# A distance geometry procedure using the Levenberg-Marquardt algorithm and with applications in biology but not only

Douglas Gonçalves, Antonio Mucherino

# A distance geometry procedure using the Levenberg-Marquardt algorithm and with applications in biology but not only

D.S. Gonçalves[1] and A. Mucherino[2]

[1]Department of Mathematics, UFSC, Florianópolis, SC, Brazil.
douglas.goncalves@ufsc.br

[2]IRISA, University of Rennes 1, Rennes, France.
antonio.mucherino@irisa.fr

**Abstract.** We revisit a simple, yet capable to provide good solutions, procedure for solving the Distance Geometry Problem (DGP). This procedure combines two main components: the first one identifying an initial approximated solution via semidefinite programming, which is thereafter projected to the target dimension via PCA; and another component where this initial solution is refined by locally minimizing the Smooth STRESS function. In this work, we propose the use of the projected Levenberg-Marquart algorithm for this second step. In spite of the simplicity, as well as of its heuristic character, our experiments show that this procedure is able to exhibit good performances in terms of quality of the solutions for most of the instances we have selected for our experiments. Moreover, it seems to be promising not only for the DGP application arising in structural biology, which we considered in our computational experiments, but also in other ongoing studies related to the DGP and its applications: we finally provide a general discussion on how extending the presented ideas to other applications.

## 1 Introduction

Let $G = (V,E,d)$ be a simple weighted undirected graph, where the weight function $d$ maps every edge of the graph to a given distance value. We suppose that a unique numerical label $i \in \{1,2,\ldots,|V|\}$ is associated to every vertex in $V$, so that a vertex ordering is implicitly given. The focus of this article is a geometric problem having several real-life applications [12, 15]:

**Definition 1.1.** *Given the graph $G$ and a dimension $K > 0$, the Distance Geometry Problem (DGP) asks whether there exists any realization $x : V \longrightarrow \mathbb{R}^K$ such that the following distance constraints are satisfied:*

$$\forall \{i,j\} \in E, \quad ||x_i - x_j|| = d_{ij}, \tag{1}$$

*where $||\cdot||$ is the Euclidean norm and $x_i = x(i)$.*

Throughout this article, we will suppose that the considered problem instances admit at least one solution, which we will refer to as "valid realizations".

We remark that, in several applications, such as the one arising in the context of structural biology [5], sensor network localization [4], or even in computer graphics [11], the distance information cannot be provided with high precision. Most likely, instead of having one precise distance value $d_{ij}$, approximated lower and upper bounds are actually provided for most of the involved distances. Let us suppose therefore that our weight function $d$ in $G$ does not provide a single real number, but rather a pair of real numbers, $\underline{d}_{ij}$ and $\overline{d}_{ij}$ for every $\{i,j\} \in E$, such that $\underline{d}_{ij} < \overline{d}_{ij}$. In order to take these interval distances into consideration, we introduce new variables $y$ indexed on the edge set $E$, and modify the problem in eq. (1) as follows:

$$\forall \{i,j\} \in E, \quad \begin{cases} ||x_i - x_j||^2 - y_{ij} = 0, \\ \underline{d}_{ij}^2 \leq y_{ij} \leq \overline{d}_{ij}^2. \end{cases} \tag{2}$$

In spite of the current large efforts of the research community in finding new and efficient solution methods to the DGP, a general method has not been devised yet. In this work, we focus our attention on a rather simple procedure, which is basically composed by two main components: $(i)$ the generation of an *initial* realization that we can expect to be in a relatively small neighborhood of a DGP solution; $(ii)$ a refinement step: from the found initial realization, we locally minimize the sum of squared constraint violations, with the aim of identifying a better approximation of the DGP solution. In particular, to tackle part $(i)$, we solve a semidefinite programming relaxation [4] of the original DGP and project the obtained high-dimensional realization in $\mathbb{R}^K$; then, from the obtained initial realization, we run the projected Levenberg-Marquardt algorithm [8] to tackle the part $(ii)$ of our procedure.

