IMA Journal of Numerical Analysis (2008) **28**, 690−710
doi:10.1093/imanum/drn012
Advance Access publication on May 2, 2008

Trust-region superposition methods for protein alignment

R. ANDREANI† AND J. M. MARTÍNEZ‡
Department of Applied Mathematics, IMECC-UNICAMP, State University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil

AND

L. MARTÍNEZ§
Institute of Chemistry, State University of Campinas, SP, Brazil and Institut Pasteur, Paris, France

[Received on 31 January 2007; revised on 1 February 2008]

This work is dedicated to the 70th birthday of Prof. M. J. D. Powell in September 2006.

Protein alignment is a challenging applied optimization problem. Superposition methods are based on the maximization of a score function with respect to rigid-body modifications of relative positions. The problem of score maximization can be modelled as a continuous nonsmooth optimization problem (low order-value optimization (LOVO)). This allows one to define practical and convergent methods that produce monotone increases of the score. In this paper, trust-region methods are introduced for solving the problem. Numerical results are presented. Computer software related to the LOVO approach for protein alignment is available at www.ime.unicamp.br/∼martinez/lovoalign.

Keywords: protein alignment; trust-region methods; low order-value optimization.

1. Introduction

Proteins are large organic compounds formed by chains of α-amino acids bound by peptide bonds. They are essential parts of all living organisms and participate in most cellular processes. Hormone recognition and transport, catalysis, transcription regulation, photosynthesis, cellular respiration and many other fundamental mechanisms of life are protein mediated. Proteins can work together to achieve a particular function and can bind to different chemical structures to be functional (Voet & Voet, 2004).

The sequence of amino acids in a protein is defined by a gene. This sequence is known as the 'primary structure' of a protein. Each amino acid has particular chemical characteristics, but contributes to the main chain of the protein with identical substructures formed by one nitrogen and two carbon atoms. One of these carbon atoms is known as the Cα atom. Roughly speaking, the 3D coordinates of the Cα atoms are known as the 'tertiary structure' of a protein. Protein structures can be determined by experimental methods, such as X-ray crystallography or nuclear magnetic resonance. A large collection containing atom coordinates for most of the known proteins is the Protein Data Bank (PDB) (Berman, 2000), which contains around 35000 structures. This number increases every year.

†Email: andreani@ime.unicamp.br
‡Corresponding author. Email: martinez@ime.unicamp.br
§Email: lmartinez@iqm.unicamp.br.

During evolution, mutations promote changes in the primary structure of a protein by introducing modifications in the genetic code. These mutations may persist in a population if they do not result in impaired protein function. The functions of different proteins may be the same in spite of different sequences of amino acids when they share the same overall 3D structure. Therefore, the 'classification' of 3D structures is useful to determine the function of the proteins and to provide hints on evolutionary mechanisms.

The main ingredient of the classification procedure is a comparison (alignment) between two structures. When a new protein structure is obtained or when a protein structure is conjectured, its comparison with the whole data bank and consequent classification is often used for functional classifications (Holm & Sander, 1993).

The degree of similarity between two proteins is usually given by a 'score'. From this score, a distance-like function is usually derived and the set of distances is frequently used to produce 'structure maps'. A structure map is a 2D or 3D representation of the whole space of proteins (Holm & Sander, 1996). In a structure map, each protein is a point and the distance between two of these points reflects the similarity given by the score. Multidimensional scaling (Hou *et al.*, 2005, 2003; Vendruscolo & Dobson, 2005) and Kernel methods (Lu *et al.*, 2005) are useful tools for building the 3D representation that comes from the scores. The structure space map developed in Hou *et al.* (2005, 2003) provides good predictions of function similarities in many cases.

The primary sequence of amino acids determines the structure of a protein. Protein folding is the molecular mechanism by which a protein achieves its tertiary structure from an unfolded sequence. Some general aspects of protein-folding mechanisms are now being elucidated (Onuchic & Wolynes, 2004), but the prediction of structure from sequence remains one of the greatest challenges of contemporary biochemistry. Methods for structural modelling based on the sequence of amino acids exist and are frequently based on sequence similarities to proteins with known structure. An evaluation of the quality of the models requires a measure of their potential energy and of their similarity to the structural references used (Sali & Blundell, 1993). Therefore, a score must be a reliable measure of similarity, not only between known structures but also between potential ones.

We will see that the score that measures the similarity between two proteins may be seen as the maximum of a (continuous nonsmooth) function in the space of relative positions (displacements). The reliability of the score depends on the accuracy with which we are able to obtain this maximum; therefore, robust and fast algorithms are necessary. Algorithms for obtaining the *global* maximum may be defined but are not affordable using current computer capabilities, (Kolodny & Linial, 2004). In this paper, we rely on the mathematical characterization of the protein alignment problem given in Martinez *et al.* (2007) (see also Andreani *et al.*, 2008a). Line-search algorithms that converge to first-order stationary points were defined in Martinez *et al.* (2007) and Andreani *et al.* (2008a). Here, we introduce a trust-region approach (Conn *et al.*, 2000; Moré, 1983; Powell, 1970) to define second-order convergent algorithms.

This paper is organized as follows. In Section 2, the protein alignment problem is formulated as a low order-value optimization (LOVO) problem. In Section 3, we define a trust-region method for solving LOVO problems and we prove convergence. In Section 4, we present numerical results. Conclusions are given in Section 5.

Notation

The symbol $\|\cdot\|$ will denote the Euclidean norm.

If the symmetric matrix A is positive semidefinite, we denote $A \succcurlyeq 0$. If A is positive definite, we denote $A \succ 0$.

We denote $\mathbb{N} = \{0, 1, 2, \ldots\}$.

The Euclidean ball with centre x and radius ε is denoted by $\mathcal{B}(x, \varepsilon)$.

2. Formulation

Let $\mathcal{Q} = \{Q_1, \ldots, Q_N\} \subset \mathbb{R}^{n_q}$ and $\mathcal{P} = \{P_1, \ldots, P_M\} \subset \mathbb{R}^{n_p}$. The goal is to find a transformation D: $\mathbb{R}^{n_q} \to \mathbb{R}^{n_p}$ such that some subset of $\{D(Q_1), \ldots, D(Q_N)\}$ fits some subset of $\mathcal{P}$. In protein alignment, D generally represents rigid-body displacements, but more general transformations can be considered. For example, assume that $n_q = 3, n_p = 2$ and $\mathcal{P}$ is the set of possible 'shadows' of the points in $\mathcal{Q}$. In that case, one could wish to find the rigid-body displacement of $\mathcal{Q}$ such that a subset of the 2D points represented by the (x, y) coordinates of the displaced $\mathcal{Q}$ fits a subset of $\mathcal{P}$ in the best possible way. In that case, D would be the composition of a rigid-body movement with a projection. A lot of examples of this general problem can be given, from tissue recognition to security systems (Andreani *et al.*, 2008b). We will denote by $\mathcal{D}$ the set of 'admissible transformations'.

Let $\mathcal{C}$ be the set of 'admissible correspondences' between nonempty subsets of $\{1, \ldots, N\}$ and $\{1, \ldots, M\}$. (Sometimes admissible correspondences must be bijective, and sometimes monotonicity will be required.)

Each element $\Phi \in \mathcal{C}$ is a function

$$\Phi\colon A(\Phi) \to B(\Phi),$$

where $A(\Phi) \subset \{1, \ldots, N\}$ and $B(\Phi) \subset \{1, \ldots, M\}$. Given $\Phi \in \mathcal{C}$ and a transformation D, an associated score $S(D, \Phi) \geqslant 0$ is assumed to be defined. This score should reflect the degree of spatial similarity between the sets $\{D(Q_a)\}_{a \in A(\Phi)}$ and $\{P_b\}_{b \in B(\Phi)}$.

The goal of the general alignment problem is to maximize, both with respect to Φ and with respect to D, the score $S(D, \Phi)$. In other words, we wish to solve the problem

$$\text{Maximize}_{D \in \mathcal{D}}\ \text{Maximum}_{\Phi \in \mathcal{C}}\ S(D, \Phi). \tag{2.1}$$

Since $\mathcal{C}$ is a finite set (say, $\mathcal{C} = \{\Phi_1, \ldots, \Phi_m\}$), we may write (2.1) in the form

$$\text{Maximize}_{D \in \mathcal{D}}\ \text{Maximum}\{S(D, \Phi_1), \ldots, S(D, \Phi_m)\}. \tag{2.2}$$

Protein alignment is a particular case of the situation explained above. The goal is to find similarities between two proteins $\mathcal{P}$ and $\mathcal{Q}$, represented by the coordinates of their Cα atoms. The similarity is measured by a score. Several scores have been proposed in the protein literature. One of them is the 'Structal' score (Gerstein & Levitt, 1998; Subbiah *et al.*, 1993). The number of admissible correspondences m is exponential in the number of amino acids. Therefore, it is essential to possess a methodology for computing the maximum in (2.1) that does not involve exhaustive enumeration. Dynamic programming (DP) procedures (Subbiah *et al.*, 1993) are generally used for this purpose, having a computational cost that is quadratic in the number of Cα atoms. A different algorithm, associated with a different definition of admissible correspondences, will also be used here to compute the maximum, being quadratic in the worst case, but generally cheaper than DP.

