Noname manuscript No.
(will be inserted by the editor)

A new algorithm for the DMDGP subclass of Distance Geometry problems

Douglas S. Gonçalves* · Carlile Lavor · Leo Liberti · Michael Souza

Received: date / Accepted: date

Abstract The fundamental inverse problem in distance geometry is the one of finding positions from inter-point distances. The Discretizable Molecular Distance Geometry Problem (DMDGP) is a subclass of the Distance Geometry Problem (DGP) whose search space can be discretized and represented by a binary tree, which can be explored by a Branch-and-Prune (BP) algorithm. It turns out that this combinatorial search space possesses many interesting symmetry properties that were studied in the last decade. In this paper, we present a new algorithm for this subclass of the DGP, which exploits DMDGP symmetries more effectively than its predecessors. Computational results show that the speedup, with respect to the classic BP algorithm, is considerable for sparse DMDGP instances related to protein conformation.

Keywords Distance Geometry · Discretization · Symmetries · DMDGP

This is an extended version of an abstract published at CTW2020.

D. S. Gonçalves* (Corresponding author)
CFM, Federal University of Santa Catarina, 88.040-900, Florianópolis, Brazil
E-mail: douglas.goncalves@ufsc.br

C. Lavor
IMECC, University of Campinas, 13081-970, Campinas, Brazil
E-mail: clavor@ime.unicamp.br

L. Liberti
LIX CNRS École Polytechnique, Institut Polytechnique de Paris, 91128, Palaiseau, France
E-mail: liberti@lix.polytechnique.fr

M. Souza
Federal University of Ceará, 60440-900, Fortaleza, Brazil
E-mail: michael@ufc.br

1 Introduction

Given a simple, undirected, weighted graph $G = (V, E, d)$, with weight function $d : E \to (0, \infty)$ and an integer $K > 0$, the *Distance Geometry Problem* (DGP) consists in finding a *realization* $x : V \to \mathbb{R}^K$ such that,

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

where $|| \cdot ||$ denotes the Euclidean norm, $x_v := x(v), \forall v \in V$ and $d_{uv} := d(\{u, v\}), \forall \{u, v\} \in E$. Each equation in (1) is called a distance constraint. We say that a realization x satisfies d_{uv} if the corresponding distance constraint is verified. A realization satisfying all distance constraints in (1) is called a *valid realization.* We shall call a pair (G, K) a DGP instance.

There are many applications of Distance Geometry, mainly related to $K \in \{1, 2, 3\}$ [3,4,27]. An application to Data Science can be found in [16], and a very recent survey on this subject is [15]. An important class of the DGP arises in the context of 3D protein structure calculations ($K = 3$), with distance information provided by Nuclear Magnetic Resonance (NMR) experiments [6, 24,31].

Existence and uniqueness of DGP solutions, among other theoretical aspects of the problem, are discussed in [19]. Henceforth, we will consider that the DGP admits a solution.

Assumption 1 *The solution set of* (1) *is non-empty.*

The DGP is naturally cast as a search in continuous space. Depending on the graph structure, however, combinatorial search algorithms can be defined, notably via the identification of appropriate vertex orders [5,11,14]. Although DGP is NP-hard [30], these combinatorial approaches allowed to show that it is Fixed Parameter Tractable (FPT) on certain graph structures, as those arising in protein conformation [20].

The aforementioned vertex orders define a DGP subclass, called the *Discretizable Molecular Distance Geometry Problem* (DMDGP) [12,13], formally given as follows.

Definition 1 A DGP instance (G, K) is a ^{K}DMDGP if there is a vertex order $v_1, ..., v_n \in V$, such that

1. $G[\{v_1, ..., v_K\}]$ is a clique;
2. (a) For every $i > K$, v_i is adjacent to $v_{i-1}, , ..., v_{i-K}$,
   (b) $CM(v_{i-1}, ..., v_{i-K}) \neq 0$.

In the above definition, $G[\cdot]$ denotes the induced subgraph and $CM(v_{i-1}, ..., v_{i-K})$ is the Cayley-Menger determinant of $v_{i-1}, ..., v_{i-K}$ [19, Sec. 2]. Its squared value is proportional to the $(K-1)$-volume of a realization $x_{i-1}, \ldots, x_{i-K}$ for $v_{i-1}, ..., v_{i-K}$. Condition $CM(v_{i-1}, ..., v_{i-K}) \neq 0$ means that the points $x_{i-1}, \ldots, x_{i-K}$ span an affine subspace of dimension $K - 1$.

Although Definition 1 applies to any dimension K, therefore covering other applications rather than molecular conformation where $K = 3$, the term "molecular" is commonly kept in the related literature [12,19,5], regardless of the dimension, to enforce the property that the adjacent predecessors of v_i are *contiguous* (the term "contiguous K-lateration order" to mean ^{K}DMDGP is used in [5]), a desirable property when ordering atoms of a protein [12,11].

When the dimension K is clear from the context, we shall simply use DMDGP rather than ^{K}DMDGP. Moreover, without loss of generality, whenever we denote an edge by $\{v_i, v_j\} \in E$, we will assume that $i < j$, i.e v_i precedes v_j in the vertex order of Definition 1.

Properties 1 and 2(a) of Definition 1 says that G is composed by a chain of contiguous $(K+1)$-cliques. Moreover, properties 1 and 2 allow us to turn the search space into a binary tree, in the following way.

After fixing the positions for the first K vertices, for each new vertex v_i, with $i > K$, property 2(a) ensures that the possible positions x_i for vertex v_i lie in the intersection of K spheres centered at $x_{i-1}, \dots, x_{i-K}$ with radii $d_{i-1,i}, \dots, d_{i-K,i}$, respectively. Property 2(b) guarantees that there are at most two points, let us say $\{x_i^+, x_i^-\}$, in such intersection [23]. This spheres intersection can be computed in many different ways that we will not cover in this paper but are well studied in the literature [1,23].

Remark 1 The above process is known in the literature as K-lateration [19].

Thus, following the vertex order, after fixing the first K vertices, each new vertex has at most 2 possible positions, which of course depend on the position of its K immediate adjacent predecessors, leading to a binary tree of possible positions, where each path, from the root to a leaf node, corresponds to a *possible realization* for the graph G.

However, not all of these possible realizations (paths on the tree) are valid, because G may contain other edges $\{h, i\}$, with $|h - i| > K$, associated to distance constraints that are not satisfied by such realizations. The edges given in Definition 1 are called *discretization edges* and the others, that may be (or not) available, are called *pruning edges.*

Henceforth, let us partition $E = E_D \cup E_P$, where E_D is the set of discretization edges and E_P the set of pruning edges. Clearly, we can also partition the equations in (1) in discretization edge constraints and pruning edge constraints. We remark that, according to Definition 1, $E_D = \{\{i,j\} \in E \mid |i-j| \leq K\}$ and therefore $E_P = \{\{i,j\} \in E \mid |i-j| > K\}$.

The *Branch-and-Prune* (BP) algorithm [18] explores the DMDGP binary tree in a depth first manner and validates possible positions for vertices as soon as a pruning edge appears. A pseudo-code is given in Algorithm 1.

In Algorithm 1, the phrase "x_i^+ is feasible" means that the equations

$$\forall h \ : \ h < i, \{v_h, v_i\} \in E_P, \quad \|x_i^+ - x_h\| = d_{hi},$$

are satisfied up to a certain tolerance. In Step 5 positions x_i^+ and x_i^- are computed via K-lateration. See [19,1,23] for details.

Algorithm 1 BP

```
1: BP($i, n, G, x$)                                              # ($i > K$)
2: if ($i > n$) then
3:     return $x$
4: else
5:     Find solutions $\{x_i^+, x_i^-\}$ for the system: $\|x_\ell - x_i\|^2 = d_{\ell,i}^2, \ell = i-K, \ldots, i-1$.
6:     if $x_i^+$ is feasible then
7:         Set $x_i = x_i^+$ and call BP($i+1, n, G, x$).          # 1st candidate position
8:     end if
9:     if $x_i^-$ is feasible then
10:        Set $x_i = x_i^-$ and call BP($i+1, n, G, x$).          # 2nd candidate position
11:    end if
12: end if
```

Computational experiments in [12] showed that BP outperforms methods based on global continuation [25] and semidefinite programming [10] on instances of the DMDGP subclass, suggesting BP as the method of choice for this subclass of DGPs.

In addition to the discretization of the DGP search space, the DMDGP order also implies *symmetry* properties of such discrete space [21,17,29]. From the computational point of view, one of the most important of such properties, in the context of this paper, is that all DMDGP solutions can be determined from just one solution. This property is related to the DMDGP symmetry vertices, which can be identified *a priori*, based on the input graph (see next section). Once a first solution is found, the others can be obtained by partial reflections of the first, based on symmetry hyperplanes associated to these vertices.

Previous works [26,20] exploited symmetry to reconstruct all valid realizations from the first one found and to prove that the BP algorithm is fixed-parameter tractable. Others [9,8], considered decomposition-based variants of BP which leverage DMDGP symmetry information.

In this work, we exploit DMDGP symmetry in order to find the first valid realization more quickly. We handle the DMDGP as a sequence of nested subproblems, each one defined by a pruning edge $\{i, j\} \in E_P$. For each subproblem, we can exploit *any* realization x (valid or not) for building the symmetry hyperplanes (which will define partial reflections). Once we have them, we apply compositions of such partial reflections *only* to x_j to find its correct position. Only after finding the correct combination of partial reflections do we use it to obtain the positions of other vertices. After a subproblem is solved, the set of valid partial reflections is reduced and a single symmetry hyperplane is enough to handle positions $x_{i+K}, \ldots, x_j$ in the next subproblem.