We point out that the general structure of our procedure is not new. One example can be found in [1], where semidefinite programming also comes to play; another example can be found in [6]. To the best of our knowledge, however, the procedure used in our article is the first one that employs the Levenberg-Marquardt algorithm. The main motivation to use this algorithm is that it provides better convergence results when compared to other methods (such as gradient descent methods), as our computational experiments will show. As a result, despite the simplicity of our procedure, we can report successful computational experiments on relatively small-sized instances (but not really *tiny* instances, as in the experiments presented in other works). The use of our procedure appears therefore to be promising for future studies in the context of the DGP.

The rest of the paper is organized as follows. The DGP procedure main structure will be briefly introduced in Section 2. This short section will then contain two subparts, one (Section 2.1) focusing on a semidefinite programming relaxation of our target problem, and another (Section 2.2) describing the Projected Levenberg-Marquardt (PLM) algorithm. Computational experiments on a set of protein-like instances will be reported in Section 3. Finally, we will conclude the paper in Section 4 with an extensive discussion on the possibilities of use for the described procedure, as well as on the use of its components in other algorithmic frameworks. A particular emphasis will be given to the impact of the uncertainty on the distances on sets of DGP solutions.

## 2 Our DGP procedure

Given a pair $(G,K)$ representing a DGP instance, our procedure for its solution can be simply summarized in the following two steps:

**Step A.** Find a realization via a Semidefinite Programming (SDP) relaxation, following by a Principal Component Analysis (PCA) projection. The realization this way obtained is our "initial realization" (see Section 2.1);

**Step B.** Improve the quality of the initial realization found at the previous step by running the Projected Levenberg-Marquardt (PLM) algorithm (see Section 2.2).

Notice that, from now on, we will be using the acronyms SDP, PCA and PLM for referring to the methods mentioned above.

### 2.1 Semidefinite programming relaxation

Let $X \in \mathbb{R}^{K\times n}$ be a matrix with the vectors $x_i \in \mathbb{R}^K$ as its columns. We have:

$$\|x_i - x_j\|^2 = (e_i - e_j)^\top X^\top X (e_i - e_j) =: (e_i - e_j)^\top Y (e_i - e_j),$$

where $e_i$ stands for the $i^{th}$ canonical vector of $\mathbb{R}^n$. As a consequence, problem (2) is equivalent to find a positive semidefinite matrix $Y$ of rank $K$ such that

$$\underline{d}_{ij}^2 \le \langle E_{ij}, Y\rangle \le \overline{d}_{ij}^2, \quad \forall\{i,j\} \in E,$$

where $\langle A,B\rangle := \text{trace}(A^\top B)$ is the trace inner product and $E_{ij} := (e_i - e_j)(e_i - e_j)^\top$. This reformulation could be cast as a linear SDP except for the (nonconvex) rank constraint. Following [1,4], we suppress the rank constraint and consider the following SDP relaxation:

$$\begin{aligned} \min_{Y=Y^\top} \quad & -\gamma\langle I, Y\rangle \\ \text{s.t.} \quad & \underline{d}_{ij}^2 \le \langle E_{ij}, Y\rangle \le \overline{d}_{ij}^2, \quad \forall\{i,j\} \in E \\ & Y\mathbf{1} = 0, \quad Y \succeq 0, \end{aligned} \tag{3}$$

where $\mathbf{1} \in \mathbb{R}^n$ is a vector of ones, $\gamma > 0$ is a regularization parameter and $Y \succeq 0$ means that $Y$ must be positive semidefinite. The term $-\langle I, Y\rangle$ in the objective function corresponds to a rank reduction heuristic [4]. The reasoning behind it is that, under $Y\mathbf{1} = 0$, we have that

$$\langle I, Y\rangle = \text{trace}(Y) = \text{trace}(Y) - \frac{1}{n}\mathbf{1}^\top Y\mathbf{1} = \frac{1}{2n}\sum_i\sum_j \|x_i - x_j\|^2,$$

and by maximizing this quantity we force the corresponding realization to be "more flat" and hence belonging (hopefully) to a lower dimensional space.

