanned by the canonical vector $\mathbf{i}$. Thus, the second vertex lies in the position $\mathbf{x}_2 = \begin{bmatrix} -d_{1,2} & 0 & 0 \end{bmatrix}^T$;

(c) and, the second position can be mapped into the third position by the sequence of a translation with the vector $\mathbf{t}_3 = \begin{bmatrix} d_{2,3} & 0 & 0 \end{bmatrix}^T$, a rotation in $\theta = \pi - \theta_{1,3}$, around the axis spanned by $\mathbf{k}$, and a rotation in $\omega = 0$ around the axis spanned by $\mathbf{i}$. Therefore, the third vertex lie in the position

$$\mathbf{x}_3 = \begin{bmatrix} -d_{1,2} + d_{2,3}\cos(\theta_{1,3}) & d_{2,3}\sin(\theta_{1,3}) & 0 \end{bmatrix}^T.$$

All the possible configurations of G, then, have the same positions for the first three vertices.

(ii) *Branching Stage*

From the fourth vertex on ($i > 4$), the tree starts to branch: discretization assumption guarantees that there are two possibilities $\mathbf{x}_i^0$ and $\mathbf{x}_i^1$ for v_i, considering the parent node in the v_{i-1}–layer as fixed and getting the positions for the three immediate predecessors in the same path as determined. As previously discussed, both positions lie in the intersection of the three spheres $S_{i-3,i}, S_{i-2,i}$ and $S_{i-1,i}$ with centers in the already fixed positions $\mathbf{x}_{i-3}, \mathbf{x}_{i-2}$ and $\mathbf{x}_{i-1}$ and radii $d_{i-3,i}, d_{i-2,i}, d_{i-1,i} \in d(E_D)$, respectively.

Therefore, they satisfy the quadratic system

$$\begin{cases} \|\mathbf{x}_i^s - \mathbf{x}_{i-3}\|^2 &= d_{i-3,i}^2 \\ \|\mathbf{x}_i^s - \mathbf{x}_{i-2}\|^2 &= d_{i-2,i}^2 \\ \|\mathbf{x}_i^s - \mathbf{x}_{i-1}\|^2 &= d_{i-1,i}^2 \end{cases}, \quad (\text{for } s = 0 \text{ or } 1). \tag{5}$$

Instead of using iterative methods to approximate solutions for nonlinear equation system (5), as usual, BP takes advantage in the recursive structure of the DMDGP to find both positions using a discrete strategy, a composition of a translation and two rotations, based on [19, 18], very similar to what has been done in the Initialization Stage (depicted in Figure 4). A $x_iy_iz_i$-frame is defined such that the position of v_{i+1} lies in the negative x_i-axis and v_{i+2} lies in the the third or fourth quadrant of the x_iy_i-plane. So, a general position $\mathbf{x}_{i-1} = \begin{bmatrix} x_{i-1} & y_{i-1} & z_{i-1} \end{bmatrix}^T$ in the $x_{i-1}y_{i-1}z_{i-1}$-frame for v_{i-1} can walk to a general position $\mathbf{x}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T$ in the $x_iy_iz_i$-frame for v_i by the application of the following ordered sequence of transformations:

(a) a translation by the vector $\mathbf{t}_i = \begin{bmatrix} d_{i-1,i} & 0 & 0 \end{bmatrix}^T$;

(b) a planar rotation P_i^θ in $\theta = \pi - \theta_{i-2,i}$ about $span\{\mathbf{k}\}$ (counter-clockwise);

(c) and, a spatial rotation E_i^ω in $\omega = \omega_{i-3,i}$ about $span\{\mathbf{i}\}$ (counter-clockwise).

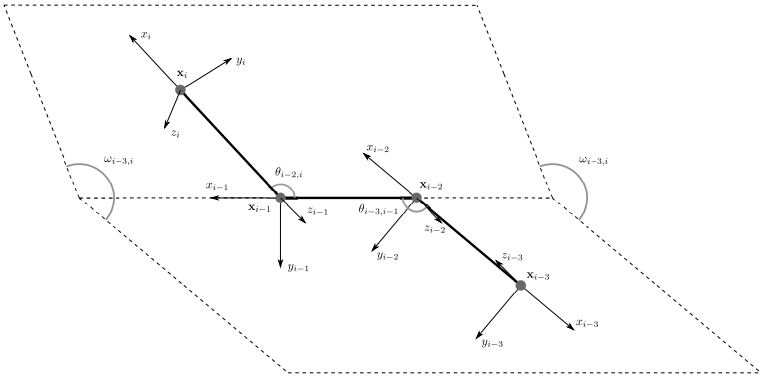

Figure 4: To include a caption

Therefore, this transformation can be summed up in the evaluation of the following composition

$$\mathbf{x}_i = E_i^\omega \circ P_i^\theta \left(\mathbf{x}_{i-1} + \mathbf{t}_i \right). \tag{6}$$