Assume that the 3D coordinates of the Cα atoms of protein $\mathcal{P}$ (in angstroms) are $P_1, \ldots, P_M$ and the coordinates of the Cα atoms of protein $\mathcal{Q}$ are $Q_1, \ldots, Q_N$. Under the rigid-body displacement D, the coordinates of the displaced protein $\mathcal{Q}$ are, therefore, $D(Q_1), \ldots, D(Q_N)$. Assume that Φ is a monotone bijection between $A(\Phi) \subset \{1, \ldots, N\}$ and $B(\Phi) \subset \{1, \ldots, M\}$. (We mean that $i < j \Rightarrow$

$\Phi(i) < \Phi(j)$.) The Structal score associated with the displacement D and the bijection Φ is

$$S(D, \Phi) = \sum_{k \in A(\Phi)} \frac{20}{1 + \|P_k - D(Q_{\Phi(k)})\|^2/5} - 10 \times \text{gaps}, \tag{2.3}$$

where 'gaps' is the number of cases in which at least one of the following situations occur:

- $\Phi(k)$ is defined, there exists $\ell > k$ such that $\Phi(\ell)$ is defined, but $\Phi(\ell + 1)$ is not defined;
- $\Phi^{-1}(k)$ is defined, there exists $\ell > k$ such that $\Phi^{-1}(\ell)$ is defined, but $\Phi^{-1}(\ell + 1)$ is not defined.

In Fig. 1, we give examples of bijective and nonbijective (NB) correspondences. The concept of gap applies only to the bijective case. The bijection on the left has no gaps. The central bijection has two gaps and the bijection on the right has one gap.

The alignment problem associated with the Structal score consists of finding Φ and D such that $S(D, \Phi)$ is maximal. A global optimization procedure for achieving this objective was given in Kolodny & Linal (2004). However, this method is not computationally affordable and, in practice, a heuristic procedure called the Structal Method (Gerstein & Levitt, 1998; Subbiah *et al.*, 1993) is generally used. In Kolodny *et al.* (2005), the Structal Method was reported as the best available practical algorithm for protein alignment. Each iteration of the Structal Method consists of two steps.

1. Update Φ: Given the positions $P_1, \ldots, P_M$ and $D(Q_1), \ldots, D(Q_N)$, the monotone bijection Φ that maximizes the score (fixing D) is computed using DP (Needleman & Wunsch, 1970).

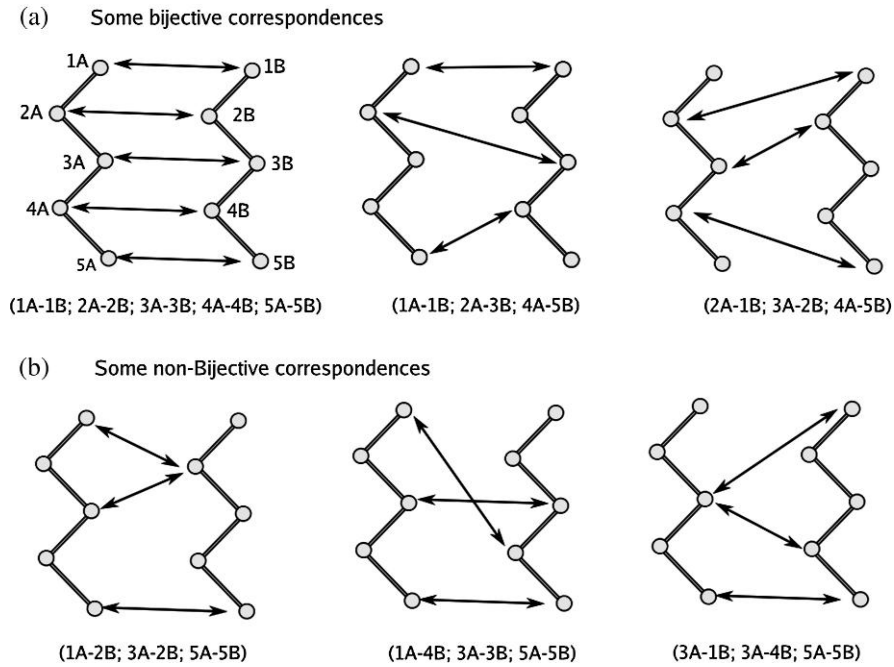

FIG. 1. Examples of correspondences that form the Φ domain: (a) Bijective correspondences and (b) NB correspondences, valid only for NB methods.

2. Update D: Assume that the graph of Φ is $\{(k_1, \Phi(k_1)), \ldots, (k_s, \Phi(k_s))\}$. Then the rigid-body displacement that minimizes $\sum_{\ell=1}^{s} \left\| P_{k_\ell} - D(Q_{\Phi(k_\ell)}) \right\|^2$ is computed.

The computation of D at the second step of the Structal Method involves the solution of the well-known Procrustes problem (Kabsch, 1978; Kearsley, 1989). The main drawback of the Structal Method is that the Update Φ step requires the optimization of a function (the Structal score) with respect to Φ and the Update D step involves the optimization of a different function (the sum of squared distances) with respect to D. This may lead to oscillation (Martinez *et al.*, 2007).

The Structal Method is the most efficient 'superposition' method for protein alignment. Superposition methods are iterative algorithms whose main iteration has two phases.

1. Update Φ: Given the positions $P_1, \ldots, P_M$ and $D(Q_1), \ldots, D(Q_N)$, the admissible correspondence Φ that maximizes S (fixing D) is computed.
2. Update D: Assume that the graph of Φ is $\{(k_1, \Phi(k_1)), \ldots, (k_s, \Phi(k_s))\}$. Then a rigid-body displacement that usually improves the score associated with this correspondence is computed.

Martinez *et al.* (2007) introduced two superposition methods that are guaranteed to improve the score by means of line-search continuous optimization algorithms. Trust-region versions of these algorithms are introduced and analyzed in the present paper.

3. LOVO algorithm

Assume that $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$. Define, for all $x \in \mathbb{R}^n$,

$$f_{\min}(x) = \min\{ f_1(x), \ldots, f_m(x)\}.$$

We will consider the optimization problem

$$\text{Minimize } f_{\min}(x). \tag{3.1}$$

This is a LOVO problem as defined in Andreani *et al.* (2008b). Let us identify the transformation D with the set of parameters by means of which D is defined (rotation angles and translation in the case of rigid-body displacements). Writing $x = D$ and $f_i(x) = -S(D, \Phi_i)$, we observe that (2.2) is a particular case of (3.1). Therefore, the protein alignment problems defined in Section 2 are order-value optimization problems in the sense of (3.1). We will assume that the second derivatives of f_i are Lipschitz continuous on a sufficiently large set, for all $i = 1, \ldots, m$. This requirement is clearly fulfilled when S is the Structal score.

For all $x \in \mathbb{R}^n$, we define

$$I_{\min}(x) = \{i \in \{1, \ldots, m\} \mid f_i(x) = f_{\min}(x)\}.$$

Here, we will define a Newtonian trust-region method for solving (3.1). This method will be applied to the protein alignment problem.

Before defining the main algorithm, let us give a technical lemma, which will be useful in proving that our method is well defined and converges. The proof of this lemma may be considered an exercise on the applications of Taylor's multidimensional formula and will be omitted here. Observe that this lemma refers to an arbitrary sequence $\{x_j\}$ converging to x, not necessarily generated by a particular algorithm.

LEMMA 3.1 Let $\{x_j\}$ be a sequence that converges to $\widehat{x} \in \mathbb{R}^n$ and let $f\colon \mathbb{R}^n \to \mathbb{R}$ possess Lipschitz continuous second derivatives on an open and convex set that contains $\{x_j\}$. We define ψ^j, the second-order quadratic approximation of $f(x)$, by

$$\psi^j(x) \equiv f(x_j) + \nabla f(x_j)^{\mathrm{T}}(x - x_j) + \tfrac{1}{2}(x - x_j)^{\mathrm{T}}\nabla^2 f(x_j)(x - x_j).$$

Assume that $\{\Delta_j\}$ is a sequence of positive numbers that tends to zero and define $\overline{x}_j$ as a global minimizer of $\psi^j(x)$ subject to $\|x - x_j\| \leqslant \Delta_j$. Finally, assume that the condition

$$\nabla f(\widehat{x}) = 0 \quad \text{and} \quad \nabla^2 f(\widehat{x}) \succcurlyeq 0 \tag{3.2}$$

does not hold, and define

$$\rho_j = \frac{f(\overline{x}_j) - f(x_j)}{\psi^j(\overline{x}_j) - \psi^j(x_j)}. \tag{3.3}$$

Then

$$\lim_{j\to\infty} \rho_j = 1.$$

Let us now introduce the main algorithm used in this work. When applied to ordinary smooth minimization ($m = 1$), this method is similar to Algorithm basic trust-region (BTR) of Conn *et al.* (2000). The precise relations will be discussed in a remark below. The specific characteristics of the algorithm were chosen so as to take account of the application to the protein alignment problem. For example, we compute the global solution of the Euclidean trust-region problem, instead of an approximate solution, because in the protein alignment application the cost of solving trust-region problems is negligible in comparison to the cost of computing the objective function. Moreover, the trust-region radius at the beginning of each iteration is chosen independently of the trust-region radius at the previous iteration. The motivation for this decision is that, in the protein alignment problem, the function f_i that defines the quadratic model changes abruptly between consecutive iterations. Therefore, there is no reason to believe that the information provided by an old trust-region radius would be useful in deciding the size of a new trust region. The independence of the trust-region radius is expressed in Algorithm 3.2 stating that $\Delta_k^0 > \Delta_{\min}$ at the beginning of each iteration (Step 3.1 below). This strategy has been used in Friedlander *et al.* (1994) in the context of first-order trust-region methods for solving smooth equations. In Section 4, we give more details on the initial trust-region choice.