Let $Y$ be a solution to problem (3). Since $Y \succeq 0$, we have that $Y = X^\top X$, where $X \in \mathbb{R}^{r\times n}$, with $r = \text{rank}(Y)$. Although $X$ satisfies all distance constraints, it may happen

that the rank $r$ is strictly larger than the desired dimension $K$. This is the reason why it is necessary to project $X$ onto $\mathbb{R}^K$: we perform this projection by PCA. Let $Y = Q\Lambda Q^\top$ be the eigendecomposition of $Y$ and assume the eigenvalues are ordered in non-increasing order $\lambda_1 \geq \ldots \lambda_n \geq 0$. If $\Lambda_K$ denotes the principal submatrix of $\Lambda$ containing the $K$ largest eigenvalues of $Y$ and $Q_K$ contains the $K$ corresponding eigenvectors in its columns, then

$$X_0 = \sqrt{\Lambda_K} Q_K^\top$$

gives us the sought "projection", which is an approximate realization in $\mathbb{R}^K$. We say *approximate* because after the projection, $X_0$ may no longer satisfy some distance constraints.

In order to recover the feasibility of the violated constraints, we consider a refinement step which consists in an iterative method for solving problem (2) using the columns of $X_0$ as starting point for the vectors $x_i$ (the additional variables $y_{ij}$ are initialized to the values $(\underline{d}_{ij}^2 + \overline{d}_{ij}^2)/2$). For this refinement step, we consider the PLM algorithm [8], which is briefly reviewed in the next subsection.

### 2.2 Projected Levenberg-Marquardt algorithm

Consider the following constrained system of nonlinear equations:

$$\begin{cases} F(z) = 0, \\ z \in C, \end{cases} \tag{4}$$

where $F : \mathbb{R}^n \to \mathbb{R}^m$ is continuously differentiable and $C$ is a convex compact set. Let $J(z)$ denote the Jacobian of $F$ at $z$ and $P_C(u)$ the Euclidean projection of $u$ onto $C$. Moreover, let us define the following function:

$$f(z) = \frac{1}{2}\|F(z)\|^2.$$

Following [8], we consider the PLM algorithm for solving eq. (4), summarized below.

**The Projected Levenberg-Marquardt** (PLM)

Given $z_0 \in C$, $\sigma, \eta_1 \in (0,1)$, $M \in \mathbb{Z}_{++}$, $\eta_2 > 0$, set $k = 0$.

**Step 1.** Set $\mu_k = \|F(z_k)\|^2$ and solve $(J(z_k)^\top J(z_k) + \mu_k I) d_k^U = -J(z_k)^\top F(z_k)$

**Step 2.** Set $d_k^C = P_C(z_k + d_k^U) - z_k$. If $d_k^C$ satisfies

$$F(z_k)^\top J(z_k) d_k^C \leq -\eta_1 \|d_k^C\|^2 \tag{5}$$

$$\|d_k^C\| \leq \eta_2 \|J(z_k)^\top F(z_k)\| \tag{6}$$

then $d_k = d_k^C$, else $d_k = P_C(z_k - J(z_k)^\top F(z_k)) - z_k$.

**Step 3.** Set $\alpha = 1$. While $f(z_k + \alpha d_k) > \max_{0 \le j \le \min\{M,k\}} f(z_{k-j}) + \sigma\alpha d_k^\top J(z_k)^\top F(z_k)$, update $\alpha = \alpha/2$.

**Step 4.** Set $\alpha_k = \alpha$ and update $z_{k+1} = z_k + \alpha_k d_k$. Go to Step 1.

Given $z_k \in C$, the unconstrained LM direction $d_k^U$ is computed at Step 1. In Step 2, the feasible direction $d_k^C$, based on the projection of $z_k + d_k^U$ onto $C$, is computed and it is chosen as search direction if it satisfies the descent conditions in eq. (5) and (6). Otherwise, the projected gradient direction is taken. A step-size $\alpha > 0$ verifying a non-monotone Armijo-like condition [9] is determined in Step 3 by a backtracking process.