The spatial and the planar rotations can be represented using *Givens Rotations* [20], respectively, by the orthogonal matrices

$$\mathbf{E}_i(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \mathrm{c}_\omega & -\mathrm{s}_\omega \\ 0 & \mathrm{s}_\omega & \mathrm{c}_\omega \end{bmatrix} \quad \text{and} \quad \mathbf{P}_i(\theta) = \begin{bmatrix} \mathrm{c}_\theta & -\mathrm{s}_\theta & 0 \\ \mathrm{s}_\theta & \mathrm{c}_\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{7}$$

where s_ω, c_ω, s_θ and c_θ respectively stand for $\sin(\omega)$, $\cos(\omega)$, $\sin(\theta)$ and $\cos(\theta)$. And, by trigonometric identities, as

$$\begin{cases} \mathrm{s}_\theta &= \sin(\pi - \theta_{i-2,i}) &= \sin(\theta_{i-2,i}) &=: \mathrm{s}_\theta^i \\ \mathrm{c}_\theta &= \cos(\pi - \theta_{i-2,i}) &= -\cos(\theta_{i-2,i}) &=: -\mathrm{c}_\theta^i \end{cases},$$

hence, the matrices in (7) get

$$\mathbf{E}_i(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \mathrm{c}_\omega^i & -\mathrm{s}_\omega^i \\ 0 & \mathrm{s}_\omega^i & \mathrm{c}_\omega^i \end{bmatrix} \quad \text{and} \quad \mathbf{P}_i(\theta) = \begin{bmatrix} -\mathrm{c}_\theta^i & -\mathrm{s}_\theta^i & 0 \\ \mathrm{s}_\theta^i & -\mathrm{c}_\theta^i & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

Thus, $\mathbf{x}_i$ can be easily computed from $\mathbf{x}_{i-1}$ by the product of

$$\mathbf{x}_i = \mathbf{E}_i(\omega)\mathbf{P}_i(\theta)\left(\mathbf{x}_{i-1} + \mathbf{t}_i\right). \tag{9}$$

In favor of better computing the whole conformation $\mathbf{x}$ by concatenating its iterations using Equation (9), homogeneous coordinates can be used (vectors have h sign overwritten). That it, $\mathbb{R}^3$ is merged as the Projective Hyperplane into the *Projective Space* $\mathbb{RP}^3 = \mathbb{R}^4 \setminus \{o\}$ (where o denotes the origin) [21], since translations can always be put in matrix terms. In the case of Equation (9), $\mathbf{x}_{i-1} + \mathbf{t}_i$ can be written as

$$\mathbf{x}^h_{i-1} + \mathbf{t}^h_i := \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ 1 \end{bmatrix} + \begin{bmatrix} d_{i-1,i} \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & d_{i-1,i} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ 1 \end{bmatrix} =: \mathbf{T}_i \mathbf{x}^h_{i-1}.$$

Consequently, the 4×4–homogeneous–matrix version of Eq. (9) gets

$$\begin{aligned} \mathbf{x}^h_i &:= \mathbf{E}^h_i(\omega)\mathbf{P}^h_i(\theta)\mathbf{T}_i\mathbf{x}^h_{i-1} \\ &:= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \mathrm{c}^i_\omega & -\mathrm{s}^i_\omega & 0 \\ 0 & \mathrm{s}^i_\omega & \mathrm{c}^i_\omega & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\mathrm{c}^i_\theta & -\mathrm{s}^i_\theta & 0 & 0 \\ \mathrm{s}^i_\theta & -\mathrm{c}^i_\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & d_{i-1,i} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ 1 \end{bmatrix}. \end{aligned}$$

