Maximum feasible subsystems of distance geometry constraints

Maurizio Bruglieri, Roberto Cordone, Leo Liberti

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

Maximum feasible subsystems of distance geometry constraints

Maurizio Bruglieri · Roberto Cordone · Leo Liberti

the date of receipt and acceptance should be inserted later

Abstract We study the problem of satisfying the maximum number of distance geometry constraints with minimum experimental error. This models the determination of the shape of proteins from atomic distance data which are obtained from nuclear magnetic resonance experiments and exhibit experimental and systematic errors. Experimental errors are represented by interval constraints on Euclidean distances. Systematic errors occur from a misassignment of distances to wrong atomic pairs: we represent such errors by maximizing the number of satisfiable distance constraints. We present many mathematical programming formulations, as well as a "matheuristic" algorithm based on reformulations, relaxations, restrictions and refinement. We show that this algorithm works on protein graphs with hundreds of atoms and thousands of distances.

Keywords Protein conformation, MINLP, diagonally dominant programming.

1 Introduction

We discuss an interesting hybrid of two problems: the Maximum Feasible Subsystem (MaxFS) [17] and the Distance Geometry Problem (DGP) [25], and its application to the problem of determining the spatial conformation of proteins from distance data derived from Nuclear Magnetic Resonance (NMR) experiments.

One of the authors (LL) was partly funded by the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement n. 764759.

Maurizio Bruglieri
Dipartimento di Design, Politecnico di Milano, Italy
E-mail: maurizio.bruglieri@polimi.it

Roberto Cordone
Dipartimento di Informatica, Università degli Studi di Milano, Italy
E-mail: roberto.cordone@unimi.it

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

The MaxFS is as follows: given a set of constraints, generally of the form

$$\forall i \in M \qquad g_i^L \leq g_i(x) \leq g_i^U, \tag{1}$$

determine a subset $S \subseteq M$ of maximum cardinality such that the set of constraints of Eq. (1) indexed by S is feasible. This problem is **NP**-hard to solve, and does not admit a polynomial-time approximation scheme unless $\mathbf{P} = \mathbf{NP}$ [5].

The (Euclidean) DGP is as follows: given an integer $K > 0$ and a simple connected edge-weighted graph $G = (V, E, d)$, where $d : E \to \mathbb{R}_+$, determine whether there exists a *realization* $x : V \to \mathbb{R}^K$ such that:

$$\forall \{i, j\} \in E \qquad \|x_i - x_j\|_2 = d_{ij}. \tag{2}$$

There are many applications of the DGP [25] and even more variants. The one we are specially interested in is the *interval* DGP (iDGP), which replaces $d : E \to \mathbb{R}_+$ with the interval weight function $d : E \to \mathbb{IR}_+$ such that $d(\{i, j\}) = [L_{ij}, U_{ij}]$ [21, 15]. Specifically, Eq. (2) becomes

$$\forall \{i, j\} \in E \qquad L_{ij} \leq \|x_i - x_j\|_2 \leq U_{ij}. \tag{3}$$

The iDGP is **NP**-hard by inclusion of the DGP: the inclusion follows from the case $L_{ij} = U_{ij}$ for all $\{i, j\} \in E$. We note that if $L_{ij} = 0$ and U_{ij} is infinite (for all $\{i, j\} \in E$) the problem is clearly tractable since any realization is feasible. The existence of a "phase transition" threshold between tractability and hardness of this problem is an open question [1]. From now on, we shall assume all norms are Euclidean (a.k.a. ℓ_2) unless stated otherwise. We also denote $n = |V|$ and $m = |E|$.

We are now in the position of stating the main problem discussed in this paper.

> Max Feasible Subsystem of Distance Geometry constraints ($\text{MaxFS}_{\text{DGP}}$). Given an integer $K > 0$ and a simple connected edge-weighted graph $G = (V, E, d)$ with $d : E \to \mathbb{IR}_+$, determine the maximum cardinality subset $S \subseteq E$ inducing a subgraph of G, such that there exists a realization $x : V[E] \to \mathbb{R}^K$ satisfying
>
> $$\forall \{i, j\} \in S \qquad L_{ij} \leq \|x_i - x_j\| \leq U_{ij}. \tag{4}$$

We recall that $d(\{i, j\}) = [L_{ij}, U_{ij}]$ for each $\{i, j\} \in E$, as explained above. The connectedness assumption, as in many problems on graphs, is without loss of generality: for a disconnected graph it suffices to find the connected components in polynomial time, and solve the problem on each connected component independently.

The $\text{MaxFS}_{\text{DGP}}$ is motivated by a specific application of the DGP, namely the determination of the shape of proteins given some of their inter-atomic distances. In principle, NMR can determine all inter-atomic distances in a given protein up to a certain length threshold (somewhere between 5Å and 6Å). In practice, reality is fuzzier than this. First, we note that proteins rarely crystallize (so X-ray crystallography does not help), but usually live in a solution. Secondly, proteins vibrate, but we assume that they do not (this is called the "molecular rigidity assumption") [28].

NMR experiments yield a probability distribution over triplets (atom label, atom label, distance value); this distribution is used to imperfectly reconstruct the weighted graph G that is the actual input to the DGP. It is not easy to understand

the methods used by NMR machinery to perform this reconstruction, other than they are mostly based on simulated annealing [31]; attempts to construct the conformations starting directly from NMR output are under way [29]. According to [8], this process induces two types of errors: experimental errors (due to the rigidity assumption), and systematic errors (due to the imperfect reconstruction). Specifically, the experimental errors are accommodated by the interval bounds on the iDGP. The systematic errors are described in [8] as consisting of a certain proportion of completely wrong distances. This induces sets of constraints in (3) that are likely to be infeasible. We propose the $\text{MaxFS}_{\text{DGP}}$ in order to address the issue and find solutions subject to such limitations.

The focus of this paper is to solve $\text{MaxFS}_{\text{DGP}}$ instances using formulations and solution methods from Mathematical Programming (MP). In particular, we present exact formulations and several reformulations thereof. We construct a practically viable solution method based on solving an approximate matrix formulation of $\text{MaxFS}_{\text{DGP}}$ followed by rank reduction and a refinement phase. Since Polynomial Programs (PP) offer more reformulation opportunities than with general Nonlinear Programs (NLP), ℓ_2 norm terms are always squared.

The rest of this paper is organized as follows. In Sect. 2 we introduce an exact formulation of the $\text{MaxFS}_{\text{DGP}}$ problem. In Sect. 3 we construct a relaxation in the same primal variables as the exact formulation. In Sect. 4 we construct some matrix relaxations, and propose methodologies for reducing solution rank and improving the quality of the low-rank solutions. In Sect. 5 we describe two computational approaches to solving $\text{MaxFS}_{\text{DGP}}$ instances, based on the formulations of Sect. 3 and 4, and discuss computational results obtained from protein instances of small and medium sizes.

2 Exact formulation

In this section we present a Mathematical Programming (MP) formulation of the $\text{MaxFS}_{\text{DGP}}$ problem.