In terms of the system of nonlinear equations (1), we solve a subset of equations and then gradually include new equations to this subset: the new equations are solved subject to the original equations in the subset. This process is repeated until all equations in (1) are satisfied.

These ideas lead to a new algorithm which deals with pruning edges, one-by-one, and takes advantage of a valid realization for already solved subproblems. Computational results illustrate the advantage of the new algorithm, compared with the classic BP.

This paper is organized as follows. Section 2 briefly reviews the main results about DMDGP symmetries and Section 3 explains how they can be used to solve a sequence of nested subproblems. The new algorithm, its correctness and implementation details are presented in Section 4, and comparisons with the classic BP in protein-like instances are given in Section 5. Concluding remarks are given in Section 6.

2 DMDGP symmetries

Before discussing the new algorithm, we shall present a theoretical background on DMDGP symmetries and recall some results from the literature [20,22,19].

Given a realization x satisfying (1), it is clear that there are uncountably many others, which satisfy the same set of distances, and which can be obtained by translations, rotations or reflections of x (because these transformations preserve all pairwise distances). Since the assumptions of Definition 1 ensure that the first K vertices form a clique, a valid realization for $G[v_1, \dots, v_K]$ in $\mathbb{R}^K$ can be found by matrix decomposition methods [7] or a sequence of spheres intersections [1], for example. Once the positions of these first K vertices are fixed, the degrees of freedom of translations and rotations are removed.

From here on, we say that two realizations are incongruent (modulo translations and rotations) if they are not translations, rotations or total reflections of each other. For technical reasons, we only allow the total reflections through the hyperplane defined by the positions of the first K vertices in the vertex order (so two realizations, one of which is a reflection of the other through this hyperplane, will both be considered members of any set of "incongruent realizations").

Definition 2 Let $\hat{X}$ be the set of all incongruent realizations satisfying distance constraints associated to discretization edges in E_D, i.e., $\|x_i - x_j\| = d_{ij}$ such that $|i-j| \leq K$. A realization $x \in \hat{X}$ is called a possible realization.

As discussed in Section 1, each $x \in \hat{X}$ corresponds to a path from the root to a leaf node in the binary tree of a DMDGP instance. Notice that $|\hat{X}| = 2^{|V|-K}$.

Definition 3 A realization $x \in \hat{X}$ is said to be valid if x is a solution of (1). Let $X \subset \hat{X}$ denote the set of all incongruent valid realizations of a DMDGP instance.

The computational experiments in [12] suggested that $|X|$ is always a power of 2. A conjecture was formulated and quickly disproved using some instances constructed by hand, until the conjecture was shown to be true with probability one in [22].

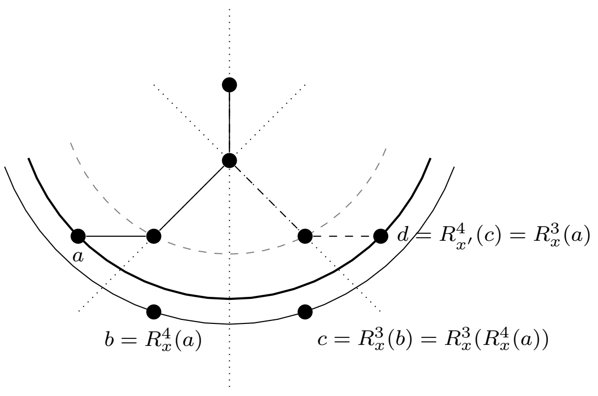
$d = R^4_{x'}(c) = R^3_x(a)$
a
$b = R^4_x(a)$
$c = R^3_x(b) = R^3_x(R^4_x(a))$

Fig. 1 The leftmost path/realization x is represented by a straight line whereas the rightmost x' by a dashed line. All 4 possible positions for the fourth vertex (denoted by a, b, c and d) can be generated by x and its induced reflections R^3_x and R^4_x. Illustration for $K = 2$.

Given $x \in \hat{X}$, for $i > K$, let $R^i_x(y)$ be the reflection of $y \in \mathbb{R}^K$ through the hyperplane defined by $x_{i-K}, \dots, x_{i-1}$, with normal p_i:

$$R^i_x(y) = (I - 2p_i p_i^T)(y - x_{i-1}) + x_{i-1},$$

assuming $\|p_i\| = 1$. Let us also define, for all $i > K$ and $x \in \hat{X}$, *partial reflection* operators:

$$g_i(x) = (x_1, x_2, \dots, x_{i-1}, R^i_x(x_i), R^i_x(x_{i+1}), \dots, R^i_x(x_n)). \tag{2}$$

Remark 2 Some direct but useful properties of reflections and partial reflections are in order:

1. A reflection $R^i_x(y)$ preserves the distance from y to any point in the hyperplane defined by $x_{i-K}, \dots, x_{i-1}$.
2. The pairwise distances for $R^i_x(x_i), R^i_x(x_{i+1}), \dots, R^i_x(x_n)$ are the same as those for $x_i, \dots, x_n$. As a consequence of this, and the fact that $R^i_x(x_\ell) = x_\ell$, for $\ell = i-K, \dots, i-1$, all pairwise distances for $x_{i-K}, \dots, x_n$ from $x \in \hat{X}$ are preserved in $g_i(x)$.
3. Partial reflections preserve distances related to discretization edges E_D, so that $g_i(x) \in \hat{X}$, for every $x \in \hat{X}$.
4. All realizations in $\hat{X}$ can be generated from a single $x \in \hat{X}$ by the composition of partial reflection operators g_i [19, Sec. 3.3.8].

Let us now recall one of the main results about ^{K}DMDGP symmetries.

Theorem 1 (Theorem 3.2 in [19]) *With probability 1, for all $j > K + i$, there is a set H^{ij} of 2^{j-i-K} real positive values such that for each $x \in \hat{X}$, we have $\|x_j - x_i\| \in H^{ij}$. Furthermore, for all $x', x \in \hat{X}$ such that $x' \neq x$ and $x'_t = x_t$, for $t \leq i+K-1$, $\|x_j - x_i\| = \|x'_j - x_i\|$ if and only if $x'_j = R^{i+K}_x(x_j)$.*

In Theorem 1, "with probability 1" means that the set of ^{K}DMDGP instances for which the statements do not hold has Lebesgue measure zero in the set of all ^{K}DMDGP instances [22].

The first part of Theorem 1 says that, for $j > K+i$, the possible realizations $x \in \hat{X}$ yield a set of 2^{j-i-K} distinct values for $\|x_i - x_j\|$. Let $\hat{X}_{i+K-1}(x)$ be the subset of possible realizations $x' \in \hat{X}$ that agree with x in the first $i+K-1$ positions. Given a possible realization x, each of these 2^{j-i-K} distinct values is associated to a pair of $2^{j-i-K+1}$ possible positions for x_j from realizations in $\hat{X}_{i+K-1}(x)$ (see Figure 1 where possible values for $\|x_1 - x_3\|$ and $\|x_1 - x_4\|$ are represented by the radii of gray and, respectively, black arcs centered at x_1).

Since $j - i > K$, if the distance d_{ij} is available, it must be a pruning distance. In view of Assumption 1, then $\|x_i - x_j\| = d_{ij}$ for some $x \in \hat{X}$. Let such a x define the set $\hat{X}_{i+K-1}(x)$. Now, from the second part of Theorem 1, we have that among the possible realizations $x' \in \hat{X}_{i+K-1}(x)$, only those such that $x'_j \in \{x_j, R_x^{i+K}(x_j)\}$ are feasible with respect to d_{ij}. If v_j is the last vertex in the order, then only two realizations in $\hat{X}_{i+K-1}(x)$ are feasible.

For every DMDGP solution, there is another one symmetric to the hyperplane defined by the positions of the first K vertices. Moreover, as a consequence of Theorem 1, the number of solutions doubles for every other *symmetry vertex* belonging to the following set [22]:

$$S := \{v_\ell \in V \mid \nexists \{v_i, v_j\} \in E \text{ with } i + K < \ell \leq j\}. \tag{3}$$

The vertex v_{K+1} is always in S, because the first K vertices define a symmetry hyperplane. The other symmetry hyperplanes are given by the positions of $v_{i-K}, \dots, v_{i-1}$, if $v_i \in S$, for $i > K+1$. As mentioned in the Section 1, S can be computed before solving a ^{K}DMDGP instance, which implies that the number of solutions is known *a priori*, and given by $2^{|S|}$, with probability one.

Theorem 2 (Theorem 3.4 in [20]) *Let (G, K) be a feasible ^{K}DMDGP and S its set of symmetry vertices. Then, with probability 1, $|X| = 2^{|S|}$.*

The $2^{|S|}$ valid realizations are incongruent modulo translations and rotations, meaning that they differ one from another only by partial reflections (or a total reflection through the first symmetry hyperplane, as explained above).

It is important to notice from (3) that the addition of new pruning edges in E may reduce the number of elements (symmetry vertices) in S.

A direct consequence of Theorem 2 is the following corollary.

Corollary 1 *Let (G, K) be a feasible ^{K}DMDGP instance where $|V(G)| = n > K$. If $\{v_1, v_n\} \in E$, then (G, K) has only two incongruent solutions which are reflections of each other through the symmetry hyperplane defined by the position of the first K vertices.*

Proof If $\{v_1, v_n\} \in E$, then $S = \{v_{K+1}\}$, which implies that the number of solutions is $2^{|S|} = 2^1 = 2$. If one of these solutions is x, then the other is x', the reflection of x through the hyperplane defined by $x_1, \dots, x_K$. □

A result that will be useful ahead is given in Proposition 1 and illustrated in Figure 1.