Performing the products, one gets a unique transformation matrix

$$\mathbf{B}_i = \mathbf{E}^h_i(\omega)\mathbf{P}^h_i(\theta)\mathbf{T}_i = \begin{bmatrix} -\mathrm{c}^i_\theta & -\mathrm{s}^i_\theta & 0 & -d_{i-1,i}\mathrm{c}^i_\theta \\ \mathrm{s}^i_\theta\mathrm{c}^i_\omega & -\mathrm{c}^i_\theta\mathrm{c}^i_\omega & -\mathrm{s}^i_\omega & d_{i-1,i}\mathrm{s}^i_\theta\mathrm{c}^i_\omega \\ \mathrm{s}^i_\theta\mathrm{s}^i_\omega & -\mathrm{c}^i_\theta\mathrm{s}^i_\omega & \mathrm{c}^i_\omega & d_{i-1,i}\mathrm{s}^i_\theta\mathrm{s}^i_\omega \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{10}$$

which is widely known in literature, specially in chemistry/physics applications [19, 22, 23, 24]. Thus, the generic position for $\mathbf{x}_i$ is driven from the previously determined position $\mathbf{x}_{i-1}$ and from the internal coordinates and it can be easily extracted from the formula

$$\begin{aligned} \mathbf{x}^h_i &= \mathbf{B}_i\mathbf{x}^h_{i-1} \\ \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} &= \begin{bmatrix} -\mathrm{c}^i_\theta & -\mathrm{s}^i_\theta & 0 & -d_{i-1,i}\mathrm{c}^i_\theta \\ \mathrm{s}^i_\theta\mathrm{c}^i_\omega & -\mathrm{c}^i_\theta\mathrm{c}^i_\omega & -\mathrm{s}^i_\omega & d_{i-1,i}\mathrm{s}^i_\theta\mathrm{c}^i_\omega \\ \mathrm{s}^i_\theta\mathrm{s}^i_\omega & -\mathrm{c}^i_\theta\mathrm{s}^i_\omega & \mathrm{c}^i_\omega & d_{i-1,i}\mathrm{s}^i_\theta\mathrm{s}^i_\omega \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ 1 \end{bmatrix}. \end{aligned} \tag{11}$$

All this development aims to show that one can determine one of the feasible positions for the i-th vertex just by knowing the internal coordinates, which can be extracted from the homogeneous-matrix product

$$\mathbf{x}_i^h = \mathbf{B}_1\mathbf{B}_2\mathbf{B}_3\mathbf{B}_4\ldots\mathbf{B}_{i-1}\mathbf{B}_i\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T, \tag{12}$$

all of the positions from vertex v_1 to vertex v_i belong to the same frame.

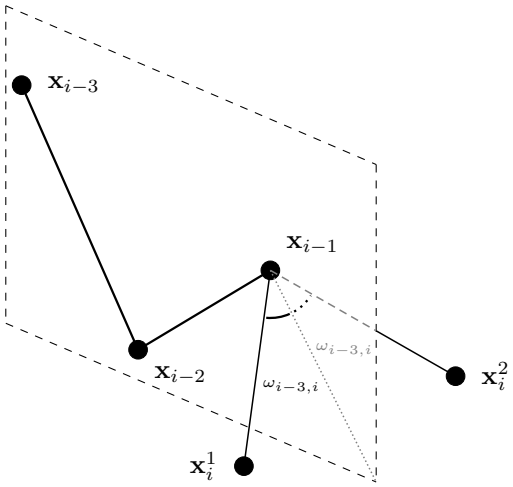

Figure 5: Possibilities $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$ form the same angle $\omega_{i-3,i}$ with the plane established by the uniquely already fixed positions $\mathbf{x}_{i-3}, \mathbf{x}_{i-2}$ and $\mathbf{x}_{i-3}$.

Now, the matricial model in (12) is supposed to be capable of representing all the same possibilities of positioning $\mathbf{x}_i$ than in (5) (which are two for each choice of three consecutive predecessor positions $\mathbf{x}_{i-3}, \mathbf{x}_{i-2}$ and $\mathbf{x}_{i-3}$). By the geometric hypotheses, the possible positions $\mathbf{x}_i^0$ and $\mathbf{x}_i^1$ are supposed to form the same angle $\omega_{i-3,i}$ with the plane determined by the unique already known positions $\mathbf{x}_{i-3}, \mathbf{x}_{i-2}$ and $\mathbf{x}_{i-3}$, as in Fig. 5.