For more details about this algorithm, the reader is referred to [8], where a detailed convergence analysis of the algorithm can also be found. It was proved, in fact, that every limit point of the sequence $\{z_k\}$ is stationary for the problem of minimizing $f(z)$ subject to $z \in C$. Furthermore, under an error bound condition, a local superlinear convergence was established.

We point out that problem (2) corresponds to problem (4) with $z = (X, y) \in \mathbb{R}^{K\times n} \times \mathbb{R}^{|E|}$, $F : \mathbb{R}^{K\times n} \times \mathbb{R}^{|E|} \to \mathbb{R}^{|E|}$ with

$$[F(X,y)]_{ij} = \|x_i - x_j\|^2 - y_{ij}, \quad \forall \{i,j\} \in E,$$

and

$$C = \{(X,y) \in \mathbb{R}^{K\times n} \times \mathbb{R}^{|E|} \mid \underline{d}_{ij}^2 \le y_{ij} \le \overline{d}_{ij}^2,\ \forall \{i,j\} \in E\}.$$

In this case, the least-squares function $f$ takes the form

$$f(X,y) = \frac{1}{2}\|F(X,y)\|^2 = \sum_{\{i,j\}\in E} (\|x_i - x_j\|^2 - y_{ij})^2, \tag{7}$$

which corresponds to the Smooth STRESS function [18], with $\underline{d}_{ij}^2 \le y_{ij} \le \overline{d}_{ij}^2$.

## 3 Computational experiments

We propose in this section some initial computational experiments with the DGP procedure sketched in Section 2. All experiments were carried out on Matlab R2018b running MacOS X 10.13.6 (personal laptop).

We consider two sets of instances, both related to protein conformations. However, in the first set that we consider, the instances will only *resemble* to typical protein instances, because we will not include any additional distance information that is likely to help DGP solvers to find solutions. This "extra" and helpful distance information would include, for example, the length of chemical bonds, as well as the angles formed by triplets of consecutively bonded atoms. Thus, we decide to take, in our instance generation procedure, no advantage from the typical chemical structure of protein conformations. We make this choice for the generation of our first set of instances with the aim of testing the effectiveness of the procedure for larger classes of DGP instances, which may be related to different applications.

| | | | | | PLM | | | SPG | | |
|---|---|---|---|---|---|---|---|---|---|---|
| pdb-id | $\|V\|$ | $\|E\|$ | $\gamma$ | $f_0$ | $k$ | $f$ | RMSD | $k$ | $f$ | RMSD |
| 2JMY | 45 | 432 | 1 | 9.16E+01 | 93 | 4.26E-09 | 0.08 | 322 | 4.53E-02 | 0.08 |
| 2LR9 | 57 | 505 | 1 | 3.72E+03 | 555 | 4.90E-09 | 1.16 | 801 | 1.73E-01 | 1.36 |
| 1HJ0 | 123 | 1210 | 1 | 8.40E+03 | 73 | 4.20E+01 | 3.86 | 1602 | 4.22E+01 | 3.83 |
| 1HJ0 | 123 | 1210 | 10 | 8.37E+03 | 778 | 4.94E-09 | 2.13 | 1498 | 5.23E-01 | 2.57 |
| 2KSL | 153 | 1398 | 1 | 4.56E+04 | 250 | 2.63E+02 | 7.59 | 1010 | 6.58E+02 | 10.23 |
| 2KXA [7] | 177 | 973 | 1 | 9.43E+02 | 172 | 4.68E-09 | 0.45 | 686 | 2.79E-02 | 0.62 |
| 1DSK [7] | 222 | 1210 | 10 | 6.29E+03 | 268 | 4.99E-09 | 2.51 | 578 | 1.79E-02 | 2.44 |
| 2ERL [7] | 323 | 1789 | 1 | 1.99E+03 | 245 | 4.80E-09 | 0.41 | 704 | 2.66E-02 | 0.41 |
| 2JWU [7] | 447 | 2413 | 1 | 6.02E+03 | 219 | 4.69E-09 | 1.03 | 824 | 4.26E-02 | 0.99 |

**Table 1.** Some experiments showing the effectiveness of our DGP procedure on the two sets of instances. In the upper row block, we consider the instances generated in this work where no extra information about the nature of the distances is exploited; in the lower row block, we present the experiments on the protein instances previously used in [7].