ALGORITHM 3.2 Assume that $\Delta_{\min} > 0$, $\sigma_1, \sigma_2 \in (0, 1)$ (with $\sigma_1 < \sigma_2$) and $\alpha \in (0, 1)$ are given independently of k. Let $x_0 \in \mathbb{R}^n$ be the initial approximation to the solution of (3.1).

For all $k \in \mathbb{N}$, $i \in \{1, \ldots, m\}$, $x \in \mathbb{R}^n$, we define

$$\psi_i^k(x) = f_i(x_k) + \nabla f_i(x_k)^{\mathrm{T}}(x - x_k) + \tfrac{1}{2}(x - x_k)^{\mathrm{T}}\nabla^2 f_i(x_k)(x - x_k).$$

Step 1. Initialize $k \leftarrow 0$.

Step 2. Choose $\nu(k) \in I_{\min}(x_k)$. If

$$\|\nabla f_{\nu(k)}(x_k)\| = 0 \quad \text{and} \quad \nabla^2 f_{\nu(k)}(x_k) \succcurlyeq 0, \tag{3.4}$$

terminate the execution of the algorithm.

Step 3. Newton trust-region step.

Step 3.1. Choose $\Delta_k^0 \geqslant \Delta_{\min}$. Set $\Delta \leftarrow \Delta_k^0$.

Step 3.2. Compute $\overline{x}(\Delta)$, a global minimizer of $\psi_{\nu(k)}^k(x)$, subject to $\|x - x_k\| \leqslant \Delta$.

Step 3.3. If

$$f_{\min}(\overline{x}(\Delta)) \leqslant f_{\min}(x_k) + \alpha[\psi_{\nu(k)}^k(\overline{x}(\Delta)) - \psi_{\nu(k)}^k(x_k)], \tag{3.5}$$

define $x_{k+1} = \overline{x}(\Delta)$, $\Delta_k = \Delta$, $k \leftarrow k+1$ and go to Step 2.
Else, choose $\Delta_{\text{new}} \in [\sigma_1\|\overline{x}(\Delta) - x_k\|, \sigma_2\Delta]$, $\Delta \leftarrow \Delta_{\text{new}}$ and go to Step 3.2.

REMARK 3.3 When applied to ordinary smooth minimization (LOVO with $m = 1$), Algorithm 3.2 is similar to Algorithm BTR of Conn *et al.* (2000). The main difference is in the choice of the initial trust-region radius employed at the beginning of each iteration. In BTR, it is imposed that $\Delta_k^0 \geqslant \gamma_2\Delta_{k-1}$, where $\gamma_2 \in (0, 1)$ is an algorithmic parameter. The reasons for our choice of $\Delta_k^0 \geqslant \Delta_{\min}$ are given above. On the other hand, our choice of the trial step by minimization of the model is a particular case of the step calculation of BTR. The remaining differences are merely formal. For example, we do not change the iteration number when the sufficient descent condition fails, whereas in BTR this failure defines an 'unsuccessful iteration' and x_{k+1} is set to be equal to x_k.

In Theorem 3.4, we prove that if (3.4) does not hold at x_k, then the iteration that computes x_{k+1} is well defined. That is, after a finite number of reductions of Δ, one obtains x_{k+1} such that the sufficient descent criterion (3.5) holds. Using the terminology of Algorithm BTR of Conn *et al.* (2000), this theorem says that if there are only finitely many successful iterations, then x_k is a critical point for all k large enough. So this theorem roughly corresponds to Theorem 6.4.4 of Conn *et al.* (2000) for smooth problems.

In the rest of the paper, we will assume that, for all $i = 1, \ldots, m$, $\nabla^2 f_i(x)$ is Lipschitz continuous in an open and convex set that contains all the iterates generated by Algorithm 3.2.

THEOREM 3.4 If $x_k, \nu(k)$ do not satisfy (3.4), then x_{k+1} is well defined and satisfies

$$f_{\min}(x_{k+1}) \leqslant f_{\min}(x_k) + \alpha[\psi_{\nu(k)}^k(x_{k+1}) - \psi_{\nu(k)}^k(x_k)] < f_{\min}(x_k). \tag{3.6}$$

Proof. Assume that $x_k, \nu(k)$ do not satisfy (3.4). Define $i = \nu(k)$. Then,

$$\nabla f_i(x_k) \neq 0 \tag{3.7}$$

or

$$\nabla f_i(x_k) = 0 \quad \text{and} \quad \nabla^2 f_i(x_k) \not\succcurlyeq 0. \tag{3.8}$$

Observe that

$$\psi_i^k(x_k) = f_i(x_k) = f_{\min}(x_k). \tag{3.9}$$

For all $\Delta > 0$, we define $\overline{x}(\Delta)$ as a minimizer of $\psi_i^k(x)$ subject to $\|x - x_k\| \leqslant \Delta$. By (3.7) and (3.8), x_k is not a minimizer of this subproblem.

Define, for all $\Delta > 0$,

$$\rho(\Delta) = \frac{f_i(\overline{x}(\Delta)) - f_i(x_k)}{\psi_i^k(\overline{x}(\Delta)) - \psi_i^k(x_k)}.$$

By Lemma 3.1, if $\{\Delta_j\}$ is a sequence of positive numbers that tends to zero, we have

$$\lim_{j\to\infty} \rho(\Delta_j) = 1.$$

Therefore, $\lim_{\Delta\to 0}\rho(\Delta) = 1$. Since $f_{\min}(\overline{x}(\Delta)) \leqslant f_i(\overline{x}(\Delta))$, this implies that for Δ sufficiently small, (3.6) will be fulfilled. So the proof is complete. □

REMARK 3.5 Theorem 3.4 says that if Algorithm 3.2 terminates at x_k, then there exists $i \in I_{\min}(x_k)$ such that x_k is a second-order stationary point of f_i. The reciprocal is not true. For example, define, with $n = 1$ and $m = 2$, $f_1(x) = x$ and $f_2(x) = x^2$. Clearly, 0 is a second-order stationary point of f_2. However, if one chooses $\nu(k) = 1$, the algorithm will not stop and, in fact, it will find a better point such that $f_{\min}(x) < f_{\min}(0)$.

The following theorem is our main global convergence result. Essentially, we prove that *all* the limit points of sequences generated by Algorithm 3.2 are first-order and second-order stationary. It is interesting to observe that in the smooth case ($m = 1$), second-order stationarity of all the limit points could not be obtained for the radius-updating rules of BTR. The additional requirement A.A.3 (Conn *et al.*, 2000, p. 158) was imposed in BTR to obtain this result. It is interesting to realize that by using the simple requirement that $\Delta_k^0 \geqslant \Delta_{\min}$, the second-order criticality of all the accumulation points may be obtained without serious difficulties.

THEOREM 3.6 Assume that, for an infinite set of indices $K \subset \mathbb{N}$, we have $\lim_{k\in K} = x_*$, where $\{x_k\}$ is an infinite sequence generated by Algorithm 3.2. Then the following hold.

1. If $i \in \{1, \ldots, m\}$ is such that $\nu(k) = i$ for infinitely many indices $k \in K$, then
$$\nabla f_i(x_*) = 0 \quad \text{and} \quad \nabla^2 f_i(x_*) \succcurlyeq 0. \tag{3.10}$$
2. There exists $i \in I_{\min}(x_*)$ such that $\nabla f_i(x_*) = 0$ and $\nabla^2 f_i(x_*) \succcurlyeq 0$.

Proof. The sequence $\{\Delta_k\}_{k\in K}$ satisfies one of the following possibilities:

$$\liminf_{k\in K} \Delta_k = 0 \tag{3.11}$$

or

$$\{\Delta_k\}_{k\in K} \text{ is bounded away from 0.} \tag{3.12}$$

Assume, initially, that (3.11) holds. Then there exists an infinite set of indices $K_1 \subset K$ such that

$$\lim_{k\in K_1} \Delta_k = 0. \tag{3.13}$$

Therefore, there exists $k_1 \in \mathbb{N}$ such that $\Delta_k < \Delta_{\min}$ for all $k \in K_2$, where $K_2 \equiv \{k \in K_1 \mid k \geqslant k_1\}$. Since, at each iteration, the initial trial trust-region radius is greater than or equal to $\Delta_{\min}$, it turns out that, for all $k \in K_2$, there exist $\overline{\Delta}_k$ and $\overline{x}(\overline{\Delta}_k)$ such that $\overline{x}(\overline{\Delta}_k)$ is a global solution of

$$\begin{aligned} &\text{Minimize } \psi_i^k(x), \\ &\|x - x_k\| \leqslant \overline{\Delta}_k, \end{aligned} \tag{3.14}$$

but

$$f_i(\overline{x}(\overline{\Delta}_k)) \geqslant f_{\min}(\overline{x}(\overline{\Delta}_k)) > f_i(x_k) + \alpha[\psi_i^k(\overline{x}(\overline{\Delta}_k)) - \psi_i^k(x_k)]. \tag{3.15}$$

Clearly, (3.14) implies that $\overline{x}(\overline{\Delta}_k)$ is a global solution of

$$\begin{array}{c}\text{Minimize } \psi_i^k(x),\\ \|x - x_k\| \leqslant \|\overline{x}(\overline{\Delta}_k) - x_k\|.\end{array} \tag{3.16}$$

By the definition of Δ_k at Step 3 of Algorithm 3.2, we have

$$\Delta_k > \sigma_1 \|\overline{x}(\overline{\Delta}_k) - x_k\|. \tag{3.17}$$

Therefore, by (3.13) and (3.17), we have

$$\lim_{k\in K_3} \|\overline{x}(\overline{\Delta}_k) - x_k\| = 0. \tag{3.18}$$

Define

$$\rho_k = \frac{f_i(\overline{x}(\overline{\Delta}_k)) - f_i(x_k)}{\psi_i^k(\overline{x}(\overline{\Delta}_k)) - \psi_i^k(x_k)}. \tag{3.19}$$

By Lemma 3.1, if (3.10) does not hold, we have that $\lim_{k\in K_2} \rho_k = 1$, which contradicts (3.15). Therefore, (3.10) is proved in the case (3.11).