In order to fullfil this, it is enough to realize that both positions are achieved just by taking angles $\omega_{i-3,i}$ and $-\omega_{i-3,i}$ in the spatial-rotation matrix $\mathbf{E}_i^h(\omega)$, respectively. It generates *two* homogeneous matrices

$$\mathbf{B}_i^0 = \mathbf{E}_i^h(\omega_{i-3,i})\mathbf{P}_i^h(\theta)\mathbf{T}_i \text{ and } \mathbf{B}_i^1 = \mathbf{E}_i^h(-\omega_{i-3,i})\mathbf{P}_i^h(\theta)\mathbf{T}_i \tag{13}$$

such that they get two possible positions for v_i by applying (12), yielding

$$\begin{bmatrix}\mathbf{x}_i^0\\1\end{bmatrix} = \mathbf{B}_1\mathbf{B}_2\ldots\mathbf{B}_{i-1}\mathbf{B}_i^0\begin{bmatrix}0\\0\\0\\1\end{bmatrix} \quad \text{and} \quad \begin{bmatrix}\mathbf{x}_i^1\\1\end{bmatrix} = \mathbf{B}_1\mathbf{B}_2\ldots\mathbf{B}_{i-1}\mathbf{B}_i^1\begin{bmatrix}0\\0\\0\\1\end{bmatrix},$$

where $\mathbf{B}_1$, $\mathbf{B}_2$ and $\mathbf{B}_3$ are always fixed: $\mathbf{B}_1$ is the identity matrix $n \times n$,

$$\mathbf{B}_2 = \begin{bmatrix}-1 & 0 & 0 & -d_{1,2}\\ 0 & 1 & 0 & 0\\ 0 & 0 & -1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad \text{and} \quad \mathbf{B}_3 = \begin{bmatrix}-\cos(\theta_{1,3}) & 0 & 0 & -d_{1,2}\\ 0 & 1 & 0 & 0\\ 0 & 0 & -1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix},$$

by using the information from item *(i)* in the matrix (10).

In summary, each possible branch, in the 2^{n-3} ones, can be iteratively determined by a choice of values for the Boolean variables $s, j_4, \ldots, j_n$

$$\begin{bmatrix}\mathbf{x}_i^s\\1\end{bmatrix} = \mathbf{B}_1\mathbf{B}_2\mathbf{B}_3\mathbf{B}_4^{j_4}\mathbf{B}_5^{j_5}\ldots\mathbf{B}_{i-1}^{j_{i-1}}\mathbf{B}_i^{j_i}\begin{bmatrix}0\\0\\0\\1\end{bmatrix}, \tag{14}$$

(iii) *Pruning*

Each determined position ought to undergo a feasibility check for deciding if it satisfies all the distance constraints in $d(E_P)$, since it satisfy $d(E_D)$ by construction. The most used feasibility device, that will be taken as standard in this paper, is the *Direct Distance Feasibility* (DDF) check: given a threshold $\varepsilon > 0$, we have got to check if

$$|\,\|\mathbf{x}_i - \mathbf{x}_j\| - d_{i,j}| < \varepsilon, \qquad \forall\{i,j\} \in E_P. \tag{15}$$

So, as the search space is a binary tree $\mathcal{T}(G)$ which is explored by a depth-first search, BP runs from the root to the leaf by taking always position *zero* for the current level. If it does not satisfy (15), the path is pruned, the search backtracks to the parent node (previous level in the tree) and position *one* is taken and so on.

2.4. Symmetry Vertices and BP-One

Let $\mathcal{X}(G)$ be the set of all valid realizations for G (modulo translations and rotations). The existence of a symmetry relation in $\mathcal{X}(G)$ was proved, based on the congruence relation from Subsection 2.1 – more details in [25, 26]. Two valid realizations are symmetric if, and only if, one can be transformed into another through partial reflections on the so-called *Symmetry Vertices*, stored in the set $S(G) = \{v \in V : \nexists \{u, w\} \in E \text{ such that } \rho(u) + 3 < \rho(v) \leq \rho(w)\}$. An important remark is that $4 \in S(G)$ always, which is called *fourth level symmetry*. And, for a practical computational handle, the Boolean structure of Eq. (14) provides a useful manner to find all conformations from the first to be found: just flip zeroes and ones from the symmetry vertex on (including it), what also gives the result that $|\mathcal{X}(G)| = 2^{sg}$, where $sg = |SG)|$.