Proposition 1 (Lemma 4.2 in [20]) *Let $x \in \hat{X}$, $k > i+1$ and $p_i \neq p_k$ be the normals to the hyperplanes defining $R_x^i(\cdot)$ and $R_x^k(\cdot)$, respectively. If $y \in \mathbb{R}^K$ is not in the hyperplanes containing the origin and normal to p_i and p_k, then*

$$R^k_{g_i(x)}(R_x^i(y)) = R_x^i(R_x^k(y)).$$

Proposition 1 tells us that compositions of partial reflections that depend on more than one realization (e.g x and $x' := g_k(x)$) can be described in terms of reflections based on a single realization. For example, for $k > i$, we have

$$\begin{aligned}(g_k \circ g_i)(x) &:= g_k(g_i(x)) \\ &= g_k(x_1, \ldots, x_{i-1}, R_x^i(x_i), \ldots, R_x^i(x_n)) \\ &= (x_1, \ldots, x_{i-1}, R_x^i(x_i), \ldots, R_x^i(x_{k-1}), R^k_{x'}(R_x^i(x_k)), \ldots, R^k_{x'}(R_x^i(x_n)) \\ &= (x_1, \ldots, x_{i-1}, R_x^i(x_i), \ldots, R_x^i(x_{k-1}), R_x^i(R_x^k(x_k)), \ldots, R_x^i(R_x^k(x_n)),\end{aligned}$$

where the last equality follows from Proposition 1.

Therefore, for a DMDGP, given $x \in \hat{X}$, problem (1) can be cast as finding a binary vector $s \in \{0,1\}^{n-K}$, such that

$$x(s) := U(x,s) = g_{K+1}^{s_1} \circ \cdots \circ g_n^{s_{n-K}}(x) \tag{4}$$

satisfies $||x_i(s) - x_j(s)|| = d_{ij}$, for all $\{i,j\} \in E$. Here, $g_i^1(\cdot) = g_i(\cdot)$ and $g_i^0(\cdot) = I(\cdot)$, where $I(x) = x$. In Section 3 we shall explain how to efficiently perform the search of this binary vector taking into account DMDGP symmetry information.

To close this section, let us describe how to generate other valid realization $x(s') \in X$ from a given one $x(s) \in X$. Let $x(s)$ be a valid realization for (G,K). The vertices in the set S determine which components of the binary vector $s \in \{0,1\}^{n-K}$ from (4) are allowed to change in order to obtain another valid realization for (G,K). In other words, the search space for the new $s' \in \{0,1\}^{n-K}$ is reduced to

$$s' \in B := \{s' \in \{0,1\}^{n-K} \mid s'_\ell = s_\ell \text{ if } v_{K+\ell} \notin S\}. \tag{5}$$

Lemma 1 *Let $S \neq \varnothing$ and $x(s)$ be a valid realization for (G,K). For every $s' \in B$, $x(s') \in X$.*

Proof Since $x(s')$ from Eq. (4) involves only partial reflections, in view of Property 3 in Remark 2, $x(s') \in \hat{X}$, i.e $\|x_i(s') - x_j(s')\| = d_{ij}, \forall \{i,j\} \in E_D$.

It remains to show that $x(s')$ does not violate distance constraints associated to pruning edges $\{i,j\} \in E_P$. Since the reflections are applied to positions x_ℓ such that $\ell \geq K+1$, edges $\{i,j\} \in E$ with $i < j \leq K$ are not affected. Thus, assume that $K+1 \leq j \leq n$.

We have that $v_{i+K+1}, \ldots, v_j \notin S$, and from (4) and (2), the positions $x_\ell, \ldots,\ x_{i+K+1}, \ldots, x_j$ are updated by reflections $R_x^\ell(x_\ell),\ \ldots,\ R_x^\ell(x_{i+K+1})$,

$\ldots$, $R_x^\ell(x_j)$, for $K+1 \leq \ell \leq i+K$ such that $v_\ell \in S$. Since either $i \leq \ell - 1$, i.e x_i is in the hyperplane associated to v_ℓ, or $i \geq \ell$, i.e. x_i comes after this hyperplane, in view of Remark 2, Property 2, these reflections are such that $\|x_i(s') - x_j(s')\| = d_{ij}$. □

3 Nested DMDGP subproblems

Given a DMDGP instance, properties 1 and 2 of Definition 1 give rise to a rich symmetric structure for the corresponding DGP problem, as discussed in Section 2.

It is interesting to observe that, on one hand, the absence of pruning edges turns the DMDGP into a trivial problem, because *any* path from the root to a leaf node of the search tree corresponds to a valid realization, i.e $\hat{X} = X$, and all other solutions can be built by partial reflections. On the other hand, one of the most challenging DMDGP instances to solve with BP is the one where the only pruning edge is $\{v_1, v_n\}$. In that case, feasibility can only be verified at a leaf node, and for a standard depth-first search (DFS), it may represent a costly backtracking process until the first valid realization is found.

Differently, given $x \in \hat{X}$, the present proposal is to iteratively handle the pruning edge constraints following a given order $<$ on E_P.

As mentioned in Section 2 (after Theorem 2), each pruning edge $\{i, j\}$ may reduce the set of valid partial reflection operations that can be applied to realizations of the vertices $v_{i+K}, \ldots, v_j$. Thus, by keeping track of valid partial reflections (or equivalently their corresponding symmetry vertices), it is possible to consistently modify a given realization satisfying a subset of distance constraints to also satisfy a new pruning edge constraint. This process is repeated until all distance constraints are satisfied.

For this, we enumerate edges in E_P as $e_1, e_2, \ldots, e_m$, with $m = |E_P|$, and use $e_k < e_\ell$ to mean that edge e_k precedes e_ℓ in this order. We define the set of pruning edges preceding edge $\{i, j\}$ by

$$P^{ij} := \{\{u, w\} = e' \in E_P \mid e' < e = \{i, j\}\}. \tag{6}$$

Then, we define a sequence of subproblems spanned by $\{i, j\} \in E_P$ following the above pruning edge order.

Definition 4 Let (G, K) be a feasible ${}^K DMDGP$ with $G = (V, E, d)$. Let $G^{ij} = (V, E^{ij}, d_{|E^{ij}})$, where $E^{ij} = E_D \cup P^{ij} \cup \{i, j\}$, $\{i, j\} \in E_P$ and $d_{|E^{ij}}$ is the restriction of d to E^{ij}. We say that (G^{ij}, K) is a ${}^K DMDGP$ subproblem of (G, K) spanned by pruning edge $\{i, j\}$.

It is clear that (G^{ij}, K) is itself a DMDGP problem. Let us denote by $X(G^{ij})$ the solution set of (G^{ij}, K).

Proposition 2 *Let $G = (V, E, d)$ and $H = (V, F, \hat{d})$ such that (G, K) and (H, K) are feasible ${}^K DMDGPs$. If $E \subset F$ and $d(\{i, j\}) = \hat{d}(\{i, j\}), \forall \{i, j\} \in E$, then $X(G) \supset X(H)$.*

Let (G^{uw}, K) and (G^{ij}, K) be DMDGP subproblems spanned by edges $\{u, w\}$ and $\{i, j\}$, respectively, such that $\{u, w\} < \{i, j\}$. In view of Proposition 2, we have $X(G^{uw}) \supset X(G^{ij})$.

Moreover, in this sequence of DMDGP subproblems, each time a new pruning edge is included, e.g $E^{ij} = E^{uw} \cup \{i, j\}$, the set of symmetry vertices (see Eq. (3)) for (G^{uw}, K) may be reduced. This motivates us to define the set of *necessary* symmetry vertices for subproblem (G^{ij}, K) as:

$$S^{ij} = \{v_\ell \in \{v_{i+K+1}, \dots, v_j\} \mid \nexists \{u, w\} \in P^{ij}, u + K < \ell \le w\}. \tag{7}$$

Let $x(s)$ be the current realization which is valid for (G^{uw}, K) and let $e_{k+1} = \{i, j\} > \{u, w\} = e_k$. The vertices in the set S^{ij} determine which components of the binary vector $s \in \{0, 1\}^{n-K}$ from Eq. (4) are allowed to change in order to obtain a valid realization for (G^{ij}, K). In other words, the search space for the new $s' \in \{0, 1\}^{n-K}$ is reduced to

$$s' \in B^{ij} := \{s' \in \{0, 1\}^{n-K} \mid s'_\ell = s_\ell \text{ if } v_{i+K+\ell} \notin S^{ij}\}. \tag{8}$$

Lemma 2 *Let $S^{ij} \neq \varnothing$ and $e_{k+1} = \{i, j\} > \{u, w\} = e_k$. Let $x(s)$ be a valid realization for (G^{uw}, K). For every $s' \in B^{ij}$, $x(s') \in X(G^{uw})$.*

Proof The proof is similar to the one of Lemma 1 and therefore is left in the Appendix. □

Remark 3 From Proposition 2 and Lemma 2, if $e_{k+1} = \{i, j\} > \{u, w\} = e_k$, then, given $x(s) \in X(G^{uw}) \supset X(G^{ij})$, to obtain $x(s') \in X(G^{ij})$ it suffices to find $s' \in B^{ij}$ such that $\|x_i(s') - x_j(s')\| = d_{ij}$.

Furthermore, in the following we show that there is a unique $s' \in B^{ij}$ satisfying such condition. For this, let us recall a simple fact that follows from Definition 1.

Proposition 3 *If (G, K) is a ^{K}DMDGP instance, so is $(G[v_i, \dots, v_j], K)$, for $j > K + i$.*

Thus, given a ^{K}DMDGP instance (G, K), any subgraph induced by at least $K + 1$ consecutive (w.r.t. the vertex order) vertices of $V(G)$ is a ^{K}DMDGP itself. Proposition 3 implies that each $\{v_i, v_j\} \in E_P$ defines a DMDGP instance based on the subgraph $G[v_i, \dots, v_j]$.