Let us consider the possibility (3.12). Since $f_{\min}(x_{k+1}) \leqslant f_{\min}(x_k)$ for all k, and $\lim_{k\in K} x_k = x_*$, by the continuity of $f_{\min}$, we have $\lim_{k\to\infty}[f_{\min}(x_{k+1}) - f_{\min}(x_k)] = 0$. Therefore, by (3.6),

$$\lim_{k\in K}(\psi_i^k(x_{k+1}) - \psi_i^k(x_k)) = 0. \tag{3.20}$$

Define $\underline{\Delta} = \inf_{k\in K_1} \Delta_k > 0$ and let $\widehat{x}$ be a global solution of

$$\begin{array}{c}\text{Minimize } \nabla f_i(x_*)^{\mathrm{T}}(x - x_*) + \frac{1}{2}(x - x_*)^{\mathrm{T}}\nabla^2 f_i(x_*)(x - x_*),\\ \|x - x_*\| \leqslant \underline{\Delta}/2.\end{array} \tag{3.21}$$

Let $k_3 \in \mathbb{N}$ such that

$$\|x_k - x_*\| \leqslant \underline{\Delta}/2 \tag{3.22}$$

for all $k \in K_4 \equiv \{k \in K \mid k \geqslant k_3\}$.

By (3.21) and (3.22), for all $k \in K_4$, we have

$$\|\widehat{x} - x_k\| \leqslant \underline{\Delta} \leqslant \Delta_k. \tag{3.23}$$

Therefore, since x_{k+1} is a global minimizer of $\psi_i^k(x)$ subject to $\|x - x_k\| \leqslant \Delta_k$, we get

$$\psi_i^k(x_{k+1}) \leqslant \psi_i^k(\widehat{x}) = \psi_i^k(x_k) + \nabla f_i(x_k)^{\mathrm{T}}(\widehat{x} - x_k) + \tfrac{1}{2}(\widehat{x} - x_k)^{\mathrm{T}}\nabla^2 f_i(x_k)(\widehat{x} - x_k). \tag{3.24}$$

So

$$\psi_i^k(x_{k+1}) - \psi_i^k(x_k) \leqslant \nabla f_i(x_k)^{\mathrm{T}}(\widehat{x} - x_k) + \tfrac{1}{2}(\widehat{x} - x_k)^{\mathrm{T}}\nabla^2 f_i(x_k)(\widehat{x} - x_k). \tag{3.25}$$

By (3.20), taking limits in (3.25) for $k \in K_3$, we have that

$$0 \leqslant \nabla f_i(x_*)^{\mathrm{T}}(\widehat{x} - x_*) + \tfrac{1}{2}(\widehat{x} - x_*)^{\mathrm{T}}\nabla^2 f_i(x_*)(\widehat{x} - x_*).$$

Therefore, x_* is a global minimizer of (3.21) for which the constraint $\|x - x_*\| < \Delta/2$ is inactive. This implies that $\nabla f_i(x_*) = 0$ and $\nabla^2 f_i(x_*) \succcurlyeq 0$.

So the first part of the statement has been proved.

Now, let us prove the second part of the statement. Since $\{1, \ldots, m\}$ is finite, there exists $i \in \{1, \ldots, m\}$ such that $i = \nu(k)$ for infinitely many indices $k \in K_1 \subset K$. So, for all $k \in K_1$,

$$f_i(x_k) \leqslant f_j(x_k) \quad \forall\, j \in \{1, \ldots, m\}.$$

Taking limits in the previous inequality and using the first part of the statement, we get

$$f_i(x_*) \leqslant f_j(x_*) \quad \forall\, j \in \{1, \ldots, m\}.$$

Therefore, $i \in I_{\min}(x_*)$. □

ASSUMPTION 3.7 We say that this assumption holds at x_* if for all $i \in I_{\min}(x_*)$ such that $\nabla f_i(x_*) = 0$, we have $\nabla^2 f_i(x_*) \succ 0$.

LEMMA 3.8 Assume that x_* is a limit point of a sequence generated by Algorithm 3.2 and that Assumption 3.7 holds at x_*. Then there exists $\varepsilon > 0$ such that the reduced ball $\mathcal{B}(x_*, \varepsilon) - \{x_*\}$ does not contain limit points of $\{x_k\}$.

Proof. If $i \in I_{\min}(x_*)$ and $\nabla f_i(x_*) = 0$, we have, by Assumption 3.7, that $\nabla^2 f_i(x_*)$ is positive definite. Therefore, by the inverse function theorem, $\nabla f_i(x) \neq 0$ for all $x \neq x_*$ in a neighbourhood of x_*.

If $i \in I_{\min}(x_*)$ and $\nabla f_i(x_*) \neq 0$, then $\nabla f_i(x) \neq 0$ in a neighbourhood of x_*.

Finally, if $i \notin I_{\min}(x_*)$, we have $f_i(x_*) > f_{\min}(x_*)$. So $f_i(x) > f_{\min}(x)$ and $i \notin I_{\min}(x)$ for all x in a neighbourhood of x_*.

Therefore, there exists $\varepsilon > 0$ such that $\nabla f_i(x) \neq 0$ whenever $i \in I_{\min}(x)$ and $x \in \mathcal{B}(x_*, \varepsilon) - \{x_*\}$. Therefore, by Theorem 3.6, x cannot be a limit point of a sequence generated by Algorithm 3.2, for all $x \in \mathcal{B}(x_*, \varepsilon) - \{x_*\}$. □

LEMMA 3.9 Suppose that x_* satisfies Assumption 3.7 and $\lim_{k \in K} x_k = x_*$, where $\{x_k\}$ is a sequence generated by Algorithm 3.2 and K is an infinite subset of indices. Then

$$\lim_{k \in K} \|x_{k+1} - x_k\| = 0.$$

Proof. Let I be the set of integers in $\{1, \ldots, m\}$ such that $i = \nu(k)$ for infinitely many indices $k \in K$. Since $f_i(x_k) = f_{\min}(x_k)$ infinitely many times, we obtain, taking limits, that $i \in I_{\min}(x_*)$ for all $i \in I$. Therefore, by Theorem 3.6, $\nabla f_i(x_*) = 0$ and, consequently, $\lim_{k \in K} \nabla f_i(x_k) = 0$ for all $k \in I$. This implies that

$$\lim_{k \in K} \nabla f_{\nu(k)}(x_k) = 0. \tag{3.26}$$

Moreover, by Assumption 3.7, $\nabla^2 f_i(x_k) \succ 0$ for all $i \in I$. So, by the continuity of the Hessians, there exists $c > 0$ such that, for all $\lambda \geqslant 0$ and $k \in K$ large enough,

$$\|[\nabla^2 f_{\nu(k)}(x) + \lambda I]^{-1}\| \leqslant \|\nabla^2 f_{\nu(k)}(x)^{-1}\| \leqslant 2 \max_{i \in I} \|\nabla^2 f_i(x_*)^{-1}\| = c. \tag{3.27}$$

Now, x_{k+1} is a solution of

$$\text{Minimize } \psi^k_{\nu(k)}(x) \text{ subject to } \|x - x_k\| \leqslant \Delta_k. \tag{3.28}$$

Therefore, by the Karush-Kuhn-Tucker (KKT) conditions of (3.28),

$$\begin{aligned}(\nabla^2 f_{v(k)}(x_k) + \lambda_k I)(x_{k+1} - x_k) + \nabla f_{v(k)}(x_k) = 0,\\ \lambda_k \|x_{k+1} - x_k\| = 0,\\ \lambda_k \geqslant 0, \quad \|x_{k+1} - x_k\| \leqslant \Delta_k.\end{aligned} \tag{3.29}$$

Therefore, by (3.27), $\|x_{k+1} - x_k\| \leqslant c\|\nabla f_{v(k)}(x_k)\|$ for $k \in K$ large enough and, by (3.26), $\lim_{k\in K} \|x_{k+1} - x_k\| = 0$, as we wanted to prove. □

THEOREM 3.10 Assume that x_* is a limit point of a sequence $\{x_k\}$ generated by Algorithm 3.2. Suppose that Assumption 3.7 holds at x_*. Then the whole sequence x_k converges quadratically to x_*.