As it is enough to find only one solution, the computational work for BP is softened and two options to BP are provided: (I) if the executor wants only one solution, he uses the *BP-One* (Alg. 1) – BP stops in the first solution found; (II) and, if it desires all solutions, he executes *symBP* algorithm – BP-One finds a solution and all the others are determined by using partial reflections in the symmetry vertices. Only a note, from these discoveries on, the classic BP to find all conformations for G is renamed for *BP-All*.

Algorithm 1 BP-One Algorithm

Require: $v \in V \setminus \{1, \ldots, n\}$ and an embedding $\overline{\mathbf{x}} = \mathbf{x}'$ for $G[\gamma_G(v)]$.

1: **function** BPONE(v, $\overline{\mathbf{x}}$)
2: $P \leftarrow \bigcap_{u \in N_G(v), u < v} S^{K-1}(\overline{\mathbf{x}}_u; d_{u,v})$
3: **for** $\mathbf{x}_v \in P$ **do**
4: $\mathbf{x} \leftarrow (\overline{\mathbf{x}}, \mathbf{x}_v)$
5: **if** $v = n$ **then**
6: **return** *success*, $\mathbf{x}$
7: **end if**
8: status, $\mathbf{y} \leftarrow$ BPONE($v + 1$, $\mathbf{x}$)
9: **if** status = *success* **then**
10: **return** *success*, $\mathbf{y}$
11: **end if**
12: **end for**
13: **return** *fail*
14: **end function**

In Fig. 6, a toy example is displayed with 20 vertices and three symmetry vertices in the shape of a tree after the execution of BP.

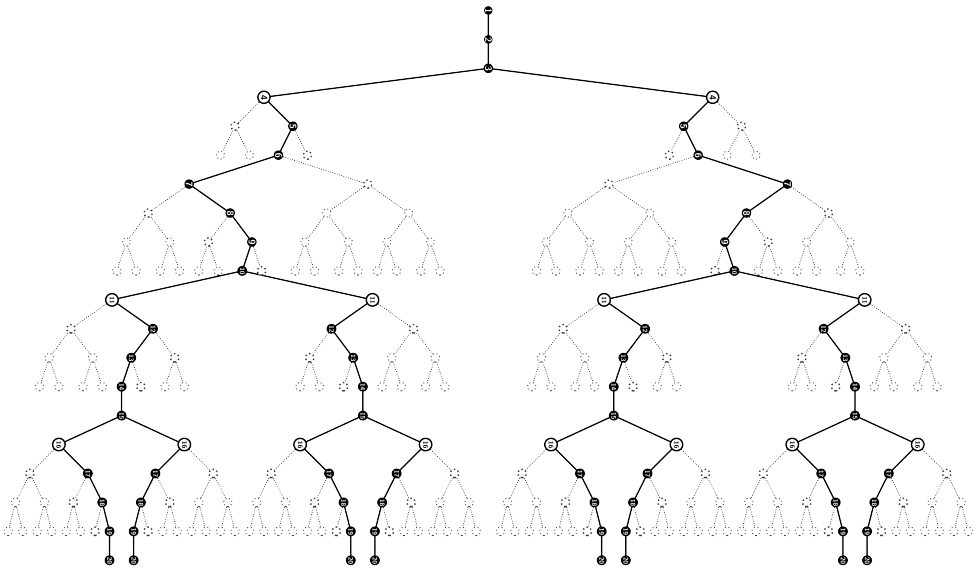

Figure 6: An exampling $\mathcal{T}_G$ of a DMDGP with $n = 20$ and $S(G) = \{4, 11, 16\}$, extracted from [27].

2.5. A hand-crafted order in a protein

To be written.

3. The Quaternionic version of the Branch-and-Prune algorithm

3.1. Quaternion Rotations

To be written.

3.2. Using Quaternions in the Branching Phase

To be written.

3.3. Quaternionic Branch-and-Prune

To be written.