Instead, the second set of instances that we consider in our experiments will include this additional information. For lack of space, we focus our attention on the main steps for generating our new instances of the first set, while the reader is referred to [7] for details on how the 4 instances of the second set were generated.

In order to generate the first set of instances, we consider models of protein conformations obtained from the Protein Data Bank (PDB) [2]. From one selected PDB model (when more than one model is available in the same PDB file, we simply pick the first one), we extract the backbone atoms N, $C_\alpha$ and C, and we generate the corresponding instance by measuring all possible distances between pairs of such backbone atoms, and by keeping only the distances shorter than 6Å. Noise is thereafter added to the distances by creating an interval $[\underline{d},\overline{d}]$ of range 0.2Å, where the computed distance is randomly placed. The procedure outputs a simple weighed graph $G$ that represents an instance of the DGP. We point out that our procedure introduces the same level of noise in all the distances, without distinguishing between distances between bonded atoms or other.

To assess the performance of PLM as a refinement tool, we compare it with the Spectral Projected Gradient (SPG) algorithm [3] for minimizing the function (7) over $C$. Notice that SPG was already successfully used in previous works as a local solver for DGP [17].

In all experiments, the SDP relaxation in eq. (3) was solved using SDPT3 solver [19] with standard parameters and tolerances. In PLM, we consider the parameters $\eta_1 = 10^{-4}$, $\eta_2 = 10^4$, $\sigma = 10^{-3}$, $M = 10$ and stop the iterations when either $\|F(X_k, y_k)\| \leq \varepsilon = 10^{-4}$ or $\|z_k - P_C(z_k - \nabla f(z_k))\| \leq \varepsilon$. For SPG, we used the same parameters as in [17], and stopped the iterations when $\|d_k\| \leq \varepsilon$. The maximum number of iterations was set to 2,000 for both SPG and PLM.

Table 1 summarizes the performed computational experiments. For every instance, we report the original PDB identifier of the protein in the PDB, together with the total number of vertices and the total number of edges in the generated graph $G$. The parameter $\gamma$ is the one involved in SDP, while $f_0$ is set to $f(X_0, y_0)$, which corresponds to the value of eq. (7) evaluated at the solution $X_0$ in **Step A**, where $y_0 = (\underline{d}^2 + \overline{d}^2)/2$. For both

PLM and SPG, we report the number of iterations $k$, the final objective function value for $f(X,y)$ (denoted $f$ in the table), and the RMSD with respect to the first model of the PDB file. Notice that only the $C_\alpha$ atoms in the solution found for our second set of instances were taken into consideration when computing the RMSD (for example, for the 2ERL instance, only 40 atoms out of 323 were selected).

The experiments show that, although after the execution of our **Step A** the value of eq. (7) for our initial realization is relatively large, such a starting point is nevertheless close enough to one of the instance solutions. In fact, the **Step B** in our procedure, when performed by running the PLM algorithm, is able to decrease the value of the Smooth STRESS function to a magnitude of $10^{-9}$ for the instances 2JMY, 2LR9, 2KXA and 2ERL belonging to our first instance set. This indicates that all involved distances are satisfied (i.e, $\|x_i - x_j\| \in [\underline{d}_{ij}, \overline{d}_{ij}]$), or close to be satisfied (i.e, $\|x_i - x_j\| \notin [\underline{d}_{ij}, \overline{d}_{ij}]$, but $\|x_i - x_j\|$ is close to one of the two bounds). Notice that we can state a similar remark for SPG, but with a final value for the STRESS function that is about six orders of magnitude larger. Even if the corresponding RMSD values are similar to those obtained by PLM, we can remark therefore that the PLM provides solutions in general capable to better satisfy the available distances.