2.1 Experimental errors

Experimental errors are addressed by minimizing the infeasibilities w.r.t. Eq. (2) or Eq. (3). A well-known box-constrained formulation targeting the DGP is:

$$\min_{x \in [x^L, x^U]} \sum_{\{i,j\} \in E} (\|x_i - x_j\|^2 - d_{ij}^2)^2, \tag{5}$$

where x^L, x^U are given lower and upper bounds for the decision variable $n \times K$ matrix $x = (x_1, \ldots, x_n)$. Eq. (5) was tested computationally in e.g. [20]. A corresponding reformulation for the iDGP can be obtained replacing each term $\|x_i - x_j\|^2 - d_{ij}^2$ of Eq. (5) with

$$\max(0, L_{ij}^2 - \|x_i - x_j\|^2) + \max(0, \|x_i - x_j\|^2 - U_{ij}^2),$$

so that setting the objective to zero corresponds to the (nonconvex) pure feasibility problem:

$$\left.\begin{array}{r} \forall \{i,j\} \in E \ \|x_i - x_j\|^2 \geq L_{ij}^2 \\ \forall \{i,j\} \in E \ \|x_i - x_j\|^2 \leq U_{ij}^2 \\ x^L \leq x \leq x^U. \end{array}\right\} \tag{6}$$

We remark that x^L, x^U may or may not be provided by the specific DGP application being tackled; but that, even when they are not explicitly provided, Prop. 2.2 below can be used to derive them. We also note that Eq. (5) and Eq. (6) can be solved without bound constraints. In the latter case, it may be advisable to impose a zero centroid constraint instead, which removes translation invariance:

$$\frac{1}{n} \sum_{i \in V} x_i = 0. \tag{7}$$

Henceforth, we shall refrain (for brevity) from mentioning bound or zero centroid constraints on realization variables in the MP formulations below, unless the context requires it.

2.2 Systematic errors

The $\text{MaxFS}_{\text{DGP}}$ can be formulated in a natural way, using so-called "big-M" techniques and binary decision variables to linearly represent disjunctions as follows [4]:

$$\left.\begin{array}{r} \max \sum_{\{i,j\} \in E} y_{ij} \\ \forall \{i,j\} \in E \ d_{ij}^2 - M(1-y_{ij}) \leq \|x_i - x_j\|^2 \leq d_{ij}^2 + M(1-y_{ij}) \\ y \in \{0,1\}^m. \end{array}\right\} \tag{8}$$

In the above, it is clear that $y_{ij} = 1$ enforces the constraint on edge $\{i,j\}$ in the DGP, i.e. Eq. (2), whereas $y_{ij} = 0$ relaxes it.

Note that, since distances are always non-negative, the LHS of the distance constraints in Eq. (8) can be tightened to $d_{ij}^2 y_{ij}$. This yields:

$$\left.\begin{array}{r} \max \sum_{\{i,j\} \in E} y_{ij} \\ \forall \{i,j\} \in E \ d_{ij}^2 y_{ij} \leq \|x_i - x_j\|^2 \leq d_{ij}^2 + M(1-y_{ij}) \\ y \in \{0,1\}^m. \end{array}\right\} \tag{9}$$

Lemma 2.1 *The y component of every optimal solution for the MaxFS_{DGP} defines a connected graph.*

Proof Suppose, by contradiction, that an optimal solution is made up by at least two connected components C_1 and C_2. Since we consider the $\text{MaxFS}_{\text{DGP}}$ defined on a connected graph, there exists a path given by the sequence of vertices v_1, $v_2, \ldots, v_k$ (with $k \geq 2$) connecting C_1 with C_2 where v_1 belongs to C_1, v_k belongs to C_2 and $v_2, \ldots, v_{k-1}$ belong neither to C_1 nor to C_2. Finding a realization for the vertices of the path is trivial and it is always possible to move as a whole both C_1

and C_2 in such a way to continue to respect their internal distance constraints and the realization of v_1 and v_k. Therefore, adding the path to the starting solution we obtain a feasible solution for the MaxFS$_{\mathrm{DGP}}$ satisfying the distances of $k-1$ (≥ 1) additional edges. This contradicts the optimality of the starting solution. □

Proposition 2.2 *If $M = (\sum_{\{i,j\}\in E} d_{ij})^2$, then the optimal solution of Eq. (8) solves the MaxFS$_{DGP}$.*

Proof First, we claim that any feasible DGP instance can be realized in a sphere of radius $R = \frac{1}{2}\sum_{\{i,j\}\in E} d_{ij}$. A cycle graph C on $V = \{1, 2, \ldots, n\}$ with $E = \{\{1,2\},\{2,3\},\ldots,\{n-1,n\},\{1,n\}\}$ with $d_{1n} = \sum_{\{i,j\}\in E\setminus\{1,n\}} d_{ij}$ can be realized on a straight segment of length $r = d_{1n}$ embedded in any Euclidean space [33]; if this segment is centered about the origin it belongs by construction to the sphere $R\mathbb{S}^{K-1}$. Any other biconnected graph on n vertices will have more cycles than C, and hence will induce realizations in $\mathbb{R}^K$ having segments shorter than $2R$ when projected on any coordinate axis. Connected but non-biconnected graphs are the same as trees: the tree yielding a realization with longest segment projection on any coordinate axis is the path on n vertices realized as a segment of length $2R$; again, by centering the segment it is easy to see that the path can be realized in a sphere of radius R. Lastly, we note that the above claim also shows that the maximum possible distance between two vertices i, j in a realization of an optimal solution for the MaxFS$_{\mathrm{DGP}}$ is $2R$, since it must induce a connected subgraph according to Lemma 2.1. This shows that if a MaxFS$_{\mathrm{DGP}}$ instance has a solution with a certain support vector y^* for the maximum cardinality set of feasible constraints, then setting $y = y^*$ in Eq. (8) will induce a valid realization x^* of the subgraph consisting of the edges $\{i, j\}$ for which $y^*_{ij} = 1$, and vice versa. □

In practice, segment realizations are extremely rare, and therefore M can be tightened w.r.t. Prop. 2.2. We remark that bounds on M can also be inferred from x^L, x^U, if they are given; and, conversely, that $[x^L, x^U]$ can be set to $[-\sqrt{M}, \sqrt{M}]$ if the application field does not explicitly provide them.

2.3 Systematic and experimental errors together

We consider Eq. (6) and employ the y binary variables as in Eq. (9) to activate/deactivate the distance constraints. This yields a valid MP formulation for the MaxFS$_{\mathrm{DGP}}$, as follows:

$$\left.\begin{array}{rl} \max \sum\limits_{\{i,j\}\in E} y_{ij} & \\ \forall \{i,j\} \in E \ \|x_i - x_j\|^2 \geq L_{ij}^2 y_{ij} & \text{(eL)} \\ \forall \{i,j\} \in E \ \|x_i - x_j\|^2 \leq U_{ij}^2 + M(1-y_{ij}) & \text{(eU)} \\ y \in \{0,1\}^m & \\ x^L \leq x \leq x^U. & \end{array}\right\} \tag{10}$$

The correctness of Eq. (10) is easy to establish: constraints (eL) and (eU) ensure that $\|x_i - x_j\|$ is in the desired interval $[L_{ij}, U_{ij}]$ as long as $y_{ij} = 1$, i.e. the $\{i,j\}$-th constraint is imposed. Otherwise $\|x_i - x_j\| \in [0, M]$, meaning that the constraint

is relaxed. The objective function ensures that as many constraints as possible are imposed.

Moreover, according to [8], the fraction p of wrong distances can be estimated statistically *a priori*. We can encode this knowledge by means of the additional cardinality constraint:

$$\sum_{\{i,j\}\in E} y_{ij} \geq (1-p)\, m, \tag{11}$$

which makes Eq. (10) infeasible whenever Eq. (11) is not satisfied.

Eq. (10) is a nonconvex Mixed-Integer Nonlinear Program (MINLP), which represents one of the hardest classes in MP. MINLP is an uncomputable class, in general [26]. For bounded decision variables (such as in Eq. (10)) it is computable, but **NP**-hard [22].