Proposition 4 *Any DMDGP instance $(G[v_i, \dots, v_j], K)$ spanned by $\{v_i, v_j\} \in E_P$ has only two solutions.*

Proof It follows from Proposition 3 and Corollary 1. □

Proposition 4 says that each DMDGP instance $(G[v_i, \dots, v_j], K)$ spanned by a pruning edge $\{i, j\}$ has only two solutions, which are reflections of each other through the hyperplane defined by $x_i, \dots, x_{i+K-1}$. These two solutions correspond to a particular configuration of the components $s'_{i+K}, \dots, s'_j$. The only difference between the two is the first component s'_{i+K}. Since $v_{i+K} \notin S^{ij}$ and the components of s'_ℓ with $\ell \le i + K$ or $\ell > j$ are kept fixed, we conclude that $s' \in B^{ij}$ is unique.

Algorithm 2 SBBU

```
1: SBBU$(G, K, (e_1, \dots, e_m), x \in \hat{X})$
2: Set $s = 0$, $x(0) = x$
3: for $k = 1, 2, \dots, m$ do
4:     $\{i, j\} = e_k$
5:     if $|S^{ij}| > 0$ then
6:         Find $s' \in B^{ij}\ :\ \|x_i(s') - x_j(s')\| = d_{ij}$
7:         Update $s = s'$ and $x(s) = U(x, s)$
8:     end if
9: end for
10: return a valid realization $x(s)$
```

4 New algorithm

Henceforth, we assume that subproblems (G^{ij}, K) spanned by pruning edges $\{i, j\}$ are solved following a given order $<$ in E_P and that a realization $x \in \hat{X}$ is given.

4.1 The conceptual algorithm

First, we present a conceptual algorithm (Algorithm 2) which summarizes the ideas discussed in the previous sections.

When solving subproblem (G^{ij}, K), if $|S^{ij}| = 0$ then this subproblem has already been solved implicitly, according to the following proposition.

Proposition 5 *Let $x(s)$ be a valid realization for (G^{uw}, K), for all $\{u, w\} \in P^{ij}$. If $S^{ij} = \varnothing$, then $x(s)$ is valid for (G^{ij}, K).*

Proof If $S^{ij} = \varnothing$, then for every $v_\ell \in \{v_{i+K+1}, \dots, v_j\}$, $\exists \{u, w\} \in P^{ij}$ such that $u + K + 1 \le \ell \le w$. Suppose that $x(s)$ is such that $\|x_i(s) - x_j(s)\| \ne d_{ij}$. (a) By Theorem 1 and Assumption 1, $\exists s' \in \{0, 1\}^{n-K}$ with some $s'_\ell \ne s_\ell$, for $u + K + 1 \le \ell \le w$, such that $\|x_i(s') - x_j(s')\| = d_{ij}$. (b) Since $\ell \ge u + K + 1$, it follows that $x_w(s') \notin \{x_w(s), R_x^{u+K}(x_w(s))\}$. Thus, by Theorem 1, $\|x_w(s') - x_u(s')\| \ne d_{uw}$. But (a) and (b) together contradict Assumption 1. Hence $\|x_i(s) - x_j(s)\| = d_{ij}$ and the assertion follows from Lemma 2. □

Otherwise, for $|S^{ij}| > 0$, in Step 6 we perform an exhaustive search to find $s' \in B^{ij}$ such that $\|x_i(s') - x_j(s')\| = d_{ij}$. In Step 7, we update the current realization to $x(s')$ according to Eq. (4).

Theorem 3 *Let (G, K) be a feasible ^{K}DMDGP instance. Considering exact arithmetic, Algorithm 2 finds $x \in X$.*

Proof Since $x(0) = x \in \hat{X}$, due to Assumption 1 and Lemma 2, Step 6 is well-defined. From Remark 3 and Step 6, it follows that $x(s') \in X(G^{ij})$, for every $e_k = \{i, j\}$. Thus, since for the last pruning edge $e_m = \{i_m, j_m\}$, we have $E^{i_m, j_m} = E$, i.e $G^{i_m, j_m} = G$, after this last subproblem is solved, $x(s) \in X(G^{i_m, j_m}) = X(G) = X$. □

4.2 A practical algorithm

In this section, based on a particular pruning edge order, we introduce a practical version of Algorithm 2 which:

1. does not required an initial realization $x \in \hat{X}$;
2. avoids the computation and storage of unnecessary reflectors $R_x^i(\cdot)$;
3. may result in less operations in the update step (Step 7) of Algorithm 2;
4. allows us to discuss a concrete implementation for the sets S^{ij}.

For this, instead of working with a full realization $x(s) \in \hat{X}$, which is updated through the binary vector s by Eq. (4), and computing and storing reflectors $R_x^i(\cdot)$ based on $x = x(0) \in \hat{X}$, the idea is to grow a partial realization $x_1, \ldots, x_t$, where $t = \arg\max\{w \mid \{u,w\} \in P^{ij}\}$, and compute the necessary reflectors on the fly based on the current partial realization and S^{ij}. This way, for each subproblem (G^{ij}, K), we do not compute full valid realizations $x_1, \ldots, x_n$ but valid partial realizations $x_1, \ldots, x_j$, with $j \le t$.

Assumption 2 *Pruning edges $\{i,j\}$, with $i < j$, are sorted in increasing order of j, followed by a decreasing order of i.*

Under this order, we can re-write the set of pruning edges preceding $\{i,j\}$ as

$$P^{ij} = \{\{u,w\} \in E_P \mid u < w < j \ \vee (w = j \wedge w > u > i)\}. \tag{9}$$

Definition 5 We say that $x_1, \ldots, x_t$, with $j \le t$, is a valid partial realization for (G^{ij}, K), if $x_1, \ldots, x_t$ satisfies all distance constraints associated to edges in $\{\{u,w\} \in E(G^{ij}) \mid w \le j\}$.

Remark 4 Recall that $E(G^{ij}) = E^{ij} = E_D \cup P^{ij} \cup \{i,j\}$ and thanks to Assumption 2, there is no $\{u,w\} \in P^{ij}$, with $w > j$. This allows us to extend a valid partial realization $x_1, \ldots, x_t$ for (G^{ij}, K) to a valid full realization $x_1, \ldots, x_n$ for (G^{ij}, K), i.e $x \in X(G^{ij})$, by simply growing $x_1, \ldots, x_t$ to $x_1, \ldots, x_n$ using discretization distances (see Subsection 4.2.1), because no distance constraint $\{u,w\} \in P^{ij} \cup \{i,j\}$ is affected by this operation.

Assumption 2, along with (9), will be used in the results that follow. Using this concepts we will show in the next subsections that when dealing with subproblem (G^{ij}, K):

1. given a partial realization $x_1, \ldots, x_t$ satisfying discretization distances and distances corresponding to pruning edges in P^{ij}, it can be extended to $x_1, \ldots, x_t, \ldots, x_j$ keeping feasibility of such distance constraints and new discretization constraints;
2. it is possible to apply partial reflections to this extended partial realization in order to fulfill $\|x_i - x_j\| = d_{ij}$ without violating the distance constraints considered so far.

4.2.1 Initialization of candidate positions (Growth)

In Section 4.2.2 we shall explain how to find a valid partial realization for DMDGP subproblems (G^{ij}, K) by composing reflections through symmetry hyperplanes and applying them to positions $x_{i+K+1}, \ldots, x_j$. This procedure assumes that candidate positions for $x_i, \ldots, x_j$ are available when we start to solve (G^{ij}, K). In this section, we describe how to initialize these positions.

From now on, we assume that initialization of candidate positions must follow the vertex order from v_1 to v_n, meaning that if (G^{ij}, K) is the current subproblem, and x_t is the last initialized position, such that $t < j$, then we initialize positions from x_{t+1} to x_j, whereas $x_1, \ldots, x_t$ remain unchanged. In other words, the candidate positions $x_{t+1}, \ldots, x_j$ are grown from the current partial realization $x_1, \ldots, x_t$ using only distance constraints associated to discretization edges. Moreover, each position x_i is initialized only once, although it can be modified later (see Section 4.2.2) in order to satisfy a distance constraint corresponding to a pruning edge $e' \geq e = \{i, j\}$. This is formalized in Proposition 6.

Proposition 6 *Assume edges in E_P are ordered as $e_1, \ldots, e_m$. Then, before solving (G^{ij}, K), positions $x_1, \ldots, x_j$ can be initialized such that*

$$\forall \{\ell, k\} \in E_D \cap E(G[v_1, \ldots, v_j]), \quad \|x_\ell - x_k\| = d_{\ell k}, \tag{10}$$

and

$$\forall \{\ell, k\} \in P^{ij}, \quad \|x_\ell - x_k\| = d_{\ell k}. \tag{11}$$

Proof We prove this by induction on the edge order. In the base case we consider $e_1 = \{i_1, j_1\} \in E_P$ spanning the first subproblem to be solved. The positions x_i, for $i = 1, \ldots, j_1$ are initialized right away. From Definition 1, $x_1, \ldots, x_K$ can be localized uniquely (up to rotations and translations) by different methods [7,1]. Hence, $\|x_\ell - x_k\| = d_{\ell k}, \forall \{\ell, k\} \in E(G[v_1, \ldots, v_K])$. Then, by K-lateration (see Remark. 1), there are at most two positions $\{x_i^+, x_i^-\}$ for $v_i \in V$ for each $K < i \leq j_1$. Notice that any partial realization $x_1, \ldots, x_{j_1}$ is enough to build partial reflections. The correct alternative will be chosen later by the appropriate partial reflection composition which satisfies constraints defined by pruning edges (Section 4.2.2 gives more details). Thus, without loss of generality, let $x_1, \ldots, x_{j_1}$ be the partial realization obtained by choosing x_i^-, for every $i = K + 1, \ldots, j_1$. Since $P^{i_1 j_1} = \varnothing$, this partial realization satisfies (10) and (11) for $\{i, j\} = \{i_1, j_1\}$.