Concerning the number of iterations of PLM and SPG, we can observe that, although the former requires fewer iterations, it is important to mention that its iterations are more expensive because requiring the solution of a positive definite linear system.

Two times the instance 1HJ0 is reported in the upper row block of Table 1. In fact, the former of the two experiments shows that the initial realization from **Step A** was not close enough to one of the instance solutions: both PLM and SPG have most certainly converged towards a local minimizer or a stationary point of (7). In the latter experiment concerning 1HJ0, however, where the value of the $\gamma$ parameter was changed from 1 to 10, we can observe a performance for our procedure which is close to the other experiments, where the initial realization is actually a good starting point for the refinement step (with PLM still beating SPG on the STRESS function value). This example is particularly interesting because, even though the use of a different value for $\gamma$ leads to a STRESS function value approaching zero, the final RMSD value in the found solutions does not change much. This indicates that the instance we have generated by using the first model in the PDB file does not have only that model in its solution set. The left side of Fig. 1 shows the model we have obtained with $\gamma = 10$.

The last line of the upper row block in Table 1 shows that, for one of the instances of our first set, we could not find any satisfactory solutions. Trying to use alternative values for the $\gamma$ parameter did improve the results in this case; the use of small values (e.g. 0.01, 0.1) or even larger than 10, did not allow us to get close enough to one of the solutions for having either SPG or PLM converge to a global minimizer. This is certainly not the only case where our procedure can fail, because of its simplicity.

Finally, the lower row block of experiments in Table 1 shows the performances of our procedure on 4 of the instances already used in [7]. As remarked above, these instances exploit some additional distance information that can help the solvers identifying the solutions, for example by fixing some of the distances to some given precise values. Our procedure seems to provide similar performances on this second set of instances, and the comparison between PLM and SPG remains the same as well. The

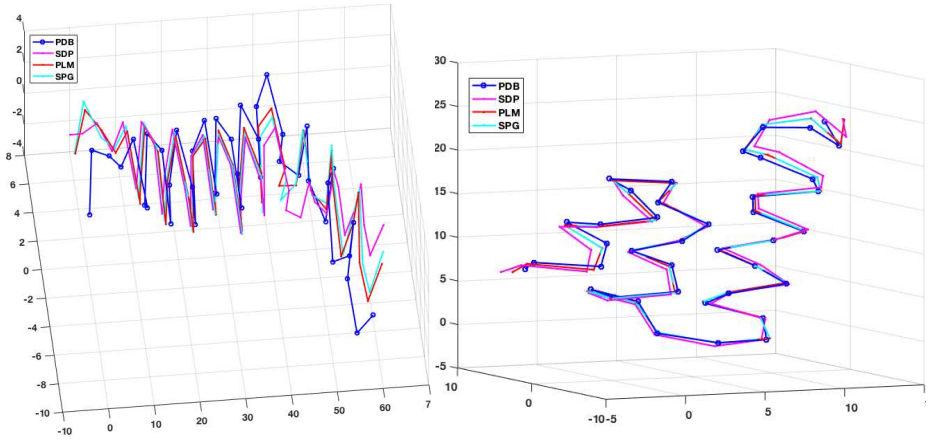


**Fig. 1.** A comparison among some obtained solutions and the original PDB model used to generate the instances: 1HJ0 ($\gamma = 10$ version, on the left-hand side), and 2ERL (only $C_\alpha$ atoms, on the right-hand side). Axis units in Angstroms.

right side of Fig. 1 shows the solution found for the instance `2ERL`: since these instances contain more atoms from the proteins (not only its backbone atoms), for clarity we only consider in the figure its $C_\alpha$ trace, which is the same considered in the computation of the RMSD.

## 4 Discussion and conclusions

We have presented and tested a simple procedure for the solution of DGPs where the value of the distances is uncertain and generally represented by a real-valued interval. As pointed out in the Introduction, the general structure of our simple procedure is not new, as it was already used in previous works, but, to the best of our knowledge, this is the first time that the projected Levenberg-Marquardt is employed in this context.