The state of the art in MINLP solution is not as advanced as for Mixed-Integer Linear Programming (MILP), for which, despite the hardness, relatively large scale instances can be solved either to optimality or at least to feasibility. Preliminary tests on Eq. (10)-(11) showed that no feasible solutions can be found in a given maximum CPU time limit. Removing Eq. (11) simply yielded the trivial solution with $y_{ij} = 0$ for all $\{i,j\} \in E$, again, with a given maximum CPU time limit.

In the rest of the paper we shall investigate reformulations of many types in order to obtain "solver-friendlier" MP formulations for the $\text{MaxFS}_{\text{DGP}}$. This investigation involves MINLP relaxations in the original nK-dimensional space of realizations (Sect. 3), as well as MILP approximations in a larger $\frac{n(n+1)}{2}$-dimensional space of symmetric matrices (Sect. 4).

The order of presentation of these formulations is dictated by a "mathematical programmer's common sense" regulated by computational experience: we believe that a certain reformulation might solve the issues of the preceding formulation; then, during preliminary testing, we discover new issues. Thus, while the whole reformulation sequence explains how we came to the one that actually works in practice, we do not systematically test every formulation we present. Specifically, in the computational results Sect. 5 we only solve Eq. (16) and test a solution method based on the pair (Eq. (23), Eq. (26)).

3 Primal relaxations and approximations

The fact that solving Eq. (10) with a CPU time limit yields a solution where all y variables are set to zero is a witness to the empirical observation that feasibility is harder to achieve than optimality. Setting $y_{ij} = 0$ and $x_i = x_j$ for all $\{i,j\} \in E$ yields a feasible solution with the worst possible objective function value. Obviously, the major contributors to the feasibility issue are the constraints (eL) and (eU) in Eq. (10). We note that, when the integrality of the y variables is relaxed, (eL) is nonconvex and (eU) is convex. We shall discuss relaxations and approximations of (eL) in Sect. 4.1. In this section, we introduce a method for approximating (eU).

First, we replace U_{ij}^2 by $U_{ij}^2 y_{ij}$, and $M(1-y_{ij})$ by an additional variable $r_{ij} \geq 0$, for $\{i,j\} \in E$. We then make sure that $r_{ij} = 0$ whenever $y_{ij} = 1$. This yields:

$$\forall \{i,j\} \in E \quad \|x_i - x_j\|^2 \leq U_{ij}^2 y_{ij} + r_{ij} \tag{12}$$

$$\forall \{i,j\} \in E \quad 0 \leq r_{ij} \leq M(1-y_{ij}). \tag{13}$$

This is an exact reformulation whose only difference with Eq. (10) is that the systematic error w.r.t. U_{ij} is represented by r_{ij} for each edge $\{i,j\} \in E$.

3.1 Bi-objective relaxation

Some preliminary experiments on the exact formulation based on Eq. (12)-(13) above presented us with the following unusual issue: our Branch-and-Bound (BB) solver of choice failed to find any feasible solution using the value of M given by Prop. 2.2, which meant that the BB algorithm could never prune by bound, which then resulted in a rapid exponential growth of the BB tree. Setting M to unreasonably large values allowed the solver to find a feasible solution rapidly. While, as is well known, a large M yields a slacker bound, it is obviously better to prune by bound ineffectively than not at all. The issue, however, is now that of finding good values of M, which — in our preliminary experiments at least — appeared to vary considerably by instance.

This motivated us to construct a formulation where Eq. (13) is turned from a hard constraint into a soft one, making sure that r_{ij} remains small. We achieve this by introducing a second objective function $\min \sum_{ij} r_{ij}$. This yields a bi-objective MINLP relaxation

$$\max \sum_{\{i,j\}\in E} y_{ij} \tag{14}$$

$$\min \sum_{\{i,j\}\in E} r_{ij} \tag{15}$$

subject to Eq. (12), (eL), and the other constraints in Eq. (10) aside from (eU). Note that Eq. (13) is relaxed. Moreover, Eq. (14) addresses the systematic error and Eq. (15) addresses the experimental error. We denote this formulation by (B).

While bi-objective programs can hardly be considered exact reformulations of single-objective ones, in the following we give some limited sufficient and necessary conditions linking optimality in Eq. (10) and Pareto optimality in (B).

Note that any solution (x^*, y^*) feasible in Eq. (10) can be extended to a solution of (B) by setting $r^*_{ij} = 0$ iff $y^*_{ij} = 1$ and $r^*_{ij} = \|x^*_i - x^*_j\|^2$ otherwise. Whenever (x^*, y^*) is feasible in Eq. (10) and the existence of r^* is implicitly referred to in the text, we assume it was computed from x^*, y^* as above. For a binary vector $y \in \{0,1\}^m$ we denote by $\mathsf{supp}(y)$ the set of indices $e = (i,j) \in E$ for which $y_{ij} = 1$.

3.1.1 Sufficient optimality conditions

We first prove that Pareto optimal values of (B) are good candidate optima for Eq. (10) as long as one knows the optimal value.

Lemma 3.1 *Suppose (x', y', r') is such that: (i) it is feasible in* (B)*; (ii) it has $r'_{ij} = 0$ for all $\{i,j\} \in E$ with $y'_{ij} = 1$. Then (x', y') is feasible in Eq.* (10)*.*

Proof This follows by inspection of Eq. (13), which is the only constraint equivalent to (eU) in Eq. (10) which is relaxed in (B). □

Proposition 3.2 *Suppose* (x', y', r') *is such that: (i) it is Pareto optimal in* (B)*; (ii) the value of Eq.* (14) *is optimal for Eq.* (10)*. Then* (x', y') *is optimal in Eq.* (10)*.*

Proof Consider any edge $\{h, \ell\} \in E$ with $y'_{h\ell} = 1$, and suppose $r'_{h\ell} > 0$. Define $\hat{r}$ s.t. $\hat{r}_{ij} = r'_{ij}$ for all $\{i, j\} \neq \{h, \ell\}$, and $\hat{r}_{h\ell} = 0$. By Eq. (12) the solution $(x', y', \hat{r})$ is feasible in (B) and dominates (x', y', r') since

$$\sum_{\{i,j\}\in E} \hat{r}_{ij} = \sum_{\substack{\{i,j\}\in E \\ \{i,j\}\neq\{h,\ell\}}} r'_{ij} < \sum_{\{i,j\}\in E} r'_{ij},$$

against the assumption. Therefore $r'_{h\ell} = 0$, which, by Lemma 3.1, implies that (x', y') is feasible in Eq. (10). Optimality follows by (ii). □

3.1.2 Necessary optimality conditions

Next, we give a support-dependent characterization of Pareto dominance in (B) w.r.t. optimality in Eq. (10).