The induction hypothesis is that (10) and (11) hold for pruning edges $e_1, \ldots, e_k$, i.e $x_1, \ldots, x_t$ is a valid partial realization for all subproblems spanned by these edges, where

$$t := \max\{w \mid \{u, w\} \in P^{ij}\} = \max\{w \mid \{u, w\} \in \{e_1, \ldots, e_k\}\}.$$

In the inductive step, let us prove that (10) and (11) also hold for pruning edge $e_{k+1} = \{i, j\}$ spanning subproblem (G^{ij}, K).

Algorithm 3 InitializePositions

```
1: InitializePositions($x, t, j$)
2: if $t \geq j$ then
3:     return $x$ and $t$.
4: end if
5: if $t = 0$ then
6:     Initialize $x_1, \dots, x_K$ as a solution of $\|x_\ell - x_i\|^2 = d_{\ell i}^2, \forall \{\ell, i\} \in E(G[v_1, \dots, v_K])$
7:     Set $t = K$
8: end if
9: for $i = t + 1, \dots, j$ do
10:    Find solutions $\{x_i^+, x_i^-\}$ for the system: $\|x_\ell - x_i\|^2 = d_{\ell,i}^2, \ell = i - K, \dots, i - 1$.
11:    Set $x_i = x_i^-$
12: end for
13: Set $t = j$
14: return $x$ and $t$.
```

Since subproblems spanned by edges in P^{ij} are solved, positions $x_1, \dots, x_t$ are already initialized and satisfy (11), and (10) with $v_j = v_t$.

If $j \leq t$, then there is nothing left to do. Thus, suppose $j > t$. Then, positions $x_{t+1}, \dots, x_j$ can be initialized by K-lateration (see Remark 1 and Algorithm 3) based on discretization distances such that (10) holds. $\square$

Remark 5 The proof of Proposition 6 describes a procedure for initialization of $x_1, \dots, x_j$ before solving (G^{ij}, K). It is important to notice that such initialization is done sequentially and depends on previously computed positions which are not recomputed in this step. Thus, after the initialization, the current partial realization continues to be valid for all already solved subproblems.

Algorithm 3 gives a pseudocode for the function IntializePositions which receives a current realization x (actually, the current partial realization $x_1, \dots, x_t$), the index of the last initialized position t, and the index j of the last vertex whose position needs initialization. Updated x and t are returned by this function.

4.2.2 Solving DMDGP subproblems (Correction)

Now we explain how to find a valid partial realization for a DMDGP subproblem (G^{ij}, K), given a valid partial realization $x_1, \dots, x_t$ for (G^{uw}, K), $\forall \{u, w\} \in P^{ij}$.

Recall from Proposition 1 that given positions $x_{i+1}, \dots, x_{i+K+1}, \dots, x_j$, *valid or not*, we can build all necessary symmetry hyperplanes and their corresponding reflection operators $R_x^{i+K+1}(\cdot), \dots, R_x^j(\cdot)$. Then, based on Theorem 1 and Proposition 4, we can apply compositions of such reflection operators *only* to x_j until we find its correct position, as illustrated in Figure 1 for the 2D case.

The only decision to be taken is whether each of the reflectors $R_x^{i+K+1}(\cdot)$, $\dots, R_x^j(\cdot)$ should be applied or not to x_j in order to fulfill $\|x_i - R(x_j, \bar{s})\| = d_{ij}$, where $R(., \bar{s})$ is a composition of the chosen reflectors:

$$R(y, \bar{s}) := (R_x^{i+K+1})^{\bar{s}_1} (R_x^{i+K+2})^{\bar{s}_2} \dots (R_x^j)^{\bar{s}_{j-i-K}}(y). \tag{12}$$

In (12), the binary vector $\bar{s}$ is of size $j - i - K$, and $(R_x^\ell)^0 := I$, the identity operator in $\mathbb{R}^K$, whereas $(R_x^\ell)^1 := R_x^\ell$, for $\ell = i + K + 1, \dots, j$.

In contrast to Algorithm 2, where s is the global binary decision variable and all the reflectors are computed based on the first realization $x(0) = x \in \hat{X}$, now the reflectors $R_x^{i+K+\ell}(\cdot)$ for which $v_{i+K+\ell} \in S^{ij}$ are computed based on the current partial realization $x_1, \dots, x_t$, and $\bar{s}$ is a local binary decision variable belonging to

$$\bar{B}^{ij} := \{\bar{s} \in \{0,1\}^{j-i-K} \mid \bar{s}_\ell = 0 \text{ if } v_{i+K+\ell} \notin S^{ij}\}. \tag{13}$$

Thus, we look for a binary vector $\bar{s} \in \bar{B}^{ij}$ such that

$$x'_j = R(x_j, \bar{s}) = (R_x^{i+K+1})^{\bar{s}_1} (R_x^{i+K+2})^{\bar{s}_2} \dots (R_x^j)^{\bar{s}_{j-i-K}}(x_j) \tag{14}$$

satisfies $\|x_i - x'_j\| = d_{ij}$. We remark that this search is exhaustive: we test all $|\bar{B}^{ij}| = 2^{|S^{ij}|}$ possible choices for $\bar{s}$ (recall that there is a unique $\bar{s}$ that works, as discussed after Proposition 4).

Once $\bar{s}$ is found, the positions of $v_{i+K+1}, \dots, v_t$ are updated according to:

$$x'_\ell = \bar{U}(x_\ell, \bar{s}) := \left(\prod_{t=1}^{\ell-i-K} (R_x^{i+K+t})^{\bar{s}_t} \right)(x_\ell), \quad \ell = i + K + 1, \dots, t. \tag{15}$$

This update maintain feasibility of $x_1, \dots, x_t$ with respect to (G^{uw}, K), for every $\{u, w\} \in P^{ij}$, because positions $x_{u+K}, \dots, x_w$ are only updated simultaneously by partial reflections $R_x^v(x_{u+K}), \dots, R_x^v(x_w)$, for $v \leq u + K$.

4.2.3 Symmetry vertex sets

The ideas of the Subsections 4.2.1 and 4.2.2 lead to Algorithm 4. This algorithm makes use of $\mathcal{C}$, a partition of $\{v_{K+1}, \dots, v_n\}$ used to obtain the sets S^{ij}. At the beginning, we set $\mathcal{C} = \{\{v_i\}\}_{i=K+1}^n$. This partition is updated in Step 14 taking into account already solved subproblems. Assume that subsets of vertices in $\mathcal{C}$ are ordered according to the vertex order of Definition 1. Let us denote by $\mathsf{first}(C^0)$ the first vertex of $C^0 \in \mathcal{C}$.

We also introduce a function $\rho_\mathcal{C} : \{K + 1, \dots, n\} \to \mathcal{C}$, parametrized by $\mathcal{C}$, such that $\rho_\mathcal{C}(\ell)$ returns the unique element of $\mathcal{C}$ containing vertex v_ℓ. The next proposition shows that this function is well-defined at every iteration of Algorithm 4.

Proposition 7 *At every iteration of Algorithm 4, $\mathcal{C}$ is a partition of the subset of vertices $\{v_{K+1}, \dots, v_n\}$.*

Algorithm 4 SBBU

```
1: $\mathsf{SBBU}(G, K)$
2: Order edges $\{v_i, v_j\} \in E_P$ in increasing order of $j$ and decreasing order of $i$ obtaining a
   sequence $(e_1, \ldots, e_m)$, with $m = |E_P|$. Set $t = 0$, $n = |V|$
3: Set $\mathcal{C} = \{\{v_i\}\}_{i=K+1}^{n}$
4: $\mathsf{InitializePositions}(x, t, K)$                # *positions for the initial clique*
5: **for** $k = 1, 2, \ldots, m$ **do**
6:     $\{i, j\} = e_k$
7:     **if** $\rho_{\mathcal{C}}(i + K) \neq \rho_{\mathcal{C}}(j)$ **then**
8:         $\mathsf{InitializePositions}(x, t, j)$
9:         Set $C^0 = \rho_{\mathcal{C}}(i + K)$ and $\mathcal{D} = \rho_{\mathcal{C}}(\{i + K + 1, \ldots, j\}) \setminus \{C^0\}$
10:        Let $S^{ij} = \cup_{C \in \mathcal{D}} \mathsf{first}(C)$            # *local symmetry vertices*
11:        Compute $R_x^{\ell}(\cdot)$ for each $v_\ell \in S^{ij}$
12:        Find $s \in \bar{B}^{ij}$ : $\|x_i - R(x_j, \bar{s})\| = d_{ij}$            # *find position $x_j$*
13:        Update $x_\ell = \bar{U}(x_\ell, \bar{s})$, for $\ell = i + K + 1, \ldots, j$
14:        Set $C^+ = (\cup_{C \in \mathcal{D}} C) \cup C^0$ and update $\mathcal{C} = (\mathcal{C} \setminus (\mathcal{D} \cup \{C^0\})) \cup C^+$
15:    **end if**
16: **end for**
17: **if** $t < n$ **then**
18:    $\mathsf{InitializePositions}(x, t, n)$
19: **end if**
20: **return** a valid realization $x$
```

Proof At the first iteration $\mathcal{C} = \{\{v_i\}\}_{i=K+1}^{n}$. Assume that, at the beginning of iteration k, $\mathcal{C}$ is a partition of $\{v_{K+1}, \ldots, v_n\}$. If $\rho_{\mathcal{C}}(i+K) = \rho_{\mathcal{C}}(j)$, then we go to the next iteration with $\mathcal{C}$ unchanged. Otherwise, from Step 9, $\mathcal{D}$ and C^0 are subsets of $\mathcal{C}$. Then, Step 14 updates $\mathcal{C}$ by removing these subsets and including their union, hence, the updated $\mathcal{C}$ is still a partition of $\{v_{K+1}, \ldots, v_n\}$. □

Proposition 8 *In Algorithm 4, after Step 14, there exists a unique $C \in \mathcal{C}$ such that $C \supset \{v_{i+K}, \ldots, v_j\}$.*

Proof Follows directly from Steps 9 and 14 of Algorithm 4. □

We remark that $\rho_{\mathcal{C}}(\{i+K+1, \ldots, j\})$ denotes the image of $\{i+K+1, \ldots, j\}$ by $\rho_{\mathcal{C}}$ in the definition of $\mathcal{D}$ (Step 9), i.e it returns elements of $\mathcal{C}$ whose union contains $v_{i+K+1}, \ldots, v_j$ and $C^0 = \rho_{\mathcal{C}}(i + K)$ is the element of $\mathcal{C}$ containing v_{i+K}.