Proof. Let K be an infinite sequence of indices such that $\lim_{k\in K} x_k = x_*$. Let us prove first that

$$\lim_{k\to\infty} x_k = x_*.$$

By Lemma 3.8, there exists $\varepsilon > 0$ such that x_* is the unique limit point in the ball with centre x_* and radius ε. Define

$$K_1 = \{k \in \mathbb{N} \mid \|x_k - x_*\| \leqslant \varepsilon/2\}.$$

The subsequence $\{x_k\}_{k\in K_1}$ converges to x_*, since x_* is its unique possible limit point. Therefore, by Lemma 3.9,

$$\lim_{k\in K_1} \|x_{k+1} - x_k\| = 0. \tag{3.30}$$

Let k_1 be such that, for all $k \in K_1$, $k \geqslant k_1$,

$$\|x_{k+1} - x_k\| \leqslant \frac{\varepsilon}{2}.$$

The set $\mathcal{B}(x_*, \varepsilon) - \mathcal{B}(x_*, \varepsilon/2)$ does not contain limit points of $\{x_k\}$. Therefore, there exists $k_2 \in \mathbb{N}$ such that, for all $k \geqslant k_2$,

$$\|x_k - x_*\| \leqslant \frac{\varepsilon}{2} \quad \text{or} \quad \|x_k - x_*\| > \varepsilon.$$

Let $k \in K_1$ such that $k \geqslant \max\{k_1, k_2\}$. Then

$$\|x_{k+1} - x_*\| \leqslant \|x_k - x_*\| + \|x_{k+1} - x_k\| \leqslant \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.$$

Since x_{k+1} cannot belong to $\mathcal{B}(x_*, \varepsilon) - \mathcal{B}(x_*, \varepsilon/2)$, it turns out that $\|x_{k+1} - x_*\| \leqslant \varepsilon/2$. Therefore, $k + 1 \in K_1$. So we may prove by induction that $x_\ell \in K_1$ for all $\ell \geqslant k$. By (3.30), this implies that

$$\lim_{k\to\infty} x_k = x_*. \tag{3.31}$$

Let us now prove the quadratic convergence.

Let $I_\infty \subset \{1, \ldots, m\}$ be the set of indices i such that $i = v(k)$ infinitely many times. Then there exists k_2 such that for all $k \geqslant k_2$, $v(k) \in I_\infty$.

Now, if $i \in I_\infty$ it turns out that $f_i(x_k) = f_{\min}(x_k)$ infinitely many times. Therefore, by (3.31) and the continuity of f_i and $f_{\min}$, we have $i \in I_{\min}(x_*)$. By Theorem 3.6, $\nabla f_i(x_*) = 0$. Therefore, by Assumption 3.7, $\nabla^2 f_i(x_*) \succ 0$. Thus, by the continuity of the Hessians, there exists $c_i > 0$ such that $\nabla^2 f_i(x)$ is positive definite and $\|\nabla^2 f_i(x)^{-1}\| \leqslant c_i$ for all x in a neighbourhood of x_*. This implies that there exists $k_3 \geqslant k_2$ such that $\nabla^2 f_{\nu(k)}(x_k)$ is positive definite and $\|\nabla^2 f_{\nu(k)}(x)^{-1}\| \leqslant \beta \equiv \max\{c_i, i \in I_\infty\}$ for all $k \geqslant k_3$.

Therefore, for all $k \geqslant k_3$, we may define

$$\overline{x}_k = x_k - \nabla^2 f_{\nu(k)}(x_k)^{-1} \nabla f_{\nu(k)}(x_k). \tag{3.32}$$

Since $\nabla f_i(x_*) = 0$ for all $i \in I_\infty$, we have $\lim_{k\to\infty} \nabla f_{\nu(k)}(x_k) = 0$. Then, by the boundedness of $\|\nabla^2 f_{\nu(k)}(x)^{-1}\|$, we have $\|\overline{x}_k - x_k\| \to 0$; therefore, for k large enough, $\|\overline{x}_k - x_k\| \leqslant \Delta_{\min}$. But $\overline{x}_k$ is, for k large enough, the unconstrained minimizer of $\psi^k_{\nu(k)}$. Therefore, since the first trust-region radius at each iteration is greater than or equal to $\Delta_{\min}$, it turns out that, for k large enough, $\overline{x}_k$ is the first trial point at each iteration of Algorithm 3.2.

By (3.32), we have

$$|\psi^k_{\nu(k)}(\overline{x}_k) - \psi^k_{\nu(k)}(x_k)| = \tfrac{1}{2}(\overline{x}_k - x_k)^{\mathrm{T}} \nabla^2 f_{\nu(k)}(x_k)(\overline{x}_k - x_k).$$

Therefore, by Assumption 3.7, there exists $c > 0$ such that for k large enough,

$$|\psi^k_{\nu(k)}(\overline{x}_k) - \psi^k_{\nu(k)}(x_k)| \geqslant c\|\overline{x}_k - x_k\|^2. \tag{3.33}$$

Define

$$\rho_k = \frac{f_{\nu(k)}(\overline{x}_k) - f_{\nu(k)}(x_k)}{\psi^k_{\nu(k)}(\overline{x}_k) - \psi^k_i(x_k)}.$$

By (3.33), and Taylor's formula, we have

$$|\rho_k - 1| = \left| \frac{f_i(\overline{x}_k) - f_i(x_k) - [\psi^k_i(\overline{x}_k) - \psi^k_i(x_k)]}{\psi^k_i(\overline{x}_k) - \psi^k_i(x_k)} \right| \leqslant \frac{o(\|\overline{x}_k - x_k\|^2)}{c\|\overline{x}_k - x_k\|^2}. \tag{3.34}$$

Since $\|\overline{x}_k - x_k\| \to 0$, we see that $\rho_k \to 1$. Therefore, for k large enough, the sufficient descent condition (3.6) is satisfied at the first trial point $\overline{x}_k$. Therefore, $x_{k+1} = \overline{x}_k$ for k large enough. This means that, for $k \geqslant k_3$ large enough,

$$x_{k+1} = x_k - \nabla^2 f_{\nu(k)}(x_k)^{-1} \nabla f_{\nu(k)}(x_k).$$

Then, by the elementary local convergence theory of Newton's method (see, e.g. Dennis & Schnabel, 1983, pp. 90–91), we find that, for k large enough,

$$\|x_{k+1} - x_*\| \leqslant \beta\gamma \|x_k - x_*\|^2,$$

where γ is a Lipschitz constant for all the Hessians $\nabla^2 f_i(x)$. □

4. Numerical results

Algorithm 3.2 was implemented with the following specifications.

1. We choose $\alpha = 0.1$, $\sigma_1 = 0.001$ and $\sigma_2 = 5/9$.
2. The initial Δ_k^0 at each iteration was chosen as the average norm of the Cα atoms of the protein $\mathcal{Q}$ at their original positions multiplied by 10. In terms of Algorithm 3.2, we may consider that $\Delta_{\min}$ is equal to Δ_k^0.
3. When (3.5) does not hold, Δ_{new} is computed in the following way. We define

$$\text{Ared} = f_{\min}(x_k) - f_{\min}(\bar{x}),$$

$$\text{Pred} = \psi^k_{\nu(k)}(x_k) - \psi^k_{\nu(k)}(\bar{x})$$

and

$$\Delta_{\text{new}} = \max\left\{0.001, \frac{\text{Pred}}{2(\text{Pred} - \text{Ared})}\right\} \times \|\bar{x} - x_k\|. \tag{4.1}$$

The fact that $\Delta_{\text{new}} \leqslant \sigma_2\|\bar{x} - x_k\|$ is guaranteed for $\sigma_2 = 5/9$ because Ared $< \alpha$Pred in the case where Δ_{new} needs to be computed and $\alpha = 0.1$ in our implementation.

We arrived to the formula (4.1) after experimentation with other possibilities, including the classical ones associated with smooth trust-region methods (see, e.g. Fletcher, 1987, pp. 95–96).

In order to optimize the behaviour of Algorithm 3.2, it was crucial to define a 'big' trust-region radius at the beginning of each iteration. The trust-region radius that defines the first trial point was chosen to be independent of the last trust-region radius employed at the previous iteration. This decision allowed the algorithm to use, very frequently, pure Newton steps and avoided artificial short steps far from the solution. Since the function f_i that defines $f_{\min}$ at a trial point may be different than the one that defines $f_{\min}$ at the current point, the quadratic model of $f_{\min}$ tends to *underestimate* the true value in many cases.

We implemented Algorithm 3.2 under the framework of the Betra standard trust-region method for box-constrained optimization (Andretta *et al.*, 2005) (see www.ime.usp.br/∼egbirgin/tango). Since our problem is unconstrained, we set artificial bounds -10^{20} and 10^{20} for each variable. The Betra code needed to be adapted to the algorithmic decisions described at the beginning of this section. In the unconstrained case, Betra is a standard trust-region method that uses the Moré–Sorensen algorithm (Moré & Sorensen, 1983). Many line-search and trust-region methods for smooth unconstrained minimization may be used for solving (3.1) if one simply ignores the nonsmoothness of the objective function 'defining' $\nabla f_{\min}(x) = \nabla f_i(x)$ and $\nabla^2 f_{\min}(x) = \nabla^2 f_i(x)$ for some $i \in I_{\min}(x)$. Of course, the theoretical properties may change for each algorithmic choice. For example, we cannot expect the thesis of Theorem 3.6 to hold if one uses the TRON (Lin & Moré, 1999) or BOX-QUACAN (Friedlander *et al.*, 1994) algorithms, since this theorem does not hold for those algorithms in the ordinary smooth case ($m = 1$).