Theorem 3.3 *No optimal solution* (x^*, y^*) *of Eq.* (10) *can be dominated in* (B) *by a Pareto solution* (x', y', r') *of* (B) *where* $\mathsf{supp}(y^*) \subseteq \mathsf{supp}(y')$*.*

Proof Suppose first that (x^*, y^*) is dominated by (x', y', r') w.r.t. the objective function Eq. (15), i.e. $\sum_{ij} r'_{ij} < \sum_{ij} r^*_{ij}$. For all $\{i, j\}$ with $y^*_{ij} = 1$ we have $r^*_{ij} = 0$ by Eq. (13) (and also $y'_{ij} = 1$ since $\mathsf{supp}(y^*) \subseteq \mathsf{supp}(y')$). So, no decrease in Eq. (15) can be achieved over the edges $\{i, j\}$ for which $y^*_{ij} = y'_{ij} = 1$. For all $\{i, j\}$ with $y^*_{ij} = y'_{ij} = 0$ we have $r^*_{ij} = \|x^*_i - x^*_j\|^2$ by definition. Note that the relaxation of Eq. (13) does not change the feasible region restricted to $y_{ij} = 0$, so we can assume without loss of generality that $x^* = x'$ over edges $\{i, j\} \in E$ for which $y^*_{ij} = y'_{ij} = 0$, whence $r'_{ij} \geq \|x'_i - x'_j\|^2 = \|x^*_i - x^*_j\|^2 = r^*_{ij}$ by Eq. (12). Because of the optimization direction of Eq. (15), we have $r'_{ij} = r^*_{ij}$ for all $\{i, j\}$ s.t. $y^*_{ij} = y'_{ij} = 0$. So, again, no decrease in Eq. (15) can be achieved over these edges. The only possible decrease in the value of Eq. (15) must therefore occur over edges $\{i, j\} \in E$ where $0 = y^*_{ij} < y'_{ij} = 1$. But then this means that (x^*, y^*) is dominated w.r.t. the objective Eq. (14).

We therefore assume that (x^*, y^*) is dominated by (x', y', r') w.r.t. Eq. (14). Then $\sum_{ij} y'_{ij} > \sum_{ij} y^*_{ij}$. By optimality of (x^*, y^*) in Eq. (10), (x', y') cannot be feasible in Eq. (10), which means that (x', y', r') does not satisfy Eq. (13), i.e. $r'_{h\ell} > 0$ for some edge $\{h, \ell\} \in E$ where $y'_{h\ell} = 1$. Since $\mathsf{supp}(y^*) \subseteq \mathsf{supp}(y')$, and $r^*_{ij} = 0$ whenever $y^*_{ij} = 1$, we have

$$\begin{aligned} &\sum_{\substack{\{i,j\}\in E \\ \{i,j\}\neq\{h,\ell\}}} r^*_{ij} \leq \sum_{\substack{\{i,j\}\in E \\ \{i,j\}\neq\{h,\ell\}}} r'_{ij} \\ \Rightarrow \quad &\sum_{\{i,j\}\in E} r^*_{ij} < \sum_{\{i,j\}\in E} r'_{ij}. \end{aligned}$$

Thus, (x', y', r') does not dominate (x^*, y^*), as claimed. We therefore must have $\sum_{ij} y^*_{ij} = \sum_{ij} y'_{ij}$. Moreover, since $\mathsf{supp}(y^*) \subseteq \mathsf{supp}(y')$, we also have $y' = y^*$, yielding $\sum_{ij} r^*_{ij} = \sum_{ij} r'_{ij}$. □

3.2 Scalarized approximation

Finally, we derive a weighted scalarization of Eq. (14)-(15), with Eq. (14) weighing more than Eq. (15). In summary, we obtain the formulation:

$$\left.\begin{array}{rl} \max & \sum\limits_{\{i,j\}\in E} y_{ij} - \alpha \sum\limits_{\{i,j\}\in E} r_{ij} \\ \forall \{i,j\} \in E & \|x_i - x_j\|^2 \geq L_{ij}^2 y_{ij} \\ \forall \{i,j\} \in E & \|x_i - x_j\|^2 \leq U_{ij}^2 y_{ij} + r_{ij} \\ & y \in \{0,1\}^m \\ & r \in \mathbb{R}^m_{\geq 0} \\ & x^L \leq x \leq x^U, \end{array}\right\} \tag{16}$$

where $\alpha \geq 0$. Eq. (16) is still a nonconvex MINLP, and thus remains very challenging to solve. However, it has no "big-M" constraint.

4 Matrix relaxations and rank reduction

In this section we look at a Mixed-Integer Semidefinite Programming (MISDP) relaxation, followed by a Mixed-Integer Diagonally Dominant Programming (MIDDP) restriction thereof, which provides a Mixed-Integer Linear Programming (MILP) approximation of Eq. (16). We then show how to reduce the rank of the $n \times n$ matrix solution of these formulations to an $n \times K$ matrix representation of the realization we look for.

4.1 MISDP relaxation

The standard derivation of a Semidefinite Programming (SDP) relaxation from MPs involving Euclidean distance terms consists in writing them as

$$\|x_i - x_j\|^2 = \|x_i\|^2 + \|x_j\|^2 - 2x_i \cdot x_j,$$

and then linearizing the nonlinear terms $\|x_i\|^2$ and $x_i \cdot x_j$ by added variables X_{ii}, X_{ij} respectively, resulting in $\|x_i - x_j\|^2$ being replaced by

$$X_{ii} + X_{jj} - 2X_{ij},$$

which is linear in X. The effect of this replacement yields an exact reformulation as long as the constraints

$$X = xx^\top \tag{17}$$

are satisfied. Note that Eq. (17) are nonconvex constraints. We relax them by

$$X \succeq xx^\top,$$

which define a convex set. We remark that, if the x variables appear nowhere else in the formulation, it suffices to enforce $X \succeq 0$ (see e.g. [13]). In order to eliminate x from the formulation, we can relax the bounds x^L, x^U in Eq. (16), which are inessential by translation invariance.

The application of the above reformulation to Eq. (16) yields

$$\left.\begin{array}{rr} \max & \sum\limits_{\{i,j\}\in E} y_{ij} - \alpha \sum\limits_{\{i,j\}\in E} r_{ij} \\ \forall\{i,j\} \in E \; X_{ii} + X_{jj} - 2X_{ij} \geq L_{ij}^2 y_{ij} \\ \forall\{i,j\} \in E \; X_{ii} + X_{jj} - 2X_{ij} \leq U_{ij}^2 y_{ij} + r_{ij} \\ y \in \{0,1\}^m \\ r \in \mathbb{R}^m_{\geq 0} \\ X \succeq 0, \end{array}\right\} \tag{18}$$

which is a MISDP formulation.

4.2 MIDDP approximations

Diagonally Dominant Programming (DDP) was proposed in [2,3] as an MP-based approximation technique for the positive semidefinite (PSD) cone, yielding both inner and outer approximating formulations in the Linear Programming (LP) or Second-Order Cone Programming (SOCP) classes. Since MILP solvers are more technologically advanced than their conic counterparts, in this section we only discuss the LP approximation.

A matrix $A = (A_{ij})$ is diagonally dominant (DD) whenever

$$\forall i \quad A_{ii} \geq \sum_{j \neq i} |A_{ij}|. \tag{19}$$

DDP rests on the observation that all DD matrices are PSD, a fact which follows easily by Gershgorin's circle theorem [14]. Since Eq. (19) can be represented by a set of linear inequalities, solving LPs over the DD cone $\mathcal{D}$ instead of the PSD cone $\mathcal{S}_+$ yields a PSD matrix solution at the cost of solving a LP problem.

On the other hand, not all PSD matrices are DD, which implies that $\mathcal{D} \subsetneq \mathcal{S}_+$: the feasible region of an SDP problem might be non-empty while the corresponding (inner) DDP problem is infeasible. In such cases, one may resort to the outer DDP problem, which is derived using the dual DD cone $\mathcal{D}^*$. On the other hand, $S_+ \subsetneq \mathcal{D}^*$, so the matrix solution of the outer DDP problem may not be PSD. More details about applying DDP to the DGP are given in [13,23].