Proposition 9 *In Algorithm 4, $\rho_{\mathcal{C}}(i + K) = \rho_{\mathcal{C}}(j)$ if and only if $S^{ij} = \varnothing$.*

Proof If $S^{ij} = \varnothing$, then

$$\forall\ v_\ell \in \{v_{i+K+1}, \ldots, v_j\}, \quad \exists \{u, w\} \in P^{ij}\ :\ u + K < \ell \leq w \leq j. \tag{16}$$

In particular, for $\ell = i + K + 1$, there exists $\{r, z\} \in P^{ij}$ such that $r + K < i + K + 1 \leq z \leq j$. Clearly, $r \leq i$. From Proposition 8 there exists a unique $C^1 \in \mathcal{C}$ such that $C^1 \supset \{v_{r+K}, \ldots, v_{i+K}, v_{i+K+1}, \ldots, v_z\}$.

Thus, if $z = j$, then $\rho_{\mathcal{C}}(i + K) = \rho_{\mathcal{C}}(j)$.

Otherwise, for $z < j$, because $i + K + 1 \leq z$, it follows that $v_{z+1} \in \{v_{i+K+1}, \ldots, v_j\}$ and from (16), there exists $\{u, w\} \in P^{ij}$ such that $u + K < z + 1 \leq w \leq j$ (clearly, $u + K \leq z$). Thus, from Proposition 8:

$$\exists! C^2 \supset \{v_{u+K}, \ldots, v_z, v_{z+1}, \ldots, v_w\}. \tag{17}$$

If $u \leq r \leq i$, then from (17), we obtain $v_{i+K} \in C^2$.

Otherwise, for $r < u \leq z - K$, then $r + K < u + K \leq z$, implying that $v_{u+K} \in C^1$. In either case, we have $\rho_{\mathcal{C}}(u+K) = \rho_{\mathcal{C}}(i+K)$. From Proposition 7, we conclude that $\rho_{\mathcal{C}}(i+K) = \rho_{\mathcal{C}}(w)$.

Hence, if $w = j$, $\rho_{\mathcal{C}}(i+K) = \rho_{\mathcal{C}}(j)$.

Otherwise $(w < j)$, in view of (16), we can apply the same argument to v_{w+1}, and repeat until we find $\{h, p\} \in P^{ij}$ with $p = j$.

On the other hand, to prove that $\rho_{\mathcal{C}}(i+K) = \rho_{\mathcal{C}}(j)$ implies $S^{ij} = \varnothing$, we use the counter-positive. Suppose there exists $v_\ell \in \{v_{i+K+1}, \ldots, v_j\}$ such that $\nexists \{u, w\} \in P^{ij}$ such that $u + K < \ell \leq w \leq j$. This means that $\forall \{u, w\} \in P^{ij}$ either (i) $w < \ell$ or (ii) $\ell \leq w \leq j$ and $u + K \geq \ell$. If $w < \ell, \forall \{u, w\} \in P^{ij}$, then $\rho_{\mathcal{C}}(j) = \{v_j\} \neq \rho_{\mathcal{C}}(i+K)$, because $w < \ell \leq j$ (v_ℓ and v_j were never reached).

Thus, let us consider $\{u, w\} \in P^{ij}$ such that $\ell \leq w \leq j$ and $u + K \geq \ell$. Without loss of generality, assume $w = j$. Since $\ell \geq i + K + 1$, then $u + K \geq i + K + 1$ (or $u \geq i + 1$), implying that $\rho_{\mathcal{C}}(i+K) \neq \rho_{\mathcal{C}}(u+K) = \rho_{\mathcal{C}}(j)$, where the last equality follows from Proposition 8. □

Proposition 9 shows that if v_{i+K} and v_j are in the same subset, i.e., $\rho_{\mathcal{C}}(i+K) = \rho_{\mathcal{C}}(j)$, then this subproblem was already solved implicitly (see Proposition 5). This is equivalent to condition $|S^{ij}| = 0$ in Algorithm 2.

Otherwise, we need to obtain the set S^{ij} of symmetry vertices for (G^{ij}, K). This is accomplished in Step 10.

Theorem 4 *If $\rho_{\mathcal{C}}(i+K) \neq \rho_{\mathcal{C}}(j)$, then $\bigcup_{C \in \mathcal{D}} \mathsf{first}(C) = S^{ij}$.*

Proof Let $v_\ell \in \cup_{C \in \mathcal{D}} \mathsf{first}(C)$. From Proposition 7, $\exists! \hat{C} \supset \{v_\ell\}$ such that $\hat{C} \in \mathcal{D}$ and $\mathsf{first}(\hat{C}) = v_\ell$. Suppose $v_\ell \notin S^{ij}$, i.e there exists $\{u, w\} \in P^{ij}$ such that $u + K < \ell \leq w$. From Proposition 8, $\exists! C \in \mathcal{C}$ such that $C \supset \{v_{u+K}, \ldots, v_w\}$ and since $\mathcal{C}$ is a partition, it follows that $C = \hat{C}$. But $\mathsf{first}(\hat{C}) = \mathsf{first}(C) = v_{u+K} \neq v_\ell$ contradicting $\mathsf{first}(\hat{C}) = v_\ell$. Therefore, $\nexists \{u, w\} \in P^{ij}$ such that $u + K < \ell \leq w$. Thus, $v_\ell \in S^{ij}$.

Conversely, let $v_\ell \in S^{ij}$. Then, for every $\{u, w\} \in P^{ij}$ either (i) $w < \ell$ or (ii) $\ell \leq w \leq j$ and $u + K \geq \ell$. If $\forall \{u, w\} \in P^{ij}$, we have $w < \ell$, then $C = \rho_{\mathcal{C}}(\ell) = \{v_\ell\} \in \mathcal{D}$ and $\mathsf{first}(C) = v_\ell$. Otherwise, there are $\{u, w\} \in P^{ij}$ such that $\ell \leq w \leq j$. For all of these, $u + K \geq \ell$. Recall from Algorithm 4 that $\mathsf{first}(C^+) = \mathsf{first}(C^0) = \mathsf{first}(\rho_{\mathcal{C}}(u+K))$. We split the analysis in three cases.

Case 1: $v_\ell \notin \rho_{\mathcal{C}}(u+K)$. Then, $\ell < u + K$, implying that $v_\ell < \mathsf{first}(u+K)$. Thus, after iteration k with $e_k = \{u, w\}$, we have $\rho_{\mathcal{C}}(\ell) = \{v_\ell\}$.

Case 2: $v_\ell \in \rho_{\mathcal{C}}(u+K)$ and $\mathsf{first}(\rho_{\mathcal{C}}(u+K)) = v_\ell$. In this case, after iteration k with $e_k = \{u, w\}$, $\rho_{\mathcal{C}}(\ell) = \rho_{\mathcal{C}}(u+K) = \{v_\ell, \ldots, v_{u+K}, \ldots, v_p\}$. Thus $\mathsf{first}(\rho_{\mathcal{C}}(\ell)) = v_\ell$.

Case 3: $v_\ell \in \rho_{\mathcal{C}}(u+K)$ but $\mathsf{first}(\rho_{\mathcal{C}}(u+K)) < v_\ell$. In this case, $C^0 = \rho_{\mathcal{C}}(\ell) = \rho_{\mathcal{C}}(u+K) = \{v_q, \ldots, v_\ell, \ldots, v_{u+K}, \ldots, v_p\}$. The set C^0 is the result of iteration k', with $e_{k'} = \{h,p\} < \{u,w\} = e_k$. Clearly $\{h,p\} \in P^{ij}$. From Proposition 8, $\exists! C \supset \{v_{h+K}, \ldots, v_p\}$. Notice that $u+K \le p \le w$. Since $q > h+K$ contradicts the fact that $\mathsf{first}(\rho_{\mathcal{C}}(u+K)) = v_q$, then $q \le h+K < \ell$. This leads to $h+K < \ell \le u+K \le p$ which implies that $v_\ell \notin S^{ij}$, a contradiction.

Therefore, only cases 1 and 2 can happen and both imply in $v_\ell \in \bigcup_{c \in \mathcal{D}} \mathsf{first}(C)$. □

Corollary 2 *Let (G,K) be a feasible ^{K}DMDGP instance. Considering exact arithmetic, Algorithm 4 finds a valid realization x for (G,K).*

Proof From Theorem 4 and Remark 4 correctness of Algorithm 4 follows from Theorem 3. □

In the end, we obtain a valid partial realization $x_1, \ldots, x_t$ for all subproblems (G^{ij}, K), with $\{i,j\} \in E_P$. If $t = n$, we are done. Otherwise, in view of Remark 4, $x_1, \ldots, x_t$ can be extended to a valid realization $x \in X$. This explains Step 18.