Even if it cannot be considered as a general solver for the DGP, our computational experiments have shown the effectiveness of our procedure on a set of artificially generated instances related to a typical biological application. The experiments show in fact that, when the first step of the procedure is able to identify an initial realization that is close enough to a valid realization (a solution for problem (2)), then its second step is able to localize that solution in the search domain.

These results open the doors for other possible uses of this procedure (or of one of its components) in more general solution methods for the DGP. For example, MDJEEP[1] is a solver for DGPs for which the discretization of the search space can be performed, by transforming the problem in a combinatorial problem [16]. When the value on the distances is uncertain, however, some nodes of the search domain cannot be associated to singletons, but rather to relatively small portions of the original continuous search

[1] `https://github.com/mucherino/mdjeep`

domain. This is the reason why, in MDJEEP, the combinatorics is coupled with a refinement step consisting in locally exploring all those small domain portions in the attempt to improve the overall solution quality [14]. The impact of the current work on the future developments of MDJEEP can be two-fold. Firstly, the first step of our procedure may be used to guess the most promising parts of the discretized search domain to enhance its performance in terms of time. The idea is only to give higher priority to the identified parts of the search domain, and to remove, a priori, none of them, so that the entire search domain may, potentially, still be explored. Secondly, since the current version of MDJEEP, the version `0.3.2` at the moment we are writing this article, uses SPG for performing its refinement step, another possible improvement may be to replace SPG with PLM.

Another interesting application falls in the context of motion adaptation [11]. Here, a skeletal structure (representing for example a human character) performs a given motion over time, and the problem of embedding the same motion (or a motion as close as possible to the original one) on another skeletal structure is considered. One of the main difficulties in solving such a problem is related to the fact that distorted self-contacts, which may be either artificially created in the new motion, or rather omitted from the original one, are likely to make the viewer perceive the motion as different when compared to the original. Self-contact is synonym of high proximity, and hence of near distances. The *dynamical* DGP (dynDGP) was introduced in [13] to tackle this class of problems, and more recently it was applied to motion adaptation in [10]. In this application, every frame of the motion can be considered as a separated (and static) DGP, where every new frame belongs to a small neighborhood, in its search domain, of the previous frame. SPG was exploited in previous works on distance-based motion adaptation: the animations created in [10] were generated by SPG for example. Again, we propose the use of PLM as a replacement for SPG, which is likely to provide interesting results in this application context as well.

We remark that, in the two cases where we propose to use PLM as a tool for refinement, the DGP instances to be solved (in both cases, these are actually sub-instances of the original problems) are rather simple if compared to the ones we used in the computational experiments in this work. When the refinement step is performed in MDJEEP, in fact, only a subset of vertices of the original instance is considered, and the local solver can benefit of a starting point where (in most of the cases) only a few distances are not satisfied (the best-case scenario being the one where only one distance is violated). In the case of motion adaptation, since the starting point is always the solution obtained at the previous frame of the motion, it is expected the new local solution to be very near the available starting point. Future works will be devoted to these very promising research directions.

## Acknowledgments

The authors are very grateful to CAPES/Brazil for the CAPES-PRINT project, process number 88887.578009/2020-00, allowing AM to visit DG at UFSC, Florianópolis (SC, Brazil) for a 2-week time in December 2021. Most of the presented work was performed during such a visit. AM is also thankful to the ANR for the support on the international France-Taiwan project MULTIBIOSTRUCT (ANR-19-CE45-0019). DG thanks

the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Grant 305213/2021-0.