4.2.1 Inner restriction

We focus on the inner approximation first, since, if it is feasible, it provides a PSD matrix as a solution, which is crucial to further processing. For the MISDP problem in Eq. (18), we simply replace $X \succeq 0$ with "X is DD", i.e.

$$\forall i \in V \quad X_{ii} \geq \sum_{j \neq i} |X_{ij}|. \tag{20}$$

This is easily reformulated to a linear form by introducing a linearizing $n \times n$ symmetric matrix T for the nonlinear term $|X|$. We then obtain:

$$\forall i \in V \quad X_{ii} \geq \sum_{j \neq i} T_{ij} \tag{21}$$

$$-T \leq X \leq T. \tag{22}$$

The exact reformulation proof between Eq. (20) and (21)-(22) is hinted at in [2] to proceed by projection of the T variables. A more direct argument (by contradiction of optimality) can be obtained by considering the objective function $\min \sum_{ij} T_{ij}$ (a corresponding term $-\sum_{ij} T_{ij}$ may optionally be added to the objective of Eq. (23)). In summary, we have the following MILP:

$$\left.\begin{array}{rr} \max & \sum\limits_{\{i,j\}\in E} y_{ij} - \alpha \sum\limits_{\{i,j\}\in E} r_{ij} \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} \geq L_{ij}^2 y_{ij} \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} \leq U_{ij}^2 y_{ij} + r_{ij} \\ \forall i < j \leq n & X_{ij} = X_{ji} \\ \forall i \leq V & \sum\limits_{j \neq i} T_{ij} \leq X_{ii} \\ & -T \leq X \leq T \\ & T, X \in \mathbb{R}^{n \times n} \\ & y \in \{0,1\}^m \\ & r \in \mathbb{R}^m_{\geq 0}. \end{array}\right\} \tag{23}$$

4.2.2 Outer relaxation

An outer relaxation of the MISDP formulation Eq. (18) can be obtained using DDP techniques and the dual DD cone $\mathcal{D}^*$.

The formulation replaces the implicit constraint $X \in \mathcal{D}$ by $X \in \mathcal{D}^*$. We remark that the PSD cone $\mathcal{S}_+$ is contained in $\mathcal{D}^*$, and that the latter is a polyhedral relaxation of the former. We also recall that $\mathcal{S}_+$ can be characterized by means of the uncountably infinite set of constraints $v^\top X v \geq 0$ for all $v \in \mathbb{R}^n$. Moreover, by [6], $\mathcal{D}$ is finitely generated by the set $\mathcal{V}$ of its extreme rays, which consists of the matrices $e_i e_i^\top$, $(e_i + e_j)(e_i + e_j)^\top$, $(e_i - e_j)(e_i - e_j)^\top$ for all $i < j \leq n$.

It is well known that if a cone is finitely generated, its dual cone is also finitely generated. By [23, Thm. 3], the polyhedral representation of $\mathcal{D}^*$ is

$$\forall v \in \mathcal{V} \quad \mathsf{trace}(v^\top X v) \geq 0, \tag{24}$$

which also shows that $\mathcal{S}_+ \subseteq \mathcal{D}^*$. Thus, an explicit formulation of $X \in \mathcal{D}^*$ is as follows:

$$\begin{array}{ll} \forall i \leq n & X_{ii} \geq 0 \\ \forall i < j \leq n & X_{ii} + X_{jj} - 2X_{ij} \geq 0 \\ \forall i < j \leq n & X_{ii} + X_{jj} + 2X_{ij} \geq 0. \end{array}$$

We can therefore derive a dual DDP formulation for the outer relaxation of Eq. (18):

$$\left.\begin{array}{rr} \max & \sum\limits_{\{i,j\}\in E} y_{ij} - \alpha \sum\limits_{\{i,j\}\in E} r_{ij} \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} \geq L_{ij}^2 y_{ij} \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} \leq U_{ij}^2 y_{ij} + r_{ij} \\ \forall i < j \leq n & X_{ij} = X_{ji} \\ \forall \{i,j\} \notin E & X_{ii} + X_{jj} - 2X_{ij} \geq 0 \\ \forall i < j \leq n & X_{ii} + X_{jj} + 2X_{ij} \geq 0 \\ \forall i \leq n & X_{ii} \geq 0 \\ & X \in \mathbb{R}^{n \times n} \\ & y \in \{0,1\}^m \\ & r \in \mathbb{R}^m_{\geq 0}. \end{array}\right\} \tag{25}$$

Since $\mathcal{S}_+ \subsetneq \mathcal{D}^*$, the optimal matrix solution X^* of Eq. (25) may (and in practice usually does) have negative eigenvalues, which makes it a poor candidate for the further rank reduction processing discussed below. On the other hand, it provides a guaranteed bound to the optimal objective function value of Eq. (18) in the optimization direction.

4.3 Rank reduction

If Eq. (23) is feasible, solving it yields a symmetric $n \times n$ PSD matrix X which has the property that $X_{ii} + X_{jj} - 2X_{ij}$ is the Euclidean distance between two points $\bar{x}_i, \bar{x}_j$, in some Euclidean space, such that $\bar{x}_i \cdot \bar{x}_i = X_{ii}$, $\bar{x}_j \cdot \bar{x}_j = X_{jj}$ and $\bar{x}_i \cdot \bar{x}_j = X_{ij}$. In other words, X is the Gram matrix of a realization $\bar{x}$ of the given graph G.

Since X is square symmetric, we can use spectral decomposition to write X as $X = P\Lambda P^\top$, where P is a matrix of eigenvectors, and Λ a diagonal matrix of eigenvalues $\lambda_1, \ldots, \lambda_n$ which we shall assume ordered largest to smallest. Since X is PSD, we have $\lambda_n \geq 0$ which implies that $\sqrt{\Lambda}$ is a real matrix. Hence we can decompose X as $(P\sqrt{\Lambda})(P\sqrt{\Lambda})^\top$, which means that we can take $\bar{x} = P\sqrt{\Lambda}$ as the realization of G.

We now recall that the DGP asks for a realization in $\mathbb{R}^K$ for a given integer K. The realization x is an $n \times n$ matrix, so it can be taken as a realization in $\mathbb{R}^n$. The intrinsic dimension of x is actually given by $\mathsf{rank}(x)$. In practice, $\mathsf{rank}(x)$ is usually n or very close to n, whereas K is usually much smaller than n. Thus, given $\bar{x} \in \mathbb{R}^{n \times n}$, we would like to find a reduced rank realization $x' \in \mathbb{R}^{n \times K}$.

We consider two rank reduction methods: the first is Principal Component Analysis (PCA) [18]. The second is Barvinok's naive algorithm [7], extended to consider arbitrary ranks [27].

4.3.1 Principal Component Analysis

With the notation of the previous section, we define

$$\Lambda^{(K)} = \mathsf{diag}(\lambda_1, \ldots, \lambda_K, 0, \ldots, 0),$$

and then we let $x' = P\sqrt{\Lambda^{(K)}}$. Although x' is still technically speaking an $n \times n$ matrix, all of the columns from the $K+1$-st to the n-th are zero vectors. This means that they can be ignored, and x' can be considered an $n \times K$ realization matrix, such that its i-th row x_i is the position of vertex i.

Since $\lambda_1 \geq \cdots \geq \lambda_K$ are the K largest eigenvalues of X, the approximate realization x' is the "closest" to $\bar{x}$ (with respect to the Schatten norm [10] considering a subset of K eigenvalues in Λ) which minimizes

$$\sum_{i \neq j} | \, \|\bar{x}_i - \bar{x}_j\|^2 - \|x'_i - x'_j\|^2 \, |,$$

for otherwise a contradiction would ensue with $\lambda_1, \ldots, \lambda_K$ being the K largest eigenvalues.