Even under Assumption 1, due to floating point arithmetic, in Step 12 we may not be able to find s such that $\|x_i - R(x_j, \bar{s})\| = d_{ij}$. Thus, instead of stopping as soon as we find a s such that $|\|x_i - R(x_j, \bar{s})\| - d_{ij}| \le \varepsilon$, for a prescribed tolerance ε, we actually consider all $2^{|S^{ij}|}$ possibilities and choose s for which $|\|x_i - R(x_j, \bar{s})\| - d_{ij}|$ is minimum. In case $|\|x_i - R(x_j, \bar{s})\| - d_{ij}| > \varepsilon$ for every $\bar{s} \in \bar{B}^{ij}$, then we actually interrupt the algorithm and return "failure". However, this never happened in the numerical experiments of Section 5.

We remark that $|S^{ij}|$ is a good indicator of the computational cost for solving subproblem (G^{ij}, K), because it determines the number of $2^{|S^{ij}|}$ reflection compositions that we need to apply to x_j in order to find its correct position. Thus, we define the corresponding *total work* to solve a ^{K}DMDGP instance as

$$W := \sum_{\{i,j\} \in \hat{E}} 2^{|S^{ij}|}, \tag{18}$$

where $\hat{E} = \{\{v_i, v_j\} \in E_P \mid |S^{ij}| > 0\}$. Let us also denote by $\bar{W} = \max_{\hat{E}} 2^{|S^{ij}|}$, the maximum work per pruning edge. We also remark that $|S^{ij}|$ depends on the order in which the pruning edges are handled and, in this paper, we consider only the order described in Assumption 2.

5 Experimental results

An efficient implementation of the function $\rho_{\mathcal{C}}$ needs to deal with its evaluation and the update of the subsets of $\mathcal{C}$. We adopted the structure proposed by Newman and Ziff [28], which allows the evaluation of $\rho_{\mathcal{C}}$ and the subsets update in time $O(\log_2(|V|))$ and memory $O(|V|)$.

In order to validate Algorithm 4 and assess its performance, we generate a set of protein-like instances ($K = 3$) whose data were extracted from Protein Data Bank (PDB) [2], and compare the results with those of the classic BP [18, 12]. For each protein, we consider only the backbone composed by the sequence of atoms $N - C_\alpha - C$ and include an edge in the corresponding graph:

1. either when the atoms are separated by at most three covalent bonds
2. or the distance between pairs of atoms is smaller than a certain cut-off value.

The resolution of NMR experiments usually varies between 5 Å and 6 Å. The smaller the cut-off value, the sparser the DMDGP instance. Each instance was generated by the first model and first chain of the PDB file.

The natural backbone order for instances generated in this way provides a vertex order satisfying the assumptions of Definition 1, implying we are working with ^{3}DMDGP instances.

In our experiments we consider two test sets: one using cut-off 6 Å and other using 5 Å . In Tables 1 and 2, we present the results obtained by both algorithms: BP is the classic Branch-and-Prune implementing a depth-first search [18,12], whereas SBBU (Symmetry-based Build-up) corresponds to Algorithm 4.

The algorithms were implemented in C++ and the experiments carried out in Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz, 8G RAM, running Linux Ubuntu 18.04.4, gcc version 7.4.0 compiler. Our codes and datasets can be found in `https://github.com/michaelsouza/sbbu`, whereas an implementation of the Branch-and-Prune algorithm is available at `https://github.com/mucherino/mdjeep`.

Both tables bring the ID of the protein in PDB, the number of atoms $|V|$, number of edges (available distances) $|E|$, CPU time in seconds for the two algorithms and the normalized Mean Distance Error (MDE):

$$MDE(X, E, d) = \frac{1}{|E|} \sum_{\{i,j\} \in E} \frac{|\ \|x_i - x_j\|_2 - d_{ij}\ |}{d_{ij}}. \tag{19}$$

Both algorithms were stopped as soon as the first solution is found and a "–" symbol means that the algorithm was not able to find a solution in less than 300 seconds. For each instance, we also present the total and maximum works W and $\bar{W}$, respectively. The last column, called "Speed-up", contains the ratio time BP / time SBBU.

From the tables, we observe that the new algorithm provides a non-trivial speed-up in most of the instances. In particular, the new algorithm is considerably faster than the classic BP for the sparser instances where it was up to 1,000 times faster.

Concerning the estimated total work of SBBU, it seems that the time varies linearly with W as depicted in Figure 2. The relationship between BP time and W and/or $\bar{W}$ is not so clear. However, we argue that while the most costly subproblem (G^{ij}, K) represents a cost of $\bar{W}$ in the total cost W for SBBU, for

			BP		SBBU				
ID	$\|V\|$	$\|E\|$	Time	MDE	Time	MDE	$\bar{W}$	W	Speed-up
1N6T	30	236	7.60E-05	8.32E-05	1.77E-05	2.72E-12	2	52	4.29
1FW5	60	558	1.30E-04	1.51E-05	3.51E-05	4.22E-12	2	112	3.70
1ADX	120	1008	2.10E-04	5.62E-12	4.49E-05	3.78E-12	2	232	4.67
1BDO	241	2167	4.10E-04	3.79E-12	9.24E-05	1.39E-11	2	474	4.44
1ALL	480	4932	8.40E-04	8.91E-13	1.90E-04	3.80E-12	2	952	4.42
6S61	522	5298	8.70E-04	6.50E-13	2.06E-04	3.09E-12	2	1036	4.23
1FHL	1002	9811	2.00E-03	6.82E-12	3.97E-04	1.93E-11	2	1996	5.04
4WUA	1033	9727	1.80E-03	1.47E-11	3.94E-04	7.73E-12	8	2060	4.57
6CZF	1494	14163	2.60E-03	1.33E-12	5.79E-04	4.18E-12	2	2980	4.49
5IJN	1950	18266	3.40E-03	1.37E-12	7.64E-04	1.76E-11	16	3908	4.45
6RN2	2052	19919	3.70E-03	1.11E-12	8.27E-04	1.54E-11	16	4104	4.48
1CZA	2694	26452	4.90E-03	1.29E-12	1.07E-03	6.22E-11	2	5380	4.59
6BCO	2856	27090	7.90E-03	4.53E-13	1.10E-03	7.91E-12	16	5730	7.15
1EPW	3861	35028	7.80E-03	1.88E-11	1.44E-03	2.50E-10	2	7714	5.40
5NP0	7584	80337	3.10E-02	6.58E-12	3.58E-03	1.35E-10	256	15562	8.66
5NUG	8760	82717	2.40E-02	1.43E-06	3.45E-03	5.33E-10	16	17592	6.96
4RH7	9015	85831	2.50E-02	1.62E-12	3.67E-03	2.22E-10	16	18054	6.82
3VKH	9126	87621	2.70E+00	3.00E-08	3.62E-03	1.15E-09	256	18556	745.03

Table 1 Computational results in some protein-like instances (cut-off: 6Å).

			BP		SBBU				
ID	$\|V\|$	$\|E\|$	Time	MDE	Time	MDE	$\bar{W}$	W	Speed-up
1N6T	30	176	7.60E-05	5.14E-05	1.04E-05	5.64E-12	2	52	7.31
1FW5	60	417	1.40E-04	7.99E-06	2.11E-05	3.08E-12	2	112	6.63
1ADX	120	659	4.70E-04	3.50E-06	3.49E-05	2.53E-12	2	232	13.48
1BDO	241	1345	3.60E-04	1.50E-07	7.05E-05	1.04E-11	2	474	5.11
1ALL	480	3443	9.80E-04	2.81E-06	1.67E-04	1.27E-12	2	952	5.88
6S61	522	3699	8.70E-04	8.10E-07	1.75E-04	1.39E-12	2	1036	4.98
1FHL	1002	6378	2.70E-03	2.56E-12	2.88E-04	1.17E-11	2	1996	9.38
4WUA	1033	6506	1.80E-03	5.34E-12	2.96E-04	2.94E-12	16	2066	6.08
6CZF	1494	9223	2.40E-03	4.62E-13	4.36E-04	2.33E-12	2	2980	5.51
5IJN	1950	11981	4.00E-03	4.43E-13	6.08E-04	4.23E-12	16	3908	6.58
6RN2	2052	13710	5.50E-03	3.89E-13	8.58E-04	9.35E-12	16	4112	6.41
1CZA	2694	17451	5.80E-03	4.51E-13	8.03E-04	3.06E-11	2	5380	7.22
6BCO	2856	18604	5.00E-03	6.00E-07	1.05E-03	6.96E-12	16	5706	4.75
1EPW	3861	23191	2.30E-02	3.00E-08	1.13E-03	9.78E-11	8	7716	20.29
5NP0	7584	59478	2.90E-01	2.56E-12	2.80E-03	4.11E-11	256	16138	103.55
5NUG	8760	56979	2.70E+00	3.60E-07	2.67E-03	1.05E-10	128	17700	1011.09
4RH7	9015	59346	3.10E-02	5.64E-13	2.97E-03	1.20E-10	32	18068	10.43
3VKH	9126	59592	–	–	2.45E-02	1.10E-09	65536	84066	

Table 2 Computational results in some protein-like instances (cut-off: 5Å).

the usual DFS recursive implementation of BP, it may contribute much more to the BP total cost because such subproblem may have to be solved several times in the occasion of backtrackings.