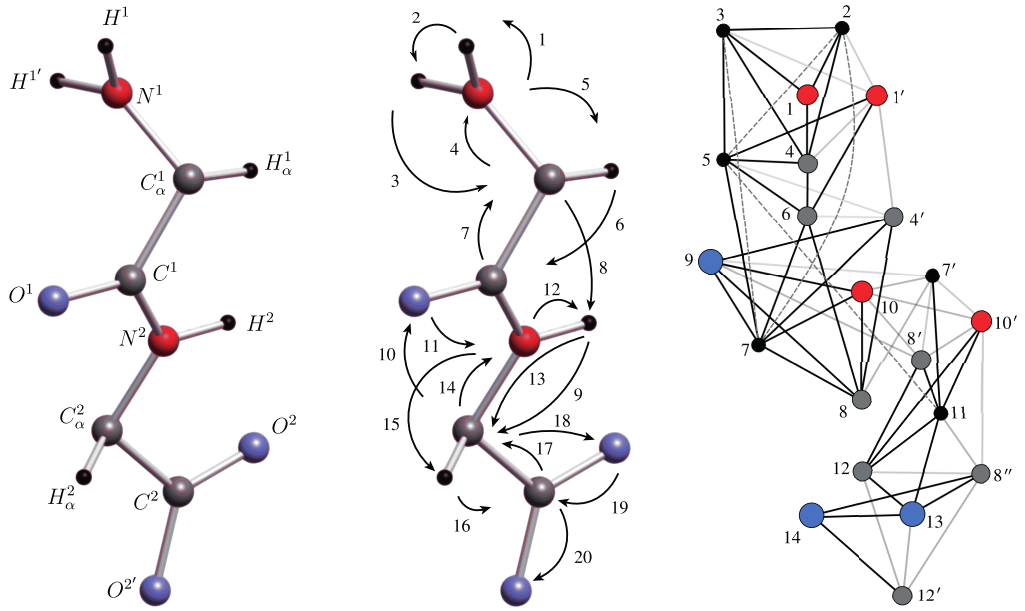

Figure 7: Figura a ser descrita depois

4. Computational Issues

To be written.

4.1. An implementation discussion

To be written.

4.2. Computational tests for artificial and real instances

In this section we will deal with the numerical aspects of the new proposal. In this sense, we divided our study into two parts:

- impact of operations on processing time;
- robustness and efficiency analysis.

Concerning the analysis of operations on processing we handle just with artificial examples. For the other point, we deal with both artificial and real. The implementations were made in Julia language [**?**] and are available in a Github repository [1]. The tests were performed in an Intel Pentium(R)

[1] https://github.com/evcastelani/MolecularConformation.jl

processor, CPU G3240 with two cores of 3.10GHz and SSD with 240GB running Ubuntu-Mate 64 bits, version 18.04 release 02.

4.3. Impact of operations on processing time

As previously explained, the number of operations performed using Quaternion is much smaller. Of course, this suggests that the implemented version of Branch and Prune becomes more efficient when using Quaternion rotations than using arrays rotations. In fact, this is true. However, practical situations show that these operational advantages can be minimized when faced with more problems of reasonable size (larger than 1000 atoms for example). The reason for this performance loss is an extra processing load in pruning testing.

To elucidate our comments, let's assume that the Algorithm X is exploit in the level $i = 1200$ of the recursive process used to find a conformation of 2000 atoms. Let's assume we have, say 1000 additional distances to test the feasibility. In this case, the number of operations in only one feasibility test will be of the order of $8 \cdot 1000 = 8000$ operations ($8 = 3$ differences $+ 3$ products $+ 2$ sums). Now, at this same level, 140 operations were performed using the matrix version and 65 operations using the Quaternion version which represents, respectively, 1.75% and 0.8125% of the total processing time at that level. As a result, it is simple to intuit that as the number of atoms increases, the processing time saved in rotations becomes a very small fraction of the total time spent. In order to illustrate our remark, we provide numerical tests in the following way. Given a distance array D, we create a new distance array denoted by $D(n_d)$ where $n_d \in \{3, 4, ..., n\}$

$$D(n_d)_{i,j} = \begin{cases} D_{i,j}, \text{ if } |i-j| \leq n_d \\ 0, \text{ otherwise} \end{cases} \tag{16}$$

and we run both algorithms X e Y to find just one solution using the matrix $D(n_d)$.

5. Conclusion and future works

Acknoledgements

GP would like to thank CNPq for financial support. FF and EVC thank, respectively, UFSC and UEM for the support.

References

[1] J. B. Saxe, Embeddability of weighted graphs in k-space is strongly np-hard, in: Proceedings of the 17th Allerton Conference in Communications, Control and Computing, University of Illinois, Monticello, USA, 1979, pp. 480–489.

[2] L. Liberti, C. Lavor, Euclidean Distance Geometry: an introduction, Springer Undergraduate Texts in Mathematics and Technology, Springer, New York, USA, 2017.