4.3.2 Barvinok's naive algorithm

The "naive algorithm" published by Barvinok in [7] is a lesser known, but effective, dimensionality reduction technique applicable to solutions of SDPs. Consider the quadratic feasibility problem of determining whether the set $\{x \in \mathbb{R}^n \mid \forall i \leq m\ x^\top Q^i x = a_i\}$ is non-empty. Let $\bar{X}$ be a solution of the corresponding SDP set $\{X \in \mathbb{R}^{n\times n} \mid \forall i \leq m\ \mathsf{trace}(Q^i X) = a_i\}$. Barvinok's naive algorithm performs the following steps:

1. let T be a factor of $\bar{X}$, so $\bar{X} = TT^\top$
2. sample each component of a vector $w \in \mathbb{R}^n$ from the normal distribution $\mathcal{N}(0,1)$
3. let $x' = Tw$.

A concentration of measure argument shows that

$$\mathsf{Prob}\left(\forall i \leq m\ \mathsf{dist}(x', \{x \mid x^\top Q^i x = a_i\}) \leq C\sqrt{\|\bar{X}\| \ln(n)}\right) \geq 0.9,$$

where C is a positive universal constant, and $\mathsf{dist}(p, S)$ is the Euclidean distance between a point p and a set S. In other words, Barvinok's naive algorithm ensures that x' is "not too far" from the feasible set of the quadratic equations.

An extension of Barvinok's naive algorithm to the iDGP was proposed in [27]. Starting from a solution $\bar{X}$ of the SDP relaxation of the iDGP, it is as follows:

1. let T be a factor of $\bar{X}$, so $\bar{X} = TT^\top$
2. sample each component of an $n \times K$ matrix w from the normal distribution $\mathcal{N}(0, 1/\sqrt{K})$
3. let $x' = Tw$.

Then

$$\mathsf{Prob}\left(\forall \{i,j\} \in E\ \mathsf{dist}(x', \{x \mid L_{ij} \leq \|x_i - x_j\| \leq U_{ij}\}) \leq C\sqrt{\|\bar{X}\| \ln(|E|)}\right) \geq 0.9.$$

Again, x' is close to being an iDGP realization of the graph G with high probability.

4.4 Refinement

Both of the rank reduction methods sketched above produce an approximate realization x' which is close to being feasible for the given iDGP instance. We therefore propose a "refinement step" where x' is given as a starting point for a local descent consisting in locally solving the following variant of Eq. (6):

$$\left.\begin{array}{rc} \min & \sum\limits_{\{i,j\}\in E} s_{ij} \\ \forall \{i,j\} \in E' & \|x_i - x_j\|^2 \geq L_{ij}^2 - s_{ij} \\ \forall \{i,j\} \in E' & \|x_i - x_j\|^2 \leq U_{ij}^2 \\ \forall \{i,j\} \in E' & s_{ij} \geq 0 \\ & x^L \leq x \leq x^U, \end{array}\right\} \tag{26}$$

where $E' = \{\{i,j\} \in E \mid \bar{y}_{ij} = 1\}$, with $\bar{y} \in \{0,1\}^m$ given by any of the mixed-integer formulations in this section.

We remark that the problematic reverse-convex constraint was relaxed in Eq. (26) by means of a slack variable to be minimized. Solving Eq. (26) yields a solution $x^* \in \mathbb{R}^{n\times K}$ improved w.r.t. x'.

5 Computational results

In this section we present some computational results concerning the formulations presented in the previous sections. More precisely, we test the following solution methods:

1. Algorithm sBB is a global spatial Branch-and-Bound (sBB) based solver deployed on the formulation in Eq. (16) for a given CPU time;
2. Algorithm DDP consists in: (i) solving an inner MIDDP restriction Eq. (23) (Sect. 4.2.1) to yield a solution $(\bar{X}, \bar{y})$, (ii) reduce the rank of $\bar{X}$ to K (Sect. 4.3) to yield a realization $\bar{x} \in \mathbb{R}^{n \times K}$, (iii) use $\bar{x}$ as a starting point for a local NLP solver deployed on Eq. (26), to obtain a realization $x^* \in \mathbb{R}^{n \times K}$.

The reason why we do not consider solving the exact MINLP formulation in Eq. (10) is that preliminary tests showed that sBB solvers cannot even find a locally optimal solution of our smallest instance. We do not consider the MISDP relaxation in Eq. (18) due to the fact that we could not find an off-the-shelf MISDP solver we could deploy on our computational platform.

5.1 Solution quality measures

Let x^* be a realization of G, and $y^* \in \{0,1\}^m$ describe the activation of the m constraints of the iDGP. We consider the following quality measures:

- the support cardinality $|\mathsf{supp}(y)| = \sum_{\{i,j\} \in E} y^*_{ij}$ of y, which is equal to the number of satisfied distance constraints, and evaluates the systematic errors;
- the mean and largest distance error (MDE, LDE) measures computed on the support (i.e., for $\{i,j\} \in E$ with $y_{ij} = 1$):

$$\mathsf{MDE}(x, y, G) = \frac{1}{m} \sum_{\{i,j\} \in E} \max(L_{ij} - \|x_i - x_j\|_2, \|x_i - x_j\|_2 - U_{ij}) y_{ij} \quad (27)$$

$$\mathsf{LDE}(x, y, G) = \max_{\{i,j\} \in E} \max(L_{ij} - \|x_i - x_j\|_2, \|x_i - x_j\|_2 - U_{ij}) y_{ij}, \quad (28)$$

  which evaluate the realization error w.r.t. the given (interval) distances;
- the CPU time taken to solve the instance.

We also employ a comparative measure between two realizations $x, y \in \mathbb{R}^{n \times K}$ called Root Mean Square Deviation (RMSD), defined as $\sqrt{\sum_{i \leq n} \|x_i - y_i\|^2 / n}$. According to [12], the RMSD is not overly meaningful on protein realizations unless one also normalizes w.r.t. partial reflections, which, however, appears hard. Along with RMSD scores between the realizations found by our method and reference realizations with zero systematic and experimental errors, we also show the plots with realization aligned according to Procrustes analysis [16] (but not w.r.t. partial reflections, see [12]).

5.2 Test set

We perform our tests on a set of small to medium-sized protein instances. We note that K is fixed to the constant 3. For a given protein `x` with known realization $\hat{x} \in \mathbb{R}^{n \times 3}$, the instance corresponding to `x` was generated as follows:

1. the $n \times n$ Euclidean distance matrix $D = (d_{ij})$ of $\hat{x}$ was computed;
2. all distances between atoms i and $j \in \{i+1, i+2\}$ on the backbone were kept as exact distances, namely $[L_{ij}, U_{ij}] = [d_{ij}, d_{ij}]$, for each $i \leq n-2$ (these distances are known as *discretization distances*);
3. all distances between atoms i and $i+3$ on the backbone were kept as interval distances, namely $[L_{ij}, U_{ij}] = [d_{ij} - \eta d_{ij}, d_{ij} + \eta d_{ij}]$, for each $i \leq n-3$ (these distances are also known as *discretization distances* [30]);
4. all other distances shorter than 5Å in D were kept as interval distances, namely $[L_{ij}, U_{ij}] = [d_{ij} - \eta d_{ij}, d_{ij} + \eta d_{ij}]$ (these distances are known as *pruning distances* [30]);
5. a given fraction σ of the pruning distances, chosen randomly, were reassigned randomly to a different pair of atoms;
6. any other distance was discarded from D.