Numerical experiments were run on an AMD Opteron 242 with 1 Gb of RAM running Linux. The software was compiled with the GNU fortran compiler version 3.3 with the '-O3 -ffast-math' options.

For each pair of proteins, our purpose is to find the displacement that maximizes similarity. We adopt the Structal score (2.3) as the similarity measure. This means that, for computing the objective function, given the relative positions of the two proteins, one needs to find the admissible bijection that

maximizes (2.3). This is done using DP. Therefore, a DP problem must be solved at each iteration of the optimization algorithm for computing the objective function. The trust-region algorithm that optimizes the Structal score using DP will be called DP-Trust. Analogously, a line-search algorithm that optimizes the same score is described in Martinez *et al.* (2007) and will be called DP-LS.

Since the DP procedure is, by far, the most expensive task of DP-Trust and DP-LS, we introduced a different problem, where similarity is defined as follows. On one hand, admissible correspondences between the proteins are merely functions between subjects of $\mathcal{Q}$ and $\mathcal{P}$, for which neither bijectivity nor monotonicity is required. The cardinality of the domain of these functions is prescribed to be 90% of the Cα atoms of $\mathcal{Q}$. On the other hand, penalty gaps are not employed in this formulation. As a consequence, the computation of the objective function does not need DP computations. A cheap procedure for computing the NB correspondences is described in Martinez *et al.* (2007) and Andreani *et al.* (2008a). The algorithm for solving this problem using the trust-region approach will be called NB-Trust) and the line-search algorithm having the same purpose will be called NB-LS.

DP-Trust (DP-LS) and NB-Trust (NB-LS) aim to solve two different mathematical problems. However, both mathematical problems are directed to the same biological purpose of getting the best possible alignment. For this reason, after finishing the execution of NB algorithms, we computed using DP (only once) the (bijective, nonmonotone) Structal score corresponding to the final displacements obtained (by NB). This allows us to compare the model simplification that leads to NB-like problems with the original DP-like formulation. We will see that, for similar proteins, the best NB correspondence found by the NB approach tends to satisfy monotonocity and bijectivity.

It is worth mentioning that the NB formulation can be applied to more general structural alignment problems, where monotonicity and bijectivity are not required (Andreani *et al.*, 2008a).

The computer time of both DP and NB approaches is dominated by the computation of the objective function. In DP-Trust, on average, 98% of the time is spent in the DP subroutine and 1.4% of the time is spent computing gradients and Hessians. Similar estimates hold for the line-search algorithm DP-LS.

All the algorithms considered here are initiated with the same initial point. Recall that 'point' in our context is equivalent to 'displacement' and is characterized by six parameters (translation and Euler angles). The initial point was obtained by means of a heuristic procedure described in Martinez *et al.* (2007) that uses internal distances and involves a DP calculation. As a consequence, each complete alignment using NB-Trust uses two DP computations, one for obtaining the initial approximation and other for computing the final (monotone, bijective) Structal score. On average, these two DP steps take 61% of the computer time. The procedure for computing the NB correspondence takes 16% of the computer time and the computation of gradients and Hessians takes 15%.

The resolution of the trust-region subproblems using the Moré–Sorensen algorithm takes, on average, less than 0.1% of the computer time, in the case of both DP-Trust and NB-Trust. Therefore, it is not worthwhile to use approximate solutions of trust-region subproblems, as many algorithms for large-scale optimization do (see Conn *et al.*, 2000; Friedlander *et al.*, 1994; Lin & Moré, 1999; Nocedal & Wright, 1999, Chapter 4, among others).

4.1 *Numerical comparison*

We compared the performances of DP-Trust, DP-LS, NB-Trust, NB-LS and Structal using 79800 alignment problems, which involve both related and unrelated proteins. A set of 400 proteins was chosen from a DALI (Holm & Sander, 1993, 1996) classification: 20 proteins were selected randomly within the DALI alignment database and the 20 best matches (according to DALI) for each of these 20 proteins were also included in the set. As a consequence, we collected 400 proteins where both different

TABLE 1 *Computer time required for an average alignment by each method*

Method	Average time per alignment (s)	One-to-all in PDB (min)	All-to-all in PDB
DP-LS	0.127	75	2.5 years
DP-Trust	0.141	82	2.8 years
NB-LS	0.033	19	7.7 months
NB-Trust	0.033	19	7.7 months
Structal	0.224	130	4.4 years

and similar structures are present, approximately grouped in sets of 20. The list of proteins used in the comparison is available at the LovoAlign site. Each alignment process was stopped when the difference between scores at two consecutive iterations was less than or equal to 10^{-6}.

4.2 *Average computer times*

Table 1 reports average computer times, disregarding the effective scores obtained by the different methods. We also show estimates of the computer time that would be used by the alignment of one protein to the whole PDB (about 35000 structures to date) and the all-on-all alignment of all PDB files ($\sim 35000 \times (35000 - 1)/2$ alignments). The numbers in the last two columns of the table were estimated using the average time per alignment and are included here only to give the reader a rough and an intuitive idea. Some single one-to-all alignments were performed and confirmed these estimates.

The methods that use NB correspondences (NB-LS and NB-Trust) are faster because computing the best NB correspondence (given the displacement) is much easier than computing the best bijective and monotone correspondence using DP. In fact, the whole application of NB methods involves two DP calls, one at the beginning, to compute the initial approximation, and the other at the end, to compute the final Structal score. As shown before, more than 60% of the computer time used by these methods is spent in these two DP calculations.

Line-search and trust-region methods turned out to be more efficient than Structal in terms of overall computer time. The introduction of the trust-region strategy in place of the line-search procedure did not improve the speed of the line-search methods.

We observe that the alignment of one protein to the whole PDB takes less than 2 h. The alignment of all the proteins in the PDB would require several months, using a single processor. However, this job may be obviously done in parallel, since different alignments are entirely independent. So, the whole task may be completed in quite affordable computer time for practical purposes.

The slightly bigger computer time required by DP-Trust relative to DP-LS is not a serious limitation, since score improvement is more important in massive alignments.

4.3 *Performance profiles*

Performance profiles (Dolan & Moré, 2002) concerning the comparison of the five methods are presented in this section. Since for each alignment there is no clear global solution known, we consider for each alignment that the score at the 'solution' is the best score obtained by the methods being compared. Then, we considered that the other method 'solved' the problem if the score obtained is similar to the best score up to a relative tolerance of 0.1%. If a method 'does not solve' a problem, we consider, that the computer time used is ∞. Let T be the total number of problems. Given the abscissa $x > 1$, the profile curve of a method takes the value y if there are yT problems in which the computer time employed

by this method is less than or equal to x times the computer time used by the best of the methods for the problem. The abscissa x is called 'relative time tolerance' in Figs 2–4 and goes from $x = 1$ to $x = 10$.

4.3.1 *DP-trust-region against Structal.* The performance profile corresponding to the comparison of DP-Trust against the Structal method is provided in Fig. 2. The performance profile curve is practically horizontal. This means that the differences detected here are almost exclusively related with the quality of the solution obtained and not with computer time. Essentially, the graphic says that DP-Trust obtained a better solution than Structal in 53% of the alignments and that Structal obtained a better score in 47% of the cases. The curve reveals that, essentially, no ties were detected in terms of final scores.

4.3.2 *DP-Trust-region against DP-LS.* In Fig. 3, the performance profile comparing DP-Trust and DP-LS is exhibited.

The differences between these algorithms are a consequence of the introduction of the trust-region strategy.

We observe that DP-Trust obtains the best scores in 71% of the alignments, while DP-LS arrives to the best solutions in 67% of the problems. Therefore, the substitution of the line-search procedure by the trust-region one improves the quality of alignments. In 38% of the problems, DP-Trust and DP-LS arrive to the same final scores. For $x > 4$, the performance profile curve is practically horizontal. This means that there are no problems in which both methods obtained the same scores being one of them more than four times faster than the other. The relative computer time in the cases in which both methods obtained the same scores can be seen for $x < 4$. For example, for $x = 2$, we observe that DP-Trust was

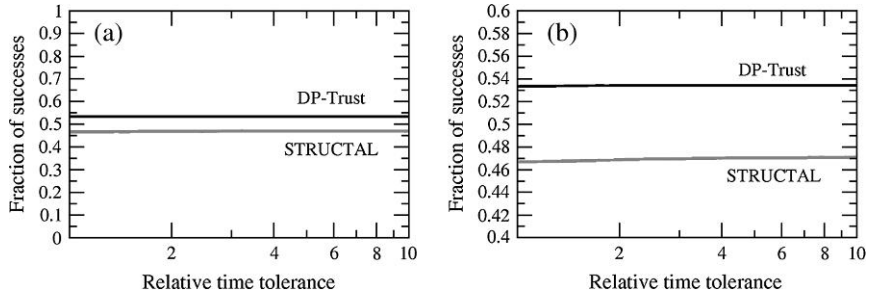

FIG. 2. Performance profile comparing the DP-Trust method with the Structal method. (b) Zoom of (a).