[3] N. Rojas, F. Thomas, Distance-based position analysis of the three seven-link assur kinematic chains, Mechanism and Machine Theory 46 (2) (2011) 112–126.

[4] L. Liberti, Distance geometry and data science, TOP accepted.

[5] G. Crippen, A novel approach to calculation of conformation: Distance geometry, Journal of Computational Physics 24 (1977) 96–107.

[6] G. Crippen, T. Havel, Distance Geometry and Molecular Conformation, Research Studies Press, 1988.

[7] J. B. Kuipers, Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality, Princeton University Press, Princeton, USA, 2002.

[8] M. Nilges, M. J. Macias, S. I. O'Donoghue, H. Oschkinat, Automated noesy interpretation with ambiguous distance restraints: the refined nmr solution structure of the pleckstrin homology domain from β-spectrin, Journal of Molecular Biology 269 (1997) 408–422.

[9] T. Schlick, Molecular modeling and simulation: an interdisciplinary guide, 2nd Edition, Vol. 21 of Interdisciplinary Applied Mathematics, Springer Science+Business Media, New York, USA, 2002.

[10] A. Cassioli, O. Gunluk, C. Lavor, L. Liberti, Discretization vertex orders in distance geometry, Discrete Applied Mathematics 137 (2015) 27–41.

[11] C. Lavor, L. Liberti, A. Mucherino, The interval branch-and-prune algorithm for the discretizable molecular distance geometry problem with inexact distances, Journal of Global Optimization 56 (2013) 855–871.

[12] C. Lavor, L. Liberti, B. Donald, B. Worley, B. Bardiaux, T. Malliavin, M. Nilges, Minimal nmr distance information for rigidity protein graphs, Discrete Applied Mathematics 256 (2019) 91–104.

[13] B. Jackson, T. Jordán, Connected rigidity matroids and unique realization of graphs, Journal of Combinatorial Theory Series B 94 (2005) 1–29.

[14] T.-S. Tay, W. Whiteley, Generating isostatic frameworks, Structural Topology 11 (1985) 20–69.

[15] L. Henneberg, Die Graphische Statik Der Starren Systeme, Nabu Press, Charleston, USA, 1886.

[16] H. Bodlaender, F. Fomin, A. Koster, D. Kratsch, D. Thilikos, A note on exact algorithms for vertex ordering problems on graphs, Theory of Computing Systems 50 (2012) 420–432.

[17] L. Liberti, C. Lavor, N. Maculan, A. Mucherino, Euclidean distance geometry and applications, SIAM Review 56 (1) (2014) 3–69.

[18] C. Lavor, L. Liberti, N. Maculan, A. Mucherino, The discretizable molecular distance geometry problem, Computational Optimization and Applications 52 (2012) 115–146.

[19] B. Thompson, Calculation of cartesian coordinates and their derivatives from internal molecular coordinates, Journal of Chemical Physics 47 (9) (1967) 3407–3410.

[20] G. Golub, C. F. Van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore, USA, 1996.

[21] A. Beutelspacher, U. Rosenbaum, Projective Geometry: from Foundations to Applications, Cambridge University Press, Cambridge, UK, 1998.

[22] H. Eyring, The resultant electric moment of complex molecules, Physical Review 39 (1932) 746–748.

[23] R. L. McCullough, P. E. McMahon, Contributions to conformational energy from interactions between nonbonded atoms and groups. part1

- general formulation, Transactions of Faraday Society 60 (1964) 2089–2096.

[24] T. Shimanouchi, S.-i. Mizushima, On the helycal configuration of a polymer chain, Journal of Chemical Physics 23 (1955) 707–711.

[25] A. Mucherino, C. Lavor, L. Liberti, Exploiting symmetry properties of the discretizable molecular distance geometry problem, Journal of Bioinformatics and Computational Biology 10 (3) (2012) 1–15.

[26] L. Liberti, B. Masson, J. Lee, C. Lavor, A. Mucherino, On the number of realizations of certain henneberg graphs arising in protein conformation, Discrete Applied Mathematics 165 (2014) 213–232.

[27] F. Fidalgo, D. S. Gonçalves, C. Lavor, L. Liberti, A. Mucherino, A symmetry-based splitting strategy for discretizable distance geometry problems, Journal of Global Optimization 71 (2018) 717–733.