In our experiments, we set $\eta = \sigma = 0.1$.

Most of the instances in Table 1 were extracted in the Protein Data Bank (PDB) [9]; a few are modifications of instances from the PDB.

Name	m	n
`tiny`	335	37
`1guu`	955	213
`1guu-1`	959	150
`1guu-4000`	968	150
`pept`	999	107
`res_2kxa`	2627	177
`2kxa`	2711	177
`C0030pkl`	3247	198
`100d`	5741	488
`helix_amber`	6265	392

Table 1 The protein instances in the test set, with the corresponding number of edges and vertices in their graphs.

5.3 Computational set-up

The sBB algorithm was implemented by the Baron solver [32,34]. As for the DDP algorithm, solutions of Eq. (23) were obtained using CPLEX 12.10 [19]. Solutions of the refinement step subproblems (Sect. 4.4) were obtained using IPOPT 3.11 [11].

In practice, the α coefficient, appearing in the scalarized formulation Eq. (16) and in the derived MIDDP formulation Eq. (23), depends on the instance. In general, we found that values of α over 0.2 often yielded trivial solutions where $y_{ij} = 0$ for every $\{i, j\} \in E$. After some preliminary experimentation, we set $\alpha = 0.15$ for both sBB and DDP.

Our implementation uses a mixture of Python3 and AMPL (using the AmplPy Python interface). The experiments were carried out on a server with four Intel Xeon E5-2620 v4 CPUs with eight cores per CPU at 2.1GHz with 64GB RAM running CentOS Linux.

5.4 Experiments

In this section we discuss the results obtained from the experiments.

5.4.1 The sBB algorithm

The sBB algorithm was able to solve none of the instances in Table 1 to guaranteed global optimality within one hour of CPU time. Moreover, it failed to find feasible solutions for any instance other than `tiny`.

On the `tiny` instance, sBB found, within one hour of CPU time, a realization x^* with $|\mathsf{supp}(y)| = 335$, $\mathsf{MDE} = 0.056$, $\mathsf{LDE} = 3.352$. Since $m = 335 = |\mathsf{supp}(y)|$, this solution neglects all systematic errors, considering them as experimental instead. This is noticeable in Fig. 1, which shows x^* optimally aligned to a reference realization without any error at all. The RMSD value for this pair is 0.084. Each

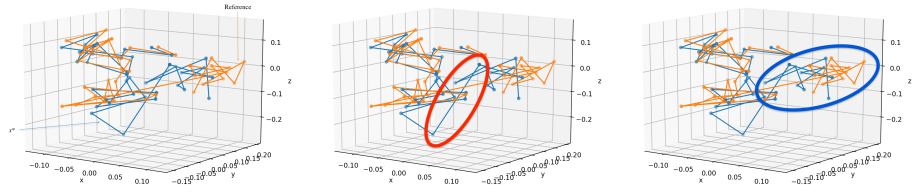

Fig. 1 Realization of `tiny` obtained using sBB compared with a correct reference realization (left). The systematic error (center). Error derived from a wrong partial reflection (right). The scaling on the axes is arbitrary.

picture in Fig. 1 shows two realizations of `tiny` aligned optimally using Procrustes analysis. One of them, labelled x^* in the left picture, was found by sBB; the other, labelled "Reference", is a realization of `tiny` without errors (experimental or systematic). We remark that the two clusters on the left appear to be well aligned. In the center picture we emphasize an edge $\{i, j\}$ in x^* which should clearly add to the systematic error rather than to the experimental one: it shows that y_{ij} should have been zero rather than one (we recall that, instead, sBB found a solution with $y_{ij} = 1$). On the right picture, we emphasize a flipped partial reflection, which contributes to the overall RMSD error, but is not an actual error, as explained in [12].

5.4.2 The DDP algorithm

The DDP algorithm was configured with a maximum CPU time of 3600s for the MILP solver deployed on the inner MIDDP formulation of Eq. (23), while the local NLP solver in the refinement phase was allowed to terminate naturally. The DDP algorithm was able to find reasonable realizations for all of the instances. The results are reported in Table 2. The columns report: the instance name, the rank reduction algorithm (PCA, denoted "pca", or Barvinok's naive algorithm, denoted "bvk"), the number m of edges in the instance, the number $|\mathsf{supp}(y)|$ of

Name	*RkRed*	m	$\lvert\mathsf{supp}(y)\rvert$	MDE	LDE	*RMSD*	*UB*	*NegEv*	*CPU*
`tiny`	pca	335	317	0.124	2.862	0.143	335	0.343	3619.34
`tiny`	bvk	335	317	0.109	3.034	0.136	335	0.343	3622.70
`1guu`	pca	955	949	0.002	0.224	0.047	955	0.391	3629.33
`1guu`	bvk	955	949	0.001	0.292	0.047	955	0.391	3683.10
`1guu-1`	pca	959	952	0.014	1.294	0.081	959	0.382	3618.80
`1guu-1`	bvk	959	952	0.022	0.854	0.078	959	0.382	3616.69
`1guu-4000`	pca	968	967	0.025	1.443	0.080	968	0.379	3623.35
`1guu-4000`	bvk	968	967	0.036	1.537	0.079	968	0.379	3636.38
`pept`	pca	999	999	0.149	2.578	0.092	999	0.387	3624.22
`pept`	bvk	999	999	0.170	3.794	0.094	999	0.387	3632.36
`res_2kxa`	pca	2627	2626	0.133	2.975	0.053	2627	0.399	3658.31
`res_2kxa`	bvk	2627	2626	0.106	3.818	0.054	2627	0.399	3898.28
`2kxa`	pca	2711	2708	0.139	3.607	0.052	2711	0.399	3684.55
`2kxa`	bvk	2711	2708	0.165	3.336	0.053	2711	0.399	3662.43
`C0030pkl`	pca	3247	3243	0.275	4.118	0.068	3247	0.405	3770.95
`C0030pkl`	bvk	3247	3243	0.249	5.330	0.070	3247	0.405	3705.82
`100d`	pca	5741	5741	0.202	3.686	0.045	5741	0.397	4488.31
`100d`	bvk	5741	5741	0.209	3.628	0.045	5741	0.397	4521.80
`helix_amber`	pca	6265	6265	0.273	3.704	0.050	6265	0.404	4229.80
`helix_amber`	bvk	6265	6263	0.264	4.176	0.050	6265	0.404	3963.57

Table 2 Results from the DDP algorithm.

edges with experimental error only, the MDE and LDE measures, the RMSD value between the realization found and a reference realization without any error, the upper bound *UB* found by the outer MIDDP relaxation in Eq. (25), the ratio *NegEv* of negative to total eigenvalue weight of the corresponding outer MIDDP solution, and the CPU time.

Table 2 allows us to make a few observations.

1. Overall, the methodology we propose is able to derive approximate solutions to small and medium-scaled instances of the $\text{MaxFS}_{\text{DGP}}$ problem in acceptable times.
2. The MDE quality measures are acceptable with respect to results obtained in the DGP literature on proteins with solution methods taken from MP (see e.g. [20, 13, 12, 27, 24]). The LDE measures, on the other hand, appear excessive, and may be a sign that the balance α between systematic and experimental error needs further tuning.
3. The time spent on the refinement phase can vary greatly. While this is not necessarily troublesome when finding the shape of proteins (which is rarely a real-time affair), it may help to set a time limit on the refinement phase too.
4. It is unclear whether PCA or Barvinok's naive algorithm is the best rank reduction method. They clearly yield different approximate realizations, which is important in view of the choice of starting point for the refinement phase. A possible idea would be to deploy them in parallel until termination of the fastest refinement phase.
5. The upper bound *UB* obtained by the outer MIDDP relaxation is always equal to the number of edges of the instance. Further analysis shows that every outer MIDDP was always solved by the presolver, which further strengthens the possibility that this upper bound may be always trivial. We do not know whether setting $y_{ij} = 1$ for each $\{i, j\} \in E$ always yields a feasible solution in

Eq. (25) with $\sum_{ij} r_{ij} = 0$; neither do we know whether such a property might be established for specific values of α.