6 Conclusion

We propose a new algorithm for the DMDGP which leverages symmetry information to find the first solution quickly. By efficiently exploiting symmetries

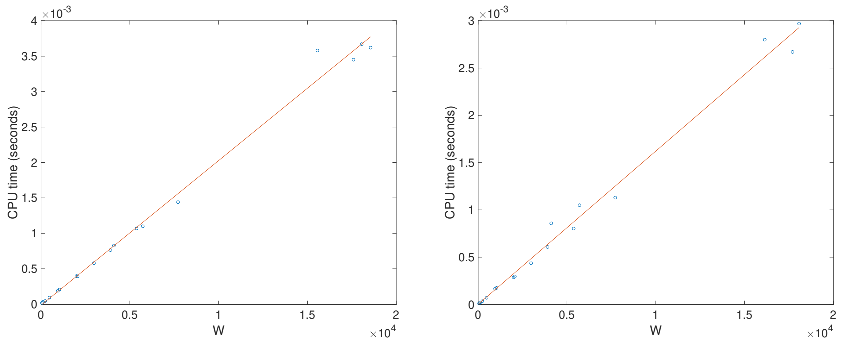

Fig. 2 Scatter plots $W\times$ time and linear regressions for the two datasets (Table 1 on the left, Table 2 on the right but not considering the last row).

of subproblems defined by pruning edges, and reducing the degrees of freedom by taking into account already solved subproblems, the resulting algorithm appears to be quite efficient in sparse DMDGP instances, sometimes giving a significant speed-up with respect to the classic BP algorithm.

We reinforce that the numerical experiments in [12] showed that the classic BP has a better performance than continuous methods [25,10] in problems of the DMDGP subclass. Thus, the new proposed algorithm can do even better and, to the best of our knowledge, figures as one of the fastest algorithms for this class of Distance Geometry problems.

In the proposed version of SBBU algorithm we consider a specific order for the pruning edges. In future works we expect to generalize SBBU in order to handle different pruning edges orderings and study the impact of these in the total cost W.

A Proof of Lemma 2

Proof Since $x(s')$ from Eq. (4) involves only partial reflections, in view of Property 3 in Remark 2, $x(s')\in\hat{X}$, i.e $\|x_u(s')-x_w(s')\|=d_{uw},\forall\{u,w\}\in E_D$.

It remains to show that $x(s')$ does not violate distance constraints associated to pruning edges in P^{ij}. Since the reflections are applied to positions x_ℓ such that $\ell\geq i+K+1$, pruning edges $\{u,w\}\in P^{ij}$ with $u<w\leq i+K$ are not affected. Thus, assume that $i+K+1\leq w\leq n$. If $u\leq i$, then for $\ell=i+K+1,\ldots,w$ there exists $\{u,w\}$ such that $u+K+1\leq\ell\leq w$, which implies that $v_{i+K+1},\ldots,v_w\notin S^{ij}$, meaning that the first symmetry vertex v_ℓ in S^{ij} is such that $\ell\geq w+1$. Thus, according to (2) and (4), partial reflections are not applied to $x_{i+K+1},\ldots,x_w$, i.e $x_\ell(s')=x_\ell(s)$, for $\ell=i+K+1,\ldots,w$ and $\|x_u(s')-x_w(s')\|=d_{uw}$ holds. Otherwise, for $u\geq i+1$, we have that $v_{u+K+1},\ldots,v_w\notin S^{ij}$, and from (4) and (2), positions $x_\ell,\ldots,x_{u+K+1},\ldots,x_w$ are updated by reflections $R^\ell_x(x_\ell),\ldots,R^\ell_x(x_{u+K+1}),\ldots,R^\ell_x(x_w)$, for $i+K+1\leq\ell\leq u+K$ such that $v_\ell\in S^{ij}$. Since either $u\leq\ell-1$, i.e x_u is in the hyperplane associated to v_ℓ, or $u\geq\ell$, i.e. x_u comes after this hyperplane, in view of Remark 2, Property 2, these reflections are such that $\|x_u(s')-x_v(s')\|=d_{uw}$. □

Acknowledgements We would like to thank Prof. Luiz M. Carvalho for valuable discussions. This work was partly supported by: (a) the Brazilian research agencies CNPq, CAPES, and FAPESP; (b) the French research agency ANR under grant ANR-19-CE45-0019 “multiBioStruct”; (c) the European Union’s Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement n. 764759 ETN ”MINOA”. Part of this work was done during the visit of DG to LL at École Polytechnique, supported by CAPES/Print Process 88887.465828/2019-00.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Alencar, J., Lavor, C., Liberti, L.: Realizing Euclidean distance matrices by sphere intersection. Discrete Applied Mathematics **256**, 5–10 (2019)
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 Research **28**, 235–242 (2000)
3. Billinge, S., Duxbury, P., Gonçalves, D., Lavor, C., Mucherino, A.: Assigned and unassigned distance geometry: applications to biological molecules and nanostructures. 4OR **14**, 337–376 (2016)
4. Billinge, S., Duxbury, P., Gonçalves, D., Lavor, C., Mucherino, A.: Recent results on assigned and unassigned distance geometry with applications to protein molecules and nanostructures. Annals of Operations Research **271**, 161–203 (2018)
5. Cassioli, A., Günlük, O., Lavor, C., Liberti, L.: Discretization vertex orders in distance geometry. Discrete Applied Mathematics **197**, 27 – 41 (2015). Distance Geometry and Applications
6. Crippen, G., Havel, T.: Distance Geometry and Molecular Conformation. Wiley (1988)
7. Dokmanic, I., Parhizkar, R., Ranieri, J., Vetterli, M.: Euclidean distance matrices: Essential theory, algorithms, and applications. Signal Processing Magazine, IEEE **32**(6), 12–30 (2015)
8. Fidalgo, F., Gonçalves, D., Lavor, C., Liberti, L., Mucherino, A.: A symmetry-based splitting strategy for discretizable distance geometry problems. Journal of Global Optimization **71**, 717–733 (2018)
9. Gramacho, W., Mucherino, A., Lavor, C., Maculan, N.: A parallel BP algorithm for the discretizable distance geometry problem. In: Proceedings of the Workshop on Parallel Computing and Optimization, pp. 1756–1762. IEEE, Piscataway (2012)
10. Krislock, N., Wolkowicz, H.: Explicit sensor network localization using semidefinite representations and facial reductions. SIAM Journal on Optimization **20**, 2679–2708 (2010)
11. Lavor, C., Liberti, L., Donald, B., Worley, B., Bardiaux, B., Malliavin, T., Nilges, M.: Minimal nmr distance information for rigidity of protein graphs. Discrete Applied Mathematics **256**, 91–104 (2019)
12. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: The discretizable molecular distance geometry problem. Computational Optimization and Applications **52**, 115–146 (2012)
13. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. European Journal of Operational Research **219**, 698–706 (2012)
14. Lavor, C., Souza, M., Carvalho, L.M., Liberti, L.: On the polynomiality of finding ^{K}DMDGP re-orders. Discrete Applied Mathematics **267**, 190–194 (2019)
15. Liberti, L.: Distance geometry and data science. TOP **28**, 271–339 (2020)
16. Liberti, L., Lavor, C.: Euclidean Distance Geometry: An Introduction. Springer (2017)
17. Liberti, L., Lavor, C., Alencar, J., Abud, G.: Counting the number of solutions of ^{k}DMDGP instances. In: F. Nielsen, F. Barbaresco (eds.) Geometric Science of Information, *Lecture Notes in Computer Science*, vol. 8085, pp. 224–230. Springer Berlin Heidelberg (2013)

18. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. International Transactions in Operational Research **15**, 1–17 (2008)
19. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. SIAM Review **56**, 3–69 (2014)
20. Liberti, L., Lavor, C., Mucherino, A.: The Discretizable Molecular Distance Geometry Problem seems easier on proteins. In: A. Mucherino, C. Lavor, L. Liberti, N. Maculan (eds.) Distance Geometry, pp. 47–60. Springer New York (2013)
21. Liberti, L., Masson, B., Lee, J., Lavor, C., Mucherino, A.: On the number of solutions of the discretizable molecular distance geometry problem. In: Combinatorial Optimization, Constraints and Applications (COCOA11), *LNCS*, vol. 6831, pp. 322–342. Springer, New York (2011)
22. Liberti, L., Masson, B., Lee, J., Lavor, C., Mucherino, A.: On the number of realizations of certain Henneberg graphs arising in protein conformation. Discrete Applied Mathematics **165**, 213–232 (2014)
23. Maioli, D., Lavor, C., Gonçalves, D.S.: A note on computing the intersection of spheres in $\mathbb{R}^n$. The ANZIAM Journal **59**(2), 271–279 (2017)
24. Malliavin, T., Mucherino, A., Lavor, C., Liberti, L.: Systematic exploration of protein conformational space using a distance geometry approach. Journal of Chemical Information and Modeling **59**, 4486–4503 (2019)
25. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. Journal of Global Optimization **15**, 219–234 (1999)
26. Mucherino, A., Lavor, C., Liberti, L.: Exploiting symmetry properties of the discretizable molecular distance geometry problem. Journal of Bioinformatics and Computational Biology **10**(3), 1242009(1–15) (2012)
27. Mucherino, A., Lavor, C., Liberti, L., Maculan, N. (eds.): Distance Geometry: Theory, Methods, and Applications. Springer, Berlin (2013)
28. Newman, M.E., Ziff, R.M.: Fast Monte Carlo algorithm for site or bond percolation. Physical Review E **64**(1), 016706 (2001)
29. Nucci, P., Nogueira, L., Lavor, C.: Solving the discretizable molecular distance geometry problem by multiple realization trees. In: Mucherino et al. [27], pp. 161–176
30. Saxe, J.B.: 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. Monticello, IL (1979)
31. Wütrich, K.: Protein structure determination in solution by nuclear magnetic resonance spectroscopy. Science **243**, 45–50 (1989)