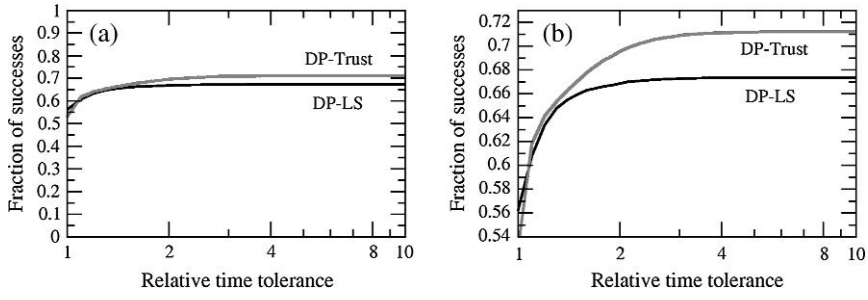

FIG. 3. Performance profile comparing the DP-Trust method with the DP-LS method. (b) Zoom of (a).

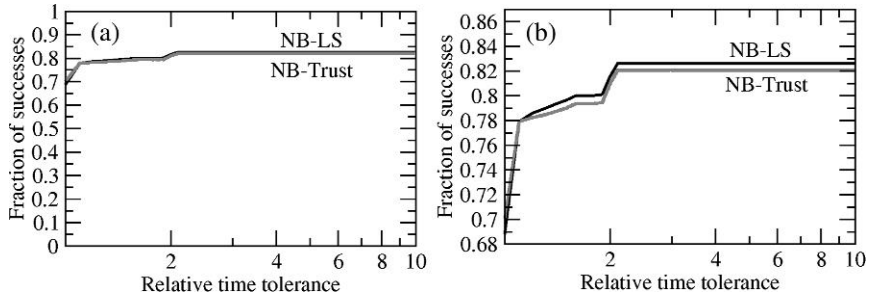

FIG. 4. Performance profile comparing the NB-Trust method with the NB-LS method. (b) Zoom of (a).

at least as good as DP-LS in 69% of the cases and that DP-LS was as least as good as DP-Trust in 65% of the problems. This means that in 34% of the problems, both methods obtained the same scores and none of them was more than two times faster than the other.

4.3.3 *NB-Trust-region against NB-LS.* There is no meaningful differences when we compare NB-Trust and NB-LS strategies, as shown in Fig. 4. In this set of alignments, NB-Trust obtains the best score (relatively 'only' to these two methods) in 81% of the problems, while NB-LS obtains best scores in 82% of the alignments.

4.4 *Robustness and score relevance*

Performance profiles illustrate the fact that protein alignment problems may have a huge number of local solutions. Of course, we prefer the best possible local optimum, but the analysis is not exhausted by this trivial observation. In protein alignment problems, one is not really interested in obtaining the best possible scores when the proteins are very dissimilar. Even getting a guaranteed global solution would be quite irrelevant in the case of very poor alignments.

What is important is to obtain good scores (if possible, the best) in the cases in which the proteins to be compared possess some reasonable degree of similarity. Recall that the alignment process returns, as a by-product, a bijection that ideally shows where is the similarity between the structures. In the case of poor alignments, the bijection found has no biological meaning at all.

With this in mind, we decided to exhibit the behaviour of each algorithm as a function of the similarity. By (2.3), the maximal Structal score that can be obtained in an alignment is 20 times the number of Cα atoms of the smallest of the two proteins being compared. For example, this is the score that one obtains comparing two identical proteins or two proteins such that one of them is a gap-free monotone subset of the other. Therefore, dividing the Structal score by the number of atoms of the smallest protein, one obtains a size-independent measure of similarity. This measure will be called 'scaled similarity'. Accordingly, we define the 'Quality' of an alignment as the best scaled similarity obtained by the algorithms DP-Trust, DP-LS, NB-Trust, NB-LS and Structal, when used to align the corresponding proteins. Consequently, the Quality of an alignment goes from 0 to 20 and we are mostly interested in obtaining the best scores for high-quality alignments. Alignments with Quality smaller than three do not have biological meaning.

In Fig. 5(a), we exhibit the percentage of problems in which each algorithm obtained the best score (up to a relative precision of 0.1%) as a function of Quality. More precisely, given the value x in the abscissa, the curves in this graphic represent the percentage of alignments with Quality greater than or equal to x in which the considered method obtained the best score.

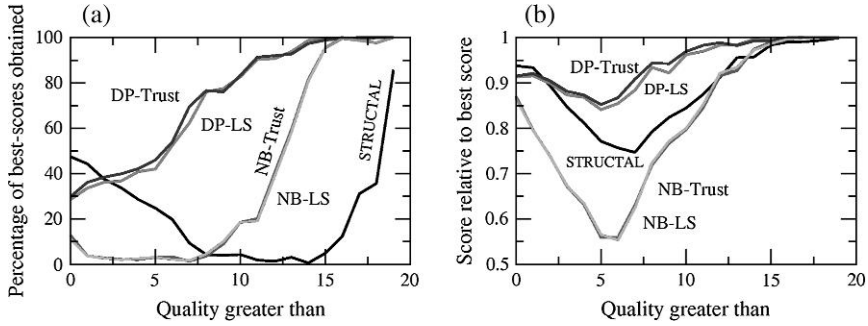

FIG. 5. Comparison of the performances of the algorithms as a function of the quality of the alignments. (a) Percentage of cases in which each algorithm obtains the best score. (b) Relative difference between the value of the score obtained by each method and the best score obtained by all methods.

Recall that, in the case of NB algorithms, although they do not optimize the original Structal score, the Structal score is computed as a post-processing step.

We observe that, including very poor alignments, the Structal strategy is able to obtain the best scores in the greatest number of cases, followed by DP methods and NB methods. This is due to the fact that NB-Trust and NB-LS tend to find the same solutions frequently, whereas the coincidence between these two methods and Structal is very rare. However, for alignments with Quality greater than only 2.5, DP-LS and DP-Trust obtain the best scores more frequently.

For alignments with Quality greater than 10, the best scores are obtained by DP-Trust and DP-LS in more than 80% of the problems, and for Quality greater than 13 this percentage grows to more than 98%.

NB procedures fail to obtain the best scores if one includes medium to poor alignments. However, for Quality greater than 14, these methods also obtain the best scores. The reason is that the best NB correspondence tends to be also monotone and bijective in these cases. Monotonicity and bijectivity may be seen as constraints defining admissibility of correspondences that tend to be inactive at the solution, in the case of good quality alignments.

In Fig. 5(b), the abscissa has the same meaning as in Fig. 5(a). Given the value x in the abscissa, we consider the set $P(x)$ of all the problems with Quality greater than or equal to x. For each method M of our study and each problem in $P(x)$, we compute the quotient between the scaled similarity obtained by M and the Quality of the alignment problem. The average of these quotients over all the problems in $P(x)$ defines the point of the curve with abscissa x corresponding to the method M in Fig. 5(b). Therefore, Fig. 5(b) shows how close to the best score each method gets, on average, as a function of the alignment quality. We can see, here, that the Structal Method obtains scores which are, on average, better than the ones obtained by NB methods. However, NB methods obtain better scores than Structal for good alignments. The use of DP in combination with smooth optimization strategies results in greater robustness, as shown by the performances of the DP-Trust and DP-LS methods, and the use of the trust-region algorithm instead of the line-search strategy improves slightly the quality of the results for all meaningful alignment qualities.

5. Final remarks

Protein alignment is a challenging area for rigorous continuous optimization. There is a lot of space for the development of algorithms with well-established convergent theories that, presumably, converge to

local optimizers and many times to global ones. We feel that line-search and trust-region methods for (2.1) are rather satisfactory, but different alternatives should be mentioned. In Andreani *et al.* (2005), problems like (2.1) were reformulated as smooth nonlinear programming problems with complementarity constraints. This reformulation should be exploited in future works.

In this paper, we showed that the trust-region approach has some advantages over the line-search algorithm in terms of robustness, at least when one deals with DP methods. We conjecture that the advantages of trust-region methods over line-search methods may be more impressive in other structural alignment problems. In particular, preliminary results for alignments in which we allow internal rotations of the objects (see Andreani *et al.*, 2008a) suggest that, in those cases, pure Newton directions are not so effective and restricted trust-region steps could help. Further research is expected with respect to flexible alignments (Li *et al.*, 2006; Shatsky *et al.*, 2002; Ye & Godzik, 2003) in the near future.

In the experiments reported here, we always used the Structal score and we recalled that the Structal Method iteration maximizes this score at its first phase and minimizes the sum of squared distances root-mean square deviation (RMSD) for the selected bijection at its second phase. This second-phase minimization admits an analytical solution (Kearsley, 1989). Our approach here has been to maintain the first phase (and the Structal score) changing the second phase to preserve coherence. The opposite choice is possible. We may preserve the Procrustes second phase, employing, at the first phase, a different score. An entirely compatible score with the Procrustes second phase may be defined by

$$S(D, \Phi) = 20 \sum_{k \in \mathcal{D}(\Phi)} \max \left[0, 1 - \left(\frac{\| P_k - D(Q_{\Phi(k)}) \|}{d_0} \right)^2 \right] - 10 \times \text{gaps},$$

where $D, \Phi, \sum$ and gaps are as in (2.3) and d_0 is a threshold distance. If the distance between two Cα atoms (associated by Φ) exceeds d_0, its contribution to this score is zero. Maximizing this score is equivalent to minimizing the RMSD for the atoms for which d_i is less than d_0. Three methods based on different d_0 values are also available in the LovoAlign software package.