A further methodological inquiry can be made by computing the RMSD between solutions of the MIDDP formulation Eq. (23) after projection (by PCA or by Barvinok's algorithm), and the solution after the refinement phase using the local NLP solver. This statistic gives us an idea of whether the refinement phase is effective. As can be seen from Table 3, this is indeed the case.

Name	pca	bvk
`tiny`	0.137	0.158
`1guu`	0.041	0.045
`1guu-1`	0.080	0.080
`1guu-4000`	0.080	0.081
`pept`	0.094	0.095
`res_2kxa`	0.048	0.053
`2kxa`	0.048	0.053
`C0030pkl`	0.068	0.071
`100`	0.045	0.045
`helix_amber`	0.050	0.050

Table 3 RMSD values between partial solutions obtained by MIDDP after dimensionality reduction, and final solutions after the refinement phase.

6 Conclusion

This paper is about the problem of finding a maximum feasible subsystem of distance geometry constraints, which models well the dual nature of the errors arising from finding protein conformations from NMR distance data. We discussed mathematical programming formulations of many different types for this problem: exact, approximate, relaxation, restrictions. We proposed a solution methodology based on a mixed integer diagonally dominant programming restriction of the original mixed-integer nonlinear programming problem. We tested our methodology on a set of protein instances of small and medium sizes obtaining reasonable solutions in acceptable CPU times.

Data availability statement

The datasets generated and analysed in this paper are available upon request from the corresponding author.

References

1. D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435(9):759–764, 2005.

2. A. Ahmadi and G. Hall. Sum of squares basis pursuit with linear and second order cone programming. In H. Harrington, M. Omar, and M. Wright, editors, *Algebraic and Geometric Methods in Discrete Mathematics*, volume 685 of *Contemporary Mathematics*, pages 27–54. AMS, Providence, RI, 2017.
3. A. Ahmadi and A. Majumdar. DSOS and SDSOS optimization: More tractable alternatives to sum of squares and semidefinite optimization. *SIAM Journal on Applied Algebra and Geometry*, 3(2):193–230, 2019.
4. E. Amaldi, M. Bruglieri, and G. Casale. A two-phase relaxation-based heuristic for the maximum feasible subsystem problem. *Computers and Operations Research*, 35:1465–1482, 2008.
5. E. Amaldi, M. Pfetsch, and L. Trotter. On the maximum feasible subsystem problem, iiss and iis-hypergraphs. *Mathematical Programming*, 95:533–554, 2003.
6. G. Barker and D. Carlson. Cones of diagonally dominant matrices. *Pacific Journal of Mathematics*, 57(1):15–32, 1975.
7. A. Barvinok. Measure concentration in optimization. *Mathematical Programming*, 79:33–53, 1997.
8. B. Berger, J. Kleinberg, and T. Leighton. Reconstructing a three-dimensional model with arbitrary errors. *Journal of the ACM*, 46(2):212–235, 1999.
9. H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I.N. Shindyalov, and P. Bourne. The protein data bank. *Nucleic Acid Research*, 28:235–242, 2000.
10. R. Bhatia. Matrix Analysis, New York, 1997.
11. COIN-OR. *Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT*, 2006.
12. C. D'Ambrosio, Ky Vu, C. Lavor, L. Liberti, and N. Maculan. New error measures and methods for realizing protein graphs from distance data. *Discrete and Computational Geometry*, 57(2):371–418, 2017.
13. G. Dias and L. Liberti. Diagonally dominant programming in distance geometry. In R. Cerulli, S. Fujishige, and R. Mahjoub, editors, *International Symposium in Combinatorial Optimization*, volume 9849 of *LNCS*, pages 225–236, New York, 2016. Springer.
14. S. Gerschgorin. Über die abgrenzung der eigenwerte einer matrix. *Izvestia Akademii Nauk USSR*, 6:749–754, 1931.
15. D. Gonçalves, A. Mucherino, C. Lavor, and L. Liberti. Recent advances on the interval distance geometry problem. *Journal of Global Optimization*, 69:525–545, 2017.
16. C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society B*, 53(2):285–339, 1991.
17. R. Greer. *Trees and hills: methodology for maximizing functions of systems of linear relations*, volume 22 of *Annals of Discrete Mathematics*. Elsevier, Amsterdam, 1984.
18. H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.
19. IBM. *ILOG CPLEX 12.9 User's Manual*. IBM, 2019.
20. C. Lavor, L. Liberti, and N. Maculan. Computational experience with the molecular distance geometry problem. In J. Pintér, editor, *Global Optimization: Scientific and Engineering Case Studies*, pages 213–225. Springer, Berlin, 2006.
21. C. Lavor, L. Liberti, and A. Mucherino. The *interval* Branch-and-Prune algorithm for the discretizable molecular distance geometry problem with inexact distances. *Journal of Global Optimization*, 56:855–871, 2013.
22. L. Liberti. Undecidability and hardness in mixed-integer nonlinear programming. *RAIRO-Operations Research*, 53:81–109, 2019.
23. L. Liberti. Distance geometry and data science. *TOP*, 28:271–339, 2020.
24. L. Liberti, G. Iommazzo, C. Lavor, and N. Maculan. A cycle-based formulation of the Distance Geometry Problem. In C. Gentile et al., editor, *Proceedings of 18th Cologne-Twente Workshop*, volume 4 of *AIRO*, New York, 2020. Springer.
25. L. Liberti, C. Lavor, N. Maculan, and A. Mucherino. Euclidean distance geometry and applications. *SIAM Review*, 56(1):3–69, 2014.
26. L. Liberti and F. Marinelli. Mathematical programming: Turing completeness and applications to software analysis. *Journal of Combinatorial Optimization*, 28(1):82–104, 2014.
27. L. Liberti and K. Vu. Barvinok's naive algorithm in distance geometry. *Operations Research Letters*, 46:476–481, 2018.
28. P. Luisi. Molecular conformational rigidity: An approach to quantification. *Naturwissenschaften*, 64:569–574, 1977.

29. T. Malliavin, A. Mucherino, C. Lavor, and L. Liberti. Systematic exploration of protein conformational space using a distance geometry approach. *Journal of Chemical Information and Modeling*, 59:4486–4503, 2019.
30. A. Mucherino, D.S. Gonçalves, L. Liberti, J-H. Lin, C. Lavor, N. Maculan. MD-JEEP: a New Release for Discretizable Distance Geometry Problems with Interval Data *Annals of Computer Science and Information Systems*, Sep 2020, Sofia, Bulgaria., 1–7, 2020.
31. M. Nilges, M. Macias, S. O'Donoghue, and 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*, 269:408–422, 1997.
32. N.V. Sahinidis and M. Tawarmalani. *BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2005.
33. J. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. *Proceedings of 17th Allerton Conference in Communications, Control and Computing*, pages 480–489, 1979.
34. M. Tawarmalani and N.V. Sahinidis. Global optimization of mixed integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563–591, 2004.