## References


1. B. Alipanahi, N. Krislock, A. Ghodsi, H. Wolkowicz, L. Donaldson, M. Li, *Determining Protein Structures from NOESY Distance Constraints by Semidefinite Programming*, Journal of Computational Biology **20**(4), 296–310, 2013.
2. H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, P. Bourne, *The Protein Data Bank*, Nucleic Acids Research **28**, 235–242, 2000.
3. E.G. Birgin, J.M. Martínez, M. Raydan, *Spectral Projected Gradient Methods: Review and Perspectives*, Journal of Statistical Software **60**(i03), 21 pages, 2014.
4. P. Biswas, T. Liang, T. Wang, Y. Ye, *Semidefinite Programming Based Algorithms for Sensor Network Localization*, ACM Transactions on Sensor Networks **2**(2), 188–220, 2006.
5. G.M. Crippen, T.F. Havel, *Distance Geometry and Molecular Conformation*, John Wiley & Sons, 1988.
6. C. D'Ambrosio, V. Ky, C. Lavor, L. Liberti, N. Maculan, *New Error Measures and Methods for Realizing Protein Graphs from Distance Data*, Discrete & Computational Geometry **57**, 371–418, 2017.
7. D.S. Gonçalves, A. Mucherino, C. Lavor, L. Liberti, *Recent Advances on the Interval Distance Geometry Problem*, Journal of Global Optimization **69**(3), 525–545, 2017.
8. D.S. Gonalves, M.L.N. Gonçalves, F.R. Oliveira, *An Inexact Projected LM Type Algorithm for Solving Convex Constrained Nonlinear Equations*, Journal of Computational and Applied Mathematics **391**(1), 113421, 2021.
9. L. Grippo, F. Lampariello, S. Lucidi, *A Truncated Newton Method with Nonmonotone Line Search for Uncontrained Optimization*, Journal of Optimization Theory and Applications **60**, 401–419, 1989.
10. S.B. Hengeveld, A. Mucherino, *On the Representation of Human Motions and Distance-based Retargeting*, IEEE Conference Proceedings, Federated Conference on Computer Science and Information Systems (FedCSIS21), Workshop on Computational Optimization (WCO21), Sofia, Bulgaria, 181–189, 2021.
11. E.S.L Ho, T. Komura, C-L. Tai, *Spatial Relationship Preserving Character Motion Adaptation*, Proceedings of the $37^{th}$ International Conference and Exhibition on Computer Graphics and Interactive Techniques, ACM Transactions on Graphics **29**(4), 8 pages, 2010.
12. L. Liberti, C. Lavor, N. Maculan, A. Mucherino, *Euclidean Distance Geometry and Applications*, SIAM Review **56**(1), 3–69, 2014.
13. A. Mucherino, D.S. Gonçalves, *An Approach to Dynamical Distance Geometry*, Lecture Notes in Computer Science **10589**, F. Nielsen, F. Barbaresco (Eds.), Proceedings of Geometric Science of Information (GSI17), Paris, France, 821–829, 2017.
14. 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*, IEEE Conference Proceedings, Federated Conference on Computer Science and Information Systems (FedCSIS20), Workshop on Computational Optimization (WCO20), Sofia, Bulgaria, 289–294, 2020.
15. A. Mucherino, C. Lavor, L. Liberti, N. Maculan (Eds.), *Distance Geometry: Theory, Methods and Applications*, 410 pages, Springer, 2013.
16. A. Mucherino, L. Liberti, C. Lavor, *MD-jeep: an Implementation of a Branch & Prune Algorithm for Distance Geometry Problems*, Lectures Notes in Computer Science **6327**, K. Fukuda et al. (Eds.), Proceedings of the $3^{rd}$ International Congress on Mathematical Software (ICMS10), Kobe, Japan, 186–197, 2010.

17. A. Mucherino, J-H. Lin, *An Efficient Exhaustive Search for the Discretizable Distance Geometry Problem with Interval Data*, IEEE Conference Proceedings, Federated Conference on Computer Science and Information Systems (FedCSIS19), Workshop on Computational Optimization (WCO19), Leipzig, Germany, 135–141, 2019.
18. Y. Takane, F.W. Young, J. de Leeuw, *Nonmetric Individual Multidimensional Scaling: an Alternating Least Squares Method with Optimal Scaling Features*, Psychometrika **42**(1), 7–67, 1977.
19. K.C. Toh, M.J. Todd, R.H. Tütüncü, *SDPT3 – a Matlab Software Package for Semidefinite Programming*, Optimization Methods and Software **11**, 545–581, 1999.