We conjecture that the LOVO methodology may be employed in connection to alignment and protein classification in a number of different related problems: conservation of residues in columns of a multiple sequence alignment (Liu *et al.*, 2006), percentage identity (Raghava & Barton, 2006), support vector machine (SVM) detection of distant structural relationships (Ogul & Mumcuoglu, 2006), protein–protein interfacial residual identification (Li *et al.*, 2006), prediction of subcellular localization (Kim *et al.*, 2006), 3D enzyme modelling (Singh *et al.*, 2006), determination of score coefficients (Kececioglu & Kim, 2006), hierarchical clustering (Gambin & Slonimski, 2005) and many others.

Acknowledgements

The authors are indebted to two anonymous referees, whose comments were very useful to improve the presentation of the paper. This paper is based on the talk given by J. M. Martínez at the International Conference on Numerical Analysis and Optimization, held in Beijing.

Funding

PRONEX-Optimization (76.79.1008-00); FAPESP (06/53768-0, 05/56773-1, 02-14203-6); PRONEX CNPq/ FAPERJ (26/171.164/2003-APQ1).

REFERENCES

ANDREANI, R., DUNDER, C. & MARTÍNEZ, J. M. (2005) Nonlinear-programming reformulation of the order-value optimization problem. *Math. Methods Oper. Res.*, **61**, 365–384.

ANDREANI, R., MARTÍNEZ, J. M., MARTÍNEZ, L. & YANO, F. (2008a) Continuous optimization methods for structural alignments. *Math. Program.*, **112**, 93–124.

ANDREANI, R., MARTÍNEZ, J. M., MARTÍNEZ, L. & YANO, F. (2008b) Low order value optimization and applications. *J. Glob. Optim.* (to appear).

ANDRETTA, M., BIRGIN, E. G. & MARTÍNEZ, J. M. (2005) Practical active set Euclidian trust-region method with spectral projected gradients for bound-constrained optimization. *Optimization*, **54**, 305–325.

BERMAN, H. M., WESTBROOK, J., FENG, Z., GILLILAND, G., BHAT, T. N., WEISSIG, H., SHINDYALOV, I. N. & BOURNE, P. E. (2000) The Protein Data Bank. *Nucleic Acids Res.*, **17**, 235–242.

CONN, A. R., GOULD, N. I. M. & TOINT, P. L. (2000) Trust-region methods. *Society of Industrial and Applied Mathematics (SIAM)*. Philadelphia, PA: SIAM.

DENNIS, J. E. & SCHNABEL, R. B. (1983) *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice-Hall. Reprinted by SIAM Publications, 1993.

DOLAN, E. & MORÉ, J. J. (2002) Benchmarking optimization software with performance profiles. *Math. Program.*, **91**, 201–213.

FLETCHER, R. (1987) *Practical Methods of Optimization*, 2nd edn. New York: John Wiley.

FRIEDLANDER, A., MARTÍNEZ, J. M. & SANTOS, S. A. (1994) A new trust-region algorithm for bound constrained minimization. *Appl. Math. Optim.*, **30**, 235–266.

GAMBIN, A. & SLONIMSKI, P. R. (2005) Hierarchical clustering based upon contextual alignment of proteins: a different way to approach phylogeny. *C. R. Biol.*, **328**, 11–22.

GERSTEIN, M. & LEVITT, M. (1998) Comprehensive assessment of automatic structural alignment against a manual standard, the Scop classification of proteins. *Protein Sci.*, **7**, 445–456.

HOLM, L. & SANDER, C. (1993) Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, **233**, 123–138.

HOLM, L. & SANDER, C. (1996) Mapping the protein universe. *Science*, **273**, 595–602.

HOU, J., JUN, S. R., ZHANG, C. & KIM, S. H. (2005) Global mapping of the protein structure space and application in structure-based inference of protein function. *Proc. Natl. Acad. Sci. USA*, **102**, 3651–3656.

HOU, J., SIMS, G. E., ZHANG, C. & KIM, S. H. (2003) A global representation of the protein fold space. *Proc. Natl. Acad. Sci. USA*, **100**, 2386–2390.

KABSCH, W. (1978) A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. A*, **34**, 827–829.

KEARSLEY, S. K. (1989) On the orthogonal transformation used for structural comparisons. *Acta Crystallogr. A*, **45**, 208–210.

KECECIOGLU, J. & KIM, E. (2006) Simple and fast inverse alignment. *Research in Computational Molecular Biology*. Proceedings Lecture Notes in Computer Science. A. Apostolico, C. Guerra, S. Istrail, P. A. Pevzner & M. S. Waterman eds) Berlin/Heidelberg: Springer. vol. 3909, pp. 441–455.

KIM, J. K., RAGHAVA, G. P. S., BANG, B. Y. & CHOI, S. J. (2006) Prediction of subcellular localization of proteins using pairwise sequence alignment and support vector machine. *Pattern Recognit. Lett.*, **27**, 996–1001.

KOLODNY, R., KOEHL, P. & LEVITT, M. (2005) Comprehensive evaluation of protein structure alignment methods: scoring by geometric measures. *J. Mol. Biol.*, **346**, 1173–1188.

KOLODNY, R. & LINIAL, N. (2004) Approximate protein structural alignment in polynomial time. *Proc. Natl. Acad. Sci. USA*, **101**, 12201–12206.

LI, J.-J., HUANG, D.-S., WANG, B. & CHEN, P. (2006) Identifying protein-protein interfacial residues in heterocomplexes using residue conservation scores. *Int. J. Biol. Macromol.*, **38**, 241–247.

LI, Z.-W., YE, Y.-Z. & GODZIK, A. (2006) Flexible structural neighborhood—a database of protein structural similarities and alignments. *Nucleic Acids Res.*, **34**, D277–D280.

LIN, C. & MORÉ, J. J. (1999) Newton's method for large bound-constrained optimization problems. *SIAM J. Optim.*, **9**, 1100–1127.

LIU, X.-S., LI, J., GUO, W.-L. & WANG, W. (2006) A new method for quantifying residue conservation and its applications to the protein folding nucleus. *Biochem. Biophys. Res. Commun.*, **351**, 1031–1036.

LU, F., KELES, S., WRIGHT, S. J. & WAHBA, G. (2005) Framework for kernel regularization with application to protein clustering. *Proc. Natl. Acad. Sci. USA*, **102**, 12332–12337.

MARTINEZ, L., ANDREANI, R. & MARTÍNEZ, J. M. (2007) Convergent algorithms for protein structural alignment. *BMC Bioinformatics*, **8**, 306.

MORÉ, J. J. (1983) Recent developments in algorithms and software for trust-region methods. *Mathematical Programming: The State of the Art* (A. Bachem, M. Grötschel & B. Korte eds). Berlin/Heidelberg: Springer, pp. 258–287.

MORÉ, J. J. & SORENSEN, D. C. (1983) Computing a trust-region step. *SIAM J. Sci. Stat. Comput.*, **4**, 553–572.

NEEDLEMAN, B. & WUNSCH, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453.

NOCEDAL, J. & WRIGHT, S. J. (1999) *Numerical Optimization*. New York: Springer.

OGUL, H. & MUMCUOGLU, E. U. (2006) SVM-based detection of distant protein structural relationships using pairwise probabilistic suffix trees. *Comput. Biol. Chem.*, **30**, 292–299.

ONUCHIC, J. N. & WOLYNES, P. G. (2004) Theory of protein folding. *Curr. Opin. Struct. Biol.*, **14**, 70–75.

POWELL, M. J. D. (1970) A new algorithm for unconstrained optimization. *Nonlinear Programming* (J. B. Rosen, O. L. Mangasarian & K. Ritter eds). New York: Academic Press, pp. 31–65.

RAGHAVA, G. P. S. & BARTON, G. J. (2006) Quantification of the variation in percentage identity for protein sequence alignments. *BMC Bioinformatics*, **7**, 415.

SALI, A. & BLUNDELL, T. L. (1993) Comparative protein modeling by satisfaction of spatial restraints. *J. Mol. Biol.*, **234**, 779–815.

SHATSKY, M., NUSSINOV, R. & WOLFSON, H. J. (2002) Flexible protein alignment and hinge detection. *Proteins*, **48**, 242–256.

SINGH, N., CHEVE, G., AVERY, M. A. & MCCURDY, C. R. (2006) Comparative protein modeling of 1-deoxy-D-xylulose-5-phosphate reductoisomerase enzyme from Plasmodium falciparum: a potential target for antimalarial drug discovery. *J. Chem. Inf. Model.*, **46,** 1360–1370.

SUBBIAH, S., LAURENTS, D. V. & LEVITT, M. (1993) Structural similarity of DNA-binding domains of bacteriophage repressors and the globin core. *Curr. Biol.*, **3**, 141–148.

VENDRUSCOLO, M. & DOBSON, C. M. (2005) A glimpse at the organization of the protein universe. *Proc. Natl. Acad. Sci. USA*, **102**, 5641–5642.

VOET, D. & VOET, J. (2004) *Biochemistry*, 3rd edn. New Jersey: John Wiley.

YE, Y.-Z. & GODZIK, A. (2003) Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, **19**, 